D.K.T.E. Society’s Textile and Engineering Institute, Ichalkaranji.

(An Autonomous Institute, Affiliated to Shivaji University, Kolhapur)

Department of Information Technology

2019-2020

Project-II Report On

Decentralized Social Media Network Using Blockchain

Under the Guidance Of

Prof. V. V. Kheradkar

Submitted By:

1.Mr. Rohit N Waldekar	16UIT12057XX
2.Miss. Rohini R Koli	16UIT11026XX
3. Mr. Ranvirsinh R Patil	16UIT12039XX
4.Miss. Shraddha S Pattanshetti	16UIT51062XX

Prof. V. V. Kheradkar	Prof. (Dr).D.V. Kodavde	Pro (Dr).P.V.KADOLE
(Project Guide)	**(H.O.D)**	**(Director)**

DKTE

Promoting Excellence in
Teaching, Learning & Research

YEAR 2019-2020

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that the project report entitles "Decentralized Social Media Network Using Blockchain" is record of project work carried out in this college by,

1.Mr. Rohit N Waldekar (16UIT12057XX)

2.Miss. Rohini R Koli (16UIT11026XX)

3.Mr. Ranvirsinh R Patil (16UIT12039XX)

4.Miss. Shraddha S Pattanshetti (16UIT51062XX)

in fulfillment of the requirement for degree of BACHELOR OF TECHNOLOGY in INFORMATION TECHNOLOGY of SHIVAJI UNIVERSITY, KOLHAPUR. This project report is record of their own work carried out under my supervision and guidance during academic year 2019-2020.

Prof. V. V. Kheradkar	Prof. (Dr.) P.V. KADOLE	Prof. (Dr.) D. V. Kodavde
[Project Guide]	**[Director]**	[**Head of the Department]**

DECLARATION

We the undersigned students of B.Tech Information Technology declare that, the field work report entitled Decentralized Social Media Network Using Blockchain written and submitted under the guidance of Prof. V. V. Kheradkar is our original work. The empirical findings in this report are based on the data collected by us. The matter assimilated in this report is not reproduction from any readymade report.

Date:
Place: Ichalkaranji.

Members	Signature
1.Mr. Rohit N Waldekar (16UIT12057XX)	
2.Miss. Rohini R Koli (16UIT11026XX)	
3.Mr. Ranvirsinh R Patil (16UIT12039XX)	
4.Miss. Shraddha S Pattanshetti (16UIT51062XX)	

ACKNOWLEDGMENT

With great pleasure we wish to express our deep sense of gratitude to prof. V. V. Kheradkar for his valuable guidelines, support and encouragement in completion of project report.

Also, we would like to take opportunity to thank our head of department Dr. D. V. Kodavde for his co-operation in preparing this project report.

We feel gratified to record our cordial thank to other staff members of Information Technology department for their support, help and assistance which they extended as and when required.

Thank you.

1. Mr. Rohit N Waldekar	16UIT12057XX
2.Miss. Rohini R Koli	16UIT11026XX
3.Mr. Ranvirsinh R Patil	16UIT12039XX
4.Miss. Shraddha S Pattanshetti	16UIT51062XX

ABSTRACT

Now a days, Social networking has influenced billions of users. Most social media sites extract the data for his or her benefit. Lack of trust, transparency and control over central authority that furnish such networks has dropped at light the adverse aspects of centralization.

Development of blockchain technology has greatly changed the network and it has made applications distributed, decentralized without loss of security. Implementation of decentralized social media platform, that provides user the power to interact, connect, share contents and message, while giving reward for his or her activities on the idea of proof of social algorithm.

Decentralized social networks have the potential to produce a stronger environment within which users can have more control over their privacy, and therefore the ownership and dissemination of their information.

Therefore, online social networking is going to be proof against censorship, monopoly, regulation, and other exercise of central authority. More importantly, a decentralized approach to online social networking breaks the boundaries between social networking sites by providing users more freedom to interact with one another.

KEYWORDS

1. Node
2. Ethereum Network
3. Distributed and Decentralized platform
4. Authentication and security
5. Transparency
6. Hash value and Transaction
7. Interplanetary File System (IPFS)
8. Angular and TypeScript
9. System Testing and Maintenance
10. Smart Contract

INDEX

CONTENTS **PAGE**

1. INTRODUCTION

1.1 Problem Statement

Implement fully decentralized social media network for users and to share contents over network in peer to peer manner without storing data on centralized server.

1.2 Need of the project with motivating example

With growing concerns a few perceived censorship is biased on social media platforms, demands for censorship-resistant social media networks is on the increase. Variety of blockchain startups aim to use a decentralized model to preserve free speech within the digital age. Example of this sort of social media platforms are Pocketnet, Steem etc.

1.3 Objective of Project

1. To study in detail existing centralized applications and need of blockchain to overcome existing problems in centralized system.
2. To implement fully decentralized social media network using blockchain.
3. To store data in decentralized storage network and give ownership to users.

1.4 Limitations and Scope

1.4.1 Existing System

Users and organizations are now connected, interacting and sharing data among themselves at a huge level. There is a crisis occurring within the social networking world. Most social media sites extract the knowledge for his or her benefit. The infrastructure of such services has been traditionally supported by centralized networks. However, lack of trust, transparency and control over central authority that furnish such networks has dropped at light the adverse aspects of centralization. Surveillance, algorithm manipulation and censorship are infecting online at unprecedented levels. As we increasingly become awake to the flip-side of the boom in social media usage, we cannot help but specialise in two pertinent concerns. One, the privacy and security of personal data. Two, the consolidation of power within the hands of some controlling the centralized systems.

1.4.2 Limitations

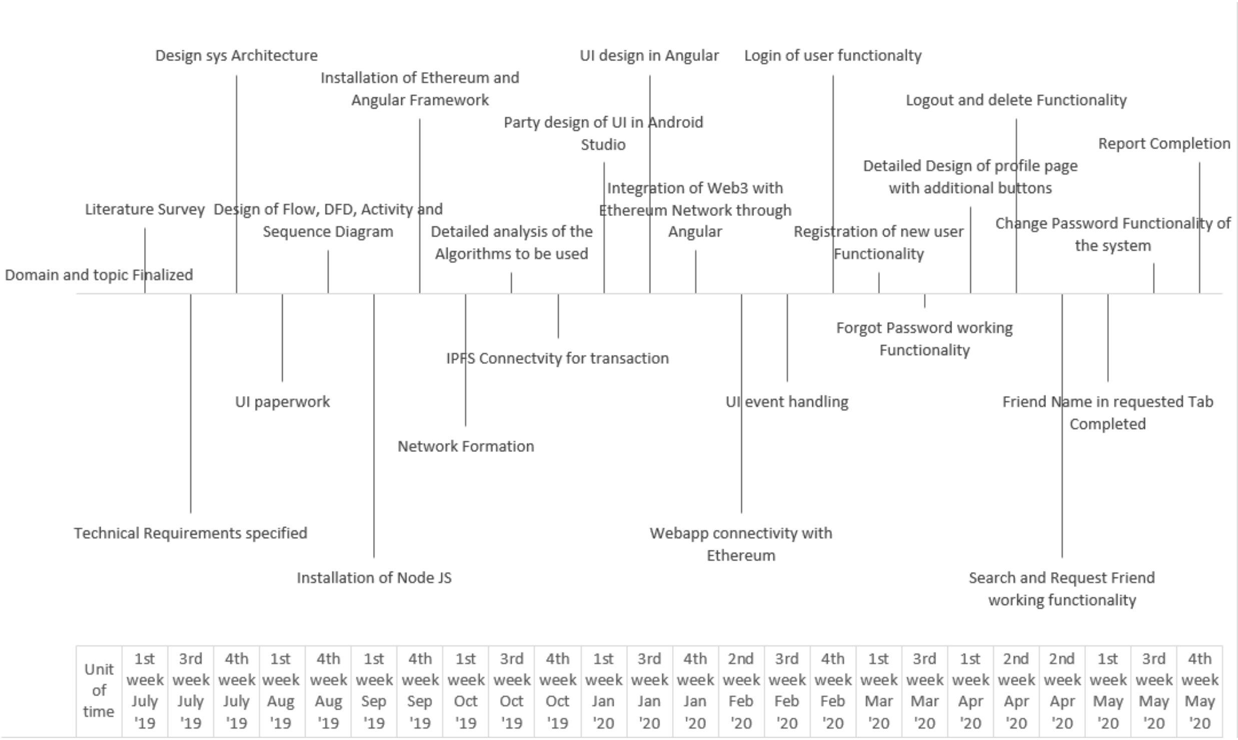
Existing system has following limitation-

- Centralization of data
- Social media piracy
- Less productivity
- Lack of Transparency and Security

1.4.3 Scope

1. Crowd Funding platform.
2. Platform for solving Environmental issues and social issues.
3. User can raise money for his original contents.
4. Gain popularity by more interactivity with platform.

1.5 Timeline for project


Design sys Architecture
Installation of Ethereum and Angular Framework
UI design in Angular
Login of user functionalty
Logout and delete Functionality
Party design of UI in Android Studio
Report Completion
Detailed Design of profile page with additional buttons
Integration of Web3 with Ethereum Network through Angular
Literature Survey
Design of Flow, DFD, Activity and Sequence Diagram
Detailed analysis of the Algorithms to be used
Registration of new user Functionality
Change Password Functionality of the system
Domain and topic Finalized
Forgot Password working Functionality
IPFS Connectvity for transaction
UI paperwork
UI event handling
Friend Name in requested Tab Completed
Network Formation
Technical Requirements specified
Webapp connectivity with Ethereum
Installation of Node JS
Search and Request Friend working functionality

Unit of time	1st week July '19	3rd week July '19	4th week July '19	1st week Aug '19	4th week Aug '19	1st week Sep '19	4th week Sep '19	1st week Oct '19	3rd week Oct '19	4th week Oct '19	1st week Jan '20	3rd week Jan '20	4th week Jan '20	2nd week Feb '20	3rd week Feb '20	4th week Feb '20	1st week Mar '20	3rd week Mar '20	1st week Apr '20	2nd week Apr '20	2nd week Apr '20	1st week May '20	3rd week May '20	4th week May '20

1.6 Cost of project:

Sr No.	**Equipment**	**Details**	Price(Rs)
1.	Electricity	Consumed 40 units	120
2.	Internet	1000 GB	2500
3.	Cloud	Virtual machine	500

Estimated cost by considering other factors will be approx. **- Rs. 3120/-**

1.5.1 COCOMO Model

In this project, the Cost Estimation based on COCOMO (Constructive Cost Model) the formula for this Model is follows:
Effort = Constant × (Size) scale factor× Effort Multiplier

– Effort in terms of person-months
– Constant: 2.45 in 1998 based on Organic Mode
– Size: Estimated Size in KLOC
– Scale Factor: combined process factors
– Effort Multiplier (EM): combined effort factors

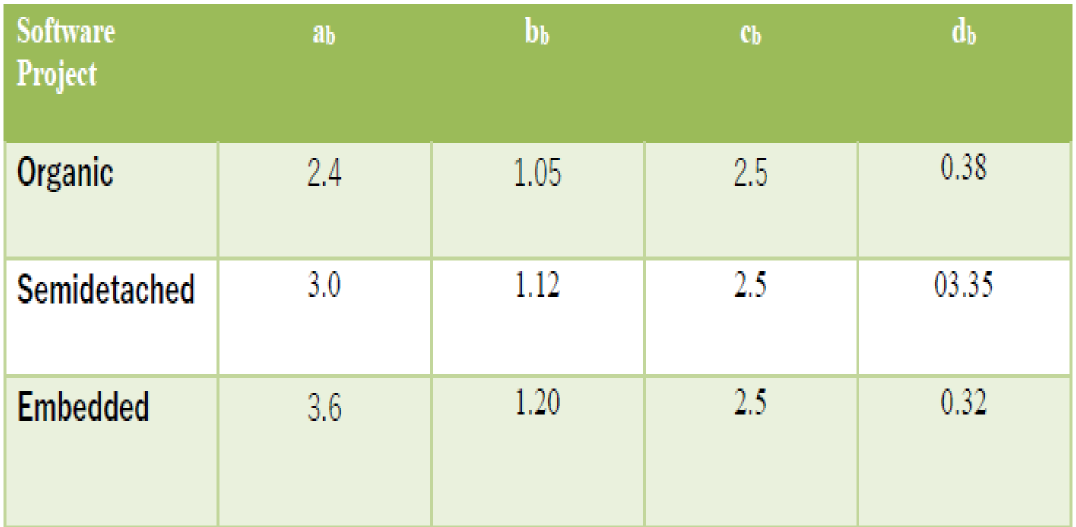
Functional Point Table

The function point range in between 1-10
Conversion of Functional point to Lines of Code (LOC)
Total function points = 6

■ Estimated Size – 7820 LOC

The basic COCOMO equations take the form
Effort Applied (E) = ab (KLOC) bb [man-months]
Development Time (D) = cb (Effort Applied) db [months]
People required (P) = Effort Applied / Development Time [count]

Where, KLOC is the estimated number of delivered lines (expressed in thousands) of code for project. The coefficients ab, bb, cb and db are given in the following table.



Software Project	a_b	b_b	c_b	d_b
Organic	2.4	1.05	2.5	0.38
Semidetached	3.0	1.12	2.5	03.35
Embedded	3.6	1.20	2.5	0.32

Organic Mode:

Effort Applied (E) = 2.4*(7120) *1.05=17942.4

Development Time (D) = 2.5*(4) *0.4= 4.0

People Required (P) = 17942.4/4.0 = 4 people

2. BACKGROUND STUDY AND LITERATURE REVIEW

2.1 Technology review

A blockchain is a list of records, called blocks, that are linked using cryptography. It's an open, distributed ledger which records transactions between two parties efficiently and in a very verifiable and permanent way.

Increasing use of blockchain technology has greatly changed the network and it has made applications more distributed and decentralized without loss of security. Implementation of decentralized social media platform offers user the power to communicate, connect, share contents and message, while rewarding his or her activities on the premise of proof of social algorithm. Decentralization can provide better privacy to the users by removing the central authority.

This may help to protect user's personal data and provides them at the most privacy. This can be safe, transparent, decentralized and rewarding platform. This decentralized social network application on the Ethereum private blockchain uses smart contract and Interplanetary File System (IPFS).

In addition to this, it also examines how blockchain and distributed storage can improve functionalities of traditional social network systems.

2.2 Literature review

We found many of the literature papers based on social media network working with blockchain technology. Some of the literatures are as follows-

2.2.1. Minds

In the Minds contribution economy, users and developers will be rewarded for a variety of contributions to the network including generation of high-quality content, account setup and verification, referring new users, maintaining an active channel, finding bugs, successfully submitting code and more. Tokenized rewards provide the key incentive to help faster growth and long-term sustainability of the network.

2.2.2. Steem

Steem aims to support social media and online communities by returning much of its value to the people who provide valuable contributions by rewarding them with cryptocurrency, and through this process create a currency that is able to reach a broad market, including people who have yet to participate in any cryptocurrency economy.

2.2.3. Pocketnet

On the Pocketnet no centralized entity exists that can disenfranchise creators by reducing their share of pay after they achieve success. Each creator earns an amount of Pocketcoin emission proportional to the success of their contributions to the platform. In addition, Pocketnet Direct Marketplace for self-serve advertising, allows advertisement (ad) from specific creators using trustless multi signature transactions. Ads can be pre designed or custom placement, where the creator has freedom in presenting the ad. This is different from traditional platforms where vast majority of wealth is now clawed back by the shareholders of platforms.

3. REQUIREMENT ANALYSIS

3.1 Functional requirements

3.1.1 User Interface requirements

1. User should be able to register him/her on the application.

2. User must create his profile in application.

3. Friend request and acceptance facility should be provided.

4. User should be able to chat with others in textual and Emoji format.

5. Initially trending groups suggestion for joining must be provided.

6. User should be capable to back up the data on his own device.

7. Keep log in facility must be there.

8. Privacy criteria like notification allow/blocked, new message, message seen etc should be provided.

9. New group formation, Search friends these functions should be provided.

10. Content - Copy, Delete, Share, Forward facility should be provided.

3.1.2 Hardware interface requirements

1. Visual studio code – it used for develop and deploy the frontend code.
2. Microsoft Azur portal- It is used for integration of project.
3. Browser – to run the application
4. Web3 JS – to connect Ethereum and angular
5. remix IDE- to develop the contracts and deploy for getting the ABI code
6. Linux OS platform- to develop and run the Ethereum network.

3.1.3 Communication protocol requirements

1. Proof of work protocol-

The early blockchains were built around the concept of proof of work. The amount of work performed is measured in terms of computational contributions, also called as mining. All nodes in the network compete to mine for a new block by solving some partial collision using hash functions. However, this type of validation could be more inefficient in term of energy. Therefore, it is expensive because the work performed by miners that do not get validated are wasted. This incentivises nodes to centralize the hashing power into pools, which obviously is not desirable for a network whose goal is to minimize the need to trust third parties.

3.1.4. Modular/ Component Requirements

Project contain following modules/components which are the backbone of social media network -

1. **Network formation** –

We have used an Ethereum platform to develop our social media application. It is a blockchain-based decentralized platform featuring smart contract functionality. It supports stateful contracts in which values can persist on the blockchain network.

Following are the steps to create the network-

1. install node -

Node is a platform built on JavaScript runtime for building fast and scalable network applications.

2. npm install or update -

npm is the package manager for the Node JavaScript platform. It puts modules in place so that nodes can find them, and manages dependency conflicts intelligently. Most commonly, it used to publish, discover, install, and develop node programs.

3.install Ethereum -

```
sudo apt-get update
sudo apt-get -y install ethereum
```

Both commands are used to install ethereum on the host machine. It will create a small readymade network.

4. create a genesis file -

The genesis block is the start of the blockchain, and the genesis.json is the file that defines it. It is like the “settings” for your blockchain. For example, the chain configuration, level of difficulty to mine blocks, etc. There are four values (config, difficulty, gas Limit, alloc) you need to specify in genesis.json.

5. Configure network –

1. create an account -

```
geth --datadir ethdata account new
```

Account has the address where anyone can Communicate.

2.Start Node –

This will initiate the network.

3.Connect-

```
admin.nodeInfo.enode
admin.peer
```

Network establishment will take place. we can check using these commands.

4.Start Mine -

```
mine.start()
mine.stop()
```

These commands will put you in loop for continuously checking request for the data over the network.

2. Smart contract - This module contains business logic and functionality in the form of smart contract.

This module contains business logic and functionality in the form of smart contract. Smart contracts are lines of code that are stored on a blockchain and automatically execute when predetermined terms and conditions are met. At the most basic level, they are programs that run as they’ve been set up to run by the people who developed them. The benefits of smart contracts are most apparent in business collaborations, in which they are typically used to enforce some type of agreement so that all participants can be certain of the outcome without an intermediary's involvement.

We have used solidity language and remix IDE for the development of smart contracts. The version specification in solidity is must because it is continuously in development stage. So still there is no its stable version available. In this we have created the functions to create a new account, register for the web app, change password, etc. after creating the Smart contract we have deployed that contract in the network. After Successful deployment, the ABI will be generated. ABI is the application binary interface. In this file, our code is encoded into binary format. we have to just copy and paste this in source code.

In this way, we can develop and deploy the number of smart contracts with different functionalities.

3. web app - We are developing social media web app.

We are have developed the web app for social media.

Steps in web app development

1. install the visual studio code IDE-

it is open Source platform. We can use any another IDE. It supports a number of programming languages and a set of features that may or may not be available for a given language. It also has an extension for Solidity which would help us in writing Smart Contracts using this IDE.

2. install angular and its dependencies-

we have installed the angular 8 version. This will help to develop our frontend. We can use java, Go language for development. We have chosen the angular because it is simple and many of the readymade tags are readymade. HTML Support is available. In angular all the files are keep separately so it's directory structure will reduce the time for development.

3. install web3.js –

```
npm install ethereum/web3.js –save
```

Using this command we can install web3.js in visual studio code environment.It will act as interface between frontend and backend. Our request is send to bkockchain network and come up with response done by web3.The time for getting response required is extra. because the miners will mine our data and give the feedback whether it is trust worthy or not.Web3 connect the network with the help of AbI of each smart contract.

4. install browser for testing purpose-

We have used the google chrome browser for the testing purpose.

4. **Interplanetary File System (IPFS)** – This module contains code of storing data on IPFS platform.

This module contains code of storing data on IPFS platform. The Interplanetary File System (IPFS) is a protocol and peer-to-peer network for storing and sharing data in a distributed file system. IPFS uses content-addressing to uniquely identify each file in a global namespace connecting all computing devices. IPFS allows users to not only receive but host content, in a similar manner to BitTorrent. As opposed to a centrally located server, IPFS is built around a decentralized system of user-operators who hold a portion of the overall data, creating a resilient system of file storage and sharing. Any user in the network can serve a file by its content address, and other peers in the network can find and request that content from any node who has it using a distributed hash table (DHT).

```
npm install ipfs-api
```

In our application, we have installed the API of ipfs and mentioned its path in the library file of Source code. Using this API we can Store our text messages, images, videos on the Storage Server. It will create a unique hash Id for each thing. Using that hash ID we can globally access that thing.

3.2 System Requirements

3.2.1 Hardware Requirements

1. Machine with minimum 4 GB of RAM for developing environment and formation of the network.

3.2.2 Operating System Requirements

1. Ubuntu version 16.04 LTS for installation of node and Ethereum framework.

3.2.3 Application or Web server requirements

1. Any type of browser which is compatible with web3 can be used to access the application.

3.2.4 Storage requirement

1. We have used the IPFS storage for store the images and chats and get there Address in the form of hash value.

3.2.5 Tools and Technologies Requirements

1.Node – Install the npm packages

2. Ethereum Blockchain - The Main backbone for public network development.

3. Visual code studio- The IDE for UI development.

4. IPFS- File system for data storage

4. SYSTEM DESIGN

4.1 Architecture Diagram

Following diagram shows the interaction between components and modules of the project

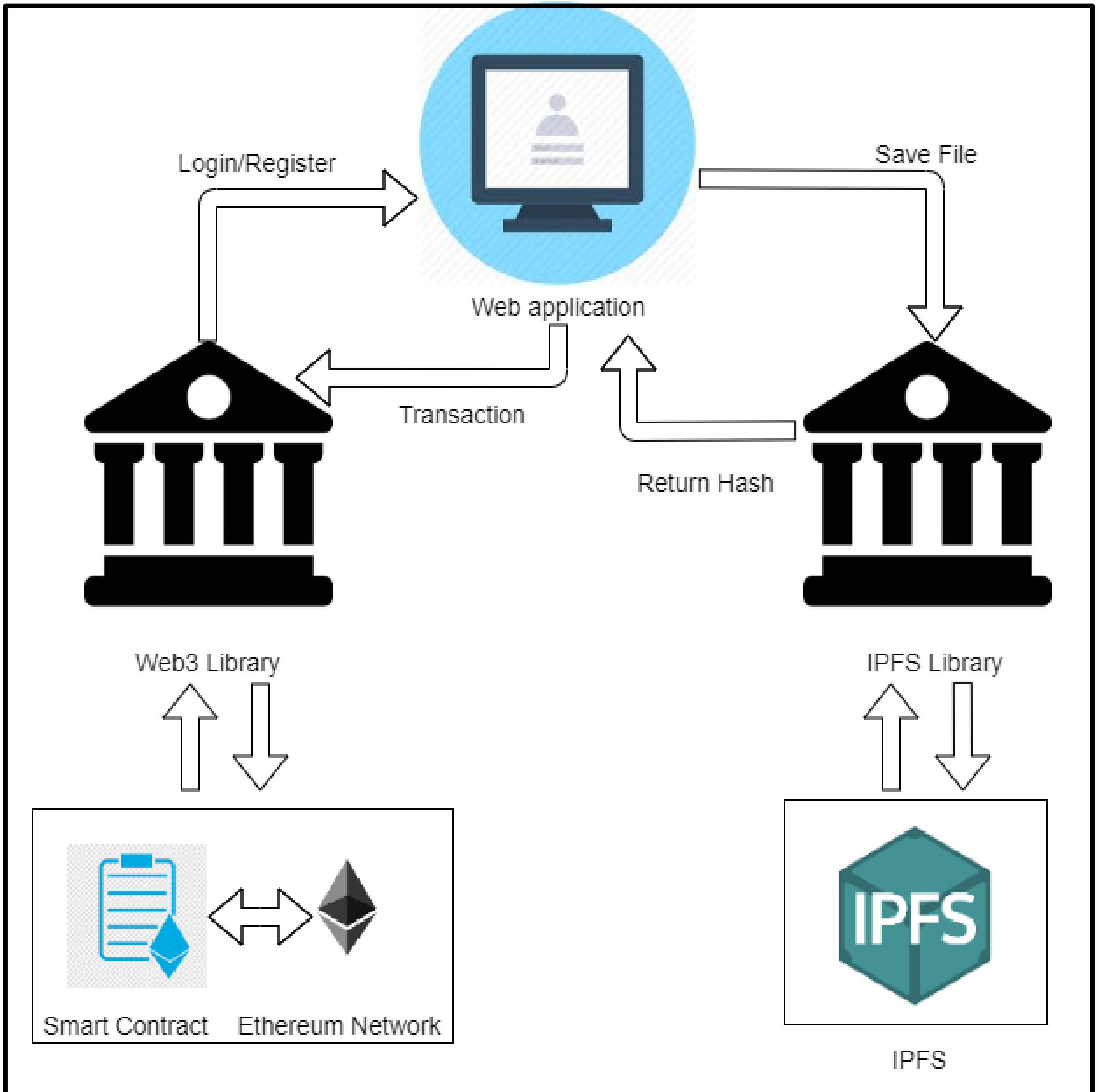


Fig 4.1 Architecture Diagram of Application

4.1.1 Components in Architecture diagram

1) User Interface

It is Web app User Interface. By using this application every user will interact with other users through network. Whenever any user performs any action or transaction, it will be added into the network in the form of ledger.

2) Web3 and IPFS Libraries

These libraries act as an interface to interact with network by using application. The Web3 library is used to connect application to decentralized network. Similarly, IPFS library is used to connect application to decentralized storage system.

3) Network

It is decentralized peer to peer network, implemented by using Ethereum full node geth. Network will automatically synchronize data between nodes. Also, every node has capability of Mining.

4) Smart Contract

It is program written in solidity language, consist of business logic or we can say that the main functionality of an application. This program is compiled. After compilation it gives two outputs, one is Application Binary Interface and other is Bytecode. These outputs are deployed on network. Application Binary Interface is used for communicating with network in the application.

5) IPFS storage

This is Blockchain based distributed web system. We can store our data, host application and many more. Any user can upload files and delete files from the IPFS web network. Distributed storage developed to reduce the cost of the server-side hardware and increase data availability. Inter Planetary File System (IPFS) is a protocol for distributed storage. IPFS stores immutable data, remove duplication, and obtains address information for storage nodes for searching files in the network.

4.2 Use Case Diagram

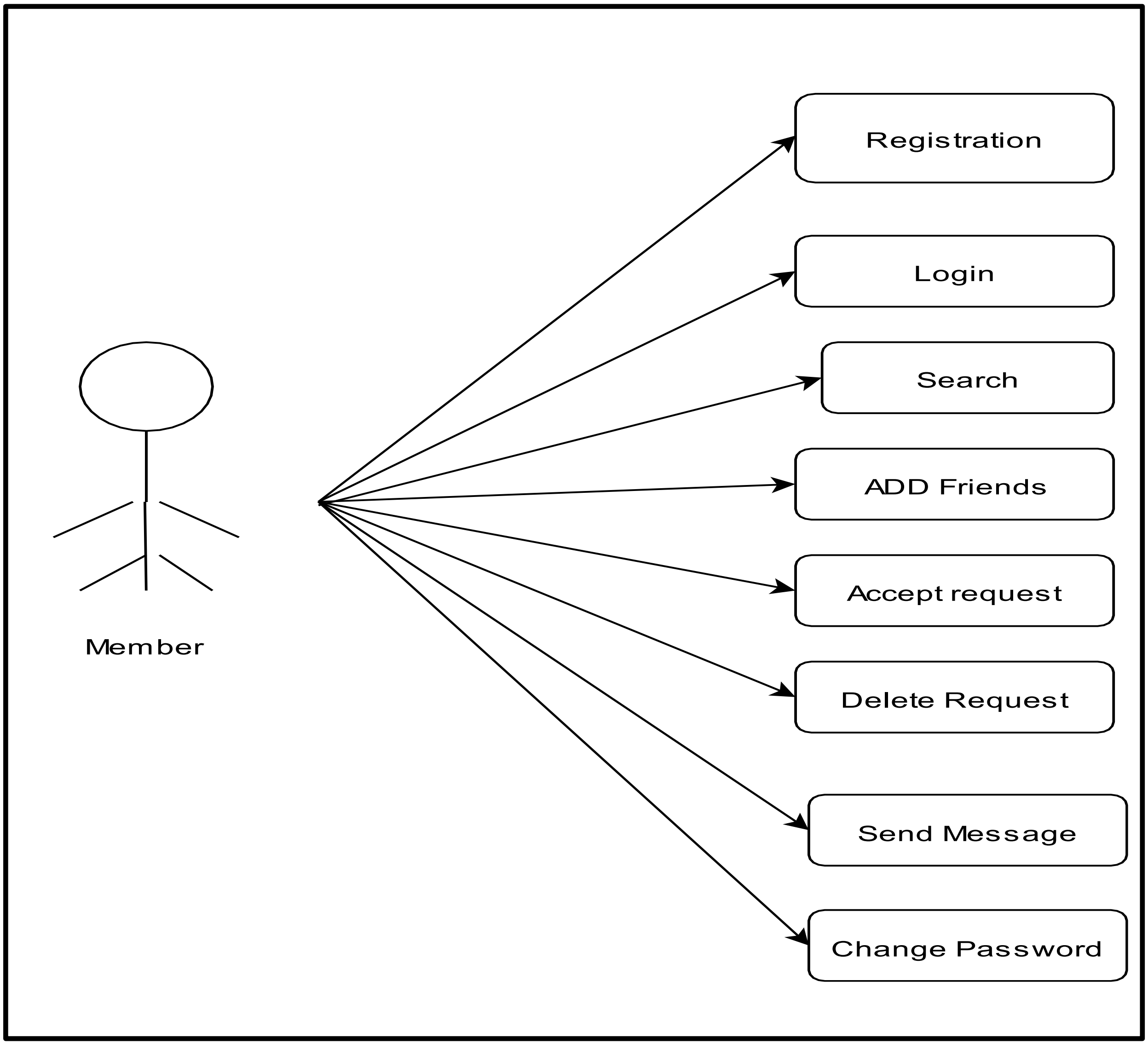


Fig 4.2 Use Case Diagram

In above diagram, main functionalities are shown. These functionalities are used by client to create the account and use it for chatting with their friends.

4.2.1 Algorithmic description of each module

1. Find friend algorithm-

   Input – user data

   Output – suggestion of new friends

   Description- This algorithm is based on common friend relationship along with interest.

2. Password recovery algorithm-

   Input – user data

   Output – successfully recovered password with change in password

   Description- This algorithm is used in case if client have forgot the password to recover his account and change password.

3. Content showing algorithm-

   Input – user data

   Output – content uploaded successfully

   Description- This algorithm is based on recently updated contents ordered by favourite friends

4. Send and Accept friend request algorithm-

   Input – User name of Friend

   Output– Successfully Send/Accept friend request

   Description– In this algorithm we are going to Send and Accept friend request by using the user name of friend to whom we want to send or we want to accept the request.

4.3 Data Flow Diagram

4.3.1 DFD level 0

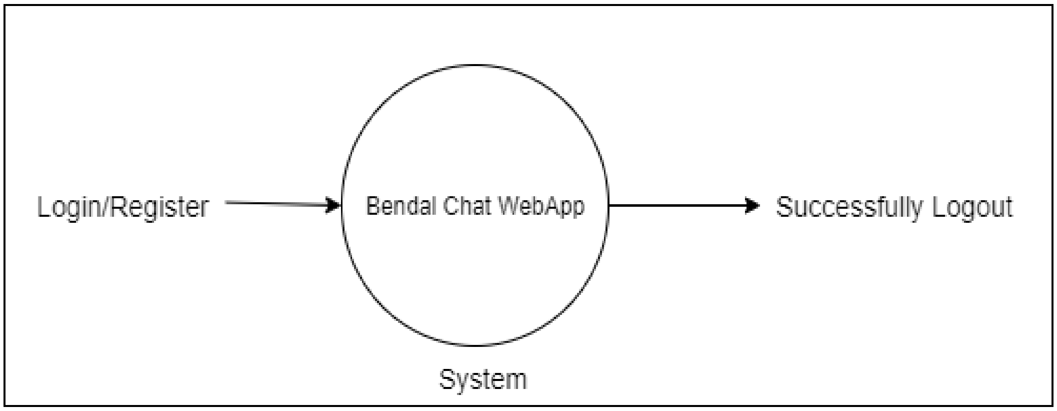


Fig 4.3.1 Diagram of DFD Level 0

As shown in diagram firstly user have to be register if his/her account does not exist, if user has registered then he/she can access the web application by giving valid login credentials.

4.3.2 DFD level 1

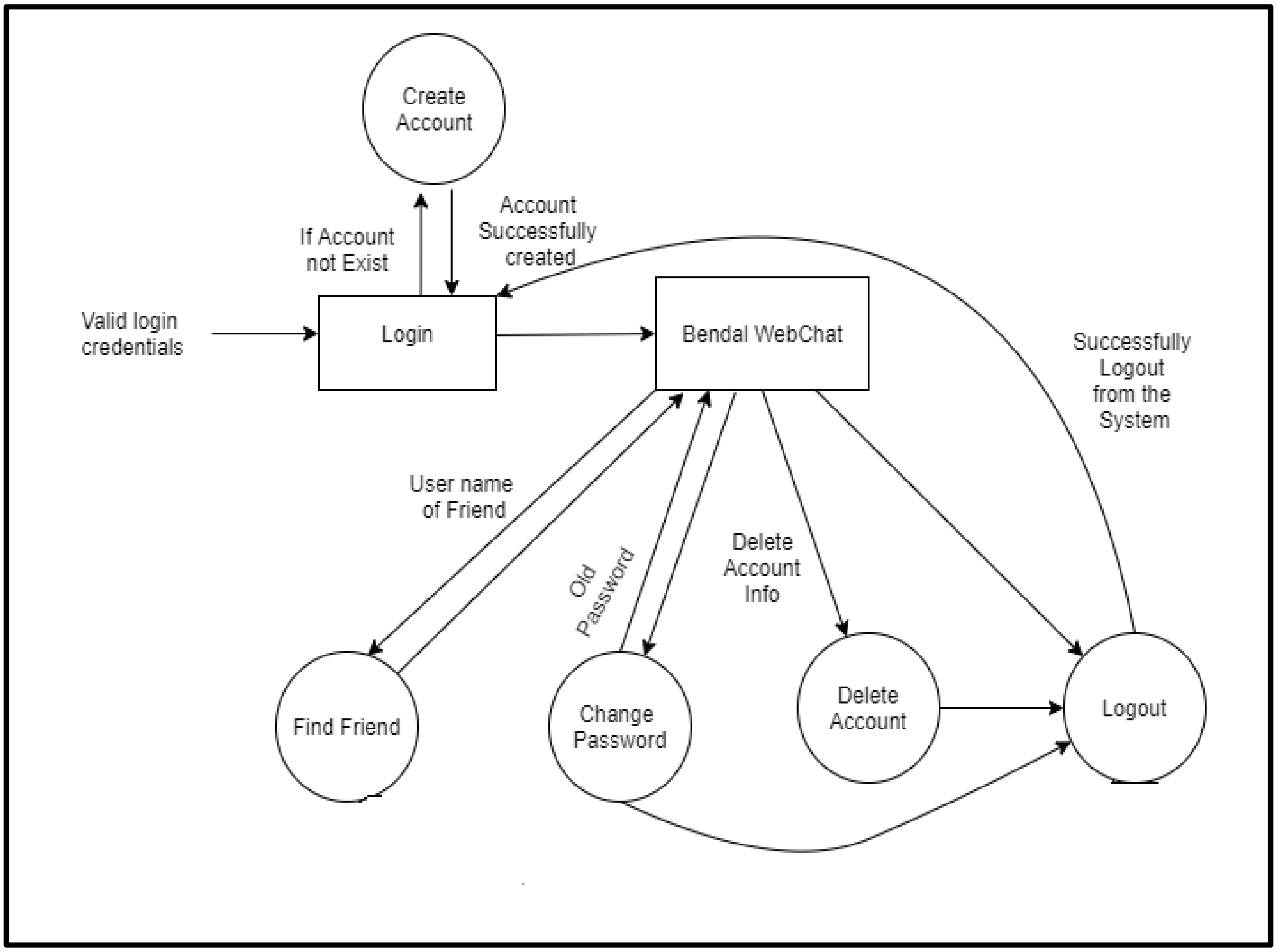


Fig 4.3.2 Diagram of DFD level 1

As shown in diagram the specific functionality of each module is given. The corresponding input and output are shown on the arrow. The arrow shows the flow of the system.

4.4 Sequence diagram

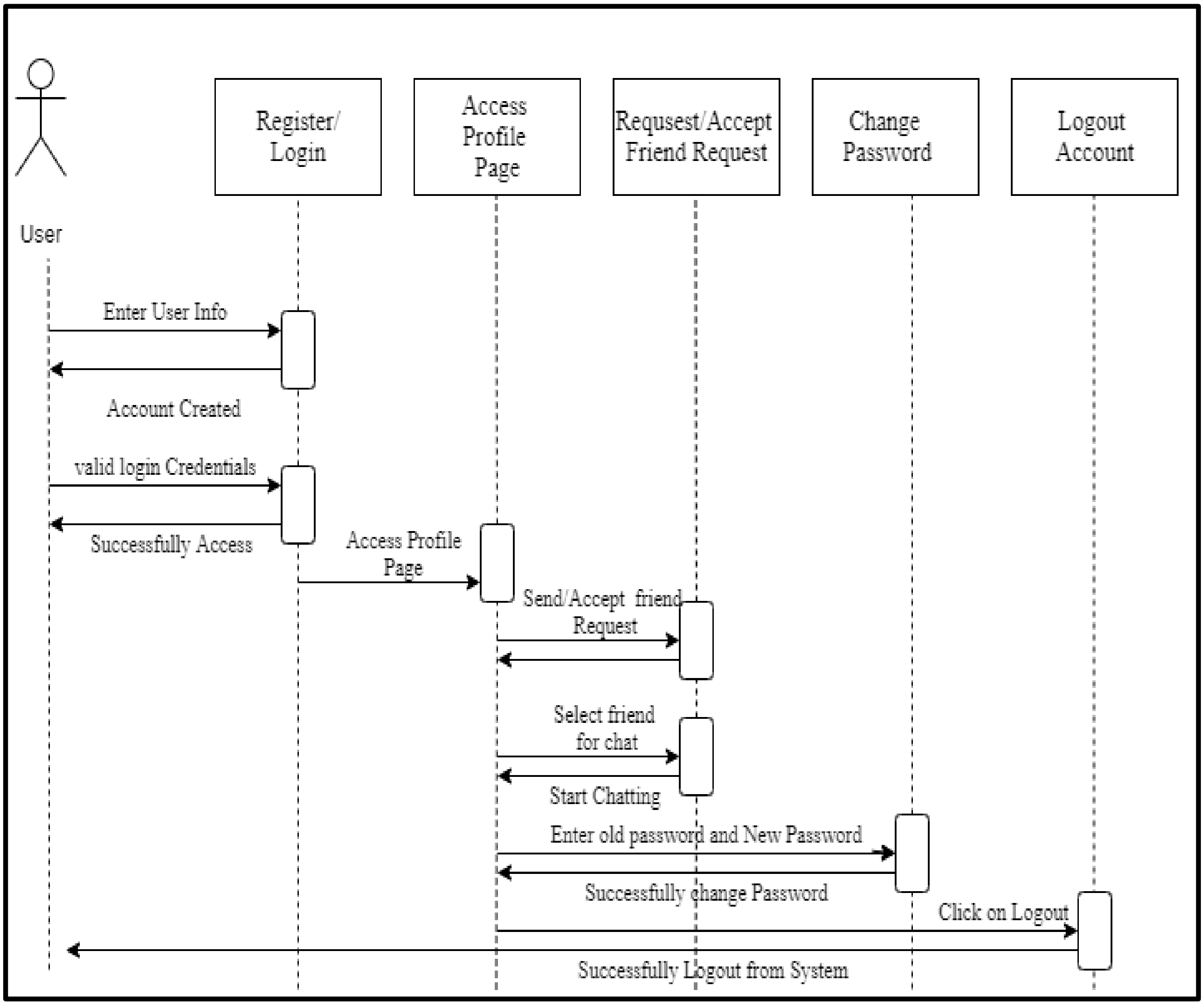


Fig 4.4 Sequence Diagram of Application.

In the above diagram the sequence of several activities are shown which are carried out in specific manner.

4.5 Class Diagram

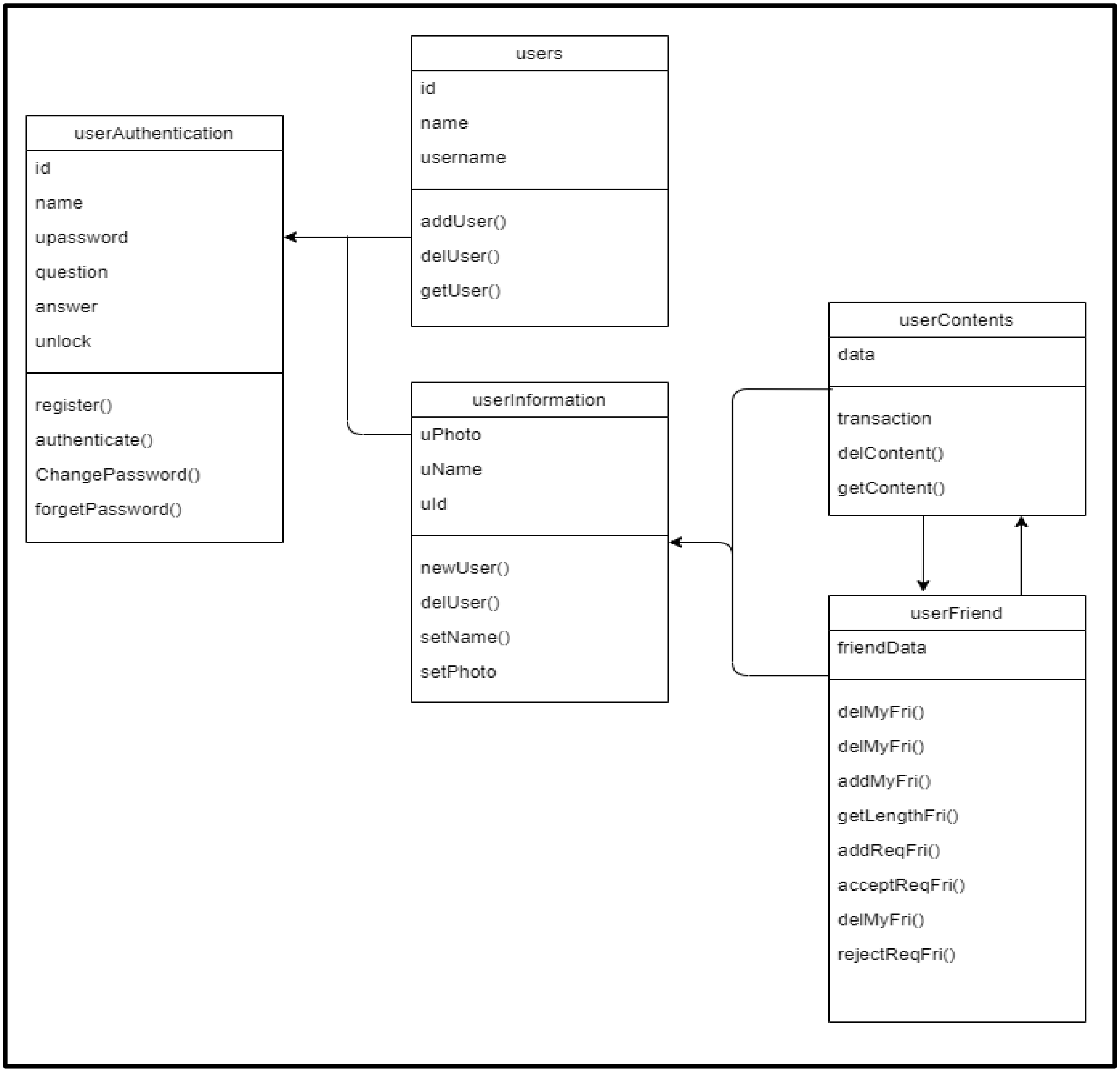


Fig 4.5 Class Diagram of Bendal Web application

There are five classes named users, userInformation, userContents, userFriend, userAuthentication containing functions as shown in above figure.

4.5.1 Description of each class in above figure is as follows,

1. **class name**: users

   Description: User has unique ID, Name and Username after registering with the app.

   Data Members: ID, Name and Username are the Data Members.

   Member Function Description: Users can register themselves, delete the account whenever they want and we can also view all the users registered to the app.

2. **class name**: userAuthentication

   Description: While registering user should enter Id, Name, Password and should provide security question and respective answer in case if user forgets username or password while signing in.

   Data Members: ID, Name and Password, Question and Answer are the Data Members.

   Member Function Description: User should firstly register with the app. Then the user can log in into the app. If in case user forgets password or username, user can reset as well as change the password. Authentication is done while registering with app.

3. **class name**: userInformation

   Description: user can also add profile picture along with name and unique userid

   Data Members: uPhoto, uName and uId are the Data Members.

   Member Function Description: Users can register themselves, delete the account whenever they want, can set as well as change their name to whatever they want and can change the profile pic whenever required.

4. **class name**: userContents

   Description: user can chat with friends, share and like posts of other friends.

   Data Members: data is the Data Member.

   Member Function Description: user can chat with friends, share and like posts of other friends, can delete the shared post and can view contents shared by other friend any time.

5. **class name**: userFriend

Description: user can chat with friends,share and like posts of other friends.

Data Members: friendData is the Data Member.

Member Function Description: user can request friend for connecting, user can unfollow the friend anytime, can add friend by accepting the request, user can reject the friend request.

5. IMPLEMENTATION

5.1 Environmental settings for running the module –

1. Install NodeJS
2. Install geth Ethereum node
3. Install Visual Studio framework
4. Form the network
5. Deploy the smart contract
6. Run the Application

5.2 Installation of Node

Node provides the basic infrastructure for installation of dependencies.
Commands to install node –

```
sudo apt-get update
sudo apt install nodejs
```

The nodejs packet contain both node and npm binaries.

5.3 Creation of Ethereum Network

Ethereum is an open source, public, blockchain-based distributed computing platform and operating system featuring smart contract (scripting) functionality. Ethereum network cab be created by following commands –

```
sudo apt-get -y install Ethereum
```

This command will create the Ethereum network automatically. Integrate this network with application created. To connect Ethereum and typescript, We are using web3. This web3 have protocols like rpc call, json call to connect with Ethereum.

5.4 Creating Account

After creation of network, account has to be created , by using this account mining will be initiated. Use following commands to create account.

- geth --datadir node1 account new

- geth --nousb --identity "node1" --datadir ./node1 --ipcdisable --port "30303" --networkid 2018 --nodiscover --rpc --rpcport "8545" --rpcaddr "0.0.0.0" --

rpccorsdomain "*" --rpcapi "eth,net,web3,miner,debug,personal,rpc" console --ws --wsport 8546 --wsaddr 127.0.0.1 --allow-insecure-unlock 0xAf62c70e88B5693D890EeEeC1d8b84471f414e84 --password rohit

- web3.personal.unlockAccount(eth.accounts[0],'rohit',15000)

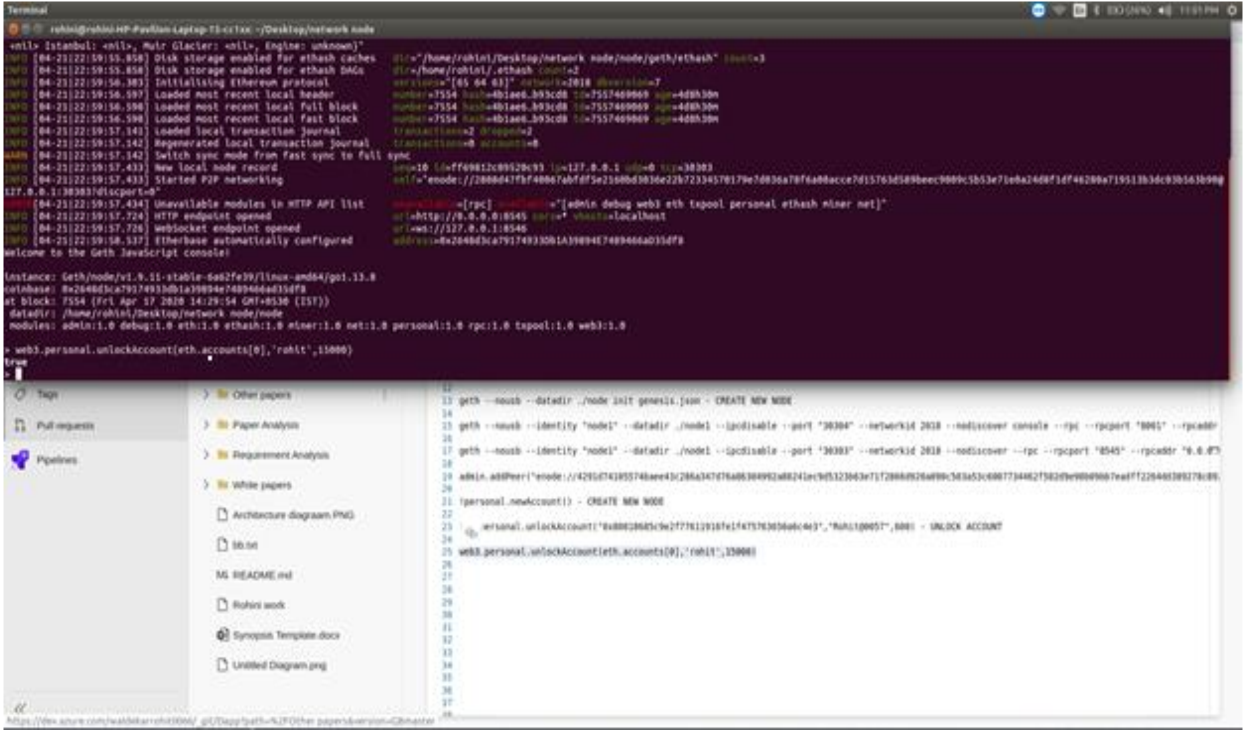
  To unlock account

  In this command first argument is the account number which is to be unlock,

  Second argument is the password for that account and last one is the time limit for which the account remains open in seconds. After that it will automatically lock the account

- After creation of account, start or stop the mining by command

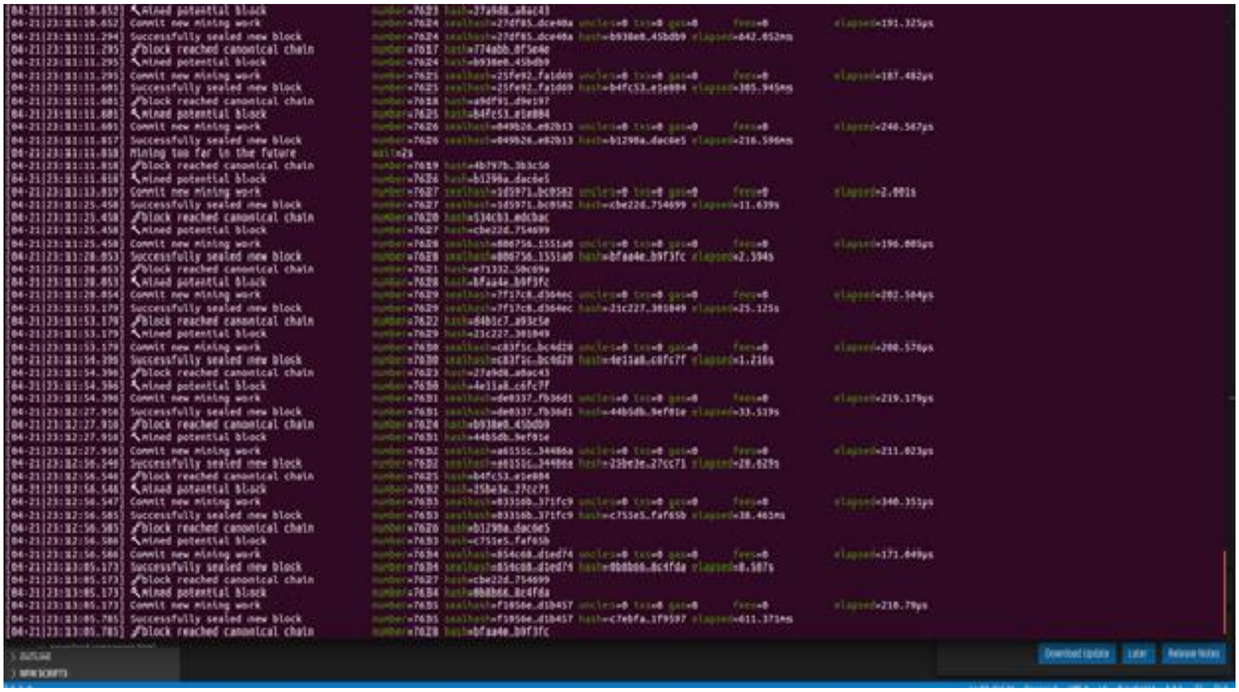
  miner.start() – to start the mining
  miner.stop()- to stop the mining



```
<nil> Istanbul: <nil>, Muir Glacier: <nil>, Engine: unknown}"
INFO [04-21|22:59:55.858] Disk storage enabled for ethash caches   dir="/home/rohini/Desktop/network node/node/geth/ethash" count=3
INFO [04-21|22:59:55.858] Disk storage enabled for ethash DAGs     dir=/home/rohini/.ethash count=2
INFO [04-21|22:59:56.383] Initialising Ethereum protocol           versions="[65 64 63]" network=2018 dbversion=7
INFO [04-21|22:59:56.597] Loaded most recent local header          number=7554 hash=4b1ae6…b93cd8 td=7557469069 age=4d8h30m
INFO [04-21|22:59:56.598] Loaded most recent local full block      number=7554 hash=4b1ae6…b93cd8 td=7557469069 age=4d8h30m
INFO [04-21|22:59:56.598] Loaded most recent local fast block      number=7554 hash=4b1ae6…b93cd8 td=7557469069 age=4d8h30m
INFO [04-21|22:59:57.141] Loaded local transaction journal         transactions=2 dropped=2
INFO [04-21|22:59:57.142] Regenerated local transaction journal    transactions=0 accounts=0
WARN [04-21|22:59:57.142] Switch sync mode from fast sync to full sync
INFO [04-21|22:59:57.433] New local node record                    seq=10 id=ff69812c09520c93 ip=127.0.0.1 udp=0 tcp=30303
INFO [04-21|22:59:57.433] Started P2P networking                   self="enode://2808d47fbf40067abfdf5e2160bd3036e22b72334570179e7d036a78f6a00acce7d15763d589beec9009c5b53e71e0a24d8f1df46280a719513b3dc03b563b90@127.0.0.1:30303?discport=0"
ERROR[04-21|22:59:57.434] Unavailable modules in HTTP API list     unavailable=[rpc] available="[admin debug web3 eth txpool personal ethash miner net]"
INFO [04-21|22:59:57.724] HTTP endpoint opened                     url=http://0.0.0.0:8545 cors=* vhosts=localhost
INFO [04-21|22:59:57.726] WebSocket endpoint opened                url=ws://127.0.0.1:8546
INFO [04-21|22:59:58.537] Etherbase automatically configured       address=0x2848d3ca79174933Db1A39894E7489466aD35df8
Welcome to the Geth JavaScript console!

instance: Geth/node/v1.9.11-stable-6a62fe39/linux-amd64/go1.13.8
coinbase: 0x2848d3ca79174933db1a39894e7489466ad35df8
at block: 7554 (Fri Apr 17 2020 14:29:54 GMT+0530 (IST))
 datadir: /home/rohini/Desktop/network node/node
 modules: admin:1.0 debug:1.0 eth:1.0 ethash:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txpool:1.0 web3:1.0

> web3.personal.unlockAccount(eth.accounts[0],'rohit',15000)
true
>
```

```
geth --nousb --datadir ./node init genesis.json - CREATE NEW NODE

geth --nousb --identity "node1" --datadir ./node1 --ipcdisable --port "30304" --networkid 2018 --nodiscover console --rpc --rpcport "8081" --rpcaddr

geth --nousb --identity "node1" --datadir ./node1 --ipcdisable --port "30303" --networkid 2018 --nodiscover --rpc --rpcport "8545" --rpcaddr "0.0.0

admin.addPeer("enode://4291d741055 [illegible]

!personal.newAccount() - CREATE NEW NODE

ersonal.unlockAccount("0x8002868 [illegible]","Rohit@0057",600) - UNLOCK ACCOUNT

web3.personal.unlockAccount(eth.accounts[0],'rohit',15000)
```

Fig 5.1 Account creation



```
04-21|23:11:10.652] 🔨 mined potential block            number=7623 hash=27a9d8…a8ac43
04-21|23:11:10.652] Commit new mining work              number=7624 sealhash=27df85…dce40a uncles=0 txs=0 gas=0     fees=0          elapsed=191.325µs
04-21|23:11:11.294] Successfully sealed new block       number=7624 sealhash=27df85…dce40a hash=b938e0…45bdb9 elapsed=642.052ms
04-21|23:11:11.295] 🔗 block reached canonical chain    number=7617 hash=774abb…0f5e4e
04-21|23:11:11.295] 🔨 mined potential block            number=7624 hash=b938e0…45bdb9
04-21|23:11:11.295] Commit new mining work              number=7625 sealhash=25fe92…fa1d69 uncles=0 txs=0 gas=0     fees=0          elapsed=187.482µs
04-21|23:11:11.601] Successfully sealed new block       number=7625 sealhash=25fe92…fa1d69 hash=b4fc53…e1e004 elapsed=305.945ms
04-21|23:11:11.601] 🔗 block reached canonical chain    number=7618 hash=a9df91…d9e197
04-21|23:11:11.601] 🔨 mined potential block            number=7625 hash=b4fc53…e1e004
04-21|23:11:11.605] Commit new mining work              number=7626 sealhash=049b26…e82b13 uncles=0 txs=0 gas=0     fees=0          elapsed=246.567µs
04-21|23:11:11.817] Successfully sealed new block       number=7626 sealhash=049b26…e82b13 hash=b1298a…dac6e5 elapsed=216.596ms
04-21|23:11:11.818] Mining too far in the future        wait=2s
04-21|23:11:11.818] 🔗 block reached canonical chain    number=7619 hash=4b797b…3b3c56
04-21|23:11:11.818] 🔨 mined potential block            number=7626 hash=b1298a…dac6e5
04-21|23:11:11.819] Commit new mining work              number=7627 sealhash=1d5971…bc0582 uncles=0 txs=0 gas=0     fees=0          elapsed=2.001s
04-21|23:11:25.458] Successfully sealed new block       number=7627 sealhash=1d5971…bc0582 hash=cbe22d…754699 elapsed=11.639s
04-21|23:11:25.458] 🔗 block reached canonical chain    number=7620 hash=534cb1…edcbac
04-21|23:11:25.458] 🔨 mined potential block            number=7627 hash=cbe22d…754699
04-21|23:11:25.458] Commit new mining work              number=7628 sealhash=806756…1551a0 uncles=0 txs=0 gas=0     fees=0          elapsed=196.085µs
04-21|23:11:28.853] Successfully sealed new block       number=7628 sealhash=806756…1551a0 hash=bfaa4e…b9f3fc elapsed=2.394s
04-21|23:11:28.853] 🔗 block reached canonical chain    number=7621 hash=e71332…50c69a
04-21|23:11:28.853] 🔨 mined potential block            number=7628 hash=bfaa4e…b9f3fc
04-21|23:11:28.854] Commit new mining work              number=7629 sealhash=7f17c8…d364ec uncles=0 txs=0 gas=0     fees=0          elapsed=202.504µs
04-21|23:11:53.179] Successfully sealed new block       number=7629 sealhash=7f17c8…d364ec hash=21c227…301049 elapsed=25.125s
04-21|23:11:53.179] 🔗 block reached canonical chain    number=7622 hash=d4b1c7…a93c5e
04-21|23:11:53.179] 🔨 mined potential block            number=7629 hash=21c227…301049
04-21|23:11:53.179] Commit new mining work              number=7630 sealhash=c83f5c…bc4d28 uncles=0 txs=0 gas=0     fees=0          elapsed=200.576µs
04-21|23:11:54.390] Successfully sealed new block       number=7630 sealhash=c83f5c…bc4d28 hash=4e11a8…c6fc7f elapsed=1.216s
04-21|23:11:54.390] 🔗 block reached canonical chain    number=7623 hash=27a9d8…a8ac43
04-21|23:11:54.396] 🔨 mined potential block            number=7630 hash=4e11a8…c6fc7f
04-21|23:11:54.396] Commit new mining work              number=7631 sealhash=de0337…fb36d1 uncles=0 txs=0 gas=0     fees=0          elapsed=219.179µs
04-21|23:12:27.910] Successfully sealed new block       number=7631 sealhash=de0337…fb36d1 hash=44b5db…9ef01e elapsed=33.519s
04-21|23:12:27.910] 🔗 block reached canonical chain    number=7624 hash=b938e0…45bdb9
04-21|23:12:27.910] 🔨 mined potential block            number=7631 hash=44b5db…9ef01e
04-21|23:12:27.910] Commit new mining work              number=7632 sealhash=a6155c…34486a uncles=0 txs=0 gas=0     fees=0          elapsed=211.023µs
04-21|23:12:56.540] Successfully sealed new block       number=7632 sealhash=a6155c…34486a hash=25be3e…27cc71 elapsed=28.629s
04-21|23:12:56.540] 🔗 block reached canonical chain    number=7625 hash=b4fc53…e1e004
04-21|23:12:56.546] 🔨 mined potential block            number=7632 hash=25be3e…27cc71
04-21|23:12:56.547] Commit new mining work              number=7633 sealhash=03310b…371fc9 uncles=0 txs=0 gas=0     fees=0          elapsed=340.351µs
04-21|23:12:56.585] Successfully sealed new block       number=7633 sealhash=03310b…371fc9 hash=c755e5…faf65b elapsed=38.461ms
04-21|23:12:56.585] 🔗 block reached canonical chain    number=7626 hash=b1298a…dac6e5
04-21|23:12:56.586] 🔨 mined potential block            number=7633 hash=c755e5…faf65b
04-21|23:12:56.586] Commit new mining work              number=7634 sealhash=854c08…d1ed74 uncles=0 txs=0 gas=0     fees=0          elapsed=171.049µs
04-21|23:13:05.173] Successfully sealed new block       number=7634 sealhash=854c08…d1ed74 hash=0b8b66…8c4fda elapsed=8.587s
04-21|23:13:05.173] 🔗 block reached canonical chain    number=7627 hash=cbe22d…754699
04-21|23:13:05.173] 🔨 mined potential block            number=7634 hash=0b8b66…8c4fda
04-21|23:13:05.173] Commit new mining work              number=7635 sealhash=f1056e…d1b457 uncles=0 txs=0 gas=0     fees=0          elapsed=218.79µs
04-21|23:13:05.785] Successfully sealed new block       number=7635 sealhash=f1056e…d1b457 hash=c7ebfa…1f9597 elapsed=611.371ms
04-21|23:13:05.785] 🔗 block reached canonical chain    number=7628 hash=bfaa4e…b9f3fc
```

Fig 5.2 Ongoing Mining process

Mining is the process of adding transactions to the large distributed public ledger of existing transactions, known as the blockchain. In this image, mining is started which will authenticate the blocks.

5.5 Editor for front end Development

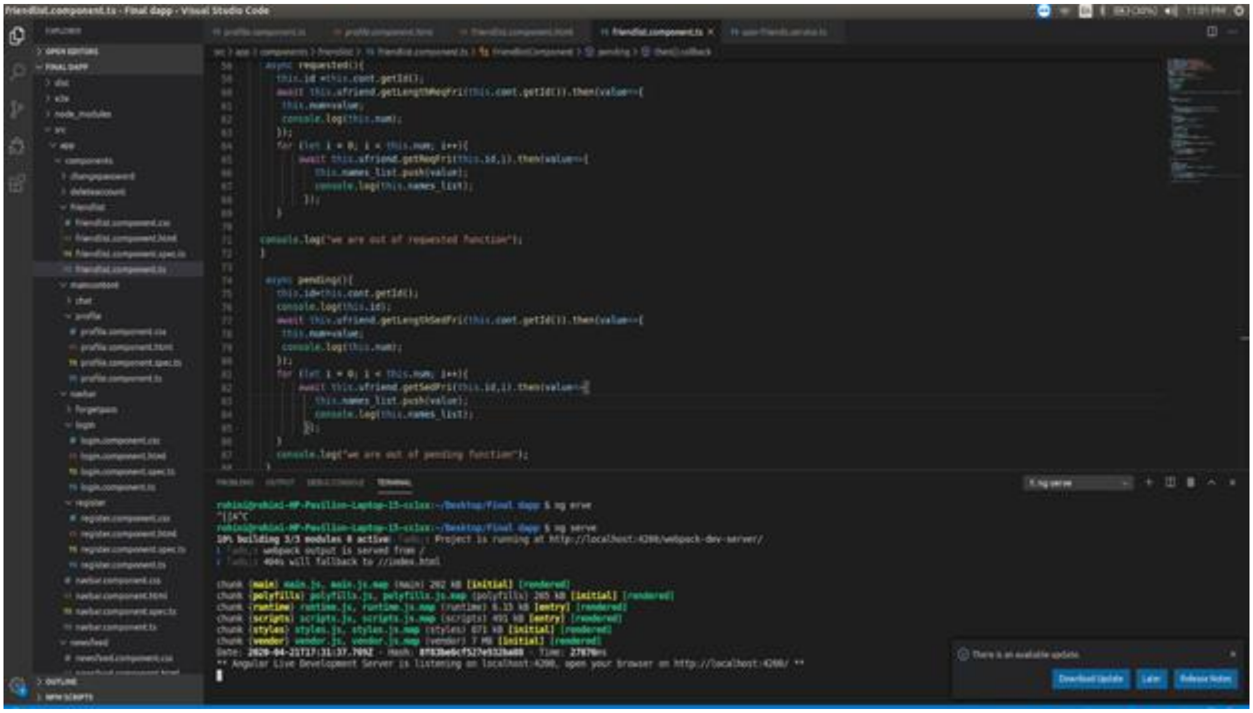
Visual Studio Code editor is used for front end development in typescript. TypeScript is an open-source programming language.
Steps to install typescript-

```
npm install -g typescript
```

After that, write the code in editor and compile-

```
ng serve
```

This will run the application on default port or on specified port. Port is specified as 4200. So, all the request by application is done by port 4200.



friendlist.component.ts - Final dapp - Visual Studio Code
EXPLORER
OPEN EDITORS
FINAL DAPP
profile.component.ts
profile.component.html
friendlist.component.html
friendlist.component.ts
user-friends.service.ts
src > app > components > friendlist > friendlist.component.ts > FriendlistComponent > pending > then() callback
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
1: ng serve
OUTLINE
NPM SCRIPTS
There is an available update.
Download Update
Later
Release Notes

```
  async requested(){
    this.id =this.cont.getId();
    await this.ufriend.getLengthReqFri(this.cont.getId()).then(value=>{
     this.num=value;
     console.log(this.num);
    });
    for (let i = 0; i < this.num; i++){
        await this.ufriend.getReqFri(this.id,i).then(value=>{
          this.names_list.push(value);
          console.log(this.names_list);
         });
    }

  console.log("we are out of requested function");
  }

   async pending(){
    this.id=this.cont.getId();
    console.log(this.id);
    await this.ufriend.getLengthSedFri(this.cont.getId()).then(value=>{
     this.num=value;
     console.log(this.num);
    });
    for (let i = 0; i < this.num; i++){
        await this.ufriend.getSedFri(this.id,i).then(value=>{
          this.names_list.push(value);
          console.log(this.names_list);
         });
    }
    console.log("we are out of pending function");
  }
```

```
rohini@rohini-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/Final dapp $ ng erve
^[[A^C
rohini@rohini-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/Final dapp $ ng serve
10% building 3/3 modules 0 active ｢wds｣: Project is running at http://localhost:4200/webpack-dev-server/
ℹ ｢wds｣: webpack output is served from /
ℹ ｢wds｣: 404s will fallback to //index.html

chunk {main} main.js, main.js.map (main) 202 kB [initial] [rendered]
chunk {polyfills} polyfills.js, polyfills.js.map (polyfills) 265 kB [initial] [rendered]
chunk {runtime} runtime.js, runtime.js.map (runtime) 6.15 kB [entry] [rendered]
chunk {scripts} scripts.js, scripts.js.map (scripts) 491 kB [entry] [rendered]
chunk {styles} styles.js, styles.js.map (styles) 671 kB [initial] [rendered]
chunk {vendor} vendor.js, vendor.js.map (vendor) 7 MB [initial] [rendered]
Date: 2020-04-21T17:31:37.709Z - Hash: 8f63be6cf527e532ba88 - Time: 27870ms
** Angular Live Development Server is listening on localhost:4200, open your browser on http://localhost:4200/ **
```

Fig 5.2 Front End in Angular

In the image above, frontend development is done using typescript. Typescript can be easily converted into JavaScript. This is the widely used technology for web development.

6. SOFTWARE TESTING

6.1 Unit test cases generation and its testing reports

6.1.1 Test cases for registration

Test case No	Test case	Input	Expected Output	Actual Output	Status
01	Registration	Name, Password, Favorite Question & its answer	Registration successful	Registration successful	Pass

6.1.2 Test cases for Login

Test case No	Test case	Input	Expected Output	Actual Output	Status
01	User registered	Name, Password	Successful login	Successful login	Pass
02	User not registered	Name, Password	Give error message & redirect to register page	Give error message & redirect to register page	Pass

6.1.3 Test cases for send friend request

Test case No	Test case	Input	Expected Output	Actual Output	Status
01	Friend name in user list	Friend's User name	Find the friend successfully	Find the friend successfully	Pass
02	Friend name not in user list	Friend's User name	Give the message user is not present	Give the message user is not present	pass

6.1.4 Test cases for accept friend request

Test case No	Test case	Input	Expected Output	Actual Output	Status
01	Accepting friend request	Friends User name In pending tab list	Accept the request successfully & redirect to friend's profile	Accept the request successfully & redirect to friend's profile	Pass

6.1.5 Test cases for Chat

Test case No	Test case	Input	Expected Output	Actual Output	Status
01	Chat with friend	Friend name	Get chat window successfully	Get chat window successfully	Pass
02	User is not friend with person	Friend name	Give error message	Give error message	pass

6.2 Integration test cases generation and its testing reports

Test case No	Test case	Input	Expected Output	Actual Output	Status
01	Login	Username password	Login successful	Login successful	Pass
02	Registration	Name, Password, Favourite Question & its answer	Registration successful	Registration successful	Pass
03	Forgot Password	Favourite Question & its answer	Change password successfully	Change password successfully	Pass
04	Change Password	Old password, New password	Change password successfully	Change password successfully	Pass
05	Send friend request	Friend user name	Send request successfully	Send request successfully	pass
06	Accept friend request	Select names in pending tab	Accept request successfully	Accept request successfully	Pass
07	Delete Account	Select Delete Account tab	Successfully delete the account	Successfully delete the account	pass

6.3 System test cases generation and its testing reports

Test case No	Test case	Input	Expected Output	Actual Output	Status
01	Access the Application	Application name/link on network	Get application successfully	Get application successfully	Pass
02	Creation of account on Ethereum	user name	Get account Successfully And return hash value	Get account Successfully And return hash value	pass
03	Chat and privacy	Message to Chat box	Successfully chat with friends	Successfully chat with friends	pass

7. Output Screen

Given below is the screenshot that shows actual system working step by step.

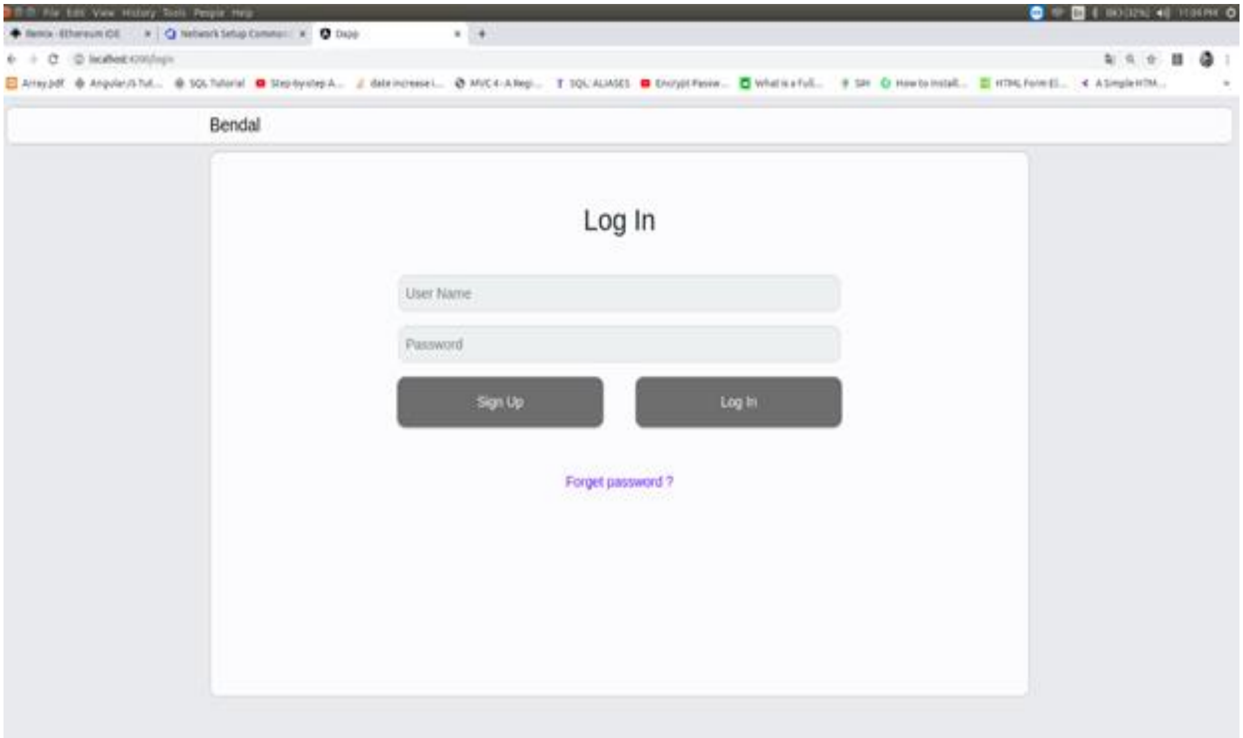


Fig 7.1 Final output screen

In the above image, after connecting to the internet and receiving the address of application, successful access of Bendal WebApp is possible.

8. PERFORMANCE ANALYSIS

Existing social media networks give users the impression that they control their data however, it is actually those companies providing those services that have sole authority over a person's information. People, without even thinking, sign and agree to privacy policies that provide these companies with the power to use this data for their advantage. This is a loss of personal privacy and has increased the desire of social media users to take back the authority to their own personal data. There's no one stopping them from using this information for their own financial advantage. There's a whole list of privacy issues when it comes to the major social networking platforms**.**

This platform can run without a central authority imposing its own rules during human interaction. The user can have control of their private data, and the freedom of expression is restored without the fear of censorship or backlash.

9. APPLICATIONS

1. Crowd Funding platform.
2. Platform for solving Environmental issues and social issues.
3. User can raise money for his original contents.
4. Gain popularity by more interactivity with platform.

10. CONCLUSION:

In current social media applications, misuse and monetization of user data, censorship issues, data security, and data availability issues due to the centralized industry are major problems seen. So, there is a need to develop A decentralized social network. That allows users more control. Unlike centralized social networking platforms, federated networks foster independence without a central authority. Benefits include censorship resistance, ownership over personal data, and improved control over user-generated content. So, we implemented a system where users will be able to choose to whom show the contents and be able to set their restrictions while determining how and where it gets distributed. They'll also have full control of their sensitive/private data. No more storing it on centralized servers and losing it when these servers go down or when their security gets breached. In the future, all applications will be moving towards decentralization as it provides transparency and privacy.

11. GUIDE TO ACCESS APPLICATION AND USER MANUAL

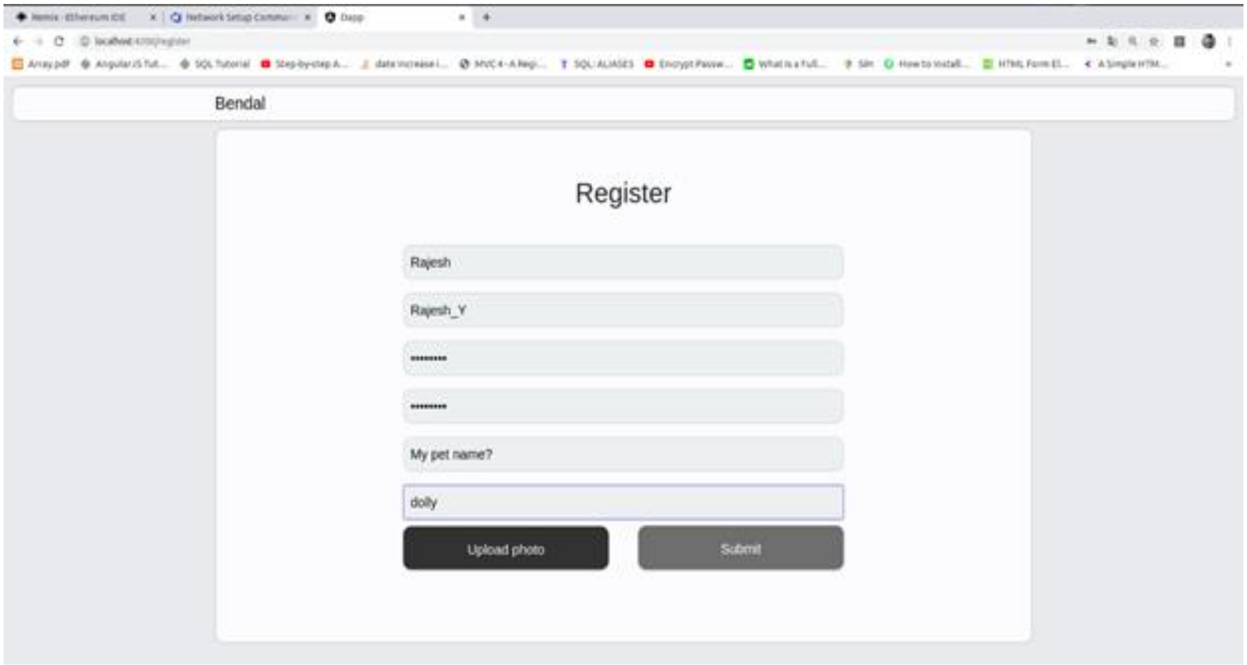
- **Guide to Access Application**

1. Connect to the network
2. Get the application address or link
3. Get the application

- **User Manual**

11.1 The Screenshots of WebApp for this project

11.1.1 Register page



Rajesh

Rajesh_Y

••••••••

••••••••

My pet name?

dolly

Fig 11.1.1 Registration page

In this image, user has to give User name, Name, Password, Profile photo, Favorite question and its answer. Using this details application will create his/her profile.

11.1.2 Login page

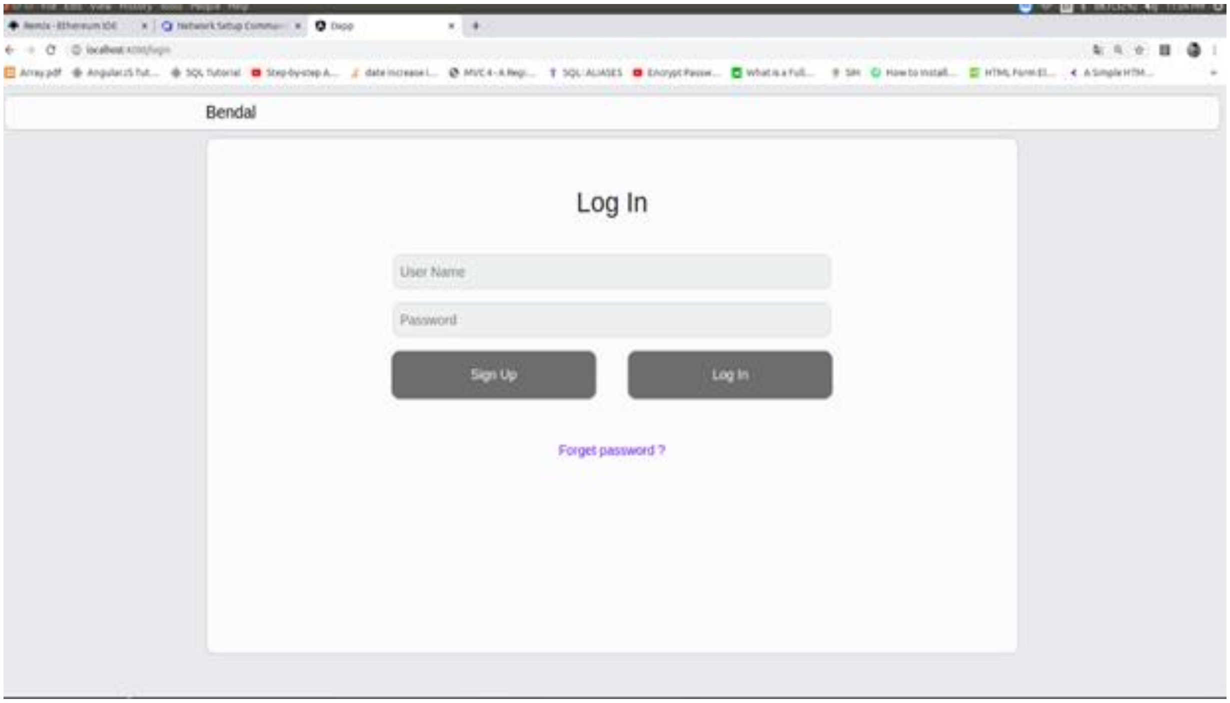


Fig 11.1.2 Login page

User should enter valid username and password for entering into the app.

11.1.3 Forgot password

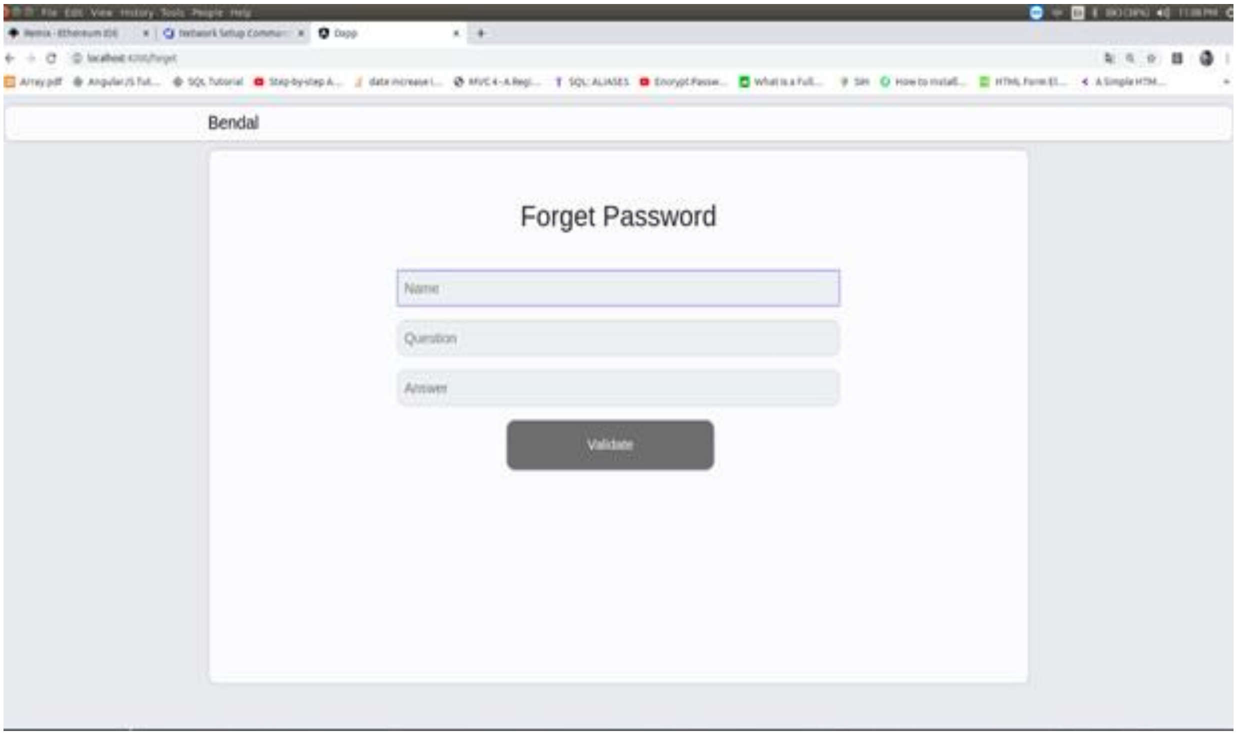


Fig 11.1.3 Forgot Password page

If password is forgotten, then facility for recovering the account is also provided

11.1.4 Profile page

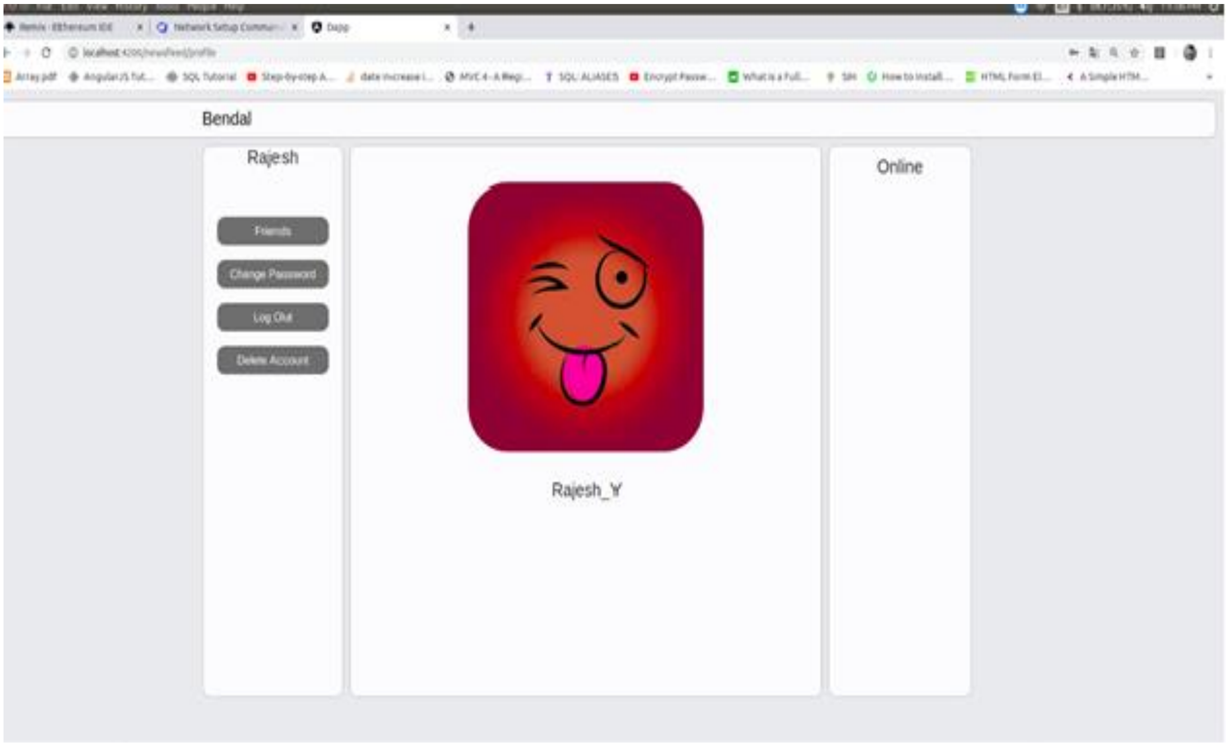
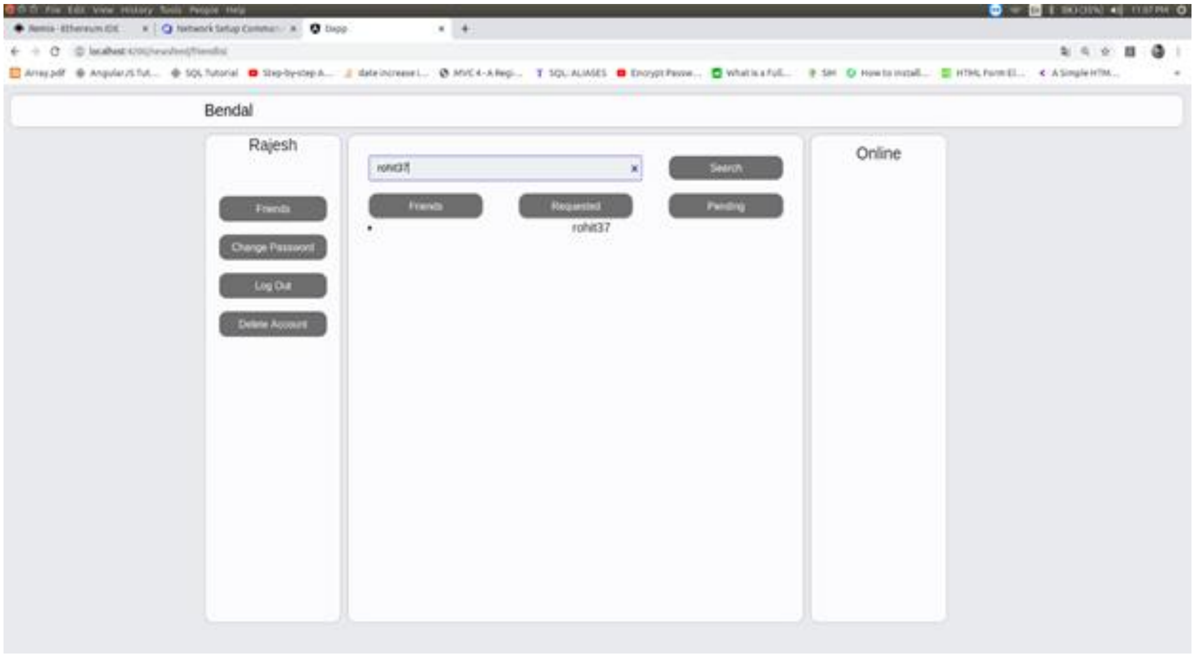


Fig 11.1.4 Profile page

Getting Details of user and profile updating id is done here.

11.1.5 Search friend page



rohit37

- rohit37

Fig 11.1.5 Search Friend page

In application, searching for friends who are present on chat application can be done here.

11.1.6 Send request page

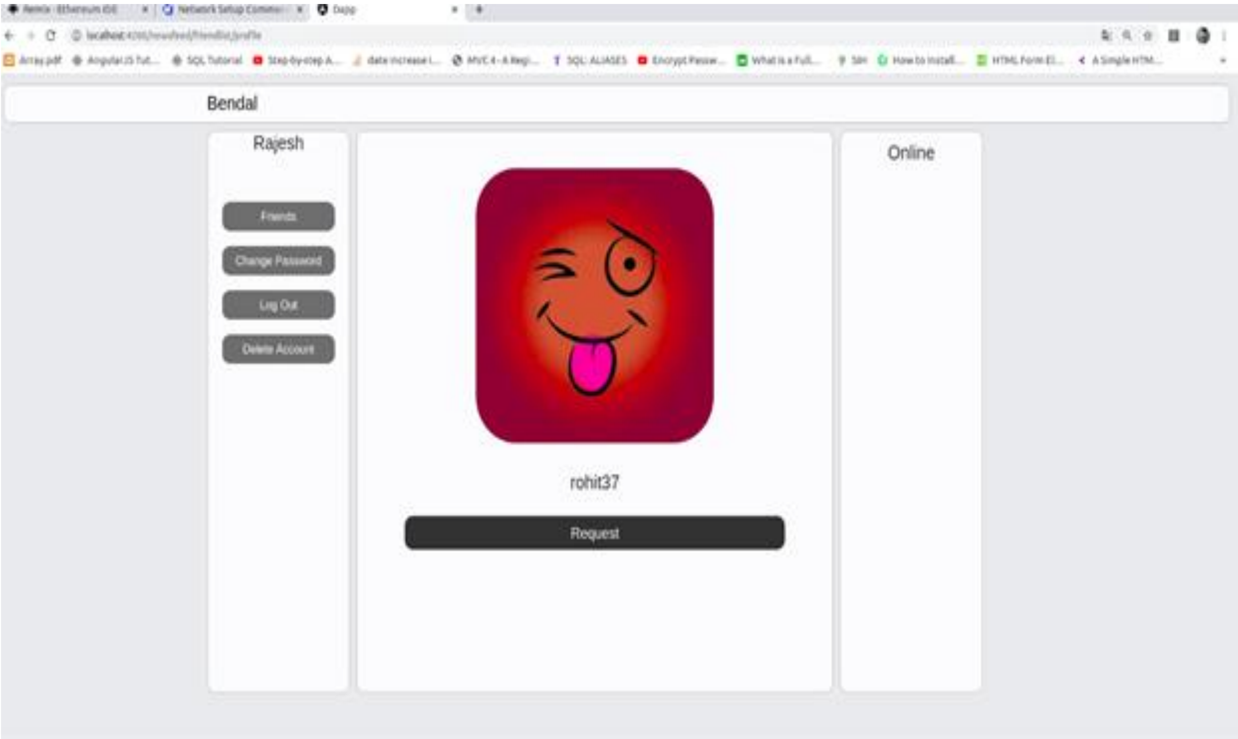


Fig 11.1.6 Send Request page

Sending a friend request to users who are present on application can be done here.

11.1.7 Friend name in requested tab

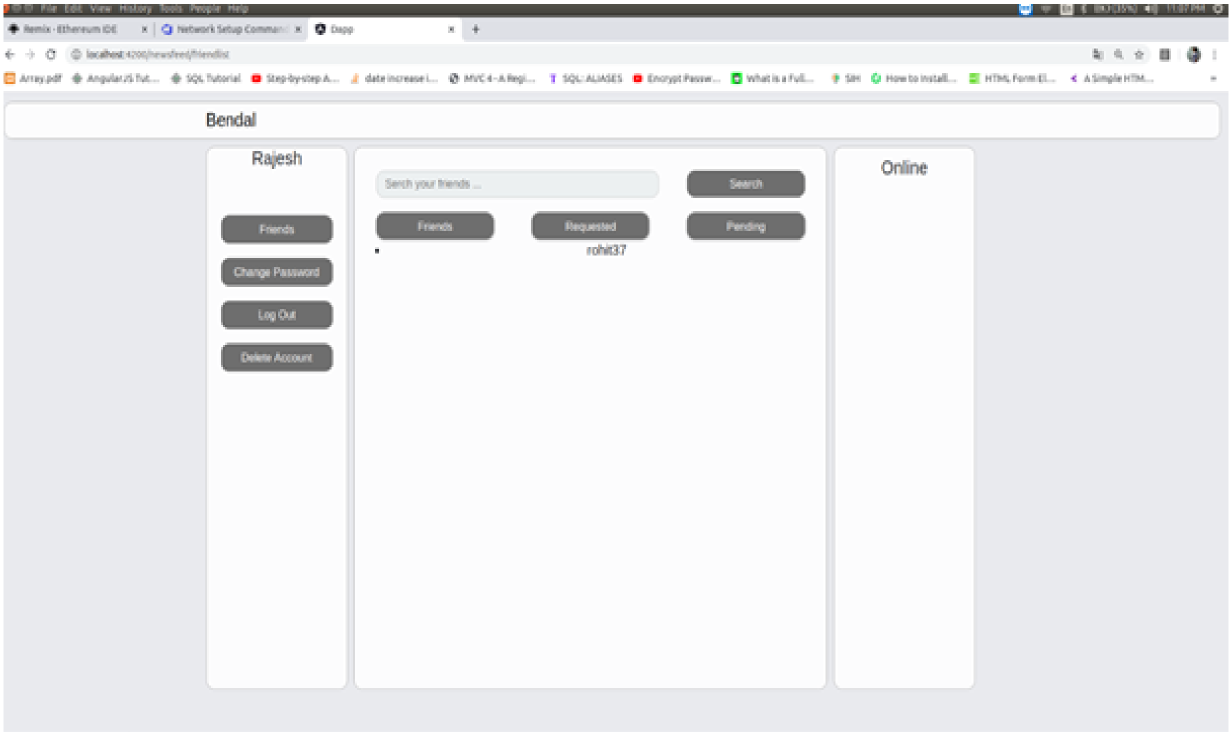


Fig 11.1.7 Friend in the requested tab page

In above image request is added to requested list.

11.1.8 Change password page

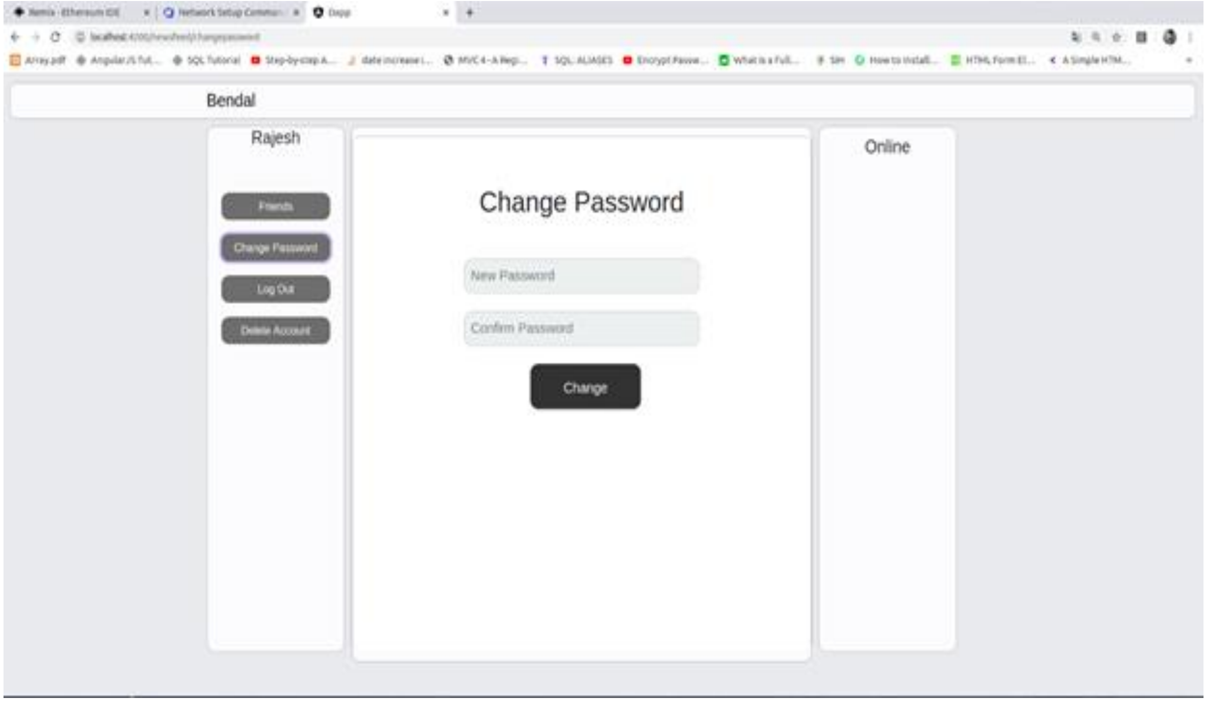
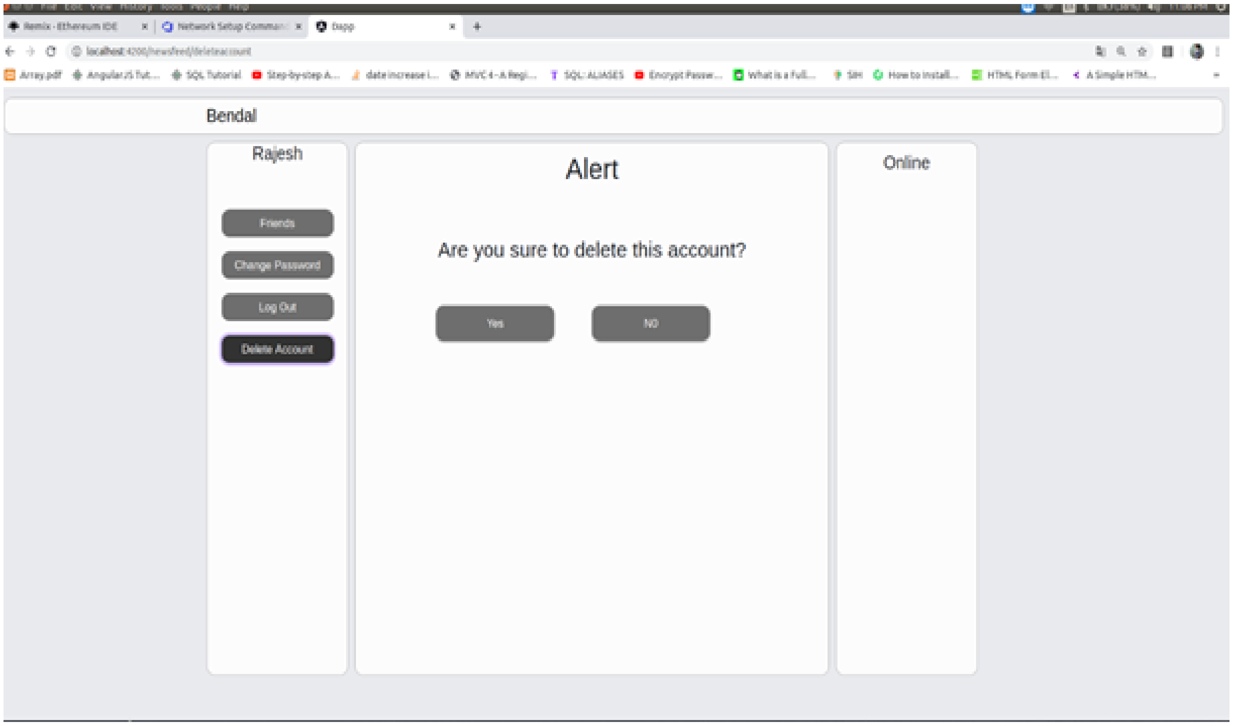


Fig 11.1.8 Change password page

User can change password y clicking on Change Password button shown in above image

11.1.9 Delete Account



Alert

Are you sure to delete this account?

Fig 11.1.9 Delete Account page

User can Delete Account by clicking on the Delete Account button in above image

12. LICENSE

Creative Common License

Attribution-Share Alike 4.0 International (CC BY-SA 4.0)

This is a human-readable summary of (and not a substitute for) the license.

You are free to:

Share — copy and redistribute the material in any medium or format

Adapt — remix, transform, and build upon the material for any purpose, even commercially. The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

Attribution — you must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

Share Alike — if you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

No additional restrictions — you may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Notices:
You do not have to comply with the license for elements of the material in the public domain or where your use is permitted by an applicable exception or limitation.

No warranties are given. The license may not give you all of the permissions necessary for your intended use. For example, other rights such as publicity, privacy, or moral rights may limit how you use the material.

13. ETHICS

1. Surf the internet for personal and non-class related purposes during classes

2. Make a copy of software for personal or commercial use.

3. Make a copy of software for a friend

4. Loan CDs of software to friends

5. Download pirated software from the internet

6. Distribute pirated software from the internet

7. Buy software with a single user license and then install it on multiple computers

8. Share a pirated copy of software

9. Install a pirated copy of software

14. REFERENCES

[1] Dongqi Fu, L. F. (2016). Blockchain-based Trusted Computing in Social Network. *2016 2nd IEEE International Conference on Computer and Communications (ICCC).* Chengdu, China.

[2] Eric Klukovich, E. E. (2016). POSN: A privacy preserving decentralized social network app for mobile devices. *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM).* San Francisco, CA, USA.

[3] Nikita V. Ghodpage, N. V. (2018). Privacy Preserving and Information Sharing in Decentralized Online Social Network. *2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT).* Coimbatore, India.

[4] Quanqing Xu, Z. S. (2018). Building an Ethereum and IPFS-based Decentralized. *2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS).* Singapore.

[5] Rammohan Narendula, T. G. (2012). A Decentralized Online Social Network with Efficient User-Driven Replication. *Privacy, Security, Risk and Trust (PASSAT), IEEE International Conference on and IEEE International Confernece on Social Computing (SocialCom).* Amsterdam, Netherlands.