CollabChain

Submitted by

Nahshal Manir (13000120095)
Dhiraj Kumar Sharma (13000120098)
Aniket Chakraborty (13000120124)
Meghadri Koley (13005320023)

7th Semester

December, 2023

Submitted for the partial fulfillment for the degree of Bachelor of Technology in Computer Science and Engineering

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

TECHNO MAIN SALT LAKE
EM 4/1, SALT LAKE, SECTOR - V, KOLKATA - 700091

ACKNOWLEDGEMENTS

We would like to express our sincere gratitude to our guide in the department of Computer Science and Engineering, whose role as project guide was invaluable for the project. We are extremely thankful for the keen interest he/she took in advising us, for the books and reference materials provided for the moral support extended to us.

Last but not the least we convey our gratitude to all the teachers for providing us the technical skill that will always remain as our asset and to all non-teaching staff for the gracious hospitality they offered us.

Place: Techno Main Salt Lake
Date:

Nahshal Manir (13000120095)

Dhiraj Kumar Sharma (13000120098)

Aniket Chakraborty (13000120124)

Meghadri Koley (13005320023)

Department of Computer Science and Engineering
Techno Main Salt Lake
Kolkata - 721302
West Bengal, India

APPROVAL

This is to certify that the project report entitled **CollabChain** prepared under my supervision by **Nahshal Manir** (Roll Number: *13000120095*), **Dhiraj Kumar Sharma** (Roll Number: *13000120098*), **Aniket Chakraborty** (Roll Number: *13000120124*) and **Meghadri Koley** (Roll Number: *13005320023*), be accepted in partial fulfillment for the degree of Bachelor of Technology in **Department of Computer Science and Engineering**.

It is to be understood that by this approval, the undersigned does not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn thereof, but approves the report only for the purpose for which it has been submitted.

Signature of
Internal Guide(s)

Signature of
the HOD

Signature of External Examiner

Contents

1 Introduction **1**
1.1 Abstract . . . 1
1.2 Problem Domain . . . 1
1.3 Glossary . . . 1

2 Problem Definition **2**
2.1 Scope . . . 2
2.2 Exclusions . . . 3
2.2.1 Functional Exclusions . . . 3
2.2.2 Technical Exclusions . . . 3
2.3 Assumptions . . . 4

3 Related Studies **4**
3.1 Survey . . . 4
3.2 Related & Existing Platforms . . . 7
3.3 Legitimacy of posted projects . . . 7
3.4 Use of CollabChain . . . 8
3.5 Hyperledger Fabric vs. Ethereum . . . 9

4 Project Planning **11**
4.1 Software Life Cycle Model . . . 11

5 Requirement Analysis **12**
5.1 Requirement Matrix . . . 12
5.1.1 User Registration [REG-1] . . . 16
5.1.1.1 REG-1.1 . . . 16
5.1.1.2 REG-1.2 . . . 17
5.1.1.3 REG-1.3 . . . 17
5.1.1.4 REG-1.4 . . . 17
5.1.2 User Login [LGN-1] . . . 17
5.1.2.1 LGN-1.1 . . . 17
5.1.2.2 LGN-1.2 . . . 17
5.1.3 User Profile [PRF-1] . . . 18
5.1.3.1 PRF-1.1 . . . 18
5.1.3.2 PRF-1.2 . . . 18
5.1.4 Platform Menu [PMNU-1] . . . 19
5.1.4.1 PMNU-1.1 . . . 19
5.1.4.2 PMNU-1.2 . . . 19
5.1.5 User Feed [UF-1] . . . 19
5.1.5.1 UF-1.1 . . . 19
5.1.5.2 UF-1.2 . . . 20
5.1.6 Completed Projects List View [LVW-1] . . . 20
5.1.7 Post Project [PSTPRJ-1] . . . 20
5.1.7.1 PSTPRJ-1.1 . . . 21
5.1.8 Owner Project View [OPVW-1] . . . 21

5.1.8.1 OPVW-1.1 . . . 22
5.1.8.2 OPVW-1.2 . . . 23
5.1.8.3 OPVW-1.3 . . . 24
5.1.9 Contributor Project View[CPVW-1] . . . 24
5.1.9.1 CPVW-1.1 . . . 25
5.1.9.2 CPVW-1.2 . . . 25
5.1.9.3 CPVW-1.3 . . . 25
5.1.9.4 CPVW-1.4 . . . 26
5.1.10 Project View [PVW-1] . . . 26
5.1.10.1 PVW-1.1 . . . 26
5.1.10.2 PVW-1.2 . . . 27
5.1.10.3 PVW-1.3 . . . 27
5.1.11 Researcher-Collaboration Page [RCOLLAB-1] . . . 27
5.1.11.1 RCOLLAB-1.1 . . . 28
5.1.11.2 RCOLLAB-1.2 . . . 28
5.1.11.3 RCOLLAB-1.3 . . . 29
5.1.11.4 RCOLLAB-1.4 . . . 29
5.1.11.5 RCOLLAB-1.5 . . . 29
5.1.12 Task/Sub-Task Modification [TMOD-1] . . . 29
5.1.13 Researcher Agreement Signing [RAS-1] . . . 30
5.1.14 Contributor Warning [CWARN-1] . . . 31
5.1.15 Contributor-Collaboration Page[CCOLLAB-1] . . . 32
5.1.15.1 CCOLLAB-1.1 . . . 32
5.1.15.2 CCOLLAB-1.2 . . . 33
5.1.15.3 CCOLLAB-1.3 . . . 33
5.1.16 Contributor Agreement Signing [CAS-1] . . . 33
5.1.17 Researcher/Contributor Chat [CHAT-1] . . . 34
5.1.18 Researcher Warning [RWARN-1] . . . 35
5.1.19 IRB Update [IRBUP-1] . . . 35
5.1.20 Foreground-IPR [FIPR-1] . . . 36
5.1.21 Exit Project [XPRJ-1] . . . 37
5.1.21.1 XPRJ-1.1 . . . 37
5.1.21.2 XPRJ-1.2 . . . 37
5.1.22 Stopped Project View [SPVW-1] . . . 37
5.1.23 Update Profile [UPF-1] . . . 38
5.1.24 Account Deletion [ACDEL-1] . . . 38
5.1.24.1 ACDEL-1.1] . . . 38
5.1.24.2 ACDEL-1.2] . . . 38
5.1.25 Log-Out [LGOUT-1]] . . . 38
5.1.25.1 LGOUT-1.1] . . . 39
5.1.25.2 LGOUT-1.2] . . . 39
5.1.26 Non-Functional Requirement: Security [NFS-1]] . . . 39
5.1.27 Non-Functional Requirement: Capacity [NFCAP-1]] . . . 39
5.1.28 Non-Functional Requirement: Compatibility [NFCOMP-1]] . . . 39
5.1.29 Non-Functional Requirement: Reliability & Availability [NFRA-1]] . . . 39

5.1.30 Non-Functional Requirement: Maintainability & Manageability [NFMM-1]] . . . 39
5.1.31 Non-Functional Requirement: Usability [NFU-1]] . . . 39
5.1.32 Non-Functional Requirement: Scalability [NFSC-1]] . . . 40
5.1.33 Non-Functional Requirement: Platform Dependent Agreement [NFPDAT-1]] . . . 40

6 Design **43**
6.1 Technical Environment . . . 43
6.2 Hierarchy of Modules . . . 43
6.3 Detailed Design . . . 54
6.3.1 User Registration [UREG] . . . 55
6.3.2 User Login [ULGN] . . . 55
6.3.3 Profile View and Platform Menu [PVPM] . . . 56
6.3.4 User Feed View & Access-ability [UFA] . . . 57
6.3.5 Project Creation & Maintainability [PCM] . . . 58
6.3.6 Contributor Project Overview [CPO] . . . 60
6.3.7 Platform Collaboration [PC] . . . 62

7 Test plan **64**
7.1 Figma Prototype Test Plan . . . 64

8 Conclusion **71**
8.1 Project Benefits . . . 71
8.2 Future Scope for improvements . . . 73

Bibliography **75**

A APPENDIX - Prototypes **76**
A.1 Environment Setup . . . 76
A.2 Coded Prototype . . . 77
A.2.1 Project APIs . . . 77
A.2.1.1 getAllProjects API [Similar to UF-1] . . . 77
A.2.1.2 createProject API [Similar to PSTPRJ-1] . . . 78
A.2.1.3 getProjectDetails API [Similar to OPVW-1/CPVW-1/PVW-1] . . . 80
A.2.1.4 editProject API [Similar to OPVW-1.1] . . . 81
A.2.1.5 applyToProject API [Similar to PVW-1.1] . . . 82
A.2.1.6 withdrawApplication API [Similar to PVW-1.1] . . . 83
A.2.1.7 updateMenteeStatus API [Similar to OPVW-1.1] . . . 84
A.2.1.8 addTask API [Similar to RCOLLAB-1.2] . . . 85
A.2.1.9 addTaskContributor API [Similar to RCOLLAB-1.2] 87
A.2.1.10 removeTaskContributor API . . . 89
A.2.1.11 uploadTaskWork API [Similar to CCOLLAB-1.3] . . . 91
A.2.1.12 getAssignedTasks API[Similar to RCOLLAB-1] . . . 93
A.2.1.13 markTaskComplete API [Similar to TMOD-1] . . . 94
A.2.2 User APIs . . . 96

A.2.2.1 registerUser API [Similar to REG-1.1] 96
A.2.2.2 loginUser API [Similar to LGN-1] 97
A.2.2.3 logout API [Similar to LGOUT-1] 98
A.2.2.4 updatePassword API [Similar to UPF-1] 99
A.2.2.5 updateProfile API [Similar to UPF-1] 100
A.2.3 Blockchain 101
A.2.3.1 createTask Function [Similar to PSTPRJ-1] 101
A.2.3.2 createTask Function [Similar to RCOLLAB-1.3] . . . 102
A.2.3.3 assignUser Function 103
A.2.3.4 removeUser Function [Similar to RCOLLAB-1.3] . . 103
A.2.3.5 createDocument Function [Similar to RCOLLAB-1.3] 104
A.2.3.6 completeTask Function [Similar to TMOD-1] 104
A.2.3.7 checkTaskStatus Function 105

1 Introduction

1.1 Abstract

CollabChain is a confidentiality based research collaboration platform. The platform allows interested individuals and researchers all around the world to collaborate on research works but at the main time maintaining confidentiality through block-chain technology. Online research and/or collaboration platforms existing till date, lack in the area of providing a proper medium for research content sharing and collaboration with confidentiality not only among different research teams but also within each team. CollabChain overcomes this limitation by bringing all these functionalities together into a single platform through the help of block-chain technology and at the mean time giving researchers the opportunity to choose their team members from an exceptionally large pool of candidates including school students to research workers themselves.

1.2 Problem Domain

The problem addressed by this project belongs to the research and development domain. The project aims to solve the problem through the integration of block-chain technology with online web application.

1.3 Glossary

Abbrev.	**Full Form**
R&D	Research & Development.
RCA	Research Collaboration Agreement.
IPR	Intellectual property rights.
API	Application Programming Interface.
FIPR	Foreground Intellectual property rights.
IRB	Institutional Review Board
R&D	Research & Development.

Continued on next page

Abbrev.	Full Form
CV	Curriculum Vitae.
T&Cs	Terms & Conditions.
UI/UX	User Interface/User Experience.
iOS	iPhone Operating System.
USP	Unique Selling Proposition/Unique Selling Point.
A. I.	Artificial Intelligence.

Table 1: *Table of Glossary*

2 Problem Definition

2.1 Scope

The project aims to promote research collaboration on-line through a single platform with the mechanism of confidentiality built-in. Allowing researchers to choose individuals for a project from a large pool of candidates, segregate tasks among selected Contributors without disclosing the actual aim of the project before publishing and user review on project completion, provides an unique package of functionalities within an on-line platform for R&D related work. Refer to [3] for more details on requirements and problems on research collaborations.

The platform offers two user types to choose from, Researcher & Contributor respectively. The "Researcher" user type allows an individual to host their own project and contribute to another project on the platform. The "Contributor" user type can be considered to be more suitable to individuals who are looking to only contribute to existing projects under the respective researchers.

Overall, the goal of the project is to provide researchers and contributors all around the world a common platform for collaboration keeping in check the confidentiality involved in the process. The project aims to provide motivation to researchers to perform at least a part of their research work on-line with the surety of non-disclosure of the actual project aim and problem statement before publication of their work. It also motivates individuals from different age groups and professional backgrounds to contribute to research work irrespective of their age groups.

2.2 Exclusions

State the exclusion.

2.2.1 Functional Exclusions

The section mentions the functionalities excluded i. e., not implemented due to their complexity or expense. Those functionalities are as follows:

1. *Mechanism to confirm the avoidance of RCA breach*: The mechanism/combination of functionalities required to assure the certainty of complying with the RCA of a project is too complex and too expensive in the sense that it involves different industries such as law, cyber-security, professional research communities to have a completely separate platform to keep a track of the activity of each individual involved in a project.

2. *Keeping track of published & revised research papers*: The platform allows project *Researcher* to attach the published and/or revised research papers to the corresponding projects but does not provide a mechanism to guarantee that the revised papers will be made available at the respective project pages.

3. *Mechanism to draft RCA & IPR documents*: The platform does not provide functionalities to draft any agreement document required for a project to avoid situations which might make collaboration on the platform rigid and/or one-sided for a *Researcher* or a *Contributor*.

2.2.2 Technical Exclusions

This section mentions the technical supports excluded from the platform for time constraint, project expenses and/or unavailability of appropriate APIs:

1. The platform does not provide any third-party application implementation.

2. The platform does not provide support for proper presentation of the application accessed through web browsers on devices other than a proper laptop, desktop PC or mobile devices.

3. The Platform does not provide the view of any other file format than "¡file_name¿.pdf". Therefore, all non ".pdf" files need to be downloaded to be accessed during collaboration.

2.3 Assumptions

State the specific assumptions under which your project will work.

The assumptions stated below and the user types defined are to be followed through out the document and system:

1. **Type of research**: The aim of the platform is to maintain confidentiality in collaborative, mentor-mentee or peer-to-peer collaborative research work.

2. **User-Types**: The platform classifies a user as either a *Researcher* or a *Contributor*. The user-type *Researcher* resembles the *mentor or lead-researcher* role of a collaborative research project & the user-type *Contributor* resembles the role of a *mentee or associate-researcher*.

3. **Experience**: Either of the user-types of the platform are assumed to be familiar and/or aware about the process and tasks involving collaborative research work in the real world.

4. **IPR & NDA Agreements**: The *Researcher* and a *Contributor* of a project while drafting and authenticating the RCA & FIPR for the concerned project must include the mentioned platform dependent agreement T&Cs under non-functional requirements, platform_agreements [NFPDAT-1] and comply with them so as to avoid authenticity violation problems later during the research work.

3 Related Studies

3.1 Survey

A survey was conducted to understand the need of the platform proposed in this project. Following are the response statistics from the res-ponders:

Figure 1: Survey Question 1

Figure 2: Survey Question 2

Figure 3: Survey Question 3

To contribute to the on-going survey, click here!

Figure 4: Survey Question 4

Figure 5: Survey Question 5

Figure 6: Survey Question 6

3.2 Related & Existing Platforms

The use of blockchain technology for protective, confidential collaboration work is a potential assumed by many. There are a no. of platforms existing online, while creating this document which represent certain similar functionalities as CollabChain. The list of the relevant platforms are provided below:

1. ***ResearchHub***: Facilitates seamless data sharing, transparent tracking of contributions, and auditable collaboration among researchers. [**Missing**: Collaborative-confidential research work to support publication of research paper i. e., only published research papers are available for peer-review and pre-printing reviews through open bounties] [8]. For further reference click here!. More information on platform USP is available here!.
2. ***ScienceOpen***: A discovery platform with interactive features for scholars to enhance their research in the open, make an impact, and receive credit for it. We provide context building services for publishers, to bring researchers closer to the content. [**Missing**: Unlike CollabChain, it provides researchers with post-publication paper-review functionalities] [9]. For further reference click here!

   Other notable mentions who are working on collaborative blockchain platforms, tools and/or technologies are: Blockchain for Scientific Data Sharing (BCSD), Everledger, EnjinCoin, Storj.

3.3 Legitimacy of posted projects

The specific documents that serve as proof of legitimacy for conducting research work can vary depending on the type of research, the funding source, and the institution where the research is being conducted. However, some common documents that can serve as proof of legitimacy include:

1. Research Proposal: A well-structured research proposal outlines the research question, methodology, potential risks and benefits, and ethical considerations. It serves as a blueprint for the research project and demonstrates that the researcher has a clear plan for conducting the research [].

2. IRB Approval Letter: An IRB approval letter signifies that the research protocol has been reviewed and approved by an institutional review board (IRB), which ensures that the research adheres to ethical standards and protects the rights and welfare of human participants.

3. Data Management Plan: A data management plan outlines how the research data will be collected, stored, secured, and shared. It demonstrates the researcher's commitment to responsible data management practices and ensures the integrity and accessibility of the research findings [].

4. Citations by Other Researchers: Citations in scholarly publications indicate that the research has been recognized and valued by other researchers in the field. It demonstrates the impact and significance of the work.

5. Awards and Recognition: Receiving awards or recognition for the research further establishes its legitimacy and highlights its contributions to the field of study.

6. Publication in Peer-Reviewed Journals: Publication in reputable peer-reviewed journals is a strong indicator of research legitimacy. It signifies that the research has undergone rigorous scrutiny by experts in the field and has been deemed worthy of publication.

By taking the above information into account we can say that IRB approval letter serves as a legitimate proof for conducting a research work and is done under the sound knowledge of the institution providing the approval letter. For further information on documents that serve as proof for conducting research works, refer to [2] [7].

3.4 Use of CollabChain

The inference that can be drawn from the above two sub-sections is that CollabChain is a platform that comes into play when researchers and contributors are looking to work in a confidential environment for the sake of creation of insightful and original research work with the added advantages of document protection and peer-review after paper publication. The platforms mentioned above provide a means for commenting and reviewing published papers and indirectly and

opportunity to come up with new ideas through those review and discussions. What they do not provide is an environment to conduct research work without the risk of data breach or any possible means of preventing plagiarism.

To see how the above platforms are a subset of CollabChain, assume the following: *You being a researcher working on a highly confidential but invaluably useful project want contributors to contribute to the project. There are no existing papers on the project title and you want to be the first to publish such an article. From finding contributors, to maintaining signed RCAs and IPRs for the project, assigning tasks to each contributor, segregating research tasks, tracking the work progress by each contributor, everything needs to be done in-order for the research to be successful. After publication, you are now ready to share the paper and put it out their for peer-review.* CollabChain can help you in this entire process, from the very beginning, right up till the end. It provides all these functionalities within a single platform.

3.5 Hyperledger Fabric vs. Ethereum

Hyperledger Fabric is an open-source blockchain framework hosted by The Linux Foundation. It offers a modular architecture that supports a wide range of use cases, including supply chain management, trade finance, and healthcare. Hyperledger Fabric is a permissioned blockchain, meaning that only authorized participants can join the network. This makes it ideal for businesses that need to share sensitive data with their partners. Hyperledger Fabric is a permissioned blockchain, which means that only authorized participants can join the network. This makes it ideal for businesses that need to share sensitive data with their partners [1].

Hyperledger Fabric is a scalable platform, which means that it can be used to support a large number of users and transactions. This makes it ideal for businesses that need to grow their network over time.
Ethereum is a decentralized, open-source blockchain platform that features its own cryptocurrency, Ether. It allows anyone to deploy permanent and immutable decentralized applications (dApps) onto it, with which users can interact.

Decentralized finance (DeFi) applications provide financial instruments which do not directly rely on financial intermediaries like brokerages, exchanges, or banks [5].

Use Cases of Ethereum

1. Decentralized finance (DeFi): Ethereum is the leading platform for DeFi applications, which provide a wide range of financial services without the need for intermediaries.

2. Non-fungible tokens (NFTs): Ethereum is the most popular platform for NFTs, which are digital assets that represent ownership of unique items.

3. Supply chain management: Ethereum can be used to track the movement of goods and materials throughout a supply chain, making it more efficient and transparent.

4. Healthcare: Ethereum can be used to securely store and share medical records, making it easier for patients to access their information.

Hyperledger Fabric	**Ethereum**
Hyperledger Fabric supports private transactions, where transaction details are visible only to the parties involved, not the entire network. This feature protects confidential information, such as research findings or intellectual property, from prying eyes.	Ethereum's public nature, while beneficial for openness and transparency, poses challenges for confidentiality. All transactions and data on the Ethereum blockchain are visible to the public, potentially exposing sensitive research information.
Hyperledger Fabric utilizes channels, private subnetworks within the blockchain, to compartmentalize data and restrict access to specific participants. This allows for fine-grained control over data sharing, ensuring that only authorized researchers can access relevant information.	While Ethereum offers some privacy-preserving mechanisms, such as zero-knowledge proofs, they are not as robust or widely adopted as Hyperledger Fabric's privacy features.
Hyperledger Fabric is designed as a permissioned blockchain, meaning that participants in the network must be identified and authenticated. This makes it suitable for enterprise applications where confidentiality and controlled access are crucial.	Ethereum, in contrast, is a permissionless blockchain, meaning that anyone can participate in the network without explicit permission. This openness is more suitable for public blockchain applications, like decentralized finance (DeFi) and token issuance.

Table 2: *Comparison between hyperledger fabric & ethereum [project related]*

In conclusion, Hyperledger Fabric's focus on privacy, permissioned network structure, and modular design make it the ideal platform for developing confidentiality-based research collaboration platforms [6]. It empowers researchers

to securely share sensitive data, foster collaboration, and advance their research endeavors while safeguarding their intellectual property and complying with data privacy regulations.

4 Project Planning

4.1 Software Life Cycle Model

State the suitable model you have chosen for your project development.
The project would be able to reach it's true potential when developed following the agile software life cycle model. Following this methodology the team can respond to changes in the marketplace or feedback from customers quickly without derailing a year's worth of plans. "Just enough" planning and shipping in small, frequent increments will let the team gather feedback on each change and integrate it into future plans at minimal cost.

Figure 7: Agile Methodology [4]

5 Requirement Analysis

5.1 Requirement Matrix

As for Requirement Matrix entries, copy and paste the latest excel with all functional, non-functional and interface requirements and format under this section.

Rqmt Id	Requirement Item	Requirement Analysis Status
REG-1	User Platform Registration	Completed
REG-1.1	Allow user registration ([Select] UserType: Researcher/Contributor, Name, Organisation Name, Professional Designation, Updated CV to be uploaded (pdf format), MailID, Phone Number, Account password); CV should be stored with user record;	Completed
REG-1.2	Verify user registration MailID	Completed
REG-1.3	Verify user contact number	Completed
REG-1.4	Send welcome message to registered MailID.	Completed
LGN-1	Verify User Credentials	Completed
LGN-1.1	Verify user type, user ID (the registered mail ID) & password	Completed
LGN-1.2	Forgot password; Allow user to reset password	Completed
PRF-1	Display User Profile Information & Status	Completed
PRF-1.1	Display user respective info. : 1. Full Name; 2. Current Organization; 3. Current Professional Designation; 4. Profile Picture (If any); 5. User-Type; 6. User Ratings;	Completed
PRF-1.2	Display the project(s): 1. The User has completed contributing; 2. The User is currently contributing to; 3. Saved Projects	Completed
PMNU-1	Platform Menu	Completed
PMNU-1.1	If the user is of UserType: Contributor then display the following menu options: 1. User Feed; 2. Account Settings; 3. LogOut	Completed
PMNU-1.2	If the user is of UserType: Researcher then display the following menu options: 1. User Feed; 2. Post Project; 3. Account Settings; 4. LogOut	Completed
UF-1	User Feed	Completed
UF-1.1	List projects: 1. Currently accepting applicants; 2. Completed Projects (Up for review by the audience);	Completed
UF-1.2	Projects ongoing should direct the users to the respective Project page.	Completed
LVW-1	Completed Projects List View: Lists projects based on most-likes in: 1. Descending order from top to bottom; 2. Ascending order from top to bottom; 3. ProjectID; Allow view based on the following filters: 1. Project Domain Tag; 2. Project Title; etc.	Completed
PSTPRJ-1	Post a new project on the platform (Only for UserType: Researcher): Fill up new project details (Project Title, [Select /Enter]Project Domain Tag(s), Initial No. of candidates required, Application Start & End Date, ProjectID (Autogenerated));	Completed
PSTPRJ-1.1	Uploading Research IRB Approval Letter & Researcher digital signature.	Completed

Continued on next page

Rqmt Id	Requirement Item	Requirement Analysis Status
OPVW-1	[Project-View for UserType: Researcher & the user is also the project owner]: Display project respective info. : 1. Project Name & ID; 2. Project Domain & Topic; 3. Application Start & End Date [if still accepting application]; 4. Project Status: [Recruiting/In-Progress/Completed];	Completed
OPVW-1.1	Display the following only when [Project Status: Recruiting]: 1. Display total no. of applicants [if still accepting applications]; 2.Display Applicant-Specific Info: 2.1 Applicant Name (Full Name); 2.2 Applicant Organisation Name; 2.3 Applicant Professional Designation; 2.4 No. of applicant contributions (in the project-domain); 2.5 Applicant CV Link; 2.6 Accept [Option]: Yes, No; 3. No. of candidates remaining to be selected	Completed
OPVW-1.2	Display the following only when [Project Status: In-Progress]: 1.1 Display the list of selected candidates in tile format i.e. clicking on a contributor's tile will lead the researcher to the specific contributor's Collaboration page; 1.2 Option to remove a contributor from the project; 2. No. Of Contributors "currently" working on the project;	Completed
OPVW-1.3	Display the following only when [Project Status: Completed]: 1. Display the list of selected candidates; 2. Researcher published document link; 3. Additional links [If any]; 4. Rate each contributor [e.g. on a scale of 1 to 10]; 4. The tasks done by each contributor; 5. The updated rating of each contributor;	Completed
CPVW-1	[Project-View for UserType: Contributor/Researcher but the user is not the project owner, only a project-contributor]: Display project respective info. : 1. Project Name & ID; 2. Project Domain; 3. Application Start & End Date [if still accepting application]; 4. Project Start & End Date [if on-going]; 5. Project Status: [Recruiting/In-Progress/Completed]; 6. Researcher CV Link	Completed
CPVW-1.1	Display the following only when [Project Status: Recruiting]: 1. No. of applicants; 2. Application Status: Accepted; 3. IRB Approval Letter 4. Option to upload digital signature	Completed
CPVW-1.2	[CCOLLAB Page assigned]: Provides a link to the CCOLLAB page on successful submission of dates and digital signature under [CPVW-1.1] .	Completed
CPVW-1.3	Display the following only when [Project Status: In-Progress]: Display an option named "Collaborate" to direct contributor to the private Collaboration Page	Completed

Continued on next page

Rqmt Id	Requirement Item	Requirement Analysis Status
CPVW-1.4	Display the following only when [Project Status: Completed]: 1. Researcher published document link of the project; 2. Additional links [if any]; 3. Rate Project & Researcher [e.g. on a scale of 1 to 10]; 4. The tasks done by the contributor; 5. The updated rating of each contributor;	Completed
PVW-1	Project-View for UserType: Contributor/Researcher but the user is neither the project owner or a project-contributor [Displaying Project Title, Project Domain Tags and ID is a must]	Completed
PVW-1.1	Display the following only when [Project Status: Recruiting]: 1. No. of applicants; 2. Option to apply; 3. Application Status[if applied]: Pending, Accepted, Rejected; 4. Option to Retract [if applied]	Completed
PVW-1.2	Display the following only when [Project Status: In-Progress]: 1. Project Start & End Date; 2. Researcher CV Link; 3. Project Status: In-Progress;	Completed
PVW-1.3	Display the following only when [Project Status: Completed]: 1. Researcher published document link of the project/ The document itself; 2. Additional comments/links [if any]; 3. Rate Project [e.g. on a scale of 1 to 10]; 4. The task done by each contributor; 5. The updated rating of each contributor;	Completed
RCOLLAB-1	Collaboration Page Functionalities for [UserType: Researcher]: Display: 1. Contributor Name; 2.Tasks List with status (% completions); 3.Collaboration Menu; 4. Option for [CWARN-1]	Completed
RCOLLAB-1.1	Collaboration Menu with 4 items: 1. Post Task; 2. Tasks & Sub-Tasks; 3. Notifications; 4. Contributor Info. & Agreement(s)	Completed
RCOLLAB-1.2	Researcher can post a task with the following details: 1. Task Name; 2. Expected End-Date; 3. Priority: High, Medium, Low; 5. Remarks [If Any]	Completed
RCOLLAB-1.3	Researcher can post a sub-task under a task with the following details: 1. Sub-Task Name; 2. Due Date; 3. Priority: High, Medium, Low; 4. Attach Files;	Completed
RCOLLAB-1.4	[Confidential files]: The researcher needs to digitally sign the documents that are to be kept strictly confidential by the concerned contributor	Completed
RCOLLAB-1.5	[For each contributor]: After the completion of the latest task by the contributor, the researcher is required to digitally sign the task for confirming task completion	Completed

Continued on next page

Rqmt Id	Requirement Item	Requirement Analysis Status
TMOD-1	Researcher's possible modifications on an existing sub-task and/or task: 1. Update/Edit: Task Name; Due Date; Priority Level; Remarks; Task Status; 2. Attach/Remove files; 3. Delete task/sub-task;	Completed
RAS-1	[(User-Type: Researcher) Research Collaboration Agreement: Between the Researcher and Contributor]: 1. Uploading the RCA by the researcher and review for the contributor 2. Digital signature of the researcher and contributor, both are required to save the document and proceed to collaboration with the concerned Contributor.	Completed
CWARN-1	[User-Type: Researcher]: Researcher warns Contributor on agreement violation	Completed
CCOLLAB-1	[Collaboration Page Functionalities for UserType: contributor]: Display: 1. Project Name & ID; 2. Researcher Info.; 3.Tasks List with status; 4.Collaboration Menu; 5. Researcher Info. & Agreement(s);	Completed
CCOLLAB-1.1	Collaboration Menu with 2 items: 1. Tasks; 2. Chat; 3. RAS-1;	Completed
CCOLLAB-1.2	View: 1. Task Name; 2. Due Date; 3. Priority Level; 4. Task Status	Completed
CCOLLAB-1.3	Download Attached Files to the task; Submit a file under the task or Remove an already submitted file only by the current-user, within the due date of the task.	Completed
CAS-1	[(User-Type: Contributor) Research Collaboration Agreement: Between the Researcher and contributor]: 1. Digital signature of the researcher and contributor, both are required to save the document and proceed to collaboration with the concerned contributor.	Completed
CHAT-1	Chat conversations between the Researcher & a Contributor for discussion purposes.	Completed
RWARN-1	[User-Type: Contributor]: Contributor(s) warns Researcher on agreement violation.	Completed
IRBUP-1	[User-Type: Researcher]: Researcher can update the IRB Approval Letter	Completed
FIPR-1	Foreground IPR: Researcher-Contributor agreement	Completed
XPRJ-1	Remove contributor (UserType: Contributor & Researcher) : To be done by the consent of both the researcher and the contributor	Completed
XPRJ-1.1	The researcher accepts project exit request from the contributor.	Completed
XPRJ-1.2	The researcher denies project exit request from the contributor.	Completed
SPVW-1	Stopped Project View: On agreement violation by the researcher.	Completed
UPF-1	User can update the existing info. in the platform i.e., Name, Organization Name, Professional Designation, Upload updated CV, register a new mailID and/or replace existing one, register a new phone number and/or existing one.	Completed

Continued on next page

Rqmt Id	Requirement Item	Requirement Analysis Status
ACDEL-1	Account Deletion.	Completed
ACDEL-1.1	User account will be deleted after user re-confirms warning prompt for account deletion.	Completed
ACDEL-1.2	All User data will be removed from the record	Completed
LGOUT-1	LogOut from current session	Completed
LGOUT-1.1	User to be displayed warning prompt to confirm logging out.	Completed
LGOUT-1.2	User to be redirected to the platform LogIn Page.	Completed
NFS-1	Security	Completed
NFCAP-1	Capacity	Completed
NFCOMP-1	Compatibility	Completed
NFRA-1	Reliability & Availability	Completed
NFMM-1	Mainatainanility & Manageability	Completed
NFU-1	Usability	Completed
NFSC-1	Scalability	Completed
NFPDAT-1	Platform Dependent Agreement Terms	Completed

Table 3: *Requirement Matrix [Traceability Matrix (Columns A-C)]*

The beginning of each main requirement states the user-type and user role with respect to a project hosted in the platform [if applicable]. If none of them is mentioned, implies the fact that the functionality is available for any user of the platform irrespective of the role in any project.

5.1.1 User Registration [REG-1]

5.1.1.1 REG-1.1 User platform registration/account creation for a new user. A new user to the platform is first asked for filling the following details:

- Select User-Type: [Researcher/Contributor]
- Full name.
- Name of the institute/organization the user is currently part of.
- Professional designation/the position they hold at the institute/organization.
- Updated CV, to be uploaded in .pdf format.
- Profile Picture [optional].
- Mail ID.
- Contact no.
- Account password.

The uploaded CV is to be stored as part of user information in the database.

5.1.1.2 REG-1.2 User mail verification is to be done by sending a verification mail to the user. The content of the mail should be an account creation confirmation message followed by a link to the login page of the platform.

5.1.1.3 REG-1.3 User contact no. verification should be done by sending the user an account creation confirmation SMS with a link to the login page.

5.1.1.4 REG-1.4 After the user has successfully logged into their account for the first time, send a welcome message to the user confirming their registration was successful and they are all set to use the platform from there on.

5.1.2 User Login [LGN-1]

5.1.2.1 LGN-1.1 The user is asked for the following platform credentials for logging into their respective accounts:

- Select User-Type: [Researcher/Contributor].
- User ID which is the user registered mail id.
- Current Account password.

Login Success: On successfully entering the above credentials, the user is directed to their platform profile page.
Login Failure: Failing to enter the above credentials correctly, the user is asked to re-enter their user id and password. After each failed attempt a warning message is displayed, indicating *the no. of attempts remaining* before the account is blocked for a period of time. After *5 such failed attempts*, the user account is blocked for *20 minutes* and a warning mail is sent to the user, notifying the user of the activity.

5.1.2.2 LGN-1.2 The user might have forgotten the login password for the account and therefore can choose to reset their account password through this option.

The forgot password option asks for the user's *registered mail id or contact no.* to which a re-set password link is sent. After clicking on the link through either of the mediums, the user is *directed to a re-set password page.*

Resetting password: In the page the user is asked to *type and re-type the new password* and confirm their entry. After successfully completing the above step, the account password for the platform will be *updated to the newly entered password* in this step and the user can login to the account through this password.

5.1.3 User Profile [PRF-1]

Display User Profile Information along with completed, in-progress and saved projects list by the user.

5.1.3.1 PRF-1.1 The user after login to the platform is directed to the profile page where the following information regarding themselves is displayed:

- User Type & Image or Avatar [If any].
- Full Name
- Current Organization
- Current Professional Designation
- Profile Picture[if uploaded].
- Uploaded CV Link.
- Average User ratings: The total ratings received by working on all participated projects divided by the total no. of projects they have worked on ("Completed" projects).

5.1.3.2 PRF-1.2 Three categories of project, based on status is displayed in this page well. The three categories being:

1. ***Completed:*** Display the list of projects the user has completed the respective contributions to along with the link to the task completed in each project.
2. ***In-Progress:*** Display the project that is the user is currently hosting/contributing to & the respective tasks completed till date (provided the user is a contributor).
3. ***Saved:*** The list of projects that the user had saved for later to review or for applying to contribute.

5.1.4 Platform Menu [PMNU-1]

The platform menu is visible & accessible for every page of the platform for which explicitly no menu has been defined.

5.1.4.1 PMNU-1.1 If the user is of type: [Contributor] then display the following options:

1. *User Feed:* For checking the lists of projects in the platform (completed/current). [Refer to UF-1]
2. *Profile*: To view the platform profile page of the user.
3. *Account Settings:* For profile updating purposes/ registration of new mail ID and/or contact number. [Refer to UPF-1 & ACDEL-1]
4. *LogOut:* For ending the current session. [Refer to LGOUT-1]

5.1.4.2 PMNU-1.2 If the user is of type: [Researcher] then display the following options:

1. *User Feed:* For checking the lists of projects in the platform (completed/current).
2. *Profile*: To view the platform profile page of the user.
3. *Post Project:* For hosting a new project on the platform.
4. *Account Settings:* For profile updating purposes/ registration of new mail ID and/or contact no.
5. *LogOut:* For ending the current session.

5.1.5 User Feed [UF-1]

5.1.5.1 UF-1.1 User Feed allows the user to view all the projects currently existing on the platform. The projects are categorized as:

1. ***On-Going:*** The projects that are currently in progress i.e., either in the "Recruiting" state or in the "In-Progress" state.

2. ***Completed:*** The projects that have been completed [Project Status: "Completed"] and are now displayed for user review.

Project view of the projects that are currently under any of the above mentioned states fall under either of these requirements: PVW-1.3, CPVW-1.3 or OPVW-1.3.

5.1.5.2 UF-1.2 The projects that are currently "Recruiting"/"In-Progress" should display contents based on either of the following requirements: PVW-1 & any of its subsections or OPVW-1 & any of its subsections or CPVW-1 & any of its subsections, depending on the user type.

5.1.6 Completed Projects List View [LVW-1]

Completed projects are in general displayed in the ascending order of project creation (oldest project first). Other than that, projects can be listed and/or searched for in the following ways:

1. *Ascending & Descending Order of ratings.*
2. *Project ID*

Projects can also be filtered out in the following ways:

1. *Project Domain Tags. [E. g. Applied filters: Engineering, Computer Science, Electronics].*
2. *Project Title.*

5.1.7 Post Project [PSTPRJ-1]

User-Type: [Researcher]

AND

The user is the project-owner.

The projects details that needs to be mentioned by the *Researcher* are as follows:

1. *Project Title*
2. *Project Domain Tag(s): [Enter new domain tag or Select from existing tags].*
3. *Initial No. Of Candidates required.*
4. *Application Start & End Date.*

5. *Project-ID: [Auto-Filled, Auto-Generated].*

6. *Project Start & End Date: [Disable entry by the Researcher].*

All the steps for successful project creation are stated under [PSTPRJ-1.1]

5.1.7.1 PSTPRJ-1.1 For successful project creation, the following tasks need to be performed:

1. *Fill-Up project details.* [PSTPRJ-1]

2. *Uploading IRB approval Letter & their own digital signature to be used for attesting project related documents through out the project.*

3. *Additional documents proving the projects legitimacy* (Refer to Legitimacy of posted projects [section 3.3] of this PRD).

5.1.8 Owner Project View [OPVW-1]

User-Type: [Researcher]
AND
The user is the project-owner

Display the following project respective info. on the project page under Project-Status: [Recruiting/In-Progress/Completed]:

1. *Project Title & ID.*

2. *Project Domain Tag(s).*

3. *Rating of the Researcher.*

4. *No. of projects the Researcher has worked on.*

5. *Other project status dependent information i. e., OPVW-1.1, OPVW-1.2 or OPVW-1.3.*

6. *Option to view Researcher CV*

Note: The status of the project should be updated in the following way:

Recruiting → In-Progress → Completed

And, the project cannot go back to a previous state, once changed to a different state.

5.1.8.1 OPVW-1.1 The following details & options are to be viewed in the project view-page for *Project Status: [Recruiting]*:

1. *Total no. of applicants.*
2. *Applicant Full Name.*
3. *Applicant Organisation Name.*
4. *Applicant Professional Designation.*
5. *Rating of the applicant.*
6. *No. of projects applicant has worked on.*
7. *No. of applicant contributions (in each of the mentioned project-domain tag(s)).*
8. *Option to view Applicant CV.*
9. *Option to accept/reject applicant.*
10. *Applicant IRB Acceptance status: [Signed/Failed].* [If applicant is accepted; Else, N/A]
11. *No. of candidates remaining to be selected.*
12. *Application start & end date.*
13. *Option to update no. of required applicants.*
14. *Option to update project status to [In-Progress].*
15. *Option to extend application end date.*
16. *Option to delete project.*
17. *Option to End Project: Remove project from the database.*

IRB Status: Signed

Once an applicant has been accepted and has uploaded the digital signature and dates from IRB approval letter successfully, [OPVW-1.1] will present the option to go to the corresponding [RCOLLAB-1] page for which a [CCOLLAB-1] page has been appointed to the applicant (now a *Contributor*).

IRB Status: Failed

If an accepted applicant fails to upload their digital signature and mention the project start & end dates from the letter, the applicant will be removed from the applicants list and will no longer be able to apply (check [CPVW-1.1]). If the application end date has been encountered, the owner (*Researcher* who created the project) will be prompted either to extend the application end date or to update project-status:"In-Progress".

Application End Date Update

After the application end date has been reached, the project page will be the same as stated above in [OPVW-1.1] but no applicant will be able to apply i. e. corresponding applying functionality will not be visible in [PVW-1.1] page. Only, after the updating (in this case extending) the application end date can other users start applying again to the project [except the ones who have been rejected or failed to sign the IRB approval letter (upload digital signature & mention dates for the letter)].

5.1.8.2 OPVW-1.2 The following details & options are to be viewed in the project view-page for *Project Status: [In-Progress]*:

1. *Option to view & update IRB approval letter for extending the date of the project.* [IRBUP-1]
2. *List of the contributors of the project & their respective [RCOLLAB-1] pages.*
3. *The updated rating of each contributor.*
4. *Project Start & End Dates.*
5. *Option to view CV of each Contributor.*
6. *Option to End-Project.* [Note: Only applicable if collaboration with all the *Contributors* have been ended. The project will automatically end after the project end date.]

5.1.8.3 OPVW-1.3 The following details & options are to be viewed in the project view-page for *Project Status: [Completed]*:

1. *List of the contributors of the project & their respective [RCOLLAB-1] pages for only viewing purposes.*
2. *Researcher published paper.*
3. *Additional Comments/links [If any by the Researcher].*
4. *The full-name, organization, professional designation & updated platform rating of each contributor & the researcher after project completion.*
5. *Project Rating.*
6. *Updated No. of Paper Reviews: Based on the no. of ratings received*
7. *Project Start & End Dates.*

5.1.9 Contributor Project View[CPVW-1]

User-Type: [Contributor]

AND

The user is one of the project-contributors

Display the following project respective info. under the Project-Status: [Recruiting/In-Progress/Completed]:

1. *Project Title & ID.*
2. *Project Domain Tag(s).*
3. *Project Status: [Recruiting/In-Progress/Completed].*
4. *Rating of the Researcher.*
5. *No. of projects the Researcher has worked on.*
6. *Option to view Researcher CV.*

5.1.9.1 CPVW-1.1 Display the following in the project page only when [Project Status: Recruiting]:

1. *No. of applicants.*
2. *Option to apply.*
3. *Application Status: [Accepted].* [If applied]
4. *Application start & end date.*
5. *IRB Approval Letter [If Application Status: Accepted].* [Go to CPVW-1.2]

<u>*IRB Review*</u>
After being able to view the IRB Letter and Other legitimacy documents [if present], the user should digitally sign the document and mention the Project Start & Date in the prompt so as to confirm agreement to the legitimacy of the project being hosted. The user will then be allocated a CCOLLAB-1 page for the concerned project and will be able to access it through [CPVW-1.2].
<u>*On failing to do the above task*</u>
If the user denies/fails to do the needful before the end of the Application End Date (mentioned under CPVW-1.1) the state of the project will return to the [Project-Status: Recruiting] as PVW-1.1 but the user will no longer be able to re-apply to the project. Display a message for the same.

5.1.9.2 CPVW-1.2 The Collaborator project view page can now display the link/option to be redirected to the [CCOLLAB-1] page assigned to the collaborator.

5.1.9.3 CPVW-1.3 Display an option to go to the CCOLLAB-1 page for the concerned project. Additionally, mention the details under PVW-1, Project-Status: [In-Progress], Researcher CV, Option to access [IRBUP-1] & Project Start & End Dates.

5.1.9.4 CPVW-1.4 Display the following only when [Project Status: Completed]: Everything under OPVW-1.3 excluding the access to [RCOLLAB-1] pages for each contributor as mentioned under item 1. Also, display the contributor specific [CCOLLAB-1] page for only viewing purposes & provide an option to Rate Paper: On a scale of 1 to 10 or any other appropriate method.

5.1.10 Project View [PVW-1]

User-Type: [Contributor/Researcher]

AND

The user is neither the project-owner or project-contributor with respect to the concerned project.

Displaying the following details is a must:

1. *Project Title.*
2. *Project Domain Name.*
3. *Project-ID.*
4. *Rating of the Researcher.*
5. *No. of projects the Researcher has worked on.*
6. *Option to view Researcher CV.*

5.1.10.1 PVW-1.1 The user satisfying the conditions above are displayed the following content when the project being viewed is in the "Recruiting" status:

1. *No. of applicants for the project*
2. *Application start & end date.*
3. *Option for the viewer to apply*
4. *Application status:* ***Pending*** *for review or* ***Application Rejected*** [if applied].
5. *Retract Application* [if applied].

5.1.10.2 PVW-1.2 The following content is displayed when the projected is "In-Progress":

1. *Project Start & End Date*
2. *Project Status:* In-Progress.

5.1.10.3 PVW-1.3 The following content is displayed when the projected is "Completed": Contributors Name, Organization, Professional Designation & Updated Rating . Also, an option to rate the project on a scale of 1 to 10/any other way that seems appropriate.

5.1.11 Researcher-Collaboration Page [RCOLLAB-1]

User-Type: [Researcher]

AND

The user is the project-owner

This page is separate for each *Contributor* of a project i. e., each *Contributor* is assigned an RCOLLAB-1 page after agreeing to the legitimacy documents of the project posted by the *Researcher*. Display the following details on the page:

1. *Project Title & ID.*
2. *Contributor Full-Name, Organization, Professional Designation.*
3. *Task list with Status (% Completion of each Task).*
4. *Collaboration Menu [RCOLLAB-1.1].*
5. *Option for [CWARN-1].*
6. *Option to End Collaboration with the Contributor [FIPR-1].* Note: [Only applicable after all the tasks listed under RCOLLAB-1 have been marked completed or project-end date has been reached and the *Contributor* has not verified updated documents (Refer to item 3 in the list under[RAS-1])]

Note: *Until the Researcher & Contributor have digitally signed the RCA between them, only items 1, 2 & 4 will be visible and accessible by the concerned Researcher. Refer to [RAS-1] item 1.*

Refer to [RAS-1], [FIPR-1] & [IRBUP-1] for legitimacy/agreement documents related activities.

5.1.11.1 RCOLLAB-1.1 The RCOLLAB-1 page menu should list the following functionalities:

1. *Tasks*: To view the overall list of tasks assigned to the concerned Contributor along with %-completion of each them.
2. *Post Tasks*: To assign a new task to the concerned Contributor. [RCOLLAB-1.2]
3. *Chat*: Option to access [CHAT-1].
4. *Contributor Agreement(s)*: [**View/Update**] Project IRB Approval Letter & other legitimacy documents [if any], RCA & Foreground IPR [RAS-1]

Note: *Until the Researcher & Contributor have digitally signed the RCA between them, only items 3 & 4 will be visible and accessible by the concerned Researcher.*

5.1.11.2 RCOLLAB-1.2 The *Researcher* can post a task with the following details:

1. *Task Name/Title.*
2. *Expected End-Date.*
3. *Priority: [High/Moderate/Late].*
4. *Remarks [if any].*
5. *Option to delete task.* [Provided no submissions have been made by the concerned contributor under any sub-task.]

Each Task [RCOLLAB-1.2] i. e., can have multiple Sub-Tasks [RCOLLAB-1.3]. The %-completion of each task will be calculated based on no. of sub-task completions. *Note: Once a task is posted under any RCOLLAB-1, the Project-Status automatically becomes [InProgress].*

5.1.11.3 RCOLLAB-1.3 A sub-task under a task needs to be posted with the following details and functionalities which is to be filled and can be accessed by the researcher at the time of sub-task creation and evaluation:

1. *Sub-Task Name/Title.*
2. *Due Date.*
3. *Priority: [High/Moderate/Late].*
4. *Option to Attach Files and classify the required ones as confidential.* [Refer to RCOLLAB-1.4]
5. *Option to View/Download contributor submitted files.* Note: [The files digitally signed and submitted by the contributor are the only files that will serve as sub-task submissions. All other files act as references or extras for the submitted files]
6. *Option to delete task.* [Provided no submissions have been made by the concerned contributor under the sub-task.]

5.1.11.4 RCOLLAB-1.4 While uploading/attaching files to a sub-task [RCOLLAB-1.3], the *Researcher* is to be given an option to classify each file as confidential by digitally signing each of them if required. The same should reflect at the *Contributor's* end.

5.1.11.5 RCOLLAB-1.5 After completion of each sub-task & task by the contributor, the researcher is required to digitally sign the same for confirming sub-task or task completion. The option to do so is to be reflected under the respective sub-tasks and tasks view [TMOD-1].

5.1.12 Task/Sub-Task Modification [TMOD-1]

User-Type: [Researcher]
AND
The user is the project-owner

The *Researcher* can modify an exiting task or sub-task in the following ways:

1. *Update/Edit: [Task/Sub-Task Name/Title, Expected End-Date/Due Date, Priority Level, Remarks, Task/Sub-Task Status(digital signature required for completion)].*

2. *Attach/Remove Files from a sub-task uploaded by the Researcher.*

3. *Delete a [task/sub-task.]* [Note: A task/sub-task marked as completed cannot be deleted by the *Researcher*

5.1.13 Researcher Agreement Signing [RAS-1]

User-Type: [Researcher]

AND

The user is the project-owner

The functionalities of this page are as follows:

1. *Initial Agreement Signing*: The *Researcher* needs to upload the RCA with respect to the concerned Contributor, mentioning the terms & conditions stated in [NFAGR-1], which needs to be signed by the concerned contributor to proceed with collaboration activities. *If signed by the contributor*: the *Researcher* will be able to access the remaining disabled options mentioned under [RCOLLAB-1] & [RCOLLAB-1.1] after project-status changes to "In-Progress". *Else If, the Contributor fails to sign within projet start date*: the *Researcher* will be returned to OPVW-1.1 [Project-Status: Recruiting] but [RCOLLAB-1] page for the Contributor will no longer be visible or accessible if no collaboration has taken place. The *Researcher* will then be prompted to either update application end date or update project-status to "In-Progress". All other functionalities for [OPVW-1.1] will be disabled till the user chooses to take either one of the mentioned actions.

2. *Researcher fails to upload document within project start date*: The [RCOLLAB-1] page will dissolve i. e. the *Researcher* will lose the *Contributor* for the project and the user will be redirected to the [OPVW-1.1] page ([Project-Status: Recruiting]). Hence, corresponding warnings should be displayed under [OPVW-1.1] for the same. The same choice of actions will be prompted as stated under item 1 of this list, for the case "*Else If, the Contributor fails to sign within projet start date*".

3. *Agreement(s) [View/Update]*: The *Researcher* will be able to view the signed RCA, IRB Apporval letter & Foreground IPR in this section and if the *Researcher* wants to update any of the above documents the procedure will be the same as stated under item no. 1. *Only in this case if the contributor has contributed i. e. finished a task/sub-task*, [RCOLLAB-1] page will not be modifiable by the *Researcher* until the *Contributor* has signed the updated document(s).
   If the Contributor does not respond to the verification before 45 days remaining of the project-end date: The *Researcher* can end collaboration with the *Contributor* and a corresponding [CWARN-1] items 2 & 3 will be applicable to the *Contributor*.

5.1.14 Contributor Warning [CWARN-1]

User-Type: [Researcher]
AND
The user is the project-owner

The *Researcher* wants to warn the *Contributor* about IRB, RCA and/or IPR violation(s). The effect of the warning(s) will be as follows:

1. After each warning, the rating of the *Contributor* will decrease by 50% of the current rating and the *Contributor* will be asked for reviewing the IRB, RCA and/or IPR and digitally sign them.

2. After 2 such warnings the *Contributor* will be blocked from editing anything under CCOLLAB-1 with respect to the project and will only be able to view the project related tasks, documents and agreements. Similarly, *Researcher* will be able to access RCOLLAB-1 with respect to the *Contributor* without any permission for modification of tasks, documents/agreements under the respective RCOLLAB-1.

3. *SPVW-1* will apply.

5.1.15 Contributor-Collaboration Page[CCOLLAB-1]

User-Type: [Contributor]
AND
The user is a project-contributor

The following details and functionalities are to be mentioned in this page:

1. *Project Title & ID.*
2. *Researcher Full-Name, Organization, Professional Designation.*
3. *Tasks List with status and option to view sub-tasks within a task.*
4. *Collaboration Menu.* [CCOLLAB-1.1]
5. *Option for [RWARN-1].*
6. *Option to exit project.* [Refer to XPRJ-1]

Note: *Until the Researcher & Contributor have digitally signed the RCA between them, only items 1, 2, 4 & 6 will be visible and accessible by the concerned Researcher. Refer to [CAS-1] item 1.*
On the end date of the project, if the *Contributor* has not digitally signed the FIPR document, digitally signed and uploaded by the *Researcher*, the user will then get an automatic [CWARN-1]. If the user fails to do the same by the end of that day, they will receive another [CWARN-1] and the project will end automatically.

5.1.15.1 CCOLLAB-1.1 The CCOLLAB-1 page menu should list the following functionalities:

1. *Tasks: To view the overall list of tasks assigned to the concerned Contributor along with %-completion of each them. Option to access each sub-task under the mentioned tasks.* [CCOLLAB-1.2]
2. *Chat: Option to access [CHAT-1].*
3. *Researcher Agreement(s)*: **[View/Verify]** Project IRB Approval Letter & other legitimacy documents [if any], RCA & Foreground IPR [CAS-1]

Note: *Until the Researcher & Contributor have digitally signed the RCA between them, only items 2 & 3 will be visible and accessible by the concerned Researcher. Refer to [CAS-1] item 1.*

5.1.15.2 CCOLLAB-1.2 Each task is assigned a [CCOLLAB-1.2]. The *Contributor* can view the following details about a task on this page:

1. *Task Name/Title.*
2. *Expected End-Date.*
3. *Priority: [High/Moderate/Low].*
4. *Remarks [if any].*
5. *The list of sub-tasks and an option to access each of them.* [Refer to CCOLLAB-1.3]

5.1.15.3 CCOLLAB-1.3 The *Contributor* can access each sub-task. Following functionalities are displayed in the page:

1. *View/Download the files attached to the sub-task.*
2. *Submit/Remove a file under the sub-task.* Note: [A signed & submitted file once signed by the researcher cannot be removed.]

Note*: A sub-task/task once marked completed by the researcher cannot be modified i. e., files cannot be removed under the corresponding sub-task/task. Also, the Contributor cannot upload/remove a file under a sub-task after the due-date.*

5.1.16 Contributor Agreement Signing [CAS-1]

User-Type: [Contributor]

AND

The user is a project-contributor

The functionalities of this page are as follows:

1. *Initial Agreement Signing*: The *Contributor* needs to digitally sign the RCA uploaded by the *Researcher* to proceed with project-collaboration. *If successfully signed by the contributor*: The *Contributor* will be able to access the remaining disabled functionalities mentioned under [CCOLLAB-1.1] after project-status is updated to "In-Progress".
   Else If, the contributor fails to sign within project start date: the *Contributor* will be returned to PVW-1.1 [Project-Status: In-Progress] but the user won't be able to re-apply to the project.

2. *Researcher fails to upload document within project start date*: The [CCOLLAB-1] page will dissolve and the *Contributor* will be redirected to the [PVW-1.1] page ([Project-Status: Recruiting]).

3. *Agreement(s) [View/Verify]*: The *Contributor* will be able to view the signed RCA, IRB Approval letter & Foreground IPR in this section and if the *Researcher* has updated any of the above documents, the *Contributor* should digitally sign the updated documents to enable the other functionalities of [CCOLLAB-1] which otherwise will be disabled until the documents are verified and signed.
   If the Contributor does not respond to the verification before 45 days remaining of the project-end date: The *Researcher* can end collaboration with the *Contributor* and a corresponding [CWARN-1] will be issued to the *Contributor*.

5.1.17 Researcher/Contributor Chat [CHAT-1]

User-Type: [Researcher/Contributor]
AND
The user is the project-owner or a project-contributor

The *Researcher* & each *Contributor* should be able have a simple asynchronous chat conversation with a contributor and keep track of past chat conversations through their respective RCOLLAB-1 & COLLAB-1 pages. The functionality to share documents and/or other forms of media may also be added (depending on resource availability of the project).

5.1.18 Researcher Warning [RWARN-1]

User-Type: [Contributor]
AND
The user is one of the project-contributors

The *Contributor* wants to warn the *Researcher* about IRB, RCA and/or IPR violation(s). The effect of the warning(s) will be as follows:

1. After each warning, the rating of the *Researcher* will decrease by 50% of the current rating and the *Researcher* will be asked for reviewing the IRB, RCA and/or IPR and digitally sign them.

2. After 2 such warnings the *Project-Status* will changed to *Agreement Violation [Stopped]*. The *Contributor*(s) will be blocked from editing anything under CCOLLAB-1 with respect to the project and will only be able to view the project related tasks, documents and agreements. Similarly, *Researcher* will be able to access RCOLLAB-1 with respect to the *Contributor* without any permission for modification of tasks, documents/agreements under the respective RCOLLAB-1.

3. *SPVW-1* will apply.

5.1.19 IRB Update [IRBUP-1]

If the user is of type *Researcher*
The Researcher will be able to upload the new IRB Letter. A pop-up message in the respective RCOLLAB-1 pages will be sent to each Contributor notfying them about the same. The Contributors would have to enter the modified dates as would be mentioned in the IRB letter and their digital signature. Only after a the *Contributor* has signed the new IRB Approval Letter would the *Researcher* be able to modify the respective [RCOLLAB-1] page.
If researcher fails to provide an updated IRB Approval letter before 45 days remaining for the project to end: The *Researcher* will be subjected to [RWARN-1], items 2 & 3 in the list will apply.

If the user is of type *Contributor*
The user needs to sign the pop-up message for verifying the modification of the IRB Approval Letter only after which the user will be able to modify the respective [CCOLLAB-1] page. If the contributor hasn't signed the IRB Approval Letter actions to be taken by the Researcher stated in the list under [CAS-1] (item 3, under *"if the Contributor does not respond to the verification before4 5 days remaining of the project-end date"*) will apply.
***Note**:*
The digital signature uploaded/used for verifying the IRB Approval Letter will be used through out the project representing the corresponding contributor's authorization.

5.1.20 Foreground-IPR [FIPR-1]

User-Type: [Researcher]
The researcher must upload & digitally sign the Foreground IPR which will then be available under RAS-1 & CAS-1. The document being signed by the corresponding contributor will mark the end of collaboration with the same. The researcher will be re-directed to the OPVW-1.2 page [Project-Status: In-Progress] and only be able to view the corresponding RCOLLAB-1 page.
If researcher fails to provide a Foreground IPR document within project end-date:
The *Researcher* will be subjected to [RWARN-1], items 2 & 3 in the list will apply.

User-Type: [Contributor]
The contributor will be notified about the availability of the Foreground-IPR through CAS-1 and will be asked to sign the document to end the project. On signing the document successfully, the contributor will be re-directed to the CPVW-1.3 [Project-Status: In-Progress] and will be able to only view the corresponding CCOLLAB-1 page. If the contributor hasn't signed the FIPR within the project end date, actions to be taken by the *Researcher* stated in the list under [CAS-1] (item 3, under *"if the Contributor does not respond to the verification before 45 days remaining of the project-end date"*) will apply.

5.1.21 Exit Project [XPRJ-1]

User-Type: [Contributor & Researcher]

AND

The users are project-contributor [requesting] & project-owner [receiving]

The contributor want's to exit the project. The *Researcher* will receive a pop-up message/notification stating that the *Contributor* wants to exit the project, which is digitally signed by the *Contributor*. The *Researcher* will be given two options:

1. *Agree & Sign* [XPRJ-1.1 will apply].
2. *Deny.* [XPRJ-1.2 will apply]

5.1.21.1 XPRJ-1.1 Two scenarios are to be handled as follows:

1. *Contributor has contributed*: CPVW-1.3 will apply as the contributor will only get the option to exit the project before the end date of the project if and only if the project-status:"In-Progress" . The *Contributor* will also be asked to rate the *Researcher*.
2. *Contributor has not contributed*: PVW-1 and all it's sub-functionalities will apply to the contributor for the concerned project.
   In either case the contributor cannot reapply or contribute to the project. Appropriate messages can be shown for clarification purposes.

5.1.21.2 XPRJ-1.2 The contributor will not be removed from the project. However, the *Researcher* & *Contributor* will be asked to review the project agreements i. e. IBR & RCA through RCOLLAB-1.2 [Menu-Item 4] and CCOLLAB-1.1 [Menu-Item 4] respectively. *Both the concerned parties should sign the prompted review(s) to continue with RCOLLAB-1 & CCOLLAB-1 with respect to the concerned Contributor for the project.* Else, the state of the CCOLLAB-1 would return to displaying the options representing XPRJ-1.1 & XPRJ-1.2.

5.1.22 Stopped Project View [SPVW-1]

Display project details as mentioned under PVW-1 and Project-Status:*Agreement Violation [Stopped]*. The project will be listed under completed projects in [UF-1].

5.1.23 Update Profile [UPF-1]

User can update the following details or perform the tasks:

1. *Full Name.*
2. *Organization Name.*
3. *Professional Designation.*
4. *CV.*
5. *Register new contact number/Replace existing contact number.*
6. *Register new Mail-ID/Replace existing Mail-ID.*
7. *Update Password.*
8. *Update Image/Avatar.*

5.1.24 Account Deletion [ACDEL-1]

User Account Deletion. The following requirements should be met under this action:

5.1.24.1 ACDEL-1.1] User should be prompted an interactive warning pop-up message. Two choices to be displayed:

1. *Yes:* Move to requirement ACDEL-1.2
2. *No:* No Changes to be made to the account.

5.1.24.2 ACDEL-1.2] All user data to be removed from the database. After this stage the platform should contain no data on the user. User profile will no longer be displayed.

5.1.25 Log-Out [LGOUT-1]]

User Log-Out from current session.

5.1.25.1 LGOUT-1.1] User should be prompted an iteractive warning pop-up message. Two choices to be displayed:

1. *Yes:* Move to requirement LGOUT-1.2
2. *No:* Current user session remains active.

5.1.25.2 LGOUT-1.2] User is logged out from the current session and re-directed to the platform log-in page.

5.1.26 Non-Functional Requirement: Security [NFS-1]]

The Database will store all the passwords and other critical information. The system should provide high security for its database & blockchain logs are to be maintained under a secure network.

5.1.27 Non-Functional Requirement: Capacity [NFCAP-1]]

The system should fall well within the limit of Atlas.

5.1.28 Non-Functional Requirement: Compatibility [NFCOMP-1]]

Should function properly on any consumer device with access to the internet and installed web browsing applications.

5.1.29 Non-Functional Requirement: Reliability & Availability [NFRA-1]]

When deployed, the system should be up and running 24x7. Providing its users access whenever and wherever required, unless in the point of a technical failure.

5.1.30 Non-Functional Requirement: Maintainability & Manageability [NFMM-1]]

In the event a technical failure occurs, the system should be back up within 1-2 days.

5.1.31 Non-Functional Requirement: Usability [NFU-1]]

The platform should be simple to use, and anyone familiar with modern websites should be able to operate with ease.

5.1.32 Non-Functional Requirement: Scalability [NFSC-1]]

The platforms used for the server, the blockchain network and the database all support various levels of scalability as per the needs of the businesses.

5.1.33 Non-Functional Requirement: Platform Dependent Agreement [NFPDAT-1]]

The platform promises to provide an online environment for confidential collaborative-research work. However, the platform does not promise or state any responsibility of drafting the documents representing the terms & conditions of research collaboration between a *Researcher* & a *Contributor*. The functionalities [RAS-1] & [CAS-1] are to be used to convey/display the platform dependent terms & conditions for conducting collaborative-research work on the platform both to the *Researcher* and the concerned *Contributor* respectively. This would act as a suggestion to both the concerned *Researcher* & *Contributor* for mentioning & checking the below mentioned T&C in their contracts/agreements i., e, the IRB Approval Letter, the RCA & Foreground-IPR to avoid any complications or hurdles while working on a project hosted on the platform.

[**Note**: The T&Cs stated below are among the essential T&Cs to be included in the corresponding RCA & Foreground-IPR documents for a project but there are several other points that can/must be stated for the purpose mentioned above under the proper consultancy of an attorney. The points below are just examples of what is to be displayed in the respective RAS-1 & CAS-1 pages]
RCA
Terms & Conditions to be included by the Researcher:

1. *Document Verification*: The agreement documents authorized & uploaded by the *Researcher*, visible to the *Contributor* under [CAS-1] should be verified by the user's digital signature for continuing to collaborate under the mentioned project-title in the platform. Failing to verifying the documents will result to consequences as stated under the respective platform functionalities. Once verified by the user's digital signature, the documents under [RAS-1] which are the same documents represented under [CAS-1] should serve as proof for

the *Contributor's* conscious authorization to abide by the T&Cs mentioned in the documents under the stated section i. e., [CAS-1].

2. *Confidential Documents*: The documents attached to a sub-task [RCOLLAB 1.3 & 1.4] by the *Researcher* with the user's digital signature & reflected under [CCOLLAB-1.3] for the concerned contributor should be considered as confidential and should be dealt with utmost care by the *Contributor* failing to do which the user will be exposed to the consequences of [CWARN-1] when used by the corresponding *Researcher*.

3. *Breach Of Contract*: If the *Contributor* violates any of the T&Cs mentioned in any of the documents under [CAS-1] will run into the risk of getting a warning by the *Researcher* [CWARN-1].

Terms & Conditions to be included by a Contributor:

1. *Document Verification*: The agreement documents authorized & uploaded by the *Researcher*, visible to the *Contributor* under [CAS-1] should be verified by the user's digital signature for continuing to collaborate under the mentioned project-title in the platform. Failing to verifying the documents will result to consequences as stated under the respective platform functionalities. Once verified by the user's digital signature, the documents under [CAS-1] which are the same documents represented under [RAS-1] should serve as proof for the *Research's* conscious authorization to abide by the T&Cs mentioned in the documents under the stated section i. e., [RAS-1].

2. *Confidential Documents*: The documents signed & submitted by the *Contributor* is to be considered for evaluation of the Foreground-IPR. The *Researcher* should treat these submissions as confidential failing to do which, the user will be exposed to the consequences of [RWARN-1] when used by the concerned *Contributor*.

3. *Breach Of Contract*: If the *Researcher* violates any of the T&Cs mentioned in any of the documents under [RAS-1] will run into the risk of getting a warning by the *Contributor* [RWARN-1].

Foreground-IPR

Terms & Conditions to be included by the Researcher:

1. *Considered Submissions*: The files digitally signed and submitted under a sub-task are the only files which are to be taken into account while calculating the Foreground-IPR for the *Contributor*. Any other attachments submitted without the contributor's digital signature will be considered as extras.

2. *Breach of Trust*: The *Contributor* failing to sign a Foreground-IPR even after providing a justified accountability for the user's contribution(s) to the project will be subjected to [CWARN-1].

3. *End Of Collaboration (EoC)*: The *Contributor* wanting to exit a project should do so considering the consequences stated under [XPRJ-1].

Terms & Conditions to be included by a Contributor:

1. *Considered Submissions*: The files digitally signed and submitted under a sub-task by the *Contributor* are the files which have to be taken into account while calculating the Foreground-IPR for the user. Failing to consider these documents for evaluating the Foreground-IPR will potentially result to the consequences of [RWARN-1] when imposed by the corresponding *Contributor*. Any other attachments submitted without the contributor's digital signature will be considered as extras.

2. *Breach of Trust*: The *Researcher* failing to provide a properly evaluated & signed document to the *Contributor* for the corresponding signed submissions with respect to the project will be subjected to [CWARN-1].

3. *End Of Collaboration (EoC)*: The *Contributor* wanting to exit a project should be allowed to do so by *"Agree & Sign"* under [XPRJ-1] by the *Researcher* considering the fact that the *Contributor* has completed all sub-tasks and has successfully signed the Foreground-IPR. Failing to allow the *Contributor* to exit the project will result to a warning by the corresponding contributor [RWARN-1].

6 Design

6.1 Technical Environment

The technical dependencies of the platform are as follows:

- *Hardware*: 2GB RAM (Minimum), 1GHz processor.
- *Operating System*: Android 6 & above, iOS 11 & above, Windows 8 & above.
- *Devices*: Any device with browser support and through any browser.

6.2 Hierarchy of Modules

The diagrams below state an overview of the platform as a whole and modes of interactions between the user and the platform through standard diagrams such as Data-Flow Diagram (DFD) & Entity-Relationship (ER) Diagram.

1. **Front-end**
   - *App*
     - Represents the entire front-end application.
     - Responsible for rendering the user interface and managing user interactions.
     - Divided into specific pages: Login, Sign-Up, Home, and Profile.
   - *Login Page*
     - Allows users to input their credentials and log in.
     - After successful login, the user is redirected to the Home Page.
   - *Home Page Options*
     (a) Side Bar
       - Allows mentors to navigate to specific functionalities.
       - Sub-options: Post Project, Profile, and Account Settings.

     (b) *User Feed*
       - Displays a feed of projects.
       - Leads to the Project List.

- *Project List*
  - Divided into Ongoing Projects and Completed Projects.
  - Ongoing Projects
    - Allows users to navigate to View Project and Apply to Project.
- Profile Page
  - Accessed after login or signup.
  - Displays user-specific information.
  - Contains options to navigate to various user-related functionalities.

2. **Back-end**
   - *APIs*
     - Receives requests from the frontend.
     - Directs requests to specific routes: User Routes or Project Routes.
   - *Middleware*
     - All requests pass through here before reaching controllers.
     - Validates and processes requests before passing them to controllers.
   - *User Routes & Project routes*
     - Direct requests to specific controllers.
     - User Controller:
       - Checks with the User Model and processes user-related requests.
   - *Project Controller*
     - Checks with the Project Model.
     - Forwards necessary requests to the Blockchain Contract API.
3. **Blockchain**
   - *Blockchain Contract API*
     - Part of the back-end but distinctly associated with the blockchain layer.
     - Serves as a bridge between the back-end and the blockchain.

  - Receives requests from the Project Controller, abstracting the complexities of interacting with the blockchain.
  - Translates requests into interactions with smart contracts on the blockchain.
  - Responsible for ensuring secure and authorized communication with the blockchain.
- *Modifiers*
  - Within the Blockchain layer.
  - Performs checks to verify if the entity initiating the action has the required role and authorization.
  - Ensures that only authorized entities can execute specific functions on smart contracts.
- *Methods*
  - Executes requests received from the Project Controller.
  - Directly interacts with smart contracts on the blockchain.
  - Responsible for creating, updating, or querying data on the blockchain.
  - Ensures the integrity and security of data transactions.
- *Events*
  - Triggered after successful execution of methods.
  - Creates events to log and notify relevant parties about changes in the blockchain state.
  - Enhances transparency and traceability on the blockchain.

4. **Data Store**
   - *Stores data resulting from successful blockchain interactions.*
   - *Ensures persistence and availability of data across the application.*

Figure 8: Modules Hierarchy in CollabChain

This refined explanation highlights the user functionality within the front-end, especially after the login process, and demonstrates how it integrates with the back-end and block-chain layers for a cohesive and interactive user experience. Refer to figure 8 for the diagrammatic representation.
The provided description outlines a Data Flow Diagram (DFD) for the Collabchain platform, illustrating the flow of data between the front-end, back-end, and blockchain. Let's break down each component and its associated functions:
Front-end to Back-end Data Flow:

1. **Create User Account Function:**
   - User provides login ID and password.
   - Receives a login key.
   - Processed by the "Create User Account" function in the central backend.
2. **Login User and Fetch Profile Function:**
   - User signs up with details.
   - Receives a login key.
   - Processed by the "Login User and Fetch Profile" function in the central backend.
3. **Edit Account Info Function:**
   - User updates account information.
   - Processed by the "Edit Account Info" function in the central backend.
4. **Reset Account Password Function:**
   - User resets the password by sending verification key and new password.
   - Processed by the "Reset Account Password" function in the central backend.
5. **Verify User Mail Function:**
   - User verifies email ID by sending a token.
   - Processed by the "Verify User Mail" function in the central backend.

6. **Logout Function:**
    - User logs out.
    - Processed by the "Logout" function in the central backend.
7. **Retrieve Info of All Projects Function:**
    - User requests to retrieve public info of all projects.
    - Receives public info of projects.
    - Processed by the "Retrieve Info of All Projects" function in the central backend.
8. **Create Project Function:**
    - Mentor sends project details and requests to create a new project.
    - Receives a confirmation.
    - Processed by the "Create Project" function in the central backend.
9. **Update Project Details Function:**
    - Mentor sends updated project details and requests to update a project.
    - Receives a confirmation.
    - Processed by the "Update Project Details" function in the central backend.
10. **Fetch Info of a Project Function:**
    - User requests to fetch info of a project.
    - Receives private info as per the user's role.
    - Processed by the "Fetch Info of a Project" function in the central backend.
11. **Apply to a Project Method:**
    - Mentee requests to apply to a project.
    - Receives confirmation.

- Executed by the "Apply to a Project" method in the central backend.

12. **Withdraw Application Method:**
    - Mentee requests to withdraw from a project.
    - Receives confirmation.
    - Executed by the "Withdraw Application" method in the central backend.
13. **Fetch Personal Task Info Method:**
    - Mentee requests to fetch personal task info.
    - Receives task-related information.
    - Executed by the "Fetch Personal Task Info" method in the central backend.
14. **Submit Task Work Method:**
    - Mentee uploads task-related documents.
    - Receives a verification token.
    - Executed by the "Submit Task Work" method in the central backend.
15. **Mark Task Complete Method:**
    - Mentor uploads the verification key after taking it from the mentee.
    - Receives task-related documents.
    - Processed by the "Mark Task Complete" method in the central backend.

Back-end to Blockchain Data Flow:

1. **Create New Task Method:**
    - Update project details based on the request of mentors.
    - Calls "Create New Task" method of the blockchain.

- Sends: Task ID, Receives: Confirmation.

2. **Assign Contributor Method:**
   - Update project details based on the request of mentors.
   - Calls "Assign Contributor" method of the blockchain.
   - Sends: Task ID, User Eth Address, Receives: Confirmation.
3. **Remove Contributor Method:**
   - Update project details based on the request of mentors.
   - Calls "Remove Contributor" method of the blockchain.
   - Sends: Task ID, User Eth Address, Receives: Confirmation.
4. **Check Task Status Method:**
   - Fetch info of project.
   - Calls "Check Task Status" method of the blockchain smart contract.
   - Sends: Request, Receives: Task Status.
5. **Create Submission Method:**
   - Submit Task Work.
   - Calls "Create Submission" method of the blockchain.
   - Sends: Task ID, Document ID, Document Hash, Receives: Confirmation.
6. **Mark Task Completion Method:**
   - Mark Task Completed.
   - Calls "Mark Task Completion" method of the blockchain smart contract.
   - Sends: Task ID, Verification Token, Receives: Confirmation.

This detailed Data Flow Diagram (Figure 9) illustrates the flow of data between the frontend, back-end, and blockchain components, highlighting the specific functions and methods involved in each step.

Figure 9: Platform Data Flow Diagram (DFD)

The ER Diagram of the platform (as shown in Figure 10) can be explained as follows:

Relationships:

1. One User Mentors Many Projects (One-to-Many):
   - **Description:**
     - A user, represented by `USER.UserId`, can mentor multiple projects.
     - Each project can have only one mentor, referenced by the foreign key `PROJECT.Mentor` pointing to `USER.UserId`.
2. One User Assigned to One Task (One-to-One):
   - **Description:**
     - A user, represented by `USER.UserId`, can be assigned to only one task.
     - Ensures that a user is associated with a unique task, helping in task management.
3. One Project has Many Tasks (One-to-Many):
   - **Description:**
     - A project, represented by `PROJECT.ProjectId`, can have multiple tasks.
     - Each task is associated with only one project, referenced by the foreign key `TASK.ProjectId` pointing to `PROJECT.ProjectId`.
4. One Project has Many Contributors (One-to-Many):
   - **Description:**
     - A project, represented by `PROJECT.ProjectId`, can have multiple contributors.
     - Each contributor is a user, referenced by the foreign key `USER.UserId` (Contributors), contributing to the project.

Foreign Key References:

- Mentor in PROJECT entity:
  - **Description:**
    - The `PROJECT.Mentor` field is a foreign key referencing the `USER.UserId` field.
    - Establishes a connection between a project and its mentor, ensuring referential integrity.
- Contributors in PROJECT entity:
  - **Description:**
    - The `USER.UserId` (Contributors) field is a foreign key referencing the `PROJECT.ProjectId` field.
    - Associates users as contributors to a specific project.

Figure 10: Platform Entity-Relationship (ER) Diagram

6.3 Detailed Design

The remaining columns of the requirement matrix involving the designs related to each module and test cases (if any) are included below along with detailed designs of modules that are implemented as either UI/UX or Coded prototypes.

Rqmt Sec.	Design Module	Design Section	Test Case No.	Technical Platform of Implementation	Prototype Prepared?
5.1.1	UREG	6.3.1	-	Figma	Yes
5.1.1.1	UREG	6.3.1	-	Javascript	Yes
5.1.2	-	-	-	Javascript	Yes
5.1.2.1	ULGN	6.3.2	T-ULGN-1.1	Figma	Yes
5.1.2.2	ULGN	6.3.2	-	Javascript	Yes
5.1.3	PVPM	6.3.3	-	Figma	Yes
5.1.3.1	PVPM	6.3.3	T-PVPM-1.1	Figma, Javascript	Yes
5.1.3.2	PVPM	6.3.3	T-PVPM-1.2	Figma	Yes
5.1.4	PVPM	6.3.3	-	Figma	Yes
5.1.4.1	PVPM	6.3.3	-	Figma	Yes
5.1.4.2	PVPM	6.3.3	T-PVPM-2.2	Figma	Yes
5.1.5	UFA	6.3.4	-	Figma, Javascript	Yes
5.1.5.1	UFA	6.3.4	T-UFA-1.1	Figma, Javascript	Yes
5.1.5.2	UFA	6.3.4	T-UFA-1.2	Figma	Yes
5.1.7	PCM	6.3.5	T-PCM-1	Figma, Javascript & Solidity	Yes
5.1.7.1	PCM	6.3.5	T-PCM-1.1	Figma	Yes
5.1.8	PCM	6.3.5	T-PCM-2	Figma, Javascript	Yes
5.1.8.1	PCM	6.3.5	T-OPVW-2.1	Figma, Javascript	Yes
5.1.8.2	PCM	6.3.5	T-OPVW-2.2	Figma	Yes
5.1.8.3	PCM	6.3.5	T-OPVW-2.3	Figma	Yes
5.1.9	CPO	6.3.6	T-CPO-1	Figma, Javascript	Yes
5.1.9.1	CPO	6.3.6	T-CPO-1.1	Figma	Yes
5.1.9.2	CPO	6.3.6	-	Figma	Yes
5.1.9.3	CPO	6.3.6	-	Figma	Yes
5.1.9.4	CPO	6.3.6	-	Figma	Yes
5.1.10	-	-	-	Javascript	Yes
5.1.10.1	CPO	6.3.6	-	Figma, Javascript	Yes
5.1.11	PC	6.3.7	-	Javascript	Yes
5.1.11.1	PC	6.3.7	-		No
5.1.11.2	PC	6.3.7	-	Javascript	Yes
5.1.11.3	PC	6.3.7	-	Javascript, Solidity	Yes
5.1.11.4	PC	6.3.7	-	-	No
5.1.11.5	PC	6.3.7	-	-	No
5.1.12	PC	6.3.7	-	Javascript, Solidity	Yes
5.1.13	PC	6.3.7	-	-	No
5.1.18	PC	6.3.7	-	-	No
5.1.15	PC	6.3.7	-	-	No
5.1.15.1	PC	6.3.7	-	-	No
5.1.15.2	PC	6.3.7	-	-	No
5.1.15.3	PC	6.3.7	-	Javascript, Solidity	Yes
5.1.16	PC	6.3.7	-	-	No
5.1.17	PC	6.3.7	-	-	No
5.1.14	PC	6.3.7	-	-	No
5.1.19	PC	6.3.7	-	-	No
5.1.20	PC	6.3.7	-	-	No
5.1.21	PC	6.3.7	-	-	No
5.1.21.1	PC	6.3.7	-	-	No
5.1.21.2	PC	6.3.7	-	-	No
5.1.23	-	-	-	Javascript	Yes
5.1.25	-	-	-	Javascript	Yes

Table 4: *Requirement Matrix [Traceability Matrix (Rqmt Sec. & Columns D-H)]*

6.3.1 User Registration [UREG]

The section presents a detailed design of the UI the user is presented with while registering to the platform. This design covers the requirement section 5.1.1.1 and the same is implemented in Figma (refer to Appendix-A for the figma prototype link).

Figure 11: User Registration Page (UI)

6.3.2 User Login [ULGN]

The requirement 5.1.2.1 is unique for this platform in terms of choosing the user type. Other functionalities are the same as expected from standard web application

platforms such as providing the registered user ID and password for logging into the user account.

Figure 12: User Login Page (UI)

6.3.3 Profile View and Platform Menu [PVPM]

The requirements 5.1.3.1, 5.1.3.2 & 5.1.4.2 are met through this design. For requirement 5.1.4.1, there is a single change i. e., the unavailability of an option as mentioned under the corresponding section. Apart from that the functionalities are all same for this page for both the user types. Refer to the figma prototype (link in APPENDIX-A) and the test-plan for further demonstration of the same.

Figure 13: A *Researcher* user's Profile Page (UI)

6.3.4 User Feed View & Access-ability [UFA]

The requirements of sections 5.1.5 and it's subsections are met through the UI design as shown below. Clicking on any of the ongoing projects will redirect the user to the site of the project. Refer to the figma prototype (link in APPENDIX-A) and the test-plan for further demonstration of the same.

Figure 14: User Feed as seen from a *Researcher's* Profile

6.3.5 Project Creation & Maintainability [PCM]

This section covers modules [PSTPRJ-1] & [OPVW-1] along with all their subsections. The flow-chart below (15) shows how a *Researcher* creates project, recruits candidates, appoints & signs sub-task /task completion and reaches the end of collaboration with a *Contributor* of the project. It includes the modules concerning the involvement of the *Researcher* with a project, namely section 5.1.7 & it's subsections, section 5.1.8 & it's subsections along with supportive involvement of section 5.1.11 & it's subsections respectively.

The dashed-lines in the image represent how the data flows from and to the back-end and how a deployed smart contract updates sub-task data accordingly. The arrows represented in bold show how in general a *Researcher* should perform the tasks of project creation and contribution with a single *Contributor* which then generalizes the actions taken for collaborating with all other project *Contributors*.

Figure 15: Researcher Project Creation & Collaboration Flow

6.3.6 Contributor Project Overview [CPO]

The contributor project view [CPVW-1] (5.1.9) & all its subsections is explained in this section. A user of type *Contributor* after applying and being accepted as a project contributor can view and access this section of functionalities as mentioned under section 5.1.9 and it's subsections. Initially a user of type either *Contributor&Researcher* applies to a project through PVW-1.1. So, PVW-1.1 (5.1.10.1) can be thought of as an intermediate page between the time when a user applies for a project and gets accepted as shown in figure 16.

Figure 16: [PVW-1.1] Applied & Awaiting Response

Figure 17: [CPVW-1.3] Project Status:In-Progress as viewed by the contributor

Once accepted, the user can view the IRB Approval Letter posted by the project owner & *Researcher* during project creation. After reading, stating start & end date as mentioned in the letter & uploading the digital signature (all under [CPVW-1.1]) they can proceed to collaborate. The user will be allocated a corresponding [CCOLLAB-1] page that can be accessed by [CPVW-1.2]) clicking the "Collaborate" button as shown in the figure above 17.
As the status of the project changes from Recruiting to In-Progress to Completed, the same will be reflected under [CPVW-1.1 & 1.2], [CPVW-1.3] & [CPVW-1.4] respectively. Implementing [CPVW-1] in this way provides a project *Contributor* a structured way of keeping up with the corresponding project progress in the form of an overview of the project details including project status.

6.3.7 Platform Collaboration [PC]

The section covers 5.1.11 & its subsections, 5.1.15 and its subsections along with the use cases of sections 5.1.12, 5.1.18, 5.1.14, 5.1.13, 5.1.16 & 5.1.21, 5.1.17 & 5.1.20. The figure below (fig. 18) describes how a *Researcher* & two *Contributors* collaborate in a project. The functional requirements that help achieve the collaboration work are listed under each *Contributor* with specified coloured boxes, namely blue and green, along with the *Contributor* specific RCOLLAB-1 page, represented by the yellow box which is controlled by the *Researcher*.

The *Researcher* assigns two tasks to the *Contributor* named ***Allen Smith (blue)***. The first task contains three sub-tasks and the second task contains 2 sub-tasks assigned through the RCOLLAB-1.2 & RCOLLAB-1.3 pages respectively per task. The same is reflected on the *Contributor's* end through the pages CCOLLAB-1.2 & CCOLLAB-1.3 through which the user accesses/submits files from and to the task.TMOD-1 is used by the *Researcher* to modify each task/sub-task assigned to the respective *Contributors*

Apart from the above interaction the *Researcher* and *Contributor* interact through other functioanilites such as CWARN-1 & RWARN-1 to issue warnings on RCA violations, access, modify and/or verify Agreement Documents uploaded during the starting, middle or end of the collaboration with the *Contributor* through RAS-1 & CAS-1. Apart from that other basic functionalities such as chatting with the *Contributor* or *Researcher* through CHAT-1 or exiting the project on collaboration completion through XPRJ-1 are also show.

Things are almost the same between the other *Contributor* and *Researcher* except the fact that the second *Contributor* i. e. ***Steven Jones (green)*** is assigned two tasks with two sub-tasks under each task. Also, the two *Contributors* of the project cannot see who the other contributor is let alone view the tasks they have done, thus ensuring confidentiality within the team and protecting the *Researcher's* unique approach to solving a problem from being completely exposed before publication.

Therefore, this figure itself gives an overall explanation of how collaboration is done on the platform and how different functionalities, as mentioned in the *requirement*

matrix Table 3 & analysed under *section 5* are used to achieve the same.

Figure 18: Collaboration Overview

7 Test plan

The test cases only represents testing of prototype prepared in Figma, hence, should be considered as a simulation than a review/testing of a coded logic. It is to give an idea of how the platform is expected to perform when logically coded and implemented later in Phase-2 of the overall project-plan.

7.1 Figma Prototype Test Plan

Figma Prototypes are a simulations (more like presentations) than coded logic. The test cases represented below are arranged in the order in which the prototype demonstrates platform functionalities. Many user inputs, options, uploads have already been entered and displayed to skip to the simulation of the features that stand-out from day-to-day applications.

<table>
<tr><td>Test Case ID</td><td>T-ULGN-1.1</td><td>Test Case Description</td><td colspan="2">Simulating functionality of LGN-1.1 with User-Type: Researcher</td></tr>
<tr><td colspan="3">Test Case Status(Pass/Fail)</td><td colspan="2">Pass</td></tr>
<tr><td>Sl. No.</td><td colspan="4">Prerequisites</td></tr>
<tr><td>N/A</td><td colspan="4">N/A</td></tr>
<tr><td>Sl. No.</td><td colspan="4">Test Data</td></tr>
<tr><td>1</td><td colspan="4">User-Type: Researcher</td></tr>
<tr><td>2</td><td colspan="4">Email-ID: (already entered) john.doe@gmail.com</td></tr>
<tr><td>3</td><td colspan="4">Password (already entered) but hidden</td></tr>
<tr><td colspan="2">Test Scenario</td><td colspan="3">The user is able to login as a researcher and is directed to the corresponding profile page.</td></tr>
<tr><td>Step #</td><td>Details</td><td>Expected Results</td><td>Actual Results</td><td>Status</td></tr>
<tr><td>1</td><td>Login as Researcher</td><td>Directed to corresponding account profile page</td><td>As Expected</td><td>Pass</td></tr>
</table>

Table 5: Test-Case 1

Test Case ID	T-PVPM-1.1	**Test Case Description**	Displaying information on the profile page as mentioned under [PRF-1.1]	
Test Case Status(Pass/Fail)			Pass	
Sl. No.	**Prerequisites**			
1	Should be Logged In as user-type: Researcher			
Sl. No.	**Test Data**			
1	User-Type: Researcher			
2	Name: John Doe			
3	Organization: Michigan Institute of Technology			
4	Profession : Assistant Lecturer			
5	Email ID : john.doe@gmail.com			
6	Contact : +122 8573910239			
7	User Rating			
Test Scenario		User information as listed under test data should be visible in the profile page after logging in as a user of type Researcher.		
Step #	**Details**	**Expected Results**	**Actual Results**	**Status**
1	View User Details	User Details should be displayed in the profile page	As Expected	Pass

Table 6: Test-Case 2

Test Case ID	T-PVPM-1.2	**Test Case Description**	Simulating the functionality stated under [PRF-1.2] and viewing corresponding data (if any)	
Test Case Status(Pass/Fail)			Pass	
Sl. No.	**Prerequisites**			
1	Should be Logged In as user-type:Researcher			
Sl. No.	**Test Data**			
1	Tabs: Completed, In-Progress & Saved			
2	Completed Project Code: ABC-234			
3	Saved Project Code: YZI-707			
Test Scenario		User (user-type: Researcher) can view completed, in-progress & saved projects in the profile page [if any] .		
Step #	**Details**	**Expected Results**	**Actual Results**	**Status**
1	Completed Project(s) listed	Completed Project(s) list displayed	As Expected	Pass
2	Saved Project(s) listed	Saved Project(s) list displayed	As Expected	Pass

Table 7: Test-Case 3

Test Case ID	T-PVPM-2.2	**Test Case Description**	Listing Menu Items as stated under [PMNU-1.2]	
Test Case Status(Pass/Fail)			Pass	
Sl. No.	**Prerequisites**			
1	Should be Logged In as user-type:Researcher			
Sl. No.	**Test Data**			
1	Menu Items: 1.User Feed; 2. Profile; 3. Post Project; 4. Account Settings; 4. Log Out			
Test Scenario		User (user-type: Researcher) can view the corresponding menu items.		
Step #	**Details**	**Expected Results**	**Actual Results**	**Status**
1	Platform Menu View	Menu Items Displayed	Menu Items Displayed	Pass

Table 8: Test-Case 4

Test Case ID	T-UFA-1.1	**Test Case Description**	Simulating User Feed Viwe as listed under [UF-1.1]	
Test Case Status(Pass/Fail)			Pass	
Sl. No.	**Prerequisites**			
1	Should be Logged-In to the platform account			
Sl. No.	**Test Data**			
1	Two Tabs: Ongoing & Completed are displayed			
2	Projects under corresponding columns (if any)			
Test Scenario		User (user-type: Researcher) has logged-in to the account & can view user feed with the corresponding project categories & project listings (if any)		
Step #	**Details**	**Expected Results**	**Actual Results**	**Status**
1	User Feed View	Displayed Project categories: [Ongoing & Completed] along with a single project under Ongoing projects category	As Expected	Pass

Table 9: Test-Case 5

Test Case ID	T-PCM-1	**Test Case Description**	Simulating the procedure of project creation by filling details as mentioned under [PSTPRJ-1]	
Test Case Status(Pass/Fail)			Pass	
Sl. No.	**Prerequisites**			
1	Should be Logged In as user-type:Researcher			
Sl. No.	**Test Data**			
1	Project Title: Blockcain based Collaboration System (already entered)			
2	Initial No. of Candidates Required: 2			
3	Application Start & End Date: dd-mm-yy			
4	Project ID (Auto-generated): BLK-001			
5	Project Domain Tags (Choose): Computer Science; Blockchain			
Test Scenario		User (user-type: Researcher) wants to post a new project and is filling project details after which the user proceeds to [PSTPRJ-1.1]		
Step #	**Details**	**Expected Results**	**Actual Results**	**Status**
1	Displaying Project Details	Items 1 to 4 under test data are displayed as already entered data	As Expected	Pass
2	Choose project domain tags	Computer Science & Blockchain are displayed as assigned project domain tags	As Expected	Pass
3	Proceed to PSTPRJ-1.1	Display Page for uploading IRB Approval Letter, Additional documents & Digital Signature	As Expected	Pass

Table 10: Test-Case 6

Test Case ID	T-PCM-1.1	**Test Case Description**	Simulating the procedure of project creation by uploading files as mentioned under [PSTPRJ-1.1]	
Test Case Status(Pass/Fail)			Pass	
Sl. No.	**Prerequisites**			
1	Should be Logged In as user-type:Researcher			
2	Should have filled project details			
Sl. No.	**Test Data**			
1	IRB Approval Letter: IRB_Approval.pdf			
2	Digital Signature: Signature.jpg			
Test Scenario		User (user-type: Researcher) wants to post a new project and is uploading IRB Approval Letter & Digital Signature after; On Completion will be redirected to [OPVW-1.1] & Project will apper under "Ongoing" in User Feed		
Step #	**Details**	**Expected Results**	**Actual Results**	**Status**
1	IRB Approval Letter Upload	IRB_Approval.pdf uploaded as IRB Approval Letter	As Expected	Pass
2	Digital Signature Upload	Signature.jpg uploaded as Digital Signature	As Expected	Pass

Table 11: Test-Case 7

Test Case ID	T-UFA-2.2	**Test Case Description**	Simulating redirection of user to project page as mentioned under [UF-1.2]	
Test Case Status(Pass/Fail)			Pass	
Sl. No.	**Prerequisites**			
1	Logged In as user-type:Researcher/Contributor			
2	There should be a project on the platform that is ongoing			
Sl. No.	**Test Data**			
1	User-Type: Researcher & Project Creator/Owner			
2	Project-Code: BLK-001 & Project-Status: Recruiting			
3	Redirection to: BLK-001 [OPVW-1.1] Page			
Test Scenario		User (user-type: Researcher) wants to view an ongoing project of which the user is the owner.User Feed redirects the user to the corresponding OPVW-1.1 page		
Step #	**Details**	**Expected Results**	**Actual Results**	**Status**
1	User Feed View	The project is listed with corresponding Project-Status	As Expected	Pass
2	Redirection	The user on clicking the listing is redirected to the corresponding OPVW-1.1 page	As Expected	Pass

Table 12: Test-Case 8

Test Case ID	T-PCM-2	**Test Case Description**	Display of information with respect to a project as stated under [OPVW-1]	
Test Case Status(Pass/Fail)			Pass	
Sl. No.	**Prerequisites**			
1	Logged In as user-type:Researcher			
2	There should be a project on the platform that is ongoing and is owned by the user			
Sl. No.	**Test Data**			
1	User-Type: Researcher & Project Creator/Owner			
2	Project-Code: BLK-001 & Project-Status: Recruiting			
3	Project Domain Tags: Computer Science; Blockchain;			
4	Rating of the Researcher			
5	Researcher CV link			
6	No. of Projects worked on: 1			
Test Scenario		User (user-type: Researcher) wants to view an ongoing project of which the user is the owner. General information on the project as stated under test data is displayed		
Step #	**Details**	**Expected Results**	**Actual Results**	**Status**
1	General Project Info.	Test Data is displayed to the user	As Expected	Pass

Table 13: Test-Case 9

Test Case ID	T-PCM-2.1	**Test Case Description**	Display of information & functionalities with respect to a project as stated under [OPVW-1.1]	
Test Case Status(Pass/Fail)			Pass	
Sl. No.	**Prerequisites**			
1	Logged In as user-type:Researcher			
2	There should be a project on the platform that is ongoing and is owned by the user			
2	The Project-Status should be "Recruiting"			
Sl. No.	**Test Data**			
1	Total No. of Applicants: 3			
2	Applicant name, organization, professional designation, ratings, no. of projects the applicant has worked on, no. of projects on relevant domains, view cv			
2	Option to accept or reject applicant			
Test Scenario		User (user-type: Researcher) wants to view an ongoing project of which the user is the owner. Applicant details along with options to accept or reject the same		
Step #	**Details**	**Expected Results**	**Actual Results**	**Status**
1	Applicant related info.	Applicant info. to be displayed	As Expected	Pass
2	Option to accept/reject	Accept/Reject options displayed	As Expected	Pass

Table 14: Test-Case 10

Test Case ID	T-PCM-2.2	**Test Case Description**	Display of information with respect to a project as stated under [OPVW-1.2]	
Test Case Status(Pass/Fail)			Pass	
Sl. No.	**Prerequisites**			
1	Logged In as user-type:Researcher			
2	There should be a project on the platform that is ongoing and is owned by the user			
3	The Project-Status should be "In-Progress"			
Sl. No.	**Test Data**			
1	Contributor details: Name, ratings, view CV link			
2	No. of contributors			
3	link to the corresponding [CCOLLAB-1] page			
4	Option to update IRB Approval Letter			
Test Scenario		User (user-type: Researcher) wants to view an ongoing project of which the user is the owner. The link to the corresponding [CCOLLAB-1] pages and option to update IRB Approval Letter.		
Step #	**Details**	**Expected Results**	**Actual Results**	**Status**
1	Contributor related details	Test Data listed under 1 & 2 is displayed to the user	As Expected	Pass
2	Link to CCOLLAB-1	Option to collaborate should be displayed	As Expected	Pass
3	IRB Update Option	Option to update IRB Approval Letter should be displayed	As Expected	Pass

Table 15: Test-Case 11

Test Case ID	T-PCM-2.3	**Test Case Description**	Display of information with respect to a project as stated under [OPVW-1.3]	
Test Case Status(Pass/Fail)			Pass	
Sl. No.	**Prerequisites**			
1	Logged In as user-type:Researcher			
2	There should be a project on the platform that is completed and is owned by the user			
3	The Project-Status should be "Completed"			
Sl. No.	**Test Data**			
1	Completed project should be listed in profile page			
2	Updated researcher & contributor ratings			
Test Scenario		User (user-type: Researcher) wants to view an ongoing project of which the user is the owner and the project is completed.		
Step #	**Details**	**Expected Results**	**Actual Results**	**Status**
1	Updated Ratings	Updated Ratings of Researcher & Contributor must be visible	As Expected	Pass
2	Project Listing	Project listed under "Completed" tab in profile page	As Expected	Pass
2	Redirection to OPVW-1.3	User is redirected in clicking the project lsiting in the profile page OPVW-1.3	As Expected	Pass

Table 16: Test-Case 12

Test Case ID	T-CPO-1	**Test Case Description**	Display of information & functionalities with respect to a project as stated under [CPVW-1]	
Test Case Status(Pass/Fail)			Pass	
Sl. No.	**Prerequisites**			
1	Logged In as user-type: Contributor			
2	There should be a project on the platform that is posted by a researcher			
3	The project status must be "Recruiting"			
Sl. No.	**Test Data**			
1	User-Type: Contributor & Project Contributor			
2	Project Domain Tags: Computer Science; Blockchain			
3	No. of applicants: 23; Researcher Name; Researcher Rating; Researcher CV Link			
Test Scenario		User (user-type: Collaborator) wants to view a posted project. General information on the project as stated under test data is displayed.		
Step #	**Details**	**Expected Results**	**Actual Results**	**Status**
1	General Project Info.	Test Data is displayed	As Expected	Pass

Table 17: Test-Case 13

Test Case ID	T-CPO-1.1	**Test Case Description**	Display of information & functionalities with respect to a project as stated under [CPVW-1.1]	
Test Case Status(Pass/Fail)			Pass	
Sl. No.	**Prerequisites**			
1	Logged In as user-type: Contributor			
2	There should be a project on the platform that is posted by a researcher			
3	The project status must be "Recruiting"			
4	User: Contributor must have been accepted by the Project Owner, [Researcher]			
Sl. No.	**Test Data**			
1	IRB Approval Letter link			
2	Option to upload User Digital Signature			
2				
Test Scenario		User (user-type: Contributor) agrees to the terms and conditions of the IRB Approval Letter and enters project start and end date as confirmation and clicks submit button.		
Step #	**Details**	**Expected Results**	**Actual Results**	**Status**
1	Intial CPVW display [CPVW-1.1]	Test Data displayed	As Expected	Pass
2	Signature Uploaded	Redirection to CPVW-1.2	As Expected	Pass

Table 18: Test-Case 14

Refer APPENDIX A – Prototypes A for Figma Prototype link & Coded Back-end Prototype Documentation.

8 Conclusion

8.1 Project Benefits

The project (CollabChain) aims to provide a confidential environment for online research collaboration through the use of blockchain smart contracts. Use of permission-based blockhain framework like Hyperledge Fabric provides an additional layer of confidentiality & security, restricting access to confidential data only to selected and authorized individuals which is regarded to be essential for research work. The use of this evolving technology brings in multiple advantages to the platform. Some of the most significant and relevant mentions are as follows:

1. **Security and Confidentiality:**
    - *Blockchain Technology:* Ensures high security and transparency.
    - *Smart Contracts:* Tamper-resistant and immutable for data integrity.

2. **Data Integrity and Traceability:**
    - *Immutable Record:* Blockchain maintains an audit trail for traceability.
3. **Trust and Transparency:**
    - *Decentralization:* Eliminates the need for intermediaries, fostering trust.
    - *Transparent Smart Contracts:* Allow participants to validate collaboration terms.
4. **Efficiency in Collaboration:**
    - *Smart Contracts Automation:* Streamlines tasks and reduces administrative overhead.
5. **Decentralized Data Ownership:**
    - *User Control:* Ensures researchers maintain ownership and control over their data.
6. **Enhanced Data Sharing:**
    - *Selective Data Sharing:* Facilitates efficient collaboration without compromising sensitive data.
7. **Immutable Intellectual Property Protection:**
    - *Smart Contract Terms:* Encodes intellectual property rights, ensuring an immutable record.
8. **Global Collaboration Opportunities:**
    - *Borderless Collaboration:* Enables international collaboration without intermediaries.
9. **Regulatory Compliance:**
    - *Built-in Compliance:* Smart contracts can be programmed to comply with regulations and standards.

8.2 Future Scope for improvements

The project at its bare minimum state comes with a plethora of functionalities supporting and promoting a confidential and collaborative environment for conducting research work online. From creating and collaborating in projects to reviewing and rating research papers after completion, rating peers, mentors and/or mentees after research work, the platform (CollabChain) itself is a complete package of online research collaboration. Blockchain being implemented as an abstracted technology in the platform reliefs the users of the complexities smart contract brings and provides an interactive front-end to simplify the task. However, there are considerable no. of improvements that the platform can undergo to evolve further into the domain of providing a confidential and collaborative environment. Some of the significant ones that have a higher priority are as follows:

1. **Third-Party Platform Integration:** The integration of collaborative platforms that help to prepare/analyse the data being used for the research project like Google Colab, Jupyter Notebooks, etc.

2. **Use of A. I.:** Trying to recruit a trust worthy contributor for a project can be a tiring job. To overcome the problem of filtering trust worthy applicants for the project, use of A. I. can prove beneficial. Checking applicant past reviews, quality and quantity of the projects the user has worked on and then providing a filtered list of candidates can reduce the chances of RCA violations in an ongoing project.

3. **Scalability:**

    - Implement solutions to enhance scalability for a growing user base.

4. **Interoperability:**

    - Work towards standardization and interoperability with other research platforms.

5. **Regulatory Compliance Enhancements:**

    - Stay compliant with evolving regulations and standards.

6. **Advanced Data Analytics:**

- Integrate advanced data analytics tools for meaningful insights.

7. **Enhanced Privacy Features:**
   - Introduce advanced privacy features like zero-knowledge proofs.
8. **Smart Contract Templates:**
   - Develop customizable smart contract templates for different collaborations.
9. **Decentralized Identity Management:**
   - Implement decentralized identity management solutions for enhanced security.
10. **Incentive Mechanisms:**
    - Introduce token-based incentives to encourage collaboration and reward contributors.
11. **Enhanced Governance Models:**
    - Explore decentralized governance models for community involvement in decisions.
12. **Cross-Institutional Collaboration Support:**
    - Facilitate collaboration among researchers from different institutions.
13. **Real-time Collaboration Features:**
    - Integrate real-time features like live chat and collaborative document editing.
14. **Improved Security Auditing:**
    - Implement robust security auditing mechanisms for proactive vulnerability identification.

References

[1] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. D. Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, S. Muralidharan, C. Murthy, B. Nguyen, M. Sethi, G. Singh, K. Smith, A. Sorniotti, C. Stathakopoulou, M. Vukolic, S. W. Cocco, and J. Yellick. Hyperledger fabric: A distributed operating system for permissioned blockchains. *CoRR*, abs/1801.10228, 2018.

[2] A. P. Association. *Publication manual of the American Psychological Association (6th ed.)*. American Psychological Association (APA), 2010.

[3] S. Bansal, S. Mahendiratta, S. Kumar, P. Sarma, A. Prakash, and B. Medhi. Collaborative research in modern era: Need and challenges, 2019.

[4] I. BG. Agile methodology (n.d.)., 2023.

[5] V. Buterin. Ethereum: A next generation smart contract & decentralized application platform. *Communications of the ACM*, 56.7, 2013.

[6] T. Farooq. Hyperledger fabric vs ethereum: Which should you choose?, 2023.

[7] N. I. of Health (NIH). Protection of human subjects, 2023.

[8] ResearchHub. Docs - researchhub docs. (n.d.), 2020.

[9] ScienceOpen. About scienceopen-. (n.d.)., 2013.

A APPENDIX - Prototypes

Provide the filtered part of RM showing prototype features. State the detailed steps of compilation, execution and setups.
To view the Figma prototype of the platform **click here!**
This section provides description the code used for preparing a coded prototype, representing the platforms back-end functionalities along with test outputs. *Please note* that the purpose of this prototype is to demonstrate how blockchain smart-contracts are integrated with the back-end of a web application. The prototype may or may not comply with the requirements of the platform as mentioned under the subsections of section 5 and as designed under section 6.3.

A.1 Environment Setup

Prerequisites

1. Install Node.js and Ganache.
2. Run the following command in the terminal: `npm install -g yarn`.

Frontend

1. Navigate to `/Prototype/frontend`.
2. Run the command: `yarn install` to install all dependencies locally.
3. Run the command: `yarn dev` to open the app on the localhost server.

Backend

1. Navigate to `/Prototype/backend/` and create a `.env` file with the following variables:

   `PORT`: Set which port the server will run.

   `NODE_ENV`: Detect whether the server is run in development mode or production (handled by the `yarn` commands mentioned above).

   `DB_URI`: Stores the URI for the local database.

`MONGO_URI`: Stores the URI for the production database.

`ETH_ADDRESS`, `ETH_URL`: Store the address and URL of the deployed contract.

`JWT_SECRET`, `JWT_EXPIRE`, `COOKIE_EXPIRE`, `SMPT_SERVICE`, `SMPT_MAIL`, `SMPT_PASS`.

2. Run the command: `yarn install`.

3. Run the command: `yarn dev`.

Blockchain

1. Navigate to `/Prototype/blockchain/` and run the following commands:

   `npm i truffle`.

   `truffle migrate` (make sure you run Ganache on your system before running the application).

A.2 Coded Prototype

A.2.1 Project APIs

A.2.1.1 getAllProjects API [Similar to UF-1] *Brief-explanation:*

- **Description:** Fetches completed projects from the public feed, filtering based on their end date.

- **Purpose:** Showcasing a curated list of accomplished projects to the public.

```
// Get all projects on public feed
exports.getAllProjects = catchAsyncErrors(async (req, res) => {
  const publicProjects = await Project.find({ endDate: { $lt: new Date() } });
  res.json(publicProjects);
});
```

Listing 1: Get all projects on public feed

Figure 19: Test Output of getAllProjectsAPI

A.2.1.2 createProject API [Similar to PSTPRJ-1] *Brief-explanation:*

- **Description:** Empowers mentors to initiate new projects, tying them to their unique user ID and Ethereum address.
- **Purpose:** Facilitating the seamless creation of innovative projects by mentors.

```
// Mentor creates a new project
exports.createProject = catchAsyncErrors(async (req, res) => {
  const mentor = req.user.id; // Assuming you have mentor authentication
  const mentorAddress = req.body.ethAddress;
  const project = await Project.create({ ...req.body, mentor });
  createProject(project._id.toString(), mentorAddress);
  res.status(201).json(project);
});
```

Listing 2: Mentor creates a new project

Figure 20: Output(a) of createProject API test

Figure 21: Output(b) of createProject API test

A.2.1.3 getProjectDetails API [Similar to OPVW-1/CPVW-1/PVW-1]

Brief-explanation:

- **Description:** Retrieves comprehensive details of a specific project, concealing certain information depending on the user's role.
- **Purpose:** Delivering detailed project information while respecting user-specific access privileges.

```
// Users get project details (some details hidden based on user type: public, mentor, mentee)
exports.getProjectDetails = catchAsyncErrors(async (req, res) => {
  const projectId = req.params.projectid;
  const project = await Project.findById(projectId);

  if (!project) {
    return res.status(404).json({ message: 'Project not found.' }); }
  res.json(project);
});
```

Listing 3: Users get project details (some details hidden based on user type: public, mentor, mentee)

Figure 22: Output of getProjectDetails API test

A.2.1.4 editProject API [Similar to OPVW-1.1] *Brief-explanation:*

- **Description:** Allows mentors to dynamically modify the specifics of a designated project.
- **Purpose:** Providing mentors with the flexibility to update project details as circumstances evolve.

```
// Mentor edits project details
exports.editProject = catchAsyncErrors(async (req, res) =>
    {
  const projectId = req.params.projectid;
  const project = await Project.findByIdAndUpdate(
    projectId,
    { ...req.body },
    { new: true } );
  if (!project) {
    return res.status(404).json({ message: 'Project not
       found.' }); }
  res.json(project);
});
```

Listing 4: Mentor edits project details

Figure 23: Output of editProject API test

A.2.1.5 applyToProject API [Similar to PVW-1.1] *Brief-explanation:*

- **Description:** Permits users to express interest in a project by submitting their application details.
- **Purpose:** Streamlining the application process for users keen on participating in a project.

```
// Users applying to a project
exports.applyToProject = catchAsyncErrors(async (req, res) => {
  const projectId = req.params.projectid;
  const project = await Project.findById(projectId);
  if (!project) { return res.status(404).json({ message: 'Project not found.' });
  }
  project.menteesApplication.push(req.body);
  await project.save();
  res.status(200).json({ message: 'Application successful.' });
});
```

Listing 5: Users applying to a project

Figure 24: Output of applyToProject API test

A.2.1.6 withdrawApplication API [Similar to PVW-1.1]

Brief-explanation:

- **Description:** Offers users the capability to retract their project application seamlessly.
- **Purpose:** Providing a user-friendly mechanism for withdrawing from project applications.

```
// Users withdrawing their application
exports.withdrawApplication = catchAsyncErrors(async (req, res) => {
  const projectId = req.params.projectid;
  const project = await Project.findById(projectId);
  if (!project) {return res.status(404).json({ message: 'Project not found.' });
  }
  project.menteesApplication = project.menteesApplication.filter((application) => application.userId != req.user.id);
  await project.save();
  res.status(200).json({ message: 'Withdrawal successful.' }); });
```

Listing 6: Users withdrawing their application

Figure 25: Output of wothdrawApplication API test

A.2.1.7 updateMenteeStatus API [Similar to OPVW-1.1]

Brief-explanation:

- **Description:** Authorizes mentors to efficiently manage and update the status of mentee applications within a project.
- **Purpose:** Empowering mentors to curate and oversee the composition of their project teams.

```
// Mentor updates mentee status
exports.updateMenteeStatus = catchAsyncErrors(async (req, res) => {
  const projectId = req.params.projectid;
  const project = await Project.findById(projectId);

  if (!project) {
    return res.status(404).json({ message: 'Project not found.' });
  }

  let newMenteesList = req.body.menteesList;
  newMenteesList.forEach(application => {
    if (application.status === "approved") {
      project.menteesApproved.push({
        userId: application.userId,
        name: application.name
      })
    }
  })

  project.menteesApplication = newMenteesList;
  await project.save();

  res.status(200).json({ message: 'Changes successful.' }) ;
});
```

Listing 7: Mentor updates mentee status

Figure 26: Output of updateMenterrStatus API test

A.2.1.8 addTask API [Similar to RCOLLAB-1.2] *Brief-explanation:*

- **Description:** Enables mentors to append new tasks to a project, each uniquely identified with a task ID.
- **Purpose:** Streamlining the process for mentors to enrich their projects with diverse tasks.

```
// Mentor adds new task to project
exports.addTask = catchAsyncErrors(async (req, res) => {
  const projectId = req.params.projectid;
  const project = await Project.findById(projectId);

  if (!project) {
    return res.status(404).json({ message: 'Project not
      found.' });
  }

  let taskId = projectId + project.tasks.length;
  const mentorAddress = req.body.ethAddress;
  let task = {
```

```
    title: req.body.title,
    description: req.body.description,
    priority: req.body.priority,
    dueDate: req.body.dueDate,
    id: taskId,
    taskStatus: "pending",
  }

  project.tasks.push(task);
  createTask(projectId, taskId, mentorAddress);
  await project.save();

  res.status(200).json({ message: 'Task added successfully.' });
});
```

Listing 8: Mentor adds new task to project

Figure 27: Output(a) of addTask API test

Figure 28: Output(b) of addTask API test

A.2.1.9 addTaskContributor API [Similar to RCOLLAB-1.2]

Brief-explanation:

- **Description:** Facilitates mentors in assigning contributors to specific tasks within a project.
- **Purpose:** Enhancing collaboration by allowing mentors to involve contributors in specific project tasks.

```
// Mentor adds new contributor
exports.addTaskContributor = catchAsyncErrors(async (req, res) => {
  const projectId = req.params.projectid;
  const taskId = req.params.taskid;
  const project = await Project.findById(projectId);
  if (!project) {return res.status(404).json({ message: 'Project not found.' });
  }
  const taskIndex = project.tasks.findIndex(currTask => currTask.id == taskId);
  project.tasks[taskIndex].menteesAssigned.push({
```

```
    userId: req.body.userId,
    name: req.body.name });
  const menteeId=project.menteesApproved.findIndex(
      currMentee => currMentee.userId == req.body.userId)
  project.menteesApproved[menteeId].assignedTaskIds.push(
      taskId);
  if (project.tasks[taskIndex].taskStatus === 'pending')
    project.tasks[taskIndex].taskStatus = 'active';
  const mentorAddress = req.body.ethAddress;
  //assignUser(projectId, taskId, req.body.
      candidateEthAddress, mentorAddress);
  assignUser(projectId, taskId, process.env.MENTEEETH,
      mentorAddress);
  await project.save();
  res.status(200).json({ message: 'Task assigned
      successfully.' });
});
```

Listing 9: Mentor adds new contributor

Figure 29: Output(a) of addTaskContributor API test

Figure 30: Output(b) of addTaskContributor API test

A.2.1.10 removeTaskContributor API *Brief-explanation:*

- **Description:** Grants mentors the ability to remove contributors from assigned tasks within a project.
- **Purpose:** Providing mentors with flexibility in managing task-specific contributors.

```
// Mentor removes a contributor
exports.removeTaskContributor = catchAsyncErrors(async (req, res) => {
  const projectId = req.params.projectid;
  const taskId = req.params.taskid;
  const project = await Project.findById(projectId);
  if (!project) {
    return res.status(404).json({ message: 'Project not found.' }); }
  const mentorAddress = req.body.ethAddress;
  const taskIndex = project.tasks.findIndex(currTask => currTask.id == taskId);
```

```
  project.tasks[taskIndex].menteesAssigned = project.tasks[taskIndex].menteesAssigned.filter(mentee => { mentee.userId !== req.body.removeUserId })
  const menteeId=project.menteesApproved.findIndex(currMentee => currMentee.userId == req.body.removeUserId)
  console.log(project.menteesApproved[menteeId])
  project.menteesApproved[menteeId].assignedTaskIds = project.menteesApproved[menteeId].assignedTaskIds.filter(taskID => { taskID !== taskId })
  if (project.tasks[taskIndex].menteesAssigned.length === 0)
    project.tasks[taskIndex].taskStatus = 'pending';
  //removeUser(projectId, taskId, req.body.removeUserAddress, mentorAddress);
  removeUser(projectId, taskId, process.env.MENTEEETH, mentorAddress);
  await project.save();
  res.status(200).json({ message: 'Task unassigned successfully.' });
});
```

Listing 10: Mentor removes a contributor

Figure 31: Output(a) of removeTaskContributor API test

Figure 32: Output(b) of removeTaskContributor API test

A.2.1.11 uploadTaskWork API [Similar to CCOLLAB-1.3]

Brief-explanation:

- **Description:** Empowers selected mentees to submit their task work, associating it with a task ID and their Ethereum address.
- **Purpose:** Facilitating a seamless process for mentees to submit their work for project evaluation.

```
// Selected mentees upload their work
exports.uploadTaskWork = catchAsyncErrors(async (req, res) => {
  const projectId = req.params.projectid;
  const taskId = req.params.taskid;
  const project = await Project.findById(projectId);
  if (!project) {
    return res.status(404).json({ message: 'Project not found.' });
  }
  const taskIndex = project.tasks.findIndex(currTask => currTask.id == taskId);
  const menteeAddress = req.body.ethAddress;
  // project.tasks[taskIndex]
```

```
  const key=req.body.key;
  const file = req.files.file;
  const filename = file.name;
  const hash = crypto.createHash('sha256');
  hash.update(file.data);
  const fileHash = hash.digest('hex');
  console.log(fileHash);

  file.mv('./uploads/'+filename, function(err){
    if (err) {
      res.status(404).json(err);}else{
      res.status(200).json({ message: 'Uploaded
         successfully.' });
    }})
  createDocument(taskId, taskId+'123', fileHash, key,
     menteeAddress);
});
```

Listing 11: Selected mentees upload their work

Figure 33: Output(a) of uploadTaskWork API test

Figure 34: Output(b) of uploadTaskWork API test

A.2.1.12 getAssignedTasks API[Similar to RCOLLAB-1]

Brief-explanation:

- **Description:** Retrieves a tailored list of tasks assigned to a specific mentee within a project.
- **Purpose:** Offering mentees a personalized view of tasks assigned to them within a project.

```
// Selected mentees get their assigned tasks
exports.getAssignedTasks = catchAsyncErrors(async (req, res) => {
  const projectId = req.params.projectid;
  const project = await Project.findById(projectId);

  if (!project) {
    return res.status(404).json({ message: 'Project not found.' });
  }

```

```
  const mentee = project.menteesApproved.find(mentee =>
     mentee.userId == req.user.id)
  console.log(project.menteesApproved)
  console.log(req.user.id)
  const menteeTasks = mentee.assignedTaskIds.map(taskId =>
      {
    return project.tasks.find(ele => ele.id === taskId)
  })
  res.json(menteeTasks);
});
```

Listing 12: Selected mentees get their assigned tasks

Figure 35: Output of getAssignedTasks API test

A.2.1.13 markTaskComplete API [Similar to TMOD-1]

Brief-explanation:

- **Description:** Allows mentors to signify the completion of a specific task within a project, updating its status.
- **Purpose:** Empowering mentors to track and manage the progress of tasks within their projects.

```
// Mentor marks the task complete
exports.markTaskComplete = catchAsyncErrors(async (req, res) => {
  const projectId = req.params.projectid;
  const taskId = req.params.taskid;

  const project = await Project.findById(projectId);

  if (!project) {
    return res.status(404).json({ message: 'Project not found.' });
  }

  const taskIndex = project.tasks.findIndex(currTask => currTask.id == taskId);
  project.tasks[taskIndex].taskStatus='complete';

  const verificationKey = req.body.verificationKey;
  const mentorAddress = req.body.ethAddress;

  completeTask(projectId, taskId, verificationKey);
});
```

Listing 13: Mentor marks the task complete

Figure 36: Output(a) of markTaskComplete API test

Figure 37: Output(b) of markTaskComplete API test

A.2.2 User APIs

A.2.2.1 registerUser API [Similar to REG-1.1] *Brief-explanation:*

- **Description:** Registers a new user by creating a user profile with provided details.
- **Purpose:** Facilitating the user registration process and generating an authentication token.

```
// Register a User
exports.registerUser = catchAsyncErrors(async (req, res, next) => {
    // const myCloud = await cloudinary.v2.uploader.upload (req.body.avatar, {
    //     folder: "avatars",
    //     width: 150,
    //     crop: "scale",
    // });
    const user = await User.create({...req.body});
    sendToken(user, 201, res); });
```

Listing 14: Register a User

Figure 38: Output of registerUser API test

A.2.2.2 loginUser API [Similar to LGN-1] *Brief-explanation:*

- **Description:** Handles user login by verifying email and password, issuing an authentication token upon successful validation.
- **Purpose:** Providing secure user authentication for accessing the system.

```
// Login User
exports.loginUser = catchAsyncErrors(async (req, res, next) => {
    // ... (your code here)
    sendToken(user, 200, res);
});
```

Listing 15: User LogIn

Figure 39: Output of loginUser API test

A.2.2.3 logout API [Similar to LGOUT-1] *Brief-explanation:*

- **Description:** Logs out the user by clearing the authentication token from the cookie.
- **Purpose:** Ensuring a secure logout process for user sessions.

```
// Logout User
exports.logout = catchAsyncErrors(async (req, res, next)
   => {
    // ... (your code here)
    res.status(200).json({
        success: true,
        message: "Logged Out",
    });
});
```

Listing 16: User LogOut

Figure 40: Output of logout API test

A.2.2.4 updatePassword API [Similar to UPF-1] *Brief-explanation:*

- **Description:** Allows users to securely update their password by verifying the old password.
- **Purpose:** Empowering users to maintain the security of their accounts through periodic password changes.

```
// Update User password
exports.updatePassword = catchAsyncErrors(async (req, res, next) => {
    // ... (your code here)
    sendToken(user, 200, res);
});
```

Listing 17: Update User Password

Figure 41: Output of updatePassword API test

A.2.2.5 updateProfile API [Similar to UPF-1] *Brief-explanation:*

- **Description:** Enables users to update their profile information, such as avatar and other relevant details.
- **Purpose:** Allowing users to keep their profiles current and personalized.

```
// Update User Profile
exports.updateProfile = catchAsyncErrors(async (req, res, next) => {
    // ... (your code here)
    res.status(200).json({
        success: true,
    });
});
```

Listing 18: Update User Profile

Figure 42: Output of updateProfile API test

A.2.3 Blockchain

A.2.3.1 createTask Function [Similar to PSTPRJ-1] *Brief-explanation:*

- **Description:** Enables mentors to create a new project, associating it with a unique project ID and marking themselves as the project mentor.
- **Purpose:** Initiating new collaborative projects on the blockchain.

```
function generateKey() internal view returns (bytes32) {
    uint256 seed = uint256(block.timestamp) + uint256(
        blockhash(block.number - 1));
    return keccak256(abi.encodePacked(seed));
}

//compare verification key strings [String Comparison]
------DEBUG!------
function compareStrings(string memory a, string memory b)
   public pure returns (bool) {
```

```
    bytes memory strA = bytes(a);
    bytes memory strB = bytes(b);
    if (strA.length != strB.length) {
        return false;
    }
    for (uint i = 0; i < strA.length; i++) {
        if (strA[i] != strB[i]) {
            return false;
        }
    }
    return true;
}
------DEBUG!------
//mentor creates a project
function createProject(string memory projectId) public{
    projectMentors[projectId]=msg.sender;
    emit ProjectCreated(projectMentors[projectId]);
}
```

Listing 19: Smart Contract Code

A.2.3.2 createTask Function [Similar to RCOLLAB-1.3]

Brief-explanation:

- **Description:** Allows mentors to create tasks within a project, generating a verification key for later use.
- **Purpose:** Structuring projects by defining specific tasks and ensuring their accountability.

```
// Mentor creates a task
// Upon creation, a verification key is generated which
   will be used later

function createTask(string memory projectId, string memory
    taskId) public onlyMentor(projectId, msg.sender) {
    tasks[taskId] = Task({
        projectId: projectId,
        isComplete: false,
```

```
        verificationKey: "abc",
        assignedUsers: new address[](0),
        documentIDs: new string[](0)
    });
    emit TaskCreated();
}
```

Listing 20: Mentor creates a task in Solidity

A.2.3.3 assignUser Function *Brief-explanation:*

- **Description:** Permits mentors to assign users to contribute to a specific task within a project.
- **Purpose:** Facilitating collaboration by involving multiple contributors in various project tasks.

```
// Mentor assigns a user to contribute to the task
function assignUser(string memory projectId, string memory
    taskId, address user) public onlyMentor(projectId, msg
   .sender) onlyOngoingTasks(taskId) {
    tasks[taskId].assignedUsers.push(user);
    emit UserAssigned();
}
```

Listing 21: Mentor assigns a user to contribute to the task

A.2.3.4 removeUser Function [Similar to RCOLLAB-1.3]

Brief-explanation:

- **Description:** Allows mentors to remove users from assigned tasks, ensuring flexibility in task management.
- **Purpose:** Providing mentors with the ability to adjust task assignments as needed.

```
// Mentor removes a user from the task
// Will add checks to verify from the user before removing
function removeUser(string memory projectId, string memory
    taskId, address user) public onlyMentor(projectId, msg
   .sender) onlyOngoingTasks(taskId) {
```

```
    address[] storage users = tasks[taskId].assignedUsers;
    for (uint256 i = 0; i < users.length; i++) {
        if (users[i] == user) {
            users[i] = users[users.length - 1];
            users.pop(); // Remove the last element
            break;
        }
    }
    emit UserRemoved();
}
```

Listing 22: Mentor removes a user from the task

A.2.3.5 createDocument Function [Similar to RCOLLAB-1.3]

Brief-explanation:

- **Description:** Enables assigned users to add documents to a project, associating them with a verification key.
- **Purpose:** Facilitating the submission of work and documentation by project contributors.

```
// Assigned users add documents to the project
// When they upload the documents, they will be given the verification key
function createDocument(string memory taskId, string memory documentId, string memory content, string memory key) public onlyAssigned(taskId, msg.sender) onlyOngoingTasks(taskId) {
    tasks[taskId].documentIDs.push(documentId);
    tasks[taskId].verificationKey = key;
    documentHash[documentId] = content;
    emit DocumentCreated(tasks[taskId].verificationKey);
}
```

Listing 23: Assigned users add documents to the project

A.2.3.6 completeTask Function [Similar to TMOD-1]

Brief-explanation:

- **Description:** Allows mentors to mark a task as complete by verifying a provided verification key against the original key.

- **Purpose:** Verifying and acknowledging the completion of specific tasks within a project.

```
// Mentor marks the task complete by entering the verification key given to them by the user
// Verifying this key against our original key to check authenticity

function completeTask(string memory projectId, string memory taskId, string memory key) public onlyMentor(projectId, msg.sender) onlyOngoingTasks(taskId) {

    // require(keccak256(abi.encodePacked(key)) == keccak256(abi.encodePacked(tasks[taskId].verificationKey)), "Invalid verification key");
    require(compareStrings(key, tasks[taskId].verificationKey), "Invalid verification key");
    tasks[taskId].isComplete = true;
    emit TaskCompleted();
}
```

Listing 24: Mentor marks the task complete by entering the verification key given to them by the user

A.2.3.7 checkTaskStatus Function

Brief-explanation: This does not comply with the system requirements of the platform and is only is used for prototype functionality.

- **Description:** Provides a function for the central server to check the status of a specific task before granting access to associated documents.
- **Purpose:** Ensuring that only completed tasks are eligible for document access, maintaining security and integrity.

```
// Central server uses this function to check task status before allowing mentor to access the documents
function checkTaskStatus(string memory taskId) public view returns (bool) { return tasks[taskId].isComplete; }
```

Listing 25: Central server uses this function to check task status before allowing mentor to access the documents