I.J. Information Engineering and Electronic Business, **2023, 1, 1-19**
Published Online on February 8, 2023 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijieeb.2023.01.01

CodeBlockS: Development of Collaborative Knowledge Sharing Application with Blockchain Smart Contract

Siddhant Jain
Jaypee Institute of Information Technology, A-10, Sector 62, NOIDA, India
Email: 18803010@mail.jiit.ac.in
ORCID iD: https://orcid.org/0000-0002-6656-1121

P. Raghu Vamsi*
Jaypee Institute of Information Technology, A-10, Sector 62, NOIDA, India
Email: prvonline@yahoo.co.in
ORCID iD: https://orcid.org/0000-0001-7726-6198
*Corresponding Author

Yashi Agarwal
Jaypee Institute of Information Technology, A-10, Sector 62, NOIDA, India
Email: 18803033@mail.jiit.ac.in
ORCID iD: https://orcid.org/0000-0002-3777-8997

Jayant Goel
Jaypee Institute of Information Technology, A-10, Sector 62, NOIDA, India
Email: 18103255@mail.jiit.ac.in
ORCID iD: https://orcid.org/0000-0002-5322-3414

Received: 07 September, 2022; Revised: 27 October, 2022; Accepted: 20 November, 2022; Published: 08 February, 2023

Abstract: In this paper, we present the design and development of a collaborative knowledge-sharing platform with Blockchain based smart contracts (CodeBlockS) to help increase the trust and efficiency of how developers find the solution to their problems or try to learn new things. The popularity of Question-and-Answer websites such as StackOverflow, Ask, and Yahoo, as well as online course websites like as Udemy, is gradually expanding. Given this increased popularity, the quality and efficiency of user interaction must be improved such that users can try to connect with each other, ask questions about technical problems they are experiencing, or if they want to learn a topic in exchange for a fee and potentially collaborate on a project, or simply share their thoughts on a topic and improve their knowledge and network at the same time. Because these contracts will contain money, CodeBlockS has employed Ethereum Blockchain-based smart contracts to manage the data and money, as blockchain-based smart contracts are immutable and handle payments very securely. In general, social networking websites there are very few people sharing valuable knowledge and many people sharing worthless, time-consuming content that creates distraction. With the CodeBlockS system, developers find the solution to their problems or try to learn new things, and users can share their thoughts and learning on the platform. The platform also provides inbuilt smart contracts functionality using which two users can create a contract where one user will teach or solve doubt of the other user and receive fees towards service rendered.

Index Terms: Blockchain, Collaborative Learning, Ethereum, Smart Contracts, Social networking, Question Answer System.

1. Introduction

The software market is vast, and software engineers encounter numerous problems when developing various apps.

The popularity of Question-and-Answer websites like StackOverflow, Ask, and Yahoo is progressively growing. Given this rising popularity, the quality of user engagement must be improved. A user-interactive question-answering (QA) platform must be created, which will be a specific type of online community with a rewarding scheme for all users that answer questions. The system suggests and supports a Consumer-to-Consumer (C2C) business model for exchanging and commercializing regular people's information. It can also be utilized as a motivator and a collaborative method to learning. Because of the business model, rapid and high-quality responses are swiftly amassed. A knowledge-sharing platform contextualizes the material so that users can comprehend and interact with it more easily. A knowledge-sharing platform allows users to communicate, ask questions, provide feedback, and make simple modifications without leaving the software. These platforms integrate learning and knowledge processes in order to improve corporate learning. Through in-person meetings, discussions, faculty development activities, industry-institute collaborations, and other means, communication, idea sharing, and information transfer all contribute to knowledge advancement, but there are very few people generously share their expertise and lot of people use to share worth less time consuming content that creates distraction. Monetizing the people sharing good content may improve the quality of collaboration [1, 2, 3, 4, 5].

1.1 Doubts resolving in the existing platforms

For solving doubts, the most widely used websites are Udemy, Coursera, Fiverr, StackOverflow and others. One can locate the ideal instructor on Udemy and Coursera. There are several themes, skill levels, and languages to choose. Join the millions of learners who have already started learning. However, one has to pay for a full skill irrespective of the interest in the entire course, so it will not help the seeker out if one just needs understanding on a certain topic. It is also same in the online marketplaces for freelance services include Fiverr (https://www.fiverr.com/). Users can use Fiverr to post and apply for small one-time jobs (gigs) online. The platform offers jobs ranging from “get a professionally designed business card” to “help with HTML, JavaScript, CSS, and jQuery”. Many developers hold one-on-one sessions with customers and mentor them, but the problem presented is entirely up to the solver. The user is unable to discover what he seeks. Additionally, these platforms were not designed with this scenario in mind. Fig. 1 shows People using Fiverr for pitching their skill. Existing Q&A sites like StackOverflow, there in addition, users ask their doubts but it is in the form of text and no one-to-one mentoring is possible [6, 7, 8, 9, 10].

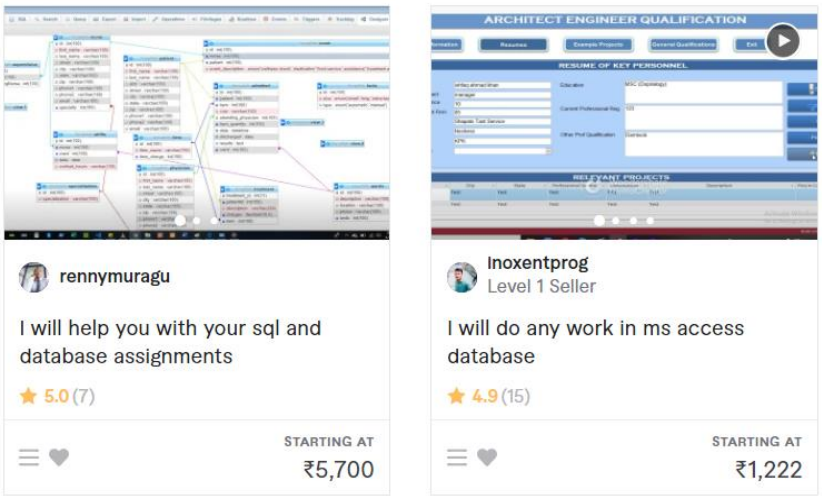

Fig. 1. People using Fiverr for pitching up their skill

1.2 Blockchian Technology

In the above-mentioned platforms, the courses there are mostly very general, most of what they are teaching is already available free, and users even after paying do not get personalized experience. If someone is paying to learn a skill, they should get some personalization because everyone is learning method is different. One more thing to consider is that it is up to the course instructor what they will teach in that course, if users want to learn a small topics of the course then there is no benefit in paying for the whole course. Users are convenient to pay only for what they want to learn. Further, the trust on the trainer or doubt solver is important for any user to proceed further [11, 12]. The trust should be in terms of the safety to the payment made and the content delivered by the trainer. Tracking the trainer's commodities and providing transparent information are two of the most important aspects of collaborative learning, helping to lay the groundwork for ongoing collaboration and trusting trainer and student relationships. These tasks are made easier with a robustly secured system. Blockchain technology can be used to preserve a safe and shareable record of every collaboration in the learning platform. Blockchain has received a lot of attention in recent years because of four essential features that helps in providing its users with trust and integrity in a collaboration system [13, 14]. They are *1) Consensus*: When numerous nodes are doing the same task, it might be difficult for the network to determine which

node has completed the task successfully. To ensure that the tasks are genuine, a common agreement between the nodes must be ensured. For example, in Bitcoin [15] this attribute is used by Blockchain to ensure that the transaction is genuine and that all nodes agree on it. *2) Provenance:* All nodes verify each block in a Blockchain network, and all information about the block is available to the network before it is attached to it. *3) Immutability:* Other network users are not permitted to delete, edit, or misrepresent their duties in the network, and only a new transaction addition is permitted once the prior information has been provided. *4) Finality:* The copies of the shared ledger all have the same consistent record of truth update the records whenever the network changes. What works for the Bitcoin network is equally applicable to any other Blockchain network.

On the other hand, smart contracts [16, 17, 18, 19] are essentially programs that are stored on Blockchian run when certain criteria are satisfied. They are often used to automate the implementation of an agreement so that all participants can be certain of the conclusion immediately, without the involvement of an intermediary or time loss. Implementing smart contracts in the Blockchain applications achieve transparency, autonomy; cost reduction, automatic updates, and improves the speed of application.

1.3 Contributions of the paper

There has not been much improvement in the current Q&A platforms that could incentivize answering the questions and improve the chances of getting solutions. There are many platforms for asking doubts and learning skills, but the main limitations of these platforms are that the doubt-solving platforms do not incentivize the user to answer other users' doubts, and when a user wants to learn a skill, there are many platforms that provide courses, but there is no option available for a user who only wants to learn a small part of the course to pay for that small part only instead of paying for the whole course, and the user can not have any customizations. Continuous collaboration depends on trust in an appropriate way. Historically, it came into observations that some users had distrust among them and this discourages them from sharing or relying over the data in a shared environment. In addition, many resource persons do not convey their expertise because of the unavailability of monetization policy. Further, learner may want to learn and get clarify selective topics or few technical issues. To solve these problems, this paper presents the design and development of Collaborative Knowledge Sharing Platform with Blockchain Smart Contract (CodeBlockS) using Ethereum as underlying blockchian, which is proved as a remedy in a shared, permission record of the ownership of the user in the network. This helps in increasing the efficiency, transparency and developed trust for any collaboration platform. Fig. 2 shows the block diagram of the proposed CodeBlockS. The proposed model is a collaborative knowledge-sharing platform using Blockchain smart contracts that is specifically built for developers/programmers to share their thoughts and ask questions/doubts, receive relevant support from others, and aid each other in their professional journey. Users can upload their doubts, which will include the main title of the doubt, a description of the doubt, tags connected to the doubt that can be used to search these doubts later, and the price they are prepared to pay to have their doubt clarified. The user will be able to see a doubt feed where he will be able to see all of the doubts asked by other users on the platform. The user will be able to see the questions they have asked in order to simply manage their doubts. The user will be able to see the doubts to which he has contributed his idea and easily keep track of them. Furthermore, users can communicate with one another using the message module to learn how the other user can assist them and to conduct discussions. After they have both decided on everything, the doubter can create a contract by paying the agreed-upon fees. The addresses of both users, as well as the title of the dispute, will then be stored into the Blockchain. The charge is collected up front to verify that the person asking the question is serious and not wasting anyone's time. Following the conclusion of the doubt session the question asker may terminate the session with satisfactory consensus and the fee is immediately transferred to the account of the doubt solver. If the session does not take place for whatever reason, the fee will be refunded to the questioner.

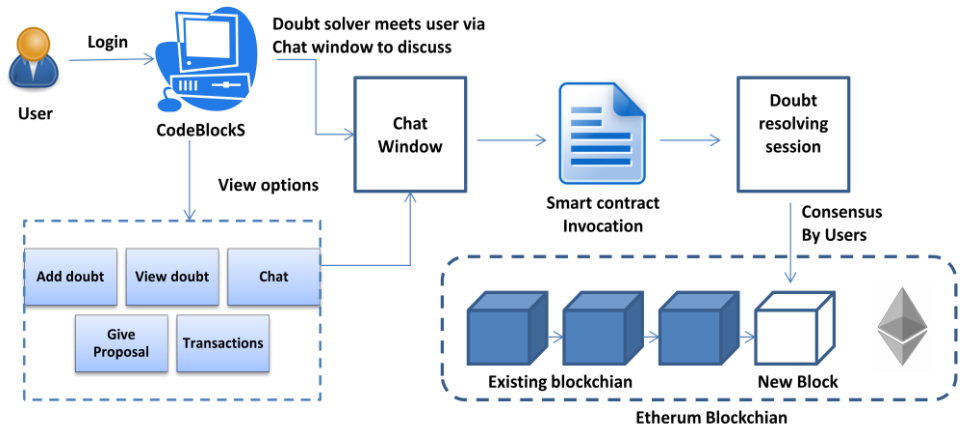

Fig. 2. Block diagram of Blockchain framework in CodeBlockS

1.4 Research methodology

Software development life cycle (SDLC) methodology is used to develop the CodeBlockS. SDLC consists of four major steps such as 1) Requirements analysis, 2) Software design, 3) Design implementation (or coding), and 4) Testing. With reference to the objectives and contributions stated in the above section, the requirements for development of CodeBlockS are analyzed. The requirements include the functional, smart contract, database, and web application requirements. The analyzed requirements are presented in the design to better understand the functionality of the proposed application. Using the design, development of application code such as smart contract code using Solidity programming language, Web application development using ReactJS, Ethereum blockchain and smart contract communication using web3js, and database tables design in MongoDB. As the final step of SDLC, testing of the developed smart contract is done by preparing the related test cases. Rest of the paper is organized as follows: After presenting the related work in Section 2, Section 3 presents the requirements analysis and design of CodeBlockS. Implementation details and testing of CodeBlockS is presented in Section 4. Finally, Section 5 concludes the paper with future scope.

2. Related Work

Muhamed et al. [4] proposed a global educational activity credit network called EduCTX, which is supported by Blockchain technology. This platform is based on the idea of the European Credit Transfer and Accumulation System (ECTS). It is a worldwide trusted, redistributed educational activity credit and grading system that can provide a globally unified perspective for students and higher education establishments (HEIs), as well as other prospective stakeholders such as corporations, establishments, and organizations. Authors have a tendency to gift an example implementation of the atmosphere, backed by the ASCII text file Ark Blockchain Platform, as a symbol of invention. EduCTX, which is supported by a globally distributed peer-to-peer network, may generate, maintain, and manage ECTX tokens, which represent credits earned by students for finished courses such as ECTS. The peers of the Blockchain network are HEIs. The platform could be a gateway to a wide range of clear and technologically advanced instructional activity systems. The EduCTX platform represents the idea of the EduCTX initiative that anticipates that varied HEIs would work so as to make a globally economical, simplified and omnipresent atmosphere so as to avoid language and body barriers. so Authors have a tendency to invite and encourage HEIs to affix the EduCTX initiative and therefore the EduCTX Blockchain network. Richa et al. [13] developed the Blockchain based system to creating programmes and strategies related to duties such as hiring, training, and pleasing employees is a delicate task that requires security. Blockchain technology is entering significant attention as a backbone technology in colorful sectors ranging from cryptocurrency to supply chain operation because of its silent features similar as decentralization, openness, translucency, and tamper evidence. The introduction of Blockchain into the being HR information system can address the being challenges encountered by the HR platoon. Smart contracts in the seeker selection process help to make the selection process more secure, and smart contracts in the payroll system help an association make easy sales, real-time payments, and outlaying hand hiring. The proposed structure is adaptable and does not have any specific tackle. Shuai et al. [14] provided the operating mechanism and idea platforms of Blockchain-enabled sensible contracts, and then planned a pursue framework for sensible contracts based on a distinct six-layer design. Second, the technological and legal problems are highlighted, with more to come as the recent analysis develops. Third, authors frequently give many typical application scenarios. Authors frequently emphasized the long-term development tendencies of smart contracts near the end. This document is intended to provide relevant guidance and references for future analysis efforts. Hiroki et al. [15] proposed a novel mechanism for protecting a Blockchain used to contract administration, similar to digital rights management. This mechanism incorporates a completely new agreement methodology that employs a credibility score and generates a hybrid Blockchain by utilizing this new methodology and proof-of-stake alternatively. This makes it possible to prevent an attacker from monopolizing resources and to keep Blockchains secure. Urshila et al. [20] proposed a secure Blockchain based finance application to demonstrate that Blockchain has the potential to deliver a high-tech banking solution that can meet the major requirements of any banking environment, namely security, regulation, immutability, and the storing of the customer's transaction records. It can ensure that future operational costs of the entire banking system are reduced, as well as an increased level of openness, leading to increased client satisfaction and trust in the bank. The authors created a loan processing system based on Ethereum smart contracts that allows users to take out loans utilizing digital mortgages in a simple and secure manner. Nitin et al. [21] presented a method is to develop a smart contract for the country of origin certificate management system in Indian customs and worldwide usage, as well as to use IPFS (Inter Planetary File System) for exchanging multilateral agreement data on the Blockchain network. In addition, this enables the company to improve and reduces the time it takes to make a contract from weeks to months. There is no requirement for third-party verification, and there are no risks of changing committed transactions, making the certificate management system secure and resilient. Authors created a system that can check certificate ownership and determine whether a certificate is genuine or fake by leveraging the capabilities of Blockchain and Blockchain-based smart contracts. The Blockchain-based issue of COO certificates will assist exporters and importers in ensuring secure and timely delivery of certificates while also preventing the siphoning of government funds through the production of fake COO certificates. Further,

Blockchian has been widely used for implementing security in applications such as Supply Chain Management [22], e-commerce [23], and others. Blockchain technology offers the establishment of a decentralized atmosphere wherever transactions and information are not underneath the administration of any third party organization. All transactions are recorded in a highly public ledger in a very verifiable and permanent manner. In recent years, the rapid development of cryptocurrencies and their underlying Blockchain technology has resurrected Szabo's original concept of sensible contracts, i.e., computer protocols designed to automatically facilitate, verify, and enforce the negotiation and implementation of digital contracts with no central authority. Sensible contracts will see a broad range of potential application scenarios in the digital economy and intelligent industries, including financial services, management, healthcare, and the internet of Things, among others, and have even been integrated into thought Blockchain-based development platforms, such as Ethereum and Hyperledger [24]. However, smart contracts are still in their infancy, and important technical obstacles such as security and privacy issues require additional research. Authors seek to provide a scholarly and complete review of Blockchain-enabled sensible contracts in this work, with the goal of motivating future research into this emerging research area.

In this way, the Blockchain is being used in many fields to provide privacy and security of transactions. It can be observed from the literature that Blockchain is a successful technology to achieve security in decentralized environment. Currently no site provides a solution to solve this problem. In this case, how will the user be able to contact a Mentor and pay him just for what he actually needs to learn? Therefore, the proposed method CodeBlockS attempts to solve these problems. The detailed description of development of CodeBlockS is presented in the next section.

3. Requirements Analysis and Design

In this section, first the architecture of the proposed CodeBolockS is presented followed by the functional requirements and design of the modules in the system. CodeBlockS aims to create a collaborative knowledge sharing platform, specifically for developers looking for quality information and having their questions answered by other users on the network, making collaborative learning trustworthy, easier and more efficient. Fig. 3 shows the system use case and Fig. 4 shows the control flow of the CodeBlocksS. CodeBlockS has several modules that help users achieve these goals, such as the doubt module, which allows users to add their doubts, finds others' doubts, and search doubts using tags. Users will also be able to add their proposals on other people's doubts by requesting how they can assist in solving the problem and the fee for doing so. All payments will be processed using Ethereum smart contracts, which make payments secure and contracts tamper-proof. There is a doubt contract module, which will create a contract between the doubt asker and doubt solver, and put it on Blockchain, this module will handle the payments. Before initiating a contract, both parties can communicate with each other to gain insight and understanding of the doubt solver, can negotiate the charging amount, share their perspectives, and eventually decide on contract commencement. Users can also communicate with one another via the message module to learn how the other user can assist them and to conduct discussions. The description of each module is as follows.

3.1 Doubt proposal module

This module is for paid questions to other users, and the Peer-to-Peer network topology employed in this module. It means, any user can express a doubt, and any user within the same group of users with that expertise can respond to that generated doubt. This is the module where users can add their doubts, access doubts asked by them or others, and search for doubts with a certain tag. Users must enter details such as the title or main question, a description of the doubt, tags, and the amount they are ready to pay to have the doubt resolved. The React Quill package is utilized to input the description because it allows users to share a link, add a piece of code, and use basic text editing capabilities such as different font sizes, bold, italic, underline, and so on. This makes the explanation more dynamic and allows the user to simply convey his question. The difference between the doubt and question module is that doubt is a paid module, i.e., it allows one to one mentoring such that users can save their time and can easily get answers to their doubts, even in a personal meeting. Features that can be availed by user in this module are 1) Add doubt, 2) Mentor doubt, 3) Initiate booking for doubt, 4) Confirm the booking and mentor, 5) Change amount, 6) Can mark doubt as solved, 7) Fetch all doubts, 8) Fetch doubts asked by user, 9) Fetch doubts one wish to mentor. To have these features this module need to be developed with the functional requirements such as

- The user should be able to create contract
- The user should be able to update contract
- The user should be able to send money to contract
- The user should be able to get money from the contract
- The user should be able to refund the money
- The user should be able to see their contracts

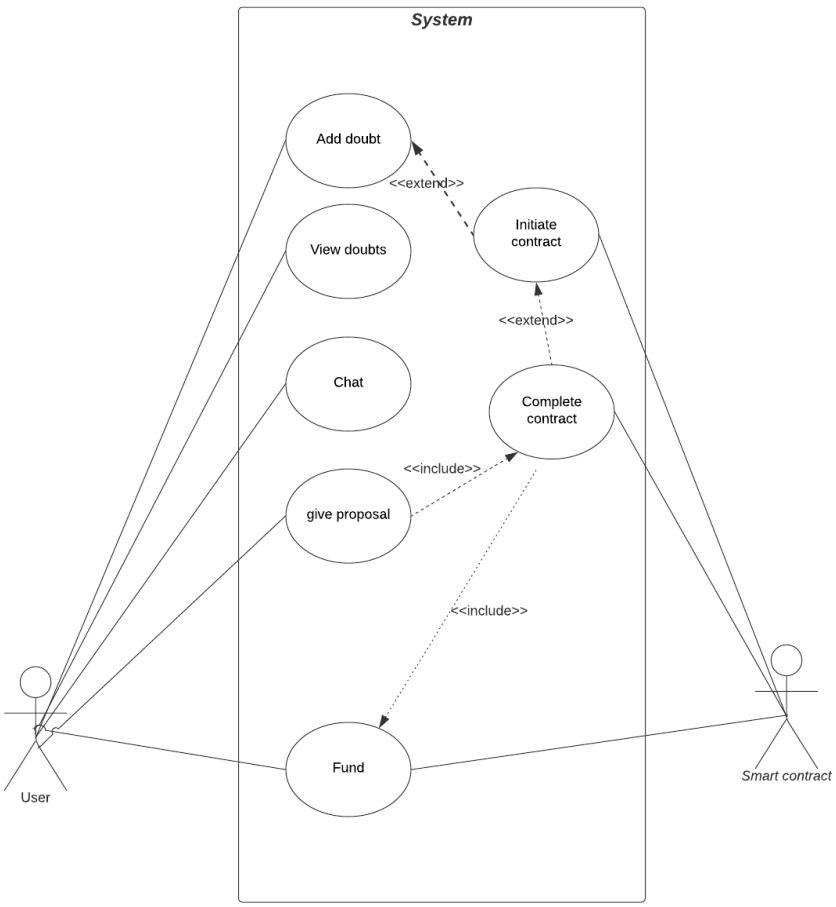

Fig. 3. Use case diagram of CodeBlockS

3.2 Doubt contract module

This is the module where the contract between the doubt asker and doubt solver is placed on the Blockchain, containing the details like the address of doubt asker, address of doubt solver, the topic of doubt, the amount they settled on and the status of contract, which indicates the contract is in progress or finished. This module will also handle all the payments. As Blockchain is used, the contract will be tamper proof and the payments will be secure. To make sure that the asker is genuinely interested in getting their doubt solved, CodeBlockS receives the payment upfront from the asker and putting it in the smart contract and once the doubt session is done, the contract will directly send the amount to the doubt solver's wallet. There is also a functionality of refunding the payment to the doubt asker, because there could be the cases that the doubt session will not take place and thereby the payment is refunded to the doubt asker. In this way, smart contract function will refund the money to the doubt asker and finish the contract. The user will be able to see the contracts that have been finished and those that are ongoing. For this, this module to be developed with the functional requirements such as

- The user should be able to add doubt
- The user should be able to delete doubt
- The user should be able to edit doubt
- The user should be able to find doubt tag wise
- The user should be able to see all doubts
- The user should be able to add proposal to doubts
- The user should be able to edit own proposal
- The user should be able to delete own proposal

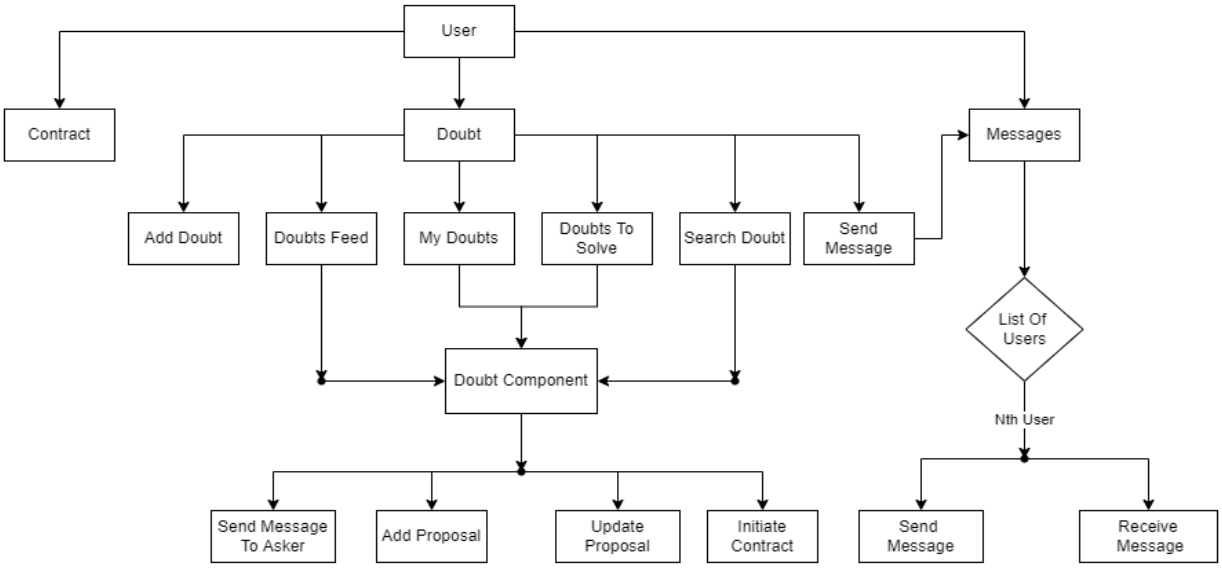

Fig. 4. Control flow of proposed modules in CodeBlockS

3.3 Message module

This module helps a doubt solver and asker to communicate regarding the doubt, amount and help each other get the best insight of other candidates. The chat box will be used to give trials before the smart contract will be made between the doubt solver and the user having doubt so that the user can ensure the solver will be able to clear the doubt. For this, the module is to be developed with the following functional requirements

- The user should be able to send a message regarding doubt to that user.
- The user should be able to get all messages.
- The user should be able to delete own messages.

4. Implementation and Testing

This section presents the technology implementation details followed by the testing of smart contract and proposed modules.

4.1 Implementation details

Several APIs (Application Programmer Interfaces) have been developed in each module. The APIs developed for the backend of doubt module is as follows.

- API for adding new doubts, which works only if the user has logged in.
- API for listing doubt by id, which works only if the user is logged in.
- API for listing all of the listed doubts, except the ones listed by the user himself.
- API for listing doubts through filtering by tags. One can enter one or more number of tags.
- API for fetching doubts raised the user himself.
- API for mentoring a doubt, user can mentor any of the doubt and can raise his proposal for that doubt.
- API for fetching all those doubts for which the user has raised his proposal.
- API for initiating the booking for a doubt with a mentor by the user who has raised the doubt.
- API for confirming the booking raised by the user by the mentor.
- API for marking the doubt as solved by the user.
- API for returning all of the doubt ids of the user, where the status of doubt is in Progress.

Table 1 shows the broader description of all actions in the doubt API performed by the users who could be either doubt Solver, or doubt asker. Table 2 shows the database design for doubt, booking, and message models used in MongoDb database. Table 3 shows the broader description of functions and actions of Session API. Table 4 shows the broader description of features and action performed by users using Message API.

Table 1. Description of the actions implemented in doubt API

Sno	Action	Route	Body
1	Create a new doubt	{{url}}/mentor/doubt	title description tags raisedAmount
2	Fetch a given doubt	{{url}}/mentor/doubt/{{doubt_id}}	
3	Get all doubts listed other than yours	{{url}}/mentor/	
4	Get all doubts by tags	{{url}}/mentor/filter/aws,c++	
5	Get all doubts raised by self	{{url}}/mentor/own/	
6	Mentor a raised doubt, add your proposal to it	{{url}}/mentor/doubt/{{doubt_id}}	amount description mentor_address
7	Get all doubts to which one have added a proposal	{{url}}/mentor/final/	
8	User initiates a booking	{{url}}/book/initiate/{{doubt_id}}/mentor/{{mentor_id}}	user_address
9	Mentor accept the booking	{{url}}/book/confirm/{{booking_id}}	
10	Create a new doubt	{{url}}/mentor/doubt	title description tags raisedAmount
11	Fetch a given doubt	{{url}}/mentor/doubt/{{doubt_id}}	

Table 2. Schema of database tables used for doubt proposal module

Double schema	Booking Schema	Message schema
User :{ type: ObjectId}, Title :{ type: String, required: true}, Description :{ type: String, required: true, minlength: 20}, Tags :{ type: [String]}, Media :{ type: String}, Date: {type: Date}, RaisedAmount: { type: Number, required: true }, Status :{ type: String}	doubtId: {type: ObjectId }, userId: { type: ObjectId}, mentorId: {type: ObjectId}, description: { type: String, minlength: 20, maxlength: 100, }, date: {type: Date}, meetLink: { type: String}, amount: { type: Number }, status: { type: String }, userMetamaskAddress: { type: String }, mentorMetamaskAddress: { type: String }	fromID: { type: String, required: true}, content: [{messenger: String, message: String, }]

There are several Issues were in deciding how to map mentors to doubts along with their transaction details in an effective way. Finding out the design of models was a challenging task. To this end, all these scenarios are designed by considering edge case conditions like for updating amount after giving a proposal to check that mentor is able to update the amount until and unless the booking has not been confirmed between the mentor and the user and is in Pending state. While allowing the mentor to change Metamask address and amount we do check if the booking user wishes to update is the owner or not. APIs for backend of the message module are as follows:

- API for sending messages.
- API for receiving messages.
- API for deleting an entire message history.

Table 3. Functions of session API

Function	Use Case	input
createSession()	Put the data of the contract on the Blockchain and accept the fee amount.	uint amount, address payable doubtSolver, address payable doubtAsker, string topic, string id
getSessionData()	Retrieve the contract data from Blockchain for specific id.	string id
endSession()	Send the fee to the doubt Solver and mark the session is completed.	string id
refund()	Refund the fee amount to the doubt Asker and mark the session as completed.	string id

Table 4. Functions of Message API

Action	Route	Body
Get Message	{{url}}/get-messages/:userid/	
Send Message	{{url}}/send-message/:to	from
Delete Message	{{url}}/delete-messages/:id	

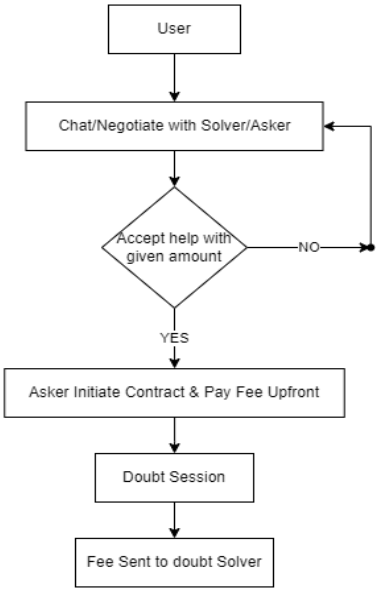

Fig. 5. Flow graph of the doubt contract

Model of messages includes fromID and content, which contains an array of messenger and message. It is like a one-on-one chat application where a message is stored in both sender and receiver messages. When Sending Message special emphasis has been on using Promises so that message is added in both sender and receiver simultaneously. While getting Messages, API is called with userID to retrieve all messages. When a user deletes a message, it is deleted only from his messages history and not from the receiver's history. Fig. 5 presents the flow of the doubt smart contract. A user can enter into chat box to negotiate with the doubt solver. If the negotiation is successful then the doubt asker initiates the contract and the fee will be deducted from asker account. Then the doubt asker and solver enter in to doubt classification session to discuss the topic. Once the discussion completed and the satisfactory consensus received from the doubt asker and the fee will be credited to the solver account.

4.2 Content delivery

Several front-end technologies have been used to implement the user interface of CodeBlockS. Brief description of the front-end technologies followed by their implementation in the system is as follows.

- React Js: React is a free and open-source front-end JavaScript library, which helps in building user interfaces based on UI components. Meta and a community of individual developers and companies maintain it. React is used to create dynamic and interactive web pages along with Redux for State management. Multiple components are created to handle different functionalities of the project like a Spinner that will be displayed during loading of content. A PrivateRoute component is created that handles the authentication on every page like even after the user login. CodeBlockS have an alert component that will alert the user if their action is completed like if the question they asked is added, if the answer they wrote is added or if we find the similar questions users have searched for, etc. CodeBlockS have used multiple react hooks like useState, useEffect. UseState is a hook that lets you add React state to function components. useEffect() tells React that a component needs to do something after render. A Link component of 'react-router-dom' which is used to create links in the application like navigating from one page to another. The useHistory component of 'react-router' is used to redirect to other pages.
- Axios: Axios is a JavaScript library, which is used to make HTTP requests from node.js or XML Http Requests from the browser, and it supports the Promise API that is native to JS ES6. We have used this module to fetch data from APIs created and send data from user to APIs.

- Redux: Redux is an open-source JavaScript library for managing and centralizing application state. It is commonly used with front-end libraries such as React or Angular for building user interfaces. We have used redux in state management where we use different reducers for modules like answer, question, alert, etc. that will handle what will be shown on the frontend without fetching from backend again and again.
- Bootstrap: Bootstrap is a free and open-source CSS framework; it is directed at responsive and mobile-first front-end web development. It has CSS- and JavaScript-based design templates for different interface components like font, tables, forms, etc. We have used bootstrap to do the styling of the web pages and make the website more appealing. We have used components like containers, buttons, badges and grid components, grid component is the most used component. The row and column are useful as they give great control over the page.
- React-Icons: It includes popular icons in the React projects easily with react-icons, this utilizes ES6 imports that allows users to include only the icons that your project is using. We have used react-icons to import high quality vector icons to make the website more appealing. We have icons for GitHub, LinkedIn, twitter, like, comment, expand, profile, etc.
- Web3.js: Web3.js is a collection of libraries which allow users to interact with a local or remote ethereum node, it uses a HTTP or IPC connection to do so. The web3 JavaScript library allows interaction with the Ethereum blockchain. It has multiple functionalities like it can retrieve user accounts, send transactions, interact with smart contracts, and much more. We have used web3 to interact with the contract on Ethereum blockchain from frontend, like retrieving the data from the blockchain and putting the data on the blockchain. The abi was taken from a local machine used with web3 to call different methods from the smart contract and send transactions to the user.

The following technologies have been used for smart contract implementation.

- Solidity: Solidity is an object-oriented programming language created specifically for constructing and designing smart contracts on Blockchain platforms by the Ethereum Network team. It is used to code the doubt smart contract, which handles the data of the contract between the doubt asker and the doubt solver. Moreover, the contract handles all the payments.
- Truffle: Truffle is a development environment, which uses the EVM (Ethereum Virtual Machine) as a basis. The slogan of Truffle is" Smart Contracts Made Sweeter", which indicates that the environment specializes in smart contract development. This environment has a number of great functionalities that help dApp developers tremendously. It is used for compiling the contract, migrating the contract on the blockchain and for testing purposes.
- Ganache: Ganache allows users to set up their own local Ethereum blockchain that they can use to deploy and test their smart contracts/dApps before launching them on an authentic chain. Ganache also enables developers to avoid paying unnecessary gas fees during the development process. It is used as a local Ethereum blockchain, for deploying and testing the smart contract.
- Chai assertion library: Chai assertion library is a JavaScript library, which is used to write assertions. Compared to what we write directly in JavaScript, chai assertion library requires less time & effort and is easy to use. It is used to test different functionalities of the smart contract, like if the contract made has correct details; the payment is getting in and out of the contract.
- Metamask: MetaMask is a software cryptocurrency wallet, which is used to interact with different blockchains. It allows users to access their Ethereum wallet using a browser extension or mobile app, which can then be used to interact with different decentralized applications. We used it to do transactions with the smart contract.

4.3 Testing details

Software testing is a very important phase of the software development life cycle as it verifies and validates the system under test. Various test cases developed and implemented for testing the CodeBlockS is as follows

A. Doubt module

The following conditions are tested while testing the Doubt Model.

- Authorized users can only add doubts.
- Doubts must have title, description and amount.
- Authorized users can only mentor doubts.
- Authorized users i.e. mentors can update the proposal only if the doubt is in pending state until a booking is confirmed.
- Only the user who created the doubt can mask it as solved.
- Fetching many doubts together, checking load balancing for it.

B. Smart contract module

Following conditions are tested while testing the smart contract using chai and truffle.

- The smart contract must have a valid address.
- The smart contract must have a name.
- The addresses of asker or solver cannot be null.
- The amount cannot be less than or equal to 0.
- The title cannot be empty.
- The data on blockchain must be consistent.
- The fee is handled correctly without getting lost during transactions.

The following are the test cases designed to test smart contract deployment and session management. Fig. shows the successful passing of all the test cases related to smart contract.

- Deployment
  - deploy successfully
    - Contract address is not equal to '0x0'.
    - Contract address is not equal to ''.
    - Contract address is not null.
    - Contract address is not undefined.
  - has a name
    - The name variable in contract is equal to "Doubt Session Contract".
- Session
  - Session contract created and fee received successfully
    - The id from the emitted event after the session created is equal to id sent to contract.
    - The address of doubt solver from the emitted event after the session created is equal to address of doubt solver sent to contract.
    - The address of doubt asker from the emitted event after the session created is equal to address of doubt asker sent to contract.
    - The topic from the emitted event after the session created is equal to the topic sent to contract.
    - The isCompleted variable from the emitted event after the session created is equal to false sent to contract.
    - The id from the emitted event after the session created is not equal to ''.
    - The topic from the emitted event after the session created is equal to ''.
    - The new balance of doubt asker is equal to the amount before contract initiation - the amount mentioned in contract.
  - Session is completed and fee transferred successfully
    - The id from the emitted event after the session created is equal to id sent to contract.
    - The address of doubt solver from the emitted event after the session created is equal to address of doubt solver sent to contract.
    - The address of doubt asker from the emitted event after the session created is equal to address of doubt asker sent to contract.
    - The topic from the emitted event after the session created is equal to the topic sent to contract.
    - The isCompleted variable from the emitted event after the session created is equal to true sent to contract.
    - The new balance of doubt solver is equal to the amount before contract initiation + the amount mentioned in contract.

C. Message module

Following conditions are tested while testing the message module

- Authorized users can only send messages and view their own messages.
- Messages cannot be empty.
- Database of sender and receiver gets updated simultaneously and has been made sure that either both gets updated or neither of them.

```
(base) siddhantjain@Siddhants-MacBook-Air client % truffle test
Using network 'development'.


Compiling your contracts...
===========================
> Everything is up to date, there is nothing to compile.



  Contract: Mycontract
    deployment
      ✓ deploy successfully
      ✓ has a name
    Session
      ✓ session contract created and fee received successfully (243ms)
      ✓ session is completed and fee transferred successfully (262ms)


  4 passing (675ms)
```

Fig. 6. Passing of test cases related to Smart contract

4.4 Final application

This section presents the user interface snapshots of the working of CodeBlockS. Fig. 7 shows the Login page, where users will enter their credentials for authentication. Fig. 8 shows the Profile page, where users will be able to access their information and can also update their information they want to display. Fig. 9 shows the state of different accounts after deployment and testing of smart contracts. Fig. 10 shows the Metamask account, where users can see the balance of Ethereum in their account and the public address. They can also check if the site is connected to the account or not. Fig. 11 shows different interactions took place while manually testing the platform functions. We can see that the user tried to refund the amount and it was successful and we can see that the user sent 0.1 eth to the contract. Fig. 12 shows the Add Doubt page, where users can add the title, description, tags and amount they are willing to pay. Fig. 13 shows the Doubts Feed, where users can see other user's doubts. Users can send messages to the doubt asker and can see detailed information about the doubt by clicking the "Open" button. Fig. 14 shows the message modal, where user can add the message they want to send to doubt solver. Fig. 15 shows the MyDoubts page, where the user can see the doubts asked by them.

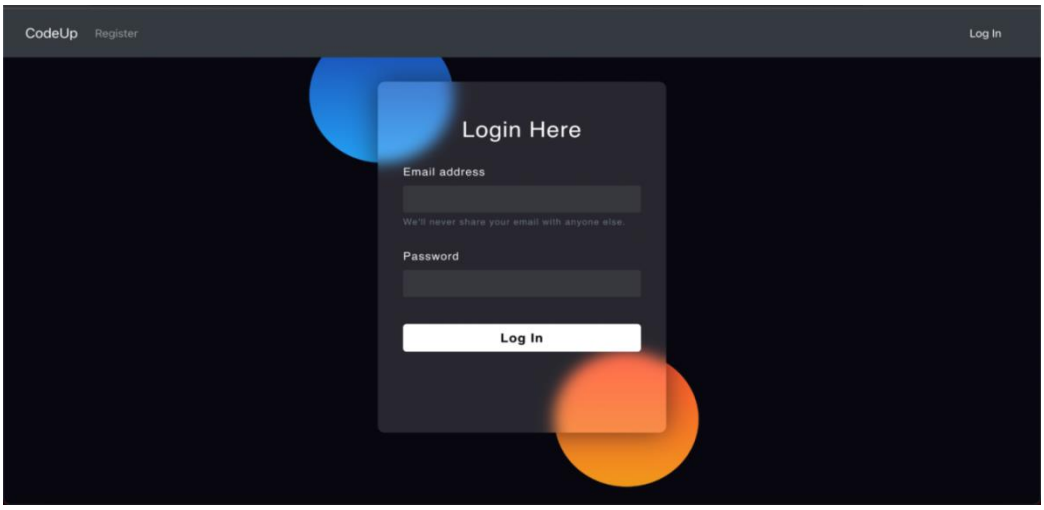

Fig. 7. Login Page

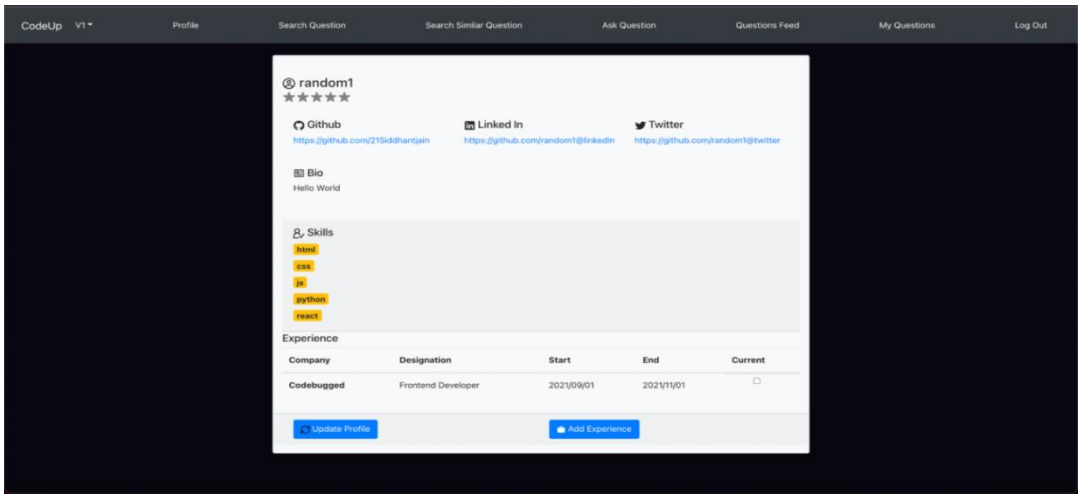

Fig. 8. User profile edit page in CodeBlockS

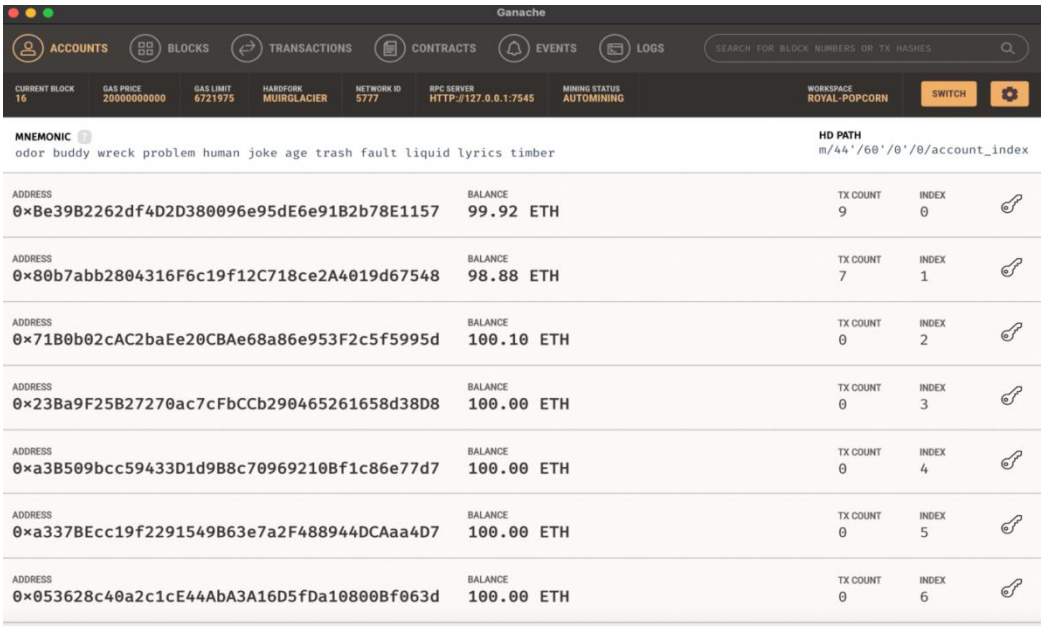

Fig. 9. Ganache Local Ethereum Blockchain

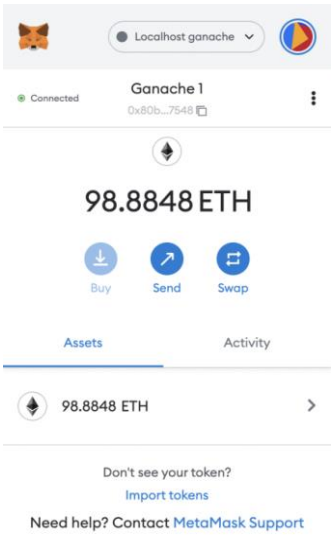

Fig. 10. Metamask wallet showing balance

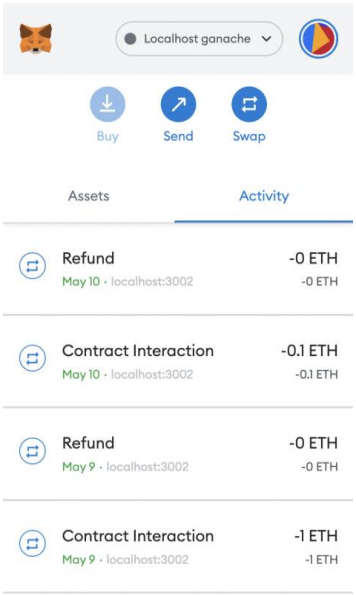

Fig. 11. Metamask wallet showing contract interactions

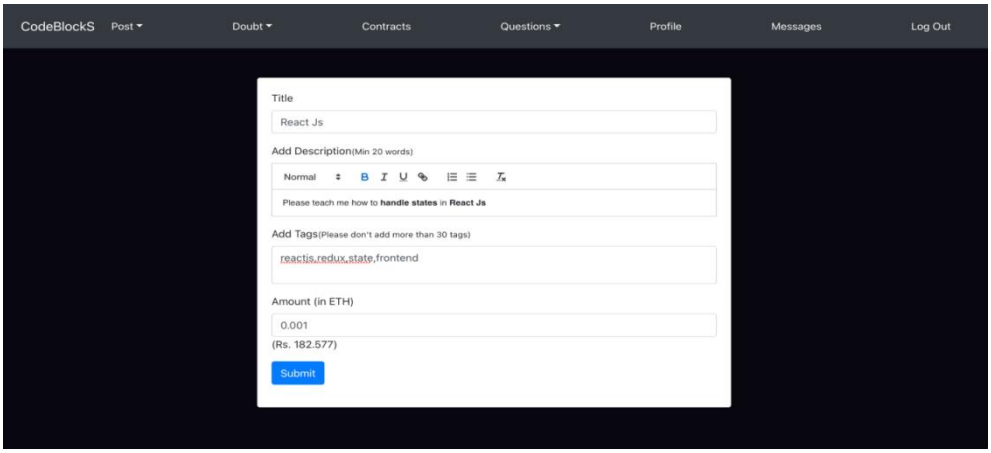

Fig. 12. Add Doubt

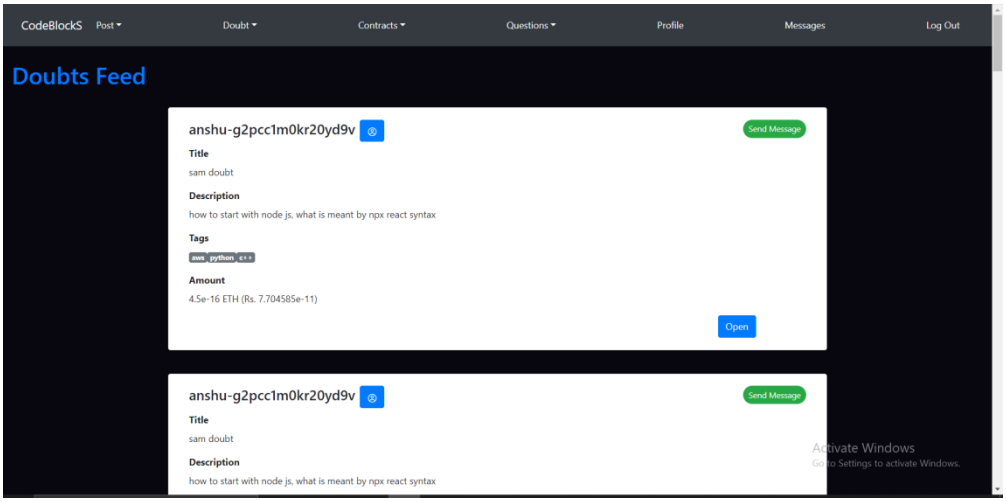

Fig. 13. Doubt Feed

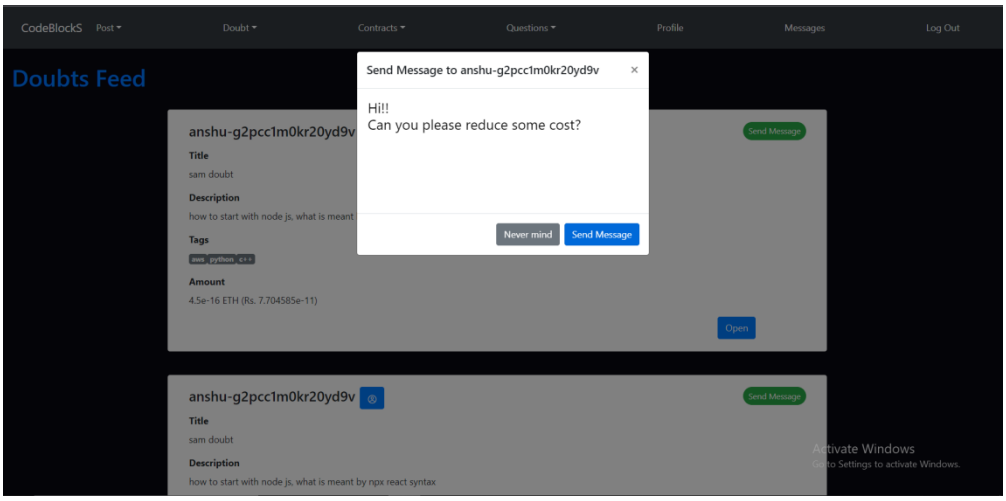

Fig. 14. Send Message

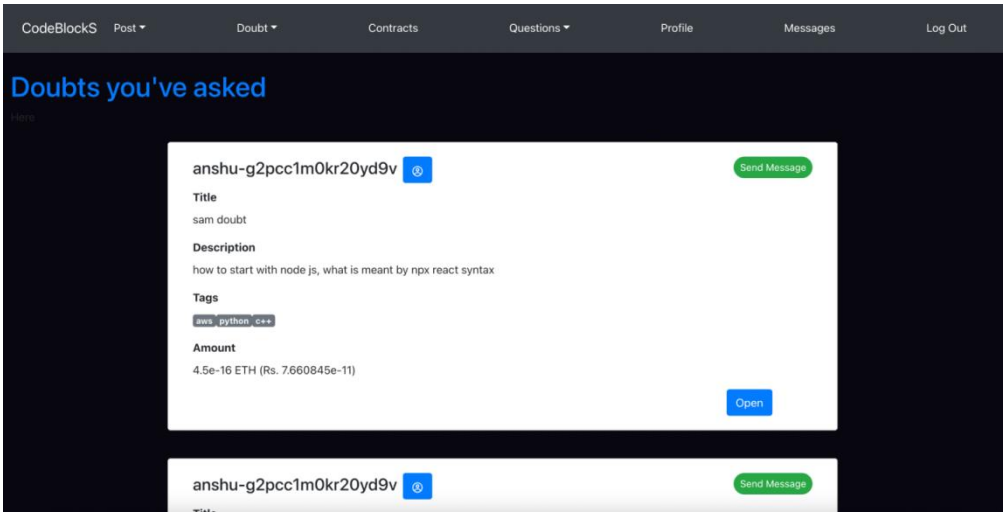

Fig. 15. Doubts asked by user

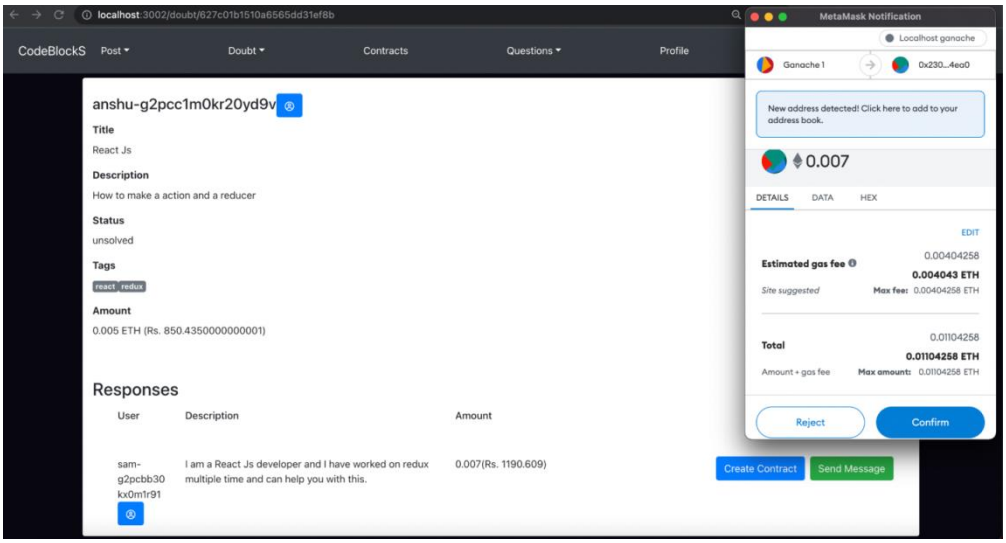

Fig. 16. Doubt Asker initiating the contract

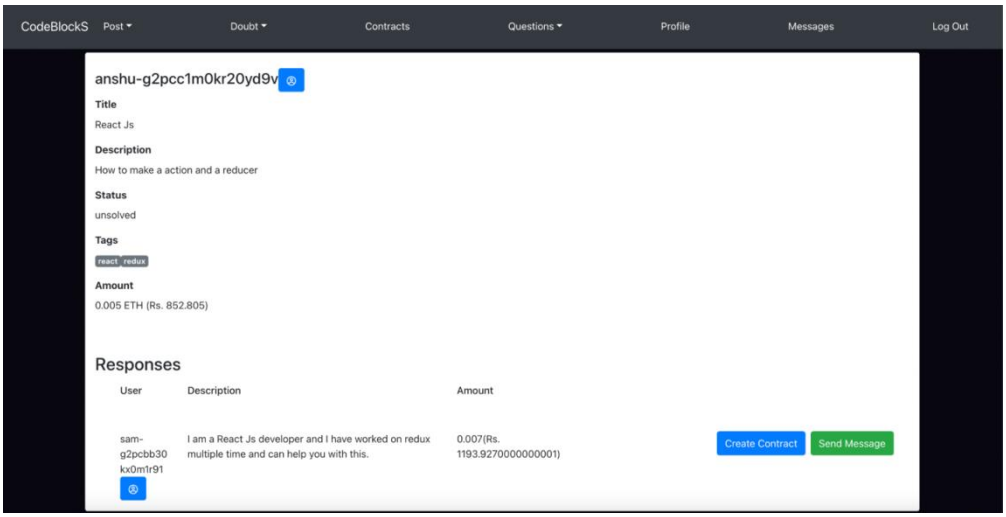

Fig. 17. Doubt Item (from doubt asker's view)

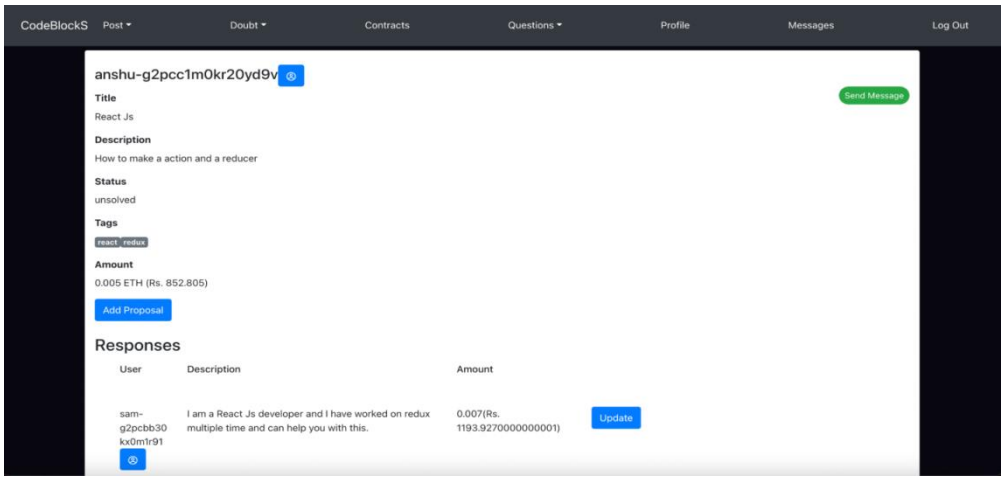

Fig. 18. Doubt Item (from doubt solver's view)

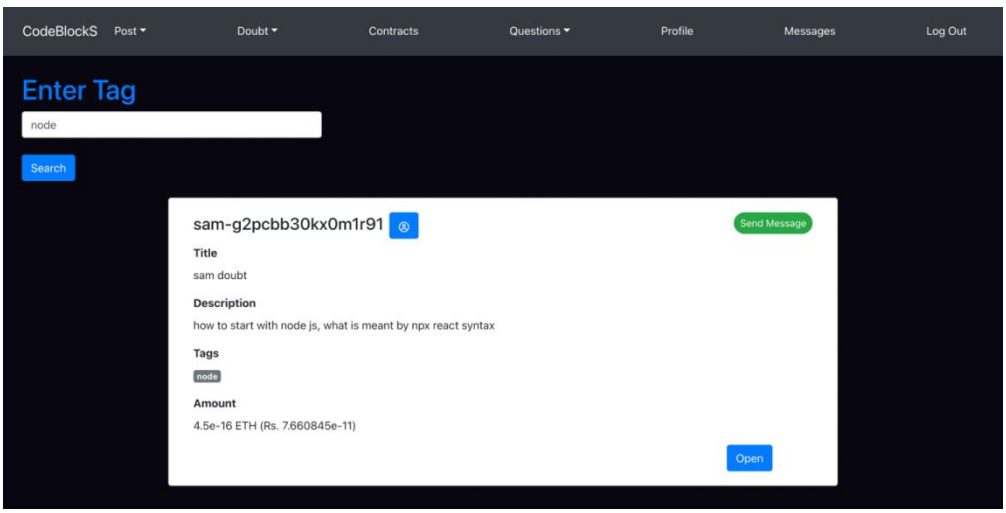

Fig. 19. Searching Doubts using tags

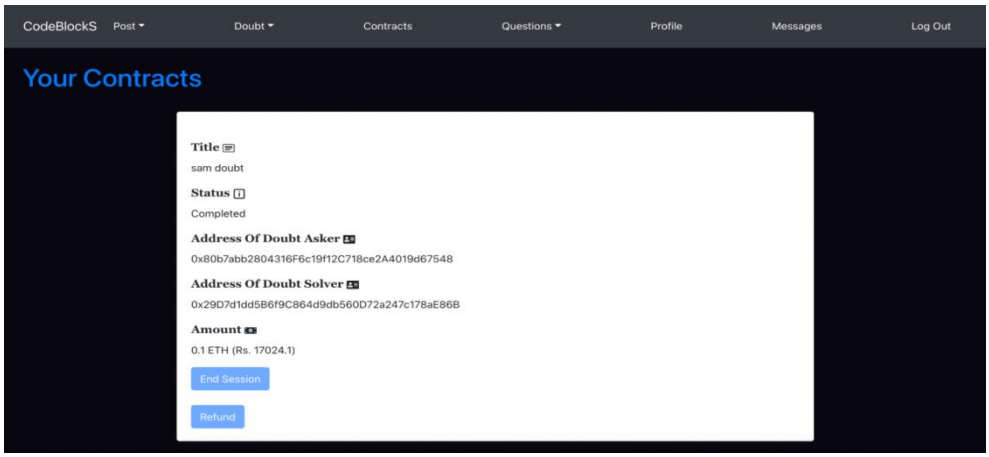

Fig. 20. Contracts page (data fetched from Blockchain)

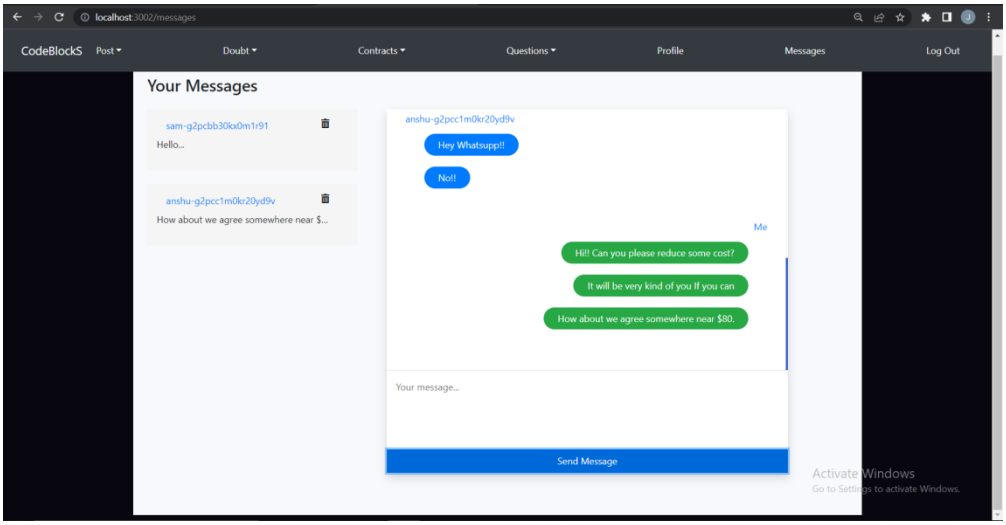

Fig. 21. Chat messages page

Fig. 16 shows the transaction request from the contract. After accepting the request, the amount will be sent to the smart contract and the data will be placed on Blockchain. Fig. 17 shows the Doubts page, where the user can see the details of doubts and the proposals by other users. We can see the option to “create contract” and “send message” to the doubt solver, the doubt asker can only see these options. Fig. 18 shows the Doubts page, where the user can see the details of doubts and the proposals by other users. Here we can see the option to update, which will allow user to change the amount or the description of their proposal. Fig. 19 shows the Search Doubt page, where users can enter a tag and get the doubts related with that tag. Fig. 20 shows the Contracts page, where users can see all the contracts they are part of. They can also either end the contract after the successful doubt session or can refund the money. These options will available to user’s according to their role in the contract i.e., doubt solver or doubt asker. Fig. 21 shows the Messages page, where users can see all the conversations with other users. Users can know about each other and can negotiate the amount of the contract here. The final application experimented by installing in Local Area Network of out University and around 30 final year graduate students are asked to use it for collaboration to clear doubts. It was found that the users were satisfied with the system towards ease of use. Readers may obtain the source code, installation and setup guide by sending email to the corresponding author for extending the work.

5. Conclusion and Future Work

In this paper, we presented the design and development of a collaborative knowledge-sharing platform with blockchain-based smart contracts (CodeBlockS) to help increase the trust and efficiency of how developers find solutions to their problems or try to learn new things. This application allows users to get their doubts cleared at reasonable fees and only pay for the skill or module they want to learn, this also incentivizes the expert and sincere mentor for solving other users' doubts via cryptocurrency. In addition, learners can learn the selected topics or clear their doubts in a trusted way, as the fee will be transferred to the smart contract before the sessions and will only be transferred to the account for the doubt after the commencement of the session. As future work, we will attempt to design and develop a specialized consensus protocol based on game theory and trust management to make the contract

fraud proof. Further, we attempt to extend the chat application to link up with various other social media platforms, such as WhatsApp, to provide notifications whenever a user receives a message from a mentor or other users.

References

[1] D. I. Castaneda and S. Cuellar, "Knowledge sharing and innovation: A systematic review," Knowledge and Process Management, vol. 27, p. 159–173, 2020.

[2] T.-M. Nguyen, "Four-dimensional model: a literature review in online organizational knowledge sharing," VINE Journal of Information and Knowledge Management Systems, 2020.

[3] Ø. Tønnessen, A. Dhir and B.-T. Flåten, "Digital knowledge sharing and creative performance: Work from home during the COVID-19 pandemic," Technological Forecasting and Social Change, vol. 170, p. 120866, 2021.

[4] J. Yao, A. Crupi, A. Di Minin and X. Zhang, "Knowledge sharing and technological innovation capabilities of Chinese software SMEs," Journal of Knowledge Management, 2020.

[5] Y. Sun, X. Zhou, A. Jeyaraj, R.-A. Shang and F. Hu, "The impact of enterprise social media platforms on knowledge sharing: An affordance lens perspective," Journal of Enterprise Information Management, 2019.

[6] J. Hao, L. Liu, A. A. von Davier, N. Lederer, D. Zapata-Rivera, P. Jakl and M. Bakkenson, "EPCAL: ETS platform for collaborative assessment and learning," ETS Research Report Series, vol. 2017, p. 1–14, 2017.

[7] S. Chatterjee, N. P. Rana and Y. K. Dwivedi, "Social media as a tool of knowledge sharing in academia: an empirical study using valance, instrumentality and expectancy (VIE) approach," Journal of Knowledge Management, 2020.

[8] V. S. Rekha and S. Venkatapathy, "Understanding the Usage of Online Forums as Learning Platforms," Procedia Computer Science, vol. 46, p. 499–506, 2015.

[9] P. Setialana, A. Fitria, R. Atika, R. N. Fadilla and others, "Development of WeShare As a Knowledge Sharing Platform to Realize the Freedom in Learning," in Journal of Physics: Conference Series, 2021.

[10] P. V. Rao and A. P. S. Kumar, "The societal communication of the Q&A community on topic modeling," The Journal of Supercomputing, vol. 78, p. 1117–1143, June 2021.

[11] R. O. de Castro, C. Sanin, A. Levula and E. Szczerbicki, "The Development of a Conceptual Framework for Knowledge Sharing in Agile IT Projects," Cybernetics and Systems, vol. 53, p. 529–540, 2022.

[12] M. S. Faisal, A. Daud, A. U. Akram, R. A. Abbasi, N. R. Aljohani and I. Mehmood, "Expert ranking techniques for online rated forums," Computers in Human Behavior, vol. 100, p. 168–176, November 2019.

[13] Richa Kushwaha, Akshit Raj Patel, P. Raghu Vamsi, "Transparent Recruitment Model with Blockchain Technology", International Journal of Information Security and Software Engineering, Vol 7, No 2, 2021, pp 11-23.

[14] S. Wang, L. Ouyang, Y. Yuan, X. Ni, X. Han, and F.-Y. Wang, "Blockchain-Enabled Smart Contracts: Architecture, Applications, and Future Trends," IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 49, no. 11, pp. 2266–2277, Nov. 2019, doi: 10.1109/tsmc.2019.2895123.

[15] Nakamoto, Satoshi. "Bitcoin: A peer-to-peer electronic cash system." Decentralized Business Review, 2008: 21260.

[16] Watanabe, S. Fujimura, A. Nakadaira, Y. Miyazaki, A. Akutsu, and J. Kishigami, "Blockchain contract: Securing a blockchain applied to smart contracts," 2016 IEEE International Conference on Consumer Electronics (ICCE), Jan. 2016, doi: 10.1109/icce.2016.7430693.

[17] Yadav, Sachin, and Surya Prakash Singh. "Blockchain Critical Success Factors for Sustainable Supply Chain." Resources, Conservation and Recycling, vol. 152, Jan. 2020, p. 104505. doi, https://doi.org/10.1016/j.resconrec.2019.104505.

[18] Wohrer, Maximilian, and Uwe Zdun. "Smart contracts: security patterns in the ethereum ecosystem and solidity." In 2018 International Workshop on Blockchain Oriented Software Engineering (IWBOSE), pp. 2-8. IEEE, 2018.

[19] Jiao, Jiao, Shuanglong Kan, Shang-Wei Lin, David Sanan, Yang Liu, and Jun Sun. "Semantic understanding of smart contracts: Executable operational semantics of solidity." In 2020 IEEE Symposium on Security and Privacy (SP), pp. 1695-1712. IEEE, 2020.

[20] U. Ravindran and P. R. Vamsi, "A Secure Blockchain based Finance Application," 2021 Thirteenth International Conference on Contemporary Computing (IC3-2021), Aug. 2021, doi: 10.1145/3474124.3474138.

[21] Tyagi, Nitin K., and Mukta Goyal. "Blockchain ‐ based Smart Contract for Issuance of Country of Origin Certificate for Indian Customs Exports Clearance.", Concurrency and Computation: Practice and Experience, Mar. 2021. doi, https://doi.org/10.1002/cpe.6249.

[22] L.-W. Wong, L.-Y. Leong, J.-J. Hew, G. W.-H. Tan, and K.-B. Ooi, "Time to seize the digital evolution: Adoption of blockchain in operations and supply chain management among Malaysian SMEs," International Journal of Information Management, vol. 52, p. 101997, Jun. 2020, doi: 10.1016/j.ijinfomgt.2019.08.005.

[23] Z. Liu and Z. Li, "A blockchain-based framework of cross-border e-commerce supply chain," International Journal of Information Management, vol. 52, p. 102059, Jun. 2020, doi: 10.1016/j.ijinfomgt.2019.102059.

[24] Z. Wan, X. Xia, and A. E. Hassan, "What Do Programmers Discuss About Blockchain? A Case Study on the Use of Balanced LDA and the Reference Architecture of a Domain to Capture Online Discussions About Blockchain Platforms Across Stack Exchange Communities," IEEE Transactions on Software Engineering, vol. 47, no. 7, pp. 1331–1349, Jul. 2021, doi: 10.1109/tse.2019.2921343.

Authors' Profiles

Siddhant Jain currently pursuing final year of Integrate B.Tech M.Tech in Computer Science and Engineering at Jaypee Institute of Information Technology, Noida, India. His programming interests include Ethereum Smart Contract development, and React application development.

P. Raghu Vamsi is currently working as Assistant Professor (Senior Grade) in the Department of Computer Science and Engineering (CSE), Jaypee Institute of Information Technology (JIIT), NOIDA, UP, India from July 2016. He received B.E in CSE from University of Madras in 2003, M.Tech (Software Engineering) from Kakatiya University in 2007, M.B.A in HRM from IGNOU in 2010, and M.A in Astrology from PSR Telugu University, Hyderabad in 2014, and PhD in CSE from JIIT, NOIDA in 2016. He has cleared technical certifications such as Core Java certification by Brain Bench, IBM DB2 Application and Fundamentals, and IBM Rational Application Developer for Web sphere. He has more than 15 years of teaching experience in various engineering institutions and 4 year of research experience. So far, he guided several undergraduate and post graduate projects and dissertations. His research interests include security in networks and communication, Wireless Adhoc Networks, IoT and Blockchain. He is a life member of Indian Society for Technical Education (ISTE) India.

Yashi Aggarwal currently pursuing final year of Integrate B.Tech M.Tech in Computer Science and Engineering at Jaypee Institute of Information Technology, Noida, India. Her programming interests include Backend Development, Cloud Computing, AWS and Competitive Programming.

Jayanth Goel received the Bachelor of Technology in Computer Science and Engineering from Jaypee Institute of Information Technology, Noida in 2022. He is currently working at Publicis Sapient, India. His research interest includes applications of machine learning, natural language process, web application development, and Blockchain.

How to cite this paper: Siddhant Jain, P. Raghu Vamsi, Yashi Agarwal, Jayant Goel, "CodeBlockS: Development of Collaborative Knowledge Sharing Application with Blockchain Smart Contract", International Journal of Information Engineering and Electronic Business(IJIEEB), Vol.15, No.1, pp. 1-19, 2023. DOI:10.5815/ijieeb.2023.01.01