A PROJECT REPORT ON

DECENTRALIZED TENDER MANAGEMENT SYSTEM FOR ROAD DEVELOPMENT USING BLOCKCHAIN

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE

OF

BACHELOR OF ENGINEERING(COMPUTER ENGINEERING)

SUBMITTED BY

STUDENT NAME	EXAM NO :
ANIKET DATAR	**B190074215**
ATHARV VASPATE	**B190074279**
SHANKAR AVAD	**B190074207**
NIRANJAN PATANKAR	**B190074253**

UNDER THE GUIDANCE OF
PROF. M.V.MARATHE

DEPARTMENT OF COMPUTER ENGINEERING

PVG's COLLEGE OF ENGINEERING & TECHNOLOGY AND GKPIM

44, VIDYANAGARI, PARVATI, PUNE 411009

SAVITRIBAI PHULE PUNE UNIVERSITY
2022-23

CERTIFICATE

This is to certify that the project report entitles

DECENTRALIZED TENDER MANAGEMENT SYSTEM FOR ROAD DEVELOPMENT USING BLOCKCHAIN

Submitted by

STUDENT NAME	EXAM NO :
ANIKET DATAR	**B190074215**
ATHARV VASPATE	**B190074279**
SHANKAR AVAD	**B190074207**
NIRANJAN PATANKAR	**B190074253**

is a bonafide student of this institute and the work has been carried out by him/her under the supervision of **Prof. M. V. Marathe** and it is approved for the partial fulfillment of the requirement of Savitribai Phule Pune University, for the award of the degree of **Bachelor of Engineering** (Computer Engineering).

(Prof. M. V. Marathe)
Guide
Department of Computer Engineering

(Prof. D. D. Sapkal)
Head
Department of Computer Engineering

(Dr. M R Tarambale)
I/C Principal,
PVG's College of Engineering and Technology GKPIM, Pune – 09

Place:Pune
Date:19/05/2023

ACKNOWLEDGEMENT

We would like to express our profound gratitude towards Prof. D.D.Sapkal of the Computer Department and for his contributions to the completion of our project titled 'Decentralized Tender Management System'.

We would like to express our special thanks to our mentor Prof. M.V.Marathe for her time and efforts she provided throughout the semester. Her useful advice and suggestions were really helpful to us during the project. In this aspect, we are eternally grateful to you.

We would also like to thank IncuGlobe Solutions Pvt. Ltd. for providing us with an opportunity to work on this project as well as providing sponsorship for the project. Your guidance and technical support were really helpful to us during the project. In this aspect, we are grateful to the company.

We would like to acknowledge that this project was completed entirely by us and not by someone else.

NAME OF THE STUDENTS
ANIKET DATAR
ATHARV VASPATE
SHANKAR AVAD
NIRANJAN PATANKAR

ABSTRACT

The tender management process is a critical aspect of procurement and plays a vital role in ensuring fairness, transparency, and efficiency in awarding contracts. However, traditional tender management systems often face challenges such as lack of trust, transparency, and vulnerability to corruption. This abstract introduces a decentralized tender management system that leverages blockchain technology to address these challenges.

Blockchain, as a decentralized and immutable ledger, provides a secure and transparent environment for recording and verifying transactions. By utilizing blockchain, the proposed tender management system enables all stakeholders, including bidders and government officers to participate in a trustless and auditable process.

In this system, tenders are published on the blockchain network, allowing interested bidders to access and review the requirements. Bidders can submit their proposals electronically ensuring the integrity and confidentiality of the submitted data. The blockchain's distributed consensus mechanism guarantees that the submitted proposals remain tamper-proof and resistant to unauthorized modifications.

The government officers can then review these bids and select the specific bid manually.

IncuGlobe Solutions Pvt. Ltd.
7, AG TC Bavdhan 411021
reach.us@incuglobe.com
+912067703300
www.incuglobe.com

To,

Whomsoever it may concern,

Given your previous record of excel and hard work, the board of directors is proud to announce that we accept your B.E. Project proposal. We give you our consent to begin working on it, applicable from August 1 2022 to April 30 2023

As received, we recognize

1. Aniket Datar
2. Shankar Avad
3. Atharv Vaspate
4. Niranjan Patankar

as the team for the project with the domain being `Public Blockchains', the core area for R&D. Subsequently we have assigned the following as the topic for the project:

"Decentralized Tender Management System for Road Development"

Our board has also decided to grant external publishing rights for the selected topic, and the mandate for the same will be discussed upon and presented on later stages. This entitlement includes esteemed journals, conferences and research articles.

Please accept our best wishes for the project, and we hope to see the positive consequences of it. We expect you to report to us on a regular basis and complete the project in the given time frame by your College/University. Should you have any queries regarding the project, please feel free to reach us at reach.us@incuglobe.com

Regards,

R&D Dept,

IncuGlobe Solutions Pvt. Ltd.

INCUGLOBE SOLUTIONS PVT. LTD. R&D Dept.

IncuGlobe Solutions Pvt. Ltd.

7, AG TC Bavdhan 411021
+912067703300
reach.us@incuglobe.com
www.incuglobe.com

01-05-2023

LETTER OF EXPERIENCE & PROJECT COMPLETION CERTIFICATE

This is to certify that Mr. Aniket Datar. Mr. Atharv Vaspate, Mr. Shankar Avad and Mr. Niranjan Patankar, final year (B.E.) students of Department of Computer Engineering in Pune Vidyarthi Griha's College Of Engineering, Pune working under the guidance of Mr. Rohan Patel successfully completed the project titled "Decentralized Tender Management System for Road Development" as part of final year B.E. project of Savitribai Phule Pune University, Pune for academic year 2022-23. They have worked for a period of 9 months starting from 1 August 2022 to 30 April 2023.

Further, we certify that the project titled "Decentralized Tender Management System for Road Development" is accepted as a service based application and will be deployed further for integrations and industrial use. The project was successfully completed by Mr. Aniket Datar, Mr. Atharv Vaspate, Mr. Shankar Avad and Mr. Niranjan Patankar within defined deadlines.

During their tenure at IncuGlobe Solutions Pvt. Ltd. all project group members have been a dedicated and valuable resource to our company. They were found to be honest, hardworking, dedicated and punctual during their project phase.

Sincerely,
Rohan Patel
E.S.C.P.
IncuGlobe Solutions Pvt. Ltd.

INCUGLOBE SOLUTIONS PVT. LTD.
R&D Dept.

CONTENTS

1 INTRODUCTION **1**

1.1 Overview 1

1.2 Motivation 1

1.3 Problem Definition and Objectives 2

1.4 Project Scope and Limitations 3

1.5 Methodologies of problem solving 5

2 LITERATURE SURVEY **8**

3 SOFTWARE REQUIREMENT SPECIFICATION **10**

3.1 Assumptions and Dependencies 10

3.1.1 Assumptions 10

3.1.2 Dependencies 10

3.2 Functional Requirements 11

3.2.1 User Registration and Authentication . . . 11

3.2.2 Tender Creation and Publication 12

3.2.3 Bid Submission 12

3.2.4 Bid Evaluation and Selection 12

3.2.5 Contract Generation and Execution 12

3.2.6 Transparency and Auditability 12

3.2.7 Notifications and Communication 13

3.3 External Interface Requirements 13

3.3.1 User Interfaces: 13

3.3.2 Blockchain Integration 13

3.3.3 Communication and Notifications 14

3.4 Nonfunctional Requirements 14

3.4.1 Performance Requirements 14

3.4.2 Security Requirements 14

3.4.3 Software Quality Attributes 15

3.5 System Requirements 16
3.5.1 Database Requirements 16
3.5.2 Software Requirements(Platform Choice) . 16
3.5.3 Hardware Requirements 17
3.6 Analysis Models: SDLC Model to be applied . . 18
3.6.1 Agile Model 18

4 SYSTEM DESIGN 24
4.1 System Architecture 24
4.2 Entity Relationship Diagrams 25
4.3 UML Diagrams 26
4.3.1 Class Diagram 26
4.3.2 Use Case 28
4.3.3 Deployment Diagram 30
4.3.4 Sequence Diagram 32

5 Project Plan 34
5.1 Project Estimate 34
5.1.1 Reconciled Estimates 34
5.1.2 Project Resources 35
5.2 Risk Management 35
5.2.1 Risk Identification 35
5.2.2 Risk Analysis 36
5.2.3 Overview of Risk Mitigation, Monitoring & Management 38
5.3 Project Schedule 39
5.3.1 Project Task Set 39
5.3.2 Task Network 41
5.3.3 Timeline Chart 43
5.4 Team Organization 43
5.4.1 Team Structure 43
5.4.2 Management Reporting and Communication 44

6 PROJECT IMPLEMENTATION **45**
6.1 Overview of Project Modules 45
6.2 Tools and Technologies used 45
6.3 Algorithm Details 47
6.3.1 Algorithm 1: Bidder Dashboard - List All Tenders 47
6.3.2 Algorithm 2: Bidder Dashboard - List My Bids 48
6.3.3 Algorithm 3: Uploader Dashboard - Show All Bids 48
6.3.4 Algorithm 4: Smart Contract (Tender.sol) 49

7 SOFTWARE TESTING **51**
7.1 Type of Testing 51
7.2 Test Cases and Test Results 53

8 RESULTS **58**
8.1 Outcomes 58
8.2 Screenshots 59

9 CONCLUSIONS **66**
9.1 Conclusions 66
9.2 Future Work 66
9.3 Applications 67

10 REFERENCES **69**

11 APPENDIX A **70**
11.1 Feasibility Study 70
11.1.1 Operational Feasibility 70
11.1.2 Technical Feasibility 70
11.2 Problem Analysis 70
11.3 Problem Type 70

12 APPENDIX C **71**

LIST OF ABBREVIATIONS

1. DTM - Decentralized Tender Management
2. RD - Road Development
3. BC - Blockchain
4. IoT - Internet of Things
5. DApp - Decentralized Application
6. UI - User Interface
7. UX - User Experience
8. API - Application Programming Interface
9. DAO - Decentralized Autonomous Organization
10. JSON - JavaScript Object Notation

List of Figures

1 Agile Model 19
2 Sprint Analysis 22
3 Scrum Meeting 23
4 Architecture diagram 24
5 ER Diagram 25
6 Class Diagram 27
7 Use Case Diagram 29
8 Deployment Diagram 31
9 Sequence Diagram 33
10 Budget 34
11 Risks and Analysis 38
12 Risks and Analysis 39
13 Network Diagram 42
14 Gannt Timeline 43
15 T.C. 1 53
16 T.C. 1 Passed 53
17 T.C. 1 Unsuccessful 54
18 T.C. 3 Passed 54
19 T.C. 3 Unsuccessful 55
20 T.C. 4 Passed 55
21 T.C. 4 Unsuccessful 56
22 T.C. 5 Passed 56
23 T.C. 5.1 Unsuccessful 57
24 Code Image 59
25 Landing Page 60
26 Login Page 60
27 Error Page 61
28 Bidder Registration Page 61

29 Government Officer Registration Page 62
30 Create Tender 62
31 My Tenders 63
32 Show all Bids 63
33 Make a bid 64
34 My Bids 64
35 Sign Transaction 65
36 Transaction Signed Successfully 65
37 Plagiarism 1 72
38 Plagiarism 2 73
39 Plagiarism 3 74
40 Plagiarism 4 75
41 Plagiarism 5 76

1 INTRODUCTION

1.1 Overview

A decentralized tender management system using blockchain is a cutting-edge platform that revolutionizes the traditional tendering process by leveraging the power of blockchain technology. It provides a transparent, secure, and efficient environment for tender issuers and bidders to engage in the tendering process. Through the use of blockchain's distributed ledger technology, all tender-related activities such as tender creation, bid submissions, and evaluation results are recorded in a tamper-proof and auditable manner. Smart contracts automate and enforce predefined rules, ensuring fairness and compliance. The decentralized nature of the system enhances trust, eliminates intermediaries, and reduces the risk of fraud. By offering transparency, security, and efficiency, a decentralized tender management system using blockchain has the potential to transform the way tenders are conducted, bringing numerous benefits to both issuers and bidders.

1.2 Motivation

The complete elimination of uncertainty is not possible; however, it can be mitigated or reduced. In situations where parties need to reach agreements, third-party institutions have historically played a crucial role in lowering uncertainty and fostering trust. For instance, when making a purchase on eBay, both the buyer and seller require a certain level of certainty regarding the transaction. In this case, they rely on eBay as a trusted intermediary to ensure a legitimate trade. However, trusting such "middleman" institutions often necessitates extensive research and knowledge. Blockchain technology emerged as a solution to address these challenges by providing decentralized and secure applications that offer a certain level of certainty. This aspect of blockchain played a significant role in its widespread adoption within a trustless society.

1.3 Problem Definition and Objectives

The absence of transparency in the tendering process and the malpractices in the assignment of tenders to agencies have resulted in inefficient road development.

Objectives: The objectives of a decentralized tender management system using blockchain can include:

Transparency: One of the primary goals is to enhance transparency in the tendering process. By leveraging blockchain's distributed ledger technology, all tender-related activities are recorded on an immutable and transparent ledger. This ensures that stakeholders have access to accurate and tamper-proof information, fostering trust and accountability.

Security: The system aims to improve the security of the tendering process. Blockchain technology provides cryptographic algorithms and consensus mechanisms that make the data stored on the blockchain tamper-resistant and secure. This reduces the risk of fraud, unauthorized modifications, and data breaches, ensuring the integrity of the tender process.

Efficiency: Decentralized tender management systems strive to increase the efficiency of the entire tendering process. By automating tasks and utilizing smart contracts, administrative overheads can be reduced, and manual errors can be minimized. This streamlines the process, saves time, and enables faster and smoother tender evaluations.

Trust and Fairness: Blockchain-based tender management systems aim to foster trust and ensure fairness in the tendering process. By eliminating the need for intermediaries and providing a transparent and auditable platform, the system reduces the potential for biased decisions and favoritism.

Cost Reduction: Another objective is to reduce costs associated with the tendering process. By eliminating manual paperwork, intermediaries, and the need for extensive audits, decentralized tender management systems can significantly re-

duce administrative costs. This benefits both tender issuers and bidders, making the process more cost-effective.

1.4 Project Scope and Limitations

The project scope of a decentralized tender management system using blockchain typically encompasses the following aspects:

Tender Creation and Publication: The system should allow tender issuers to create and publish tender requests on the blockchain platform. This includes defining tender details, requirements, timelines, and any necessary supporting documentation.

Bid Submission: The system should provide a mechanism for potential bidders to access published tenders and submit their bids electronically through the blockchain platform. This involves securely recording and time-stamping bid submissions on the blockchain.

Smart Contract Development: The implementation of smart contracts is crucial in automating various includes developing aspects of the tender process. This smart contracts to validate bid submissions, enforce eligibility criteria, and ensure compliance with tender requirements.

Bid Evaluation and Scoring: The system should facilitate the evaluation of bids based on predefined criteria. It may involve the development of algorithms or mechanisms to calculate scores and rank the bids objectively. The evaluation results should be recorded on the blockchain for transparency and auditability.

Security and Authentication: Ensuring the security of the system is essential. The project scope should include the implementation of robust security measures, such as cryptographic algorithms, to protect sensitive data, prevent unauthorized access, and validate user authentication.

Supplier Management: The system may incorporate functionalities for supplier management. This includes maintaining a decentralized database of registered suppliers, storing their information, and tracking their performance records. It

should also provide mechanisms for selecting qualified bidders for future tenders based on historical data.

Limitations: While decentralized tender management systems using blockchain offer numerous benefits, they also have some limitations that need to be considered:

Scalability: Blockchain technology, especially in its current state, may face scalability challenges. As the number of participants and transactions increases, the performance of the blockchain network may degrade, leading to slower transaction processing times and higher costs. This can potentially limit the scalability of the system, especially in scenarios involving a large number of tenders and participants.

Adoption and Integration: Implementing a decentralized tender management system using blockchain requires widespread adoption and integration among stakeholders. This can be challenging, as it requires buy-in from tender issuers, bidders, and other involved parties. Integration with existing systems and processes may also pose technical challenges and require additional effort and resources.

Regulatory and Legal Considerations: Blockchain technology is still evolving, and there may be regulatory and legal considerations that need to be addressed. Tender management processes often involve compliance with specific regulations and legal frameworks. Adapting blockchain-based systems to meet these requirements and ensuring their acceptance within the existing legal framework can be a complex task.

Technical Expertise: Developing and maintaining a decentralized tender management system using blockchain requires technical expertise in blockchain development, cryptography, and smart contract programming. Finding skilled professionals with expertise in these areas can be a challenge, leading to increased development costs and potential delays.

Privacy Concerns: While blockchain offers transparency, it may also present challenges in terms of privacy. Tender-related information recorded on the blockchain

is visible to all participants, which may not be desirable in certain cases where confidentiality is required. Balancing transparency with the need for privacy can be a challenge, especially when sensitive information is involved.

Energy Consumption: Blockchain networks, particularly those utilizing proof-of-work consensus algorithms, can be energy-intensive. The process of validating and adding transactions to the blockchain requires significant computational power, resulting in higher energy consumption. This aspect raises concerns about the environmental impact and sustainability of blockchain-based systems.

Immutability and Error Correction: The immutability of data stored on the blockchain, while a key feature for trust and transparency, can also pose challenges. Once data is recorded on the blockchain, it becomes extremely difficult to modify or correct any errors or inaccuracies. In the context of tender management, where errors or discrepancies may occur, mechanisms for handling and rectifying such situations need to be carefully designed.

1.5 Methodologies of problem solving

When it comes to problem-solving in a decentralized tender management system using blockchain, several methodologies can be employed. Here are some commonly used approaches:

Requirements Analysis: Conduct a thorough analysis of the requirements and challenges associated with the tender management process. This involves identifying pain points, understanding stakeholder needs, and defining clear objectives for the blockchain-based solution.

Research and Exploration: Conduct research on existing blockchain technologies and solutions in the tender management domain. Explore various decentralized architectures, consensus mechanisms, and smart contract frameworks to determine the most suitable approach for addressing the identified problems.

Prototyping and Iterative Development: Begin with prototyping and iterative development to validate the feasibility and effectiveness of the proposed solution.

Build small-scale prototypes or proof-of-concepts to test specific functionalities and gather feedback from stakeholders. This iterative approach allows for incremental improvements and course corrections as needed.

Collaboration and Stakeholder Engagement: Involve relevant stakeholders, including tender issuers, bidders, and regulatory bodies, in the problem-solving process. Seek their input, gather feedback, and encourage collaboration to ensure that the solution aligns with their needs and addresses their concerns.

User-Centered Design: Adopt a user-centered design approach to ensure the system is intuitive and user-friendly. Engage with end-users during the design and development stages, conduct usability tests, and incorporate user feedback to create an optimized user experience.

Agile Project Management: Implement agile project management methodologies to facilitate flexibility and adaptability. Break down the development process into smaller, manageable tasks, set clear priorities, and regularly review progress. This iterative approach allows for continuous improvement and facilitates the timely identification and resolution of any issues that arise.

Testing and Quality Assurance: Establish comprehensive testing and quality assurance processes to ensure the reliability, functionality, and security of the decentralized tender management system. Conduct unit testing, integration testing, and system testing to identify and address any bugs or vulnerabilities. Implement rigorous security measures, including vulnerability assessments and penetration testing, to safeguard the system against potential threats.

Continuous Monitoring and Feedback: Implement mechanisms for continuous monitoring and feedback collection once the decentralized tender management system is deployed. Monitor the system's performance, gather feedback from users, and actively address any issues or concerns that arise. Regularly update and enhance the system based on the evolving needs and requirements of the stakeholders.

By applying these problem-solving methodologies, the development and implementation of a decentralized tender management system using blockchain can be

guided effectively, ensuring that the system addresses the identified challenges and delivers the desired outcomes.

2 LITERATURE SURVEY

Public procurement processes often lack transparency and can be highly complex. They encompass all purchases of goods and services by public institutions within a country, including contracts with the private sector in various sectors such as healthcare, military, and construction. Government bodies have floated tenders worth substantial amounts, such as INR 82,433 crores in 2012–13, INR 189,279 crores in 2013–14, INR 212,735.5 crores in 2014–15, INR 404,176.6 crores in 2015–16, and INR 543,820.5 crores in 2016–17. In total, an enormous sum of INR 14 lakh crores has been spent in the last five years, with suppliers being selected through the tender process [1] . This constitutes approximately 30 percent of India's annual GDP. The government has come to realize that suppliers in the country are often acquainted with each other and with officials, making it easier for them to exploit weaknesses in the system. Consequently, instances of misconduct, such as large-scale scams, suspicious modifications of tender requirements and deadlines, and the selection of undeserving bidders, have occurred. Unfortunately, those who raise their voices against such misconduct often face severe consequences [2] . Given that procurement accounts for a significant portion of public resources, it is crucial that the tender process is conducted with accountability, transparency, and effective management. [3]. Traditional e-tendering, while efficient, has faced issues of trust and centralization. This research proposes a permissioned blockchain-based solution to make the e-tendering process more secure, transparent, and fair. By employing a blockchain framework, the government can invite suggestions from citizens to improve the solutions and services. This proposal would allow citizens to actively participate, track funds, and give ideas and suggestions to solve government's posted problems. The system would credit the contributions fairly and transparently.

[4]. Blockchain based smart contracts are computer programs that encode an agreement between non-trusting participants. Smart contracts are executed on a blockchain system if specified conditions are met, without the need of a trusted third party. Blockchains and smart contracts have received increasing and booming attention in recent years, also in academic circles. We carry out a systematic

mapping study of all peer-reviewed technology-oriented research in smart contracts. Our interest is twofold, namely to provide a survey of the scientific literature and to identify academic research trends and uptake. We only focus on peer-reviewed scientific publications, in order to identify how academic researchers have taken up smart contract technologies and established scientific outputs. We obtained all research papers from the main scientific databases, and using the systematic mapping method arrived at 188 relevant papers. We classified these papers into six categories, namely, security, privacy, software engineering, application, performance & scalability and other smart contract related topics. We found that the majority of the papers falls into the applications (about 64%) and software engineering (21%) categories. Compared to our 2017 survey [1], we observe that the number of relevant articles has increased about eightfold and shifted considerably towards applications of smart contracts.

The proposed framework could be applied to initiatives like the Members of Parliament Local Area Development Scheme (MPLADS) in India. The goal is to achieve a fair, transparent e-tendering system that reduces biases and corruption, thereby fostering trust between government and citizens.

3 SOFTWARE REQUIREMENT SPECIFICATION

3.1 Assumptions and Dependencies

3.1.1 Assumptions

a. Participants' Understanding of Blockchain: It is assumed that the participants involved in the tender management system have a basic understanding of blockchain technology and its principles.

b. Availability of Internet Connectivity: The system assumes that all participants have reliable access to the internet for seamless communication and interaction with the blockchain network.

c. Adequate Infrastructure: It is assumed that the necessary hardware and software infrastructure, including computers, servers, and blockchain nodes, are in place to support the decentralized tender management system.

d. Compliance with Legal and Regulatory Frameworks: The system assumes that all tender processes and activities comply with relevant legal and regulatory frameworks governing public procurement in the specific jurisdiction.

3.1.2 Dependencies

a. Blockchain Platform and Framework: The system depends on the selection and availability of a suitable blockchain platform and framework, such as Ethereum, Hyperledger Fabric, or Corda, to build and deploy the decentralized tender management system.

b. Smart Contract Development: The system relies on the development and implementation of smart contracts to automate and enforce the tender-related processes and rules on the blockchain network.

c. Integration with External Systems: If required, the system may depend on the integration with external systems, such as identity verification services, payment

gateways, or external data sources, to enhance the functionality and usability of the tender management system.

d. Blockchain Network Consensus: The system's effectiveness depends on the consensus mechanism employed by the blockchain network, ensuring agreement and validation of transactions and data across the network participants.

e. User Adoption and Participation: The success of the system depends on the active participation and adoption by tender issuers, bidders, government officers, and other stakeholders in the tender management process.

f. Security Measures: The system's security and integrity depend on the implementation of appropriate security measures, including encryption, access controls, and auditing, to protect sensitive data and prevent unauthorized access or tampering.

g. Scalability and Performance: The system's performance and scalability depend on the capacity and capability of the underlying blockchain network to handle a growing number of tenders, participants, and transactions without compromising efficiency and responsiveness.

h. Regulatory Support and Acceptance: The system's successful implementation and adoption may depend on the regulatory support and acceptance of blockchain technology and its application in the public procurement domain.

3.2 Functional Requirements

3.2.1 User Registration and Authentication

The system should allow users to register and create accounts with appropriate credentials. Users should be able to authenticate their identities securely using cryptographic techniques. User authentication should be decentralized and based on public-private key pairs.

3.2.2 Tender Creation and Publication

The system should enable tender issuers to create and publish tenders on the blockchain. Tender issuers should be able to provide details such as tender title, description, requirements, and submission deadline. Tenders should be stored on the blockchain to ensure immutability and transparency.

3.2.3 Bid Submission

Registered bidders should be able to submit their bids for the published tenders. Bidders should provide relevant information, including bid price, proposal documents, and any additional requirements specified in the tender.

Bids should be securely stored on the blockchain for transparency and auditability.

3.2.4 Bid Evaluation and Selection

The system should support the evaluation and selection of bids in a transparent and automated manner [3] . Evaluation criteria and weights should be predefined and disclosed to bidders. Smart contracts should execute the evaluation process based on the specified criteria and select the winning bid(s).

3.2.5 Contract Generation and Execution

The system should facilitate the generation of contracts based on the selected bids. Smart contracts should be utilized to automate contract execution and enforce the terms and conditions agreed upon by the parties. Contracts should be stored on the blockchain for transparency and immutability.

3.2.6 Transparency and Auditability

The system should ensure transparency throughout the tendering process, allowing all stakeholders to access and verify tender-related information stored on the

blockchain. The system should maintain an immutable record of all tender activities, including tender creation, bid submission, evaluation, and contract awarding.

3.2.7 Notifications and Communication

Email Notifications: The system should be able to send email notifications to users, informing them about tender updates, bid statuses, evaluation results, and other relevant information. Messaging System: Users should have a means of communication within the system to facilitate discussions, inquiries, and clarifications regarding tender-related activities.

3.3 External Interface Requirements

3.3.1 User Interfaces:

User Registration:

The system should provide a user-friendly interface for users to register their accounts, providing necessary information and authentication credentials. Tender Creation and Publication: Users should have an intuitive interface to create and publish tenders, including details such as title, description, requirements, and submission deadlines. Bid Submission: Bidders should have a straightforward interface to submit their bids, including bid price, proposal documents, and any additional requirements specified in the tender. Bid Evaluation and Selection: government officers should have a user interface to evaluate and select bids based on predefined criteria and weights.

3.3.2 Blockchain Integration

Wallet Integration:
Users should have the ability to connect their digital wallets to the system to facilitate secure transactions and fund management.
Smart Contract Integration:
The system should integrate with the blockchain network and enable the exe-

cution of smart contracts for automated tender processes, bid evaluations, and contract generation.

3.3.3 Communication and Notifications

Email Notifications:
The system should be able to send email notifications to users, informing them about tender updates, bid statuses, evaluation results, and other relevant information.

Messaging System:
Users should have a means of communication within the system to facilitate discussions, inquiries, and clarifications regarding tender-related activities.

3.4 Nonfunctional Requirements

3.4.1 Performance Requirements

Response Time: The system should provide prompt responses to user interactions and transactions, ensuring a seamless user experience. Throughput: The system should handle a significant number of concurrent users and transactions efficiently without performance degradation.

Scalability: The system should be able to scale horizontally or vertically to accommodate increased user and transaction loads.

3.4.2 Security Requirements

Security: Data Confidentiality: The system should ensure the confidentiality of sensitive data, such as bidder information and contract details, preventing unauthorized access or disclosure.

Authentication and Authorization: The system should employ robust authentication mechanisms to verify the identities of users and authorize their access to appropriate functionalities. Data Integrity: The system should protect data stored

on the blockchain from unauthorized modifications or tampering, ensuring its integrity and immutability.

3.4.3 Software Quality Attributes

Reliability: Availability: The system should be available and accessible to users for the majority of the time, minimizing downtime due to maintenance or system failures. Fault Tolerance: The system should continue to operate properly even in the presence of individual node failures or disruptions within the blockchain network. Disaster Recovery: The system should have measures in place to recover data and restore functionality in the event of unexpected disasters or system failures.

Usability: User Interface: The system should have an intuitive and user-friendly interface, enabling users to easily navigate and perform tasks without significant training. Accessibility: The system should comply with accessibility standards, ensuring that users with disabilities can access and use the system effectively. Multilingual Support: The system should support multiple languages to cater to users from diverse linguistic backgrounds.

Compliance: Regulatory Compliance: The system should adhere to relevant legal and regulatory requirements governing public procurement and blockchain technology. Auditability: The system should provide an audit trail of actions and transactions, allowing for traceability, transparency, and accountability in the tender management process.

Interoperability: Integration: The system should be able to integrate with other systems or external services, such as payment gateways or document management systems, to facilitate data exchange and enhance functionality. Interchangeability: The system should be designed to be adaptable to different blockchain platforms or frameworks, ensuring flexibility and future-proofing.

3.5 System Requirements

3.5.1 Database Requirements

Blockchain Integration: The system should integrate with the blockchain network and utilize the distributed ledger as a primary source of data storage for transactional information, bid details, and contract records. The database should be capable of storing and retrieving data from the blockchain network efficiently. Selection of appropriate databases such as MySQL, PostgreSQL, or NoSQL databases depending on the system's requirements. Tender Data Management: The database should store tender-related information such as tender title, description, requirements, submission deadlines, and other relevant details. It should provide efficient search and retrieval mechanisms to retrieve tenders based on various parameters, such as date, category, or keyword.

Bid Data Management: The database should store bid-related information, including bid price, proposal documents, bidder details, and any additional information provided by the bidders. It should support efficient indexing and querying to retrieve bids based on tender ID, bidder ID, or other relevant criteria.

Contract Data Management: The database should store contract details, including contract terms, parties involved, payment terms, and milestones. It should support data retrieval and querying to retrieve contracts based on tender ID, contractor ID, or other relevant parameters.

User Management: The database should store user information, including user profiles, authentication credentials, and access permissions. It should support efficient retrieval and management of user data, including user registration, login, and profile updates.

3.5.2 Software Requirements(Platform Choice)

Operating System: The system should be compatible with popular operating systems such as Windows, Linux, or macOS.

Blockchain Platform: Selection of a suitable blockchain platform such as Ethereum,

Hyperledger Fabric, or Corda, depending on the specific requirements of the tender management system.

Development Framework: The system may require the use of development frameworks like Solidity for Ethereum-based systems or other programming languages and frameworks specific to the chosen blockchain platform.

Smart Contract Development: Tools and libraries for developing and deploying smart contracts, including contract writing languages, integrated development environments (IDEs), and testing frameworks.

User Interface: Design and development of a user-friendly and intuitive interface for various user roles, including tender creators, bidders, government officers, and administrators. Technologies such as HTML, CSS, and JavaScript for web-based interfaces or platform-specific frameworks for mobile applications.

Authentication and Authorization: Implementation of authentication mechanisms for user login and access control to ensure secure and authorized access to the system. Integration with identity management systems or frameworks to handle user registration, authentication, and authorization processes.

3.5.3 Hardware Requirements

Servers: High-performance servers to host the blockchain network nodes, including validator nodes and consensus nodes. Sufficient processing power, memory, and storage capacity to handle the computational demands of the blockchain network.

Network Infrastructure: Reliable and high-bandwidth network infrastructure to support communication between the blockchain nodes and system users. Network switches, routers, and firewalls to ensure secure and uninterrupted data transmission.

Storage: Adequate storage capacity to store the blockchain data, including the distributed ledger and associated transaction records. Depending on the blockchain platform, the storage requirements may vary. For example, Ethereum-based sys-

tems require significant storage space due to the size of the blockchain.

Backup and Recovery: Backup devices or systems to regularly back up the blockchain data and ensure data integrity and availability. Redundancy measures to protect against hardware failures and enable quick recovery in case of system disruptions.

Load Balancers: Load balancing hardware to distribute network traffic efficiently and ensure optimal performance and availability of the system.

Security Measures: Hardware security modules (HSM) to provide additional security for cryptographic operations and protect private keys.

3.6 Analysis Models: SDLC Model to be applied

3.6.1 Agile Model

Here's how we applied the Agile model to the SDLC of a decentralized tender management system using blockchain [4] :

Application Backlogs: The project team, including developers, stakeholders, and users, collaboratively defines the project vision, goals, and scope. They identify the key features and prioritize them based on their importance and value to the stakeholders.

Sprint Planning: The project is divided into a series of time-boxed iterations called sprints. In each sprint, a set of user stories or requirements is selected from the product backlog. The development team estimates the effort required for each user story and commits to completing a set of them within the sprint.

Development: The development team works on implementing the selected user stories, including the blockchain components and the application layer. They follow best practices for blockchain development, such as writing smart contracts, implementing features, and integrating with the blockchain network. The development tasks are broken down into smaller tasks and assigned to individual team members.

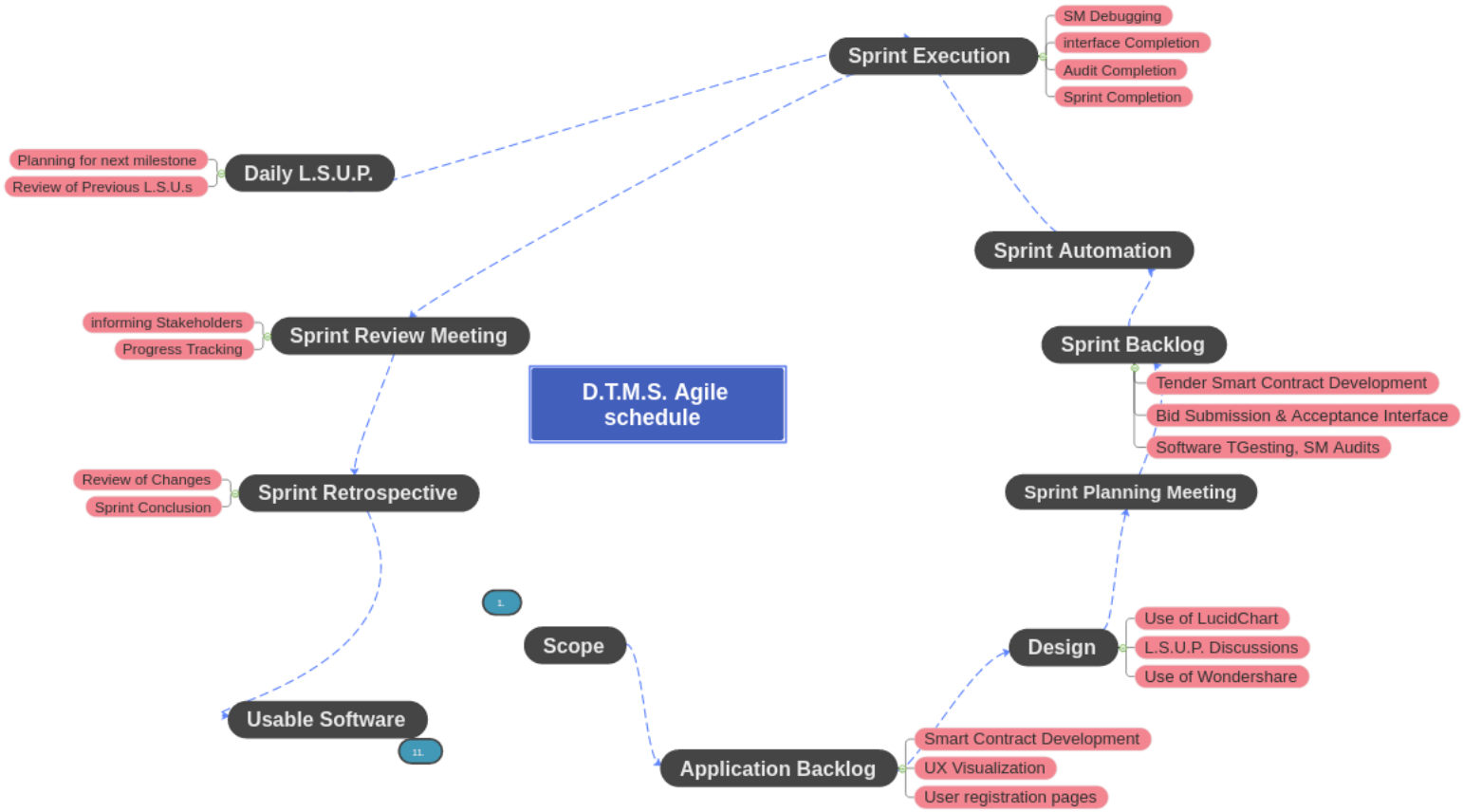

Figure 1: Agile Model

Daily Stand-up Meetings: The development team holds short daily meetings to discuss progress, challenges, and plans. Each team member provides an update on their tasks, identifies any obstacles, and collaborates on resolving them. This ensures that everyone is aligned and any issues are addressed promptly.

Continuous Integration and Testing: As the development progresses, the code is continuously integrated and tested to ensure functionality, quality, and adherence to requirements. Automated testing is used to validate the correctness of smart contracts, perform unit testing, integration testing, and system testing. Regular feedback is gathered from stakeholders and users to refine and improve the system.

Sprint Review and Demo: At the end of each sprint, a review and demo session is conducted to showcase the completed functionality to stakeholders and gather their feedback. This allows for early validation, course correction, and ensures that the system is meeting the desired objectives.

Sprint Retrospective: The team reflects on the sprint, discussing what went well, what could be improved, and any lessons learned. This feedback is used to refine processes, address challenges, and enhance teamwork and productivity.

Release and Deployment: Once the desired set of features is implemented, tested, and reviewed, a release is prepared for deployment. The deployment process involves configuring the blockchain network, deploying smart contracts, and launching the application layer in the production environment.

Ongoing Maintenance and Enhancements: After deployment, the system is maintained and supported, with regular updates, bug fixes, and enhancements based on user feedback and evolving requirements.

Application Backlog:

Smart Contract Development: Tasks related to the creation of smart contracts that will govern the tender process. This could include contracts for tender creation, bid submission, bid evaluation, and contract awarding.

UX Visualization: The UX design for a decentralized application (DApp) like this needs to consider not only the traditional principles of good UX design—simplicity, usability, and consistency—but also the unique characteristics of blockchain technology. For example, transaction confirmation times, gas fees, and wallet management are all aspects of the blockchain user experience that might be unfamiliar to users accustomed to traditional web applications.

Here are a few elements that could be included in the UX visualization for this project:

User Journey: These would outline the path that different types of users (e.g., tender issuers, bidders) take when interacting with the system. For example, the journey for a bidder includes discovering a tender, evaluating it, preparing a bid,

submitting the bid, and receiving the outcome.

Mockups and Prototypes: These are more detailed and interactive representations of the DApp. They could show the look and feel of the system, including colors, typography, and graphics. Prototypes could also demonstrate the interaction flow, allowing users to click through the different steps of submitting a bid, for example.

User Feedback Loops: Part of UX visualization could involve the design of mechanisms to collect user feedback. This could include surveys, feedback forms, or user testing sessions. Collecting and incorporating user feedback is crucial for improving the user experience.

Sprint Backlog:

In a Decentralized Tender Management System project, the Sprint Backlog is a subset of the Product Backlog items that the team commits to deliver during a particular Sprint. The exact contents of the Sprint Backlog will depend on the team's capacity, the priorities set by the Product Owner, and the overall progress of the project.

Develop Tender Creation Smart Contract: A task to develop the smart contract that will be used to create new tenders in the system.

Create Bid Submission User Interface: A task to create the part of the DApp that bidders will use to submit their bids.

Smart Contract Audits.

Sprint Execution:

In this stage, the tasks mentioned in the Sprint Backlog are executed.

Final Platform:

In this stage, final application is ready for use

Feature	Initial Estimat	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Hours Left
Categories	60	20	8	5	1	5	10	0	1	10
Synchronization	60	10	5	2	2	5	10	19	2	5
Accounts	60	5	8	2	10	5	10	5	10	5
Reminders	60	10	12	2	3	5	10	2	10	6

Setting	Start	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8
Planned Hours		30	30	30	30	30	30	30	30
Actual Hours		45	33	11	16	20	40	26	23
Remaining Effort	240	195	162	151	135	115	75	49	26
Ideal Burn down	240	210	180	150	120	90	60	30	0

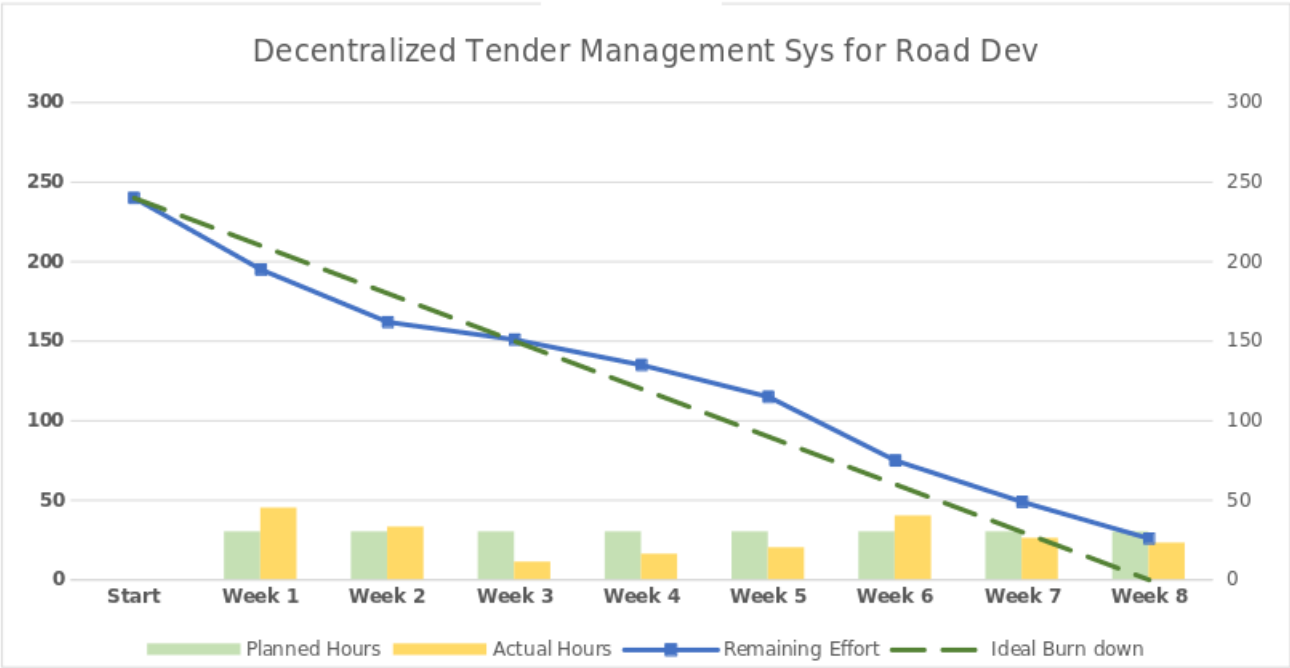

Figure 2: Sprint Analysis

INCUGLOBE SOLUTIONS LTD.

PROJECT GROUP 1- SCRUM S.U.P. REPORT

Topic: Decentralized Tender Management System for Road Development

DURATION	8 Months	TIME	Variable	FACILITATOR	Mr. Rohan Patel

GROUP MEMBERS			
Shankar Avad	Atharv Vaspate	Aniket Datar	Niranjan Patankar
Role –Design & Doc	Role – Back-end & Doc	Role-Blockchain Dev & Doc	Role- Design & Reporting

DAY DATE	T.O.D.	R.A.N.	Remarks
E- 12:30	Commencement, L.S.U.P. Reporting	Niranjan Patankar	Satisfactory
M-13:30	Development Milestone CPC	Aniket Datar	Satisfactory
M-14:00	Continuous Survey and Market Research	Shankar Avad	Satisfactory
E-14:30	Design & Analysis Study of Existing Systems	Members	Satisfactory
W-15:00	Feature Decisions & Drafting A/U/D	Members	Satisfactory
E-10:00, 17:00	Daily Report on Project Progress	Flora Berggren	Satisfactory
University Hours	Break	-	-
V-	New Ideation A. Existing Business Stakeholders meet B. Technical Expert Sessions C. Integration of approved suggestions	Members	Satisfactory
W	Committee Reports A. Design Report B. Testing & Q.A. Report C. Stakeholders suggestions reporting D. University remarks	Atharv Vaspate	Satisfactory

Agile- iLogs Report Generated by ScrumFA Master 127.0.0.3 on 03-05-2023. All Rights Reserved @IncuGlobe Solutions Pvt. Ltd.

Figure 3: Scrum Meeting

4 SYSTEM DESIGN

4.1 System Architecture

A system architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system.

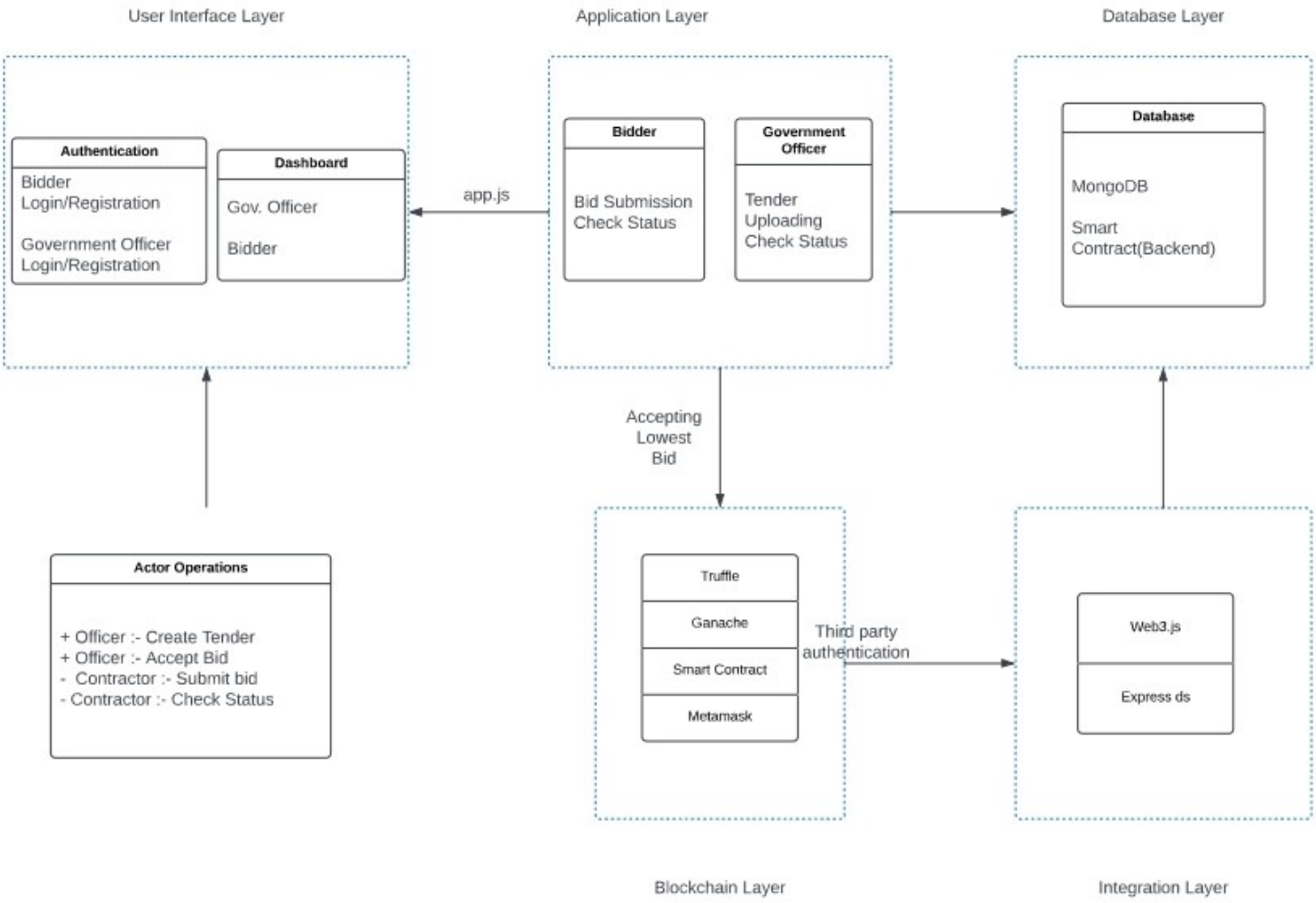

Figure 4: Architecture diagram

4.2 Entity Relationship Diagrams

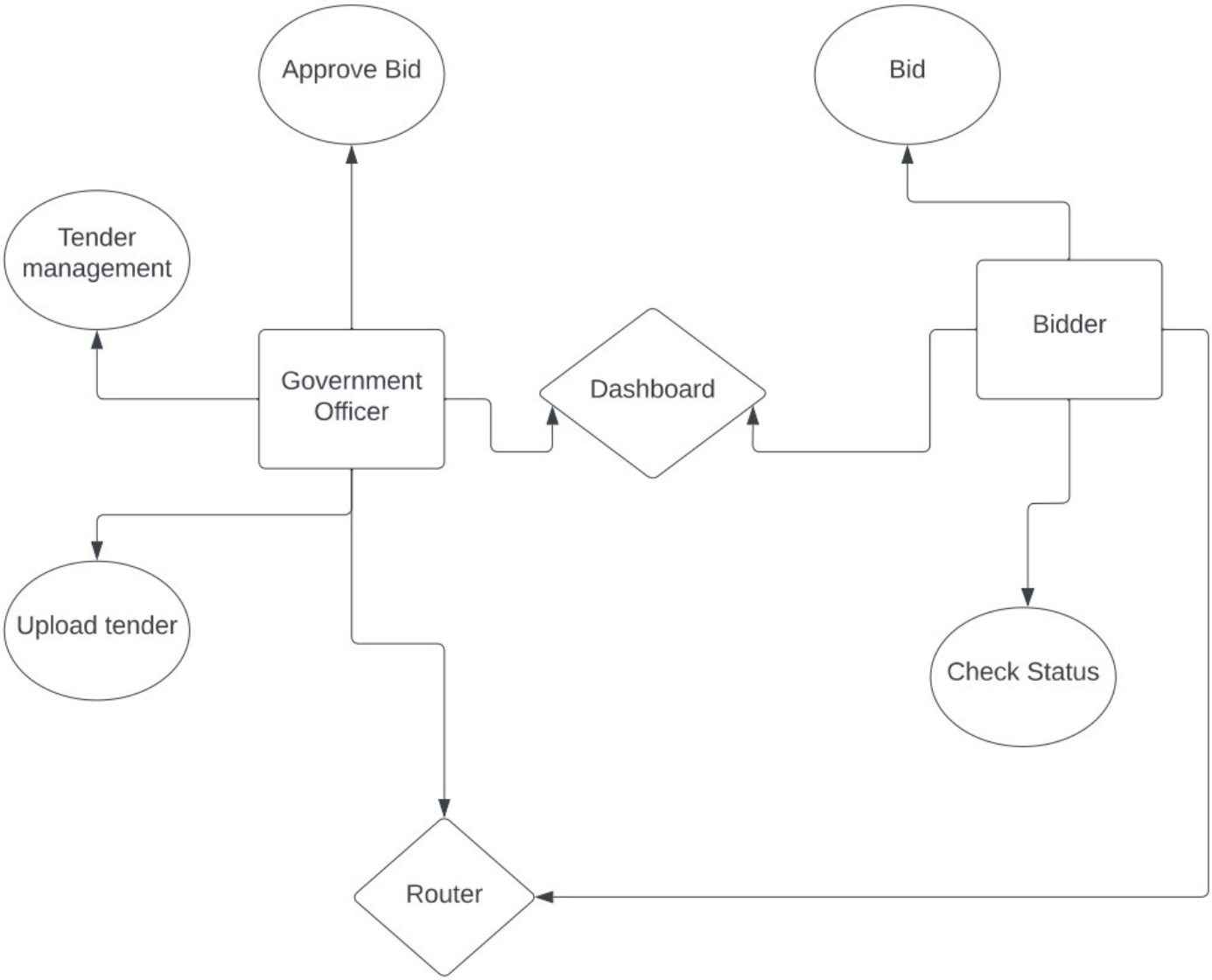

Figure 5: ER Diagram

4.3 UML Diagrams

Software industry uses some well-defined process model for developing software. This model is called Software Engineering Process(SEP) or Software Development Process (SDP). The unified Software Development Process (USDP) is a popular, widely used industry standard SDP. In short it is also called as Unified Process (UP). Unified Process co-exists with Unified Modelling Language (UML) which is a virtual language part of the system or project, UP addresses the process part of software development. UML is standard modeling language and notation for object oriented analysis and design. These notations are used to visualize the design of system and widely accepted among the software development organizations.

4.3.1 Class Diagram

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

The purpose of class diagram is to model the static view of an application. Class diagrams are the only diagrams which can be directly mapped with object-oriented languages and thus widely used at the time of construction.

UML diagrams like activity diagram, sequence diagram can only give the sequence flow of the application, however class diagram is a bit different. It is the most popular UML diagram in the coder community.

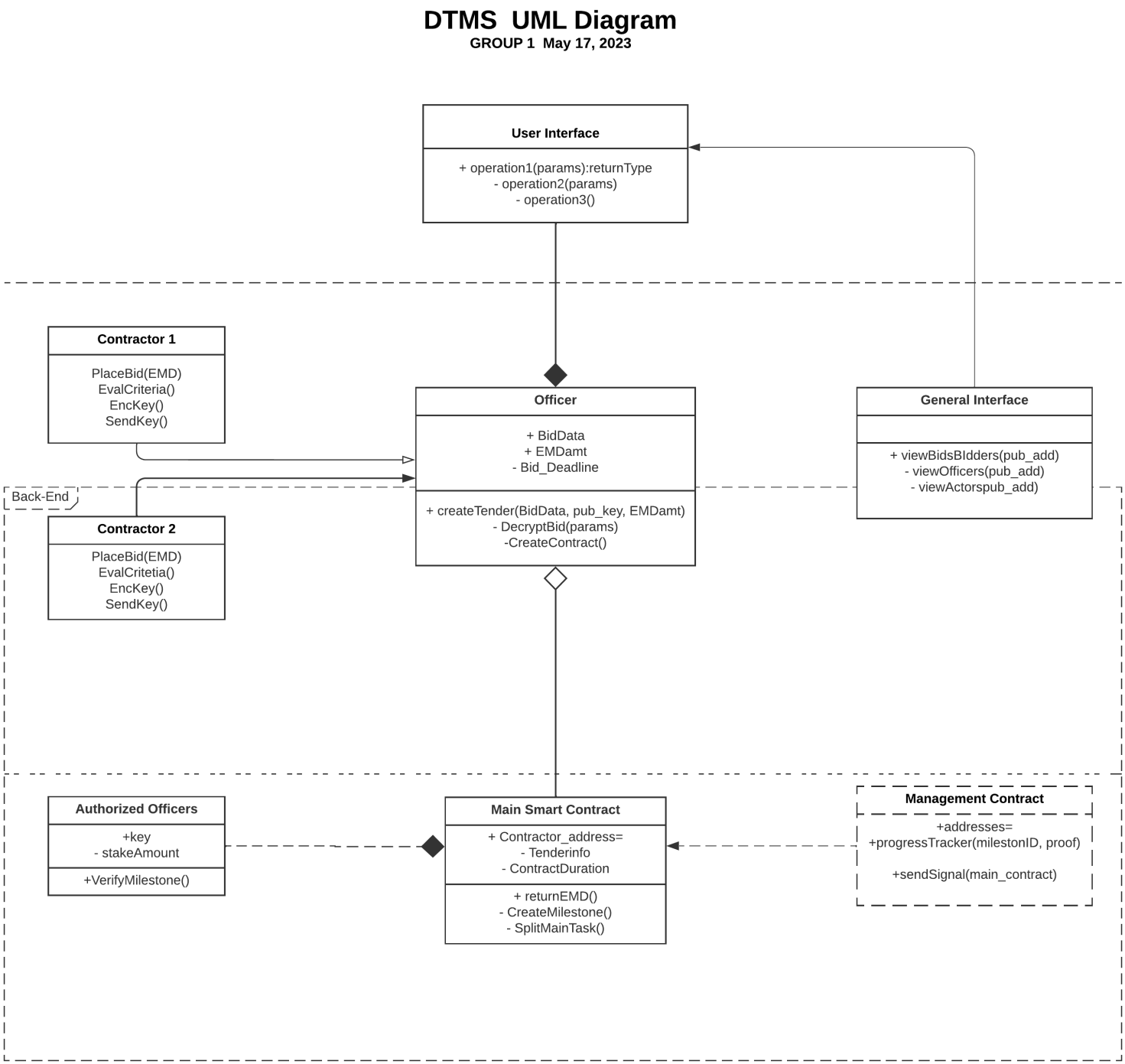

Figure 6: Class Diagram

4.3.2 Use Case

Use case diagram represents the interaction of the user with the system. It is represented as a sequence of interaction between actor and the system. Use cases are described in texts as well as in diagrams. The system is depicted as it is visible to the user. Internal working and processing of the system is not highlighted in use cases.

Use case diagram shows three aspects of the system:

- Actors: Actors are represented by arranging sticks in human shape. But not necessarily it represents the human user. It can be used to represent user, other system or devices that interacts with particular high level functionality (use case) of the system.
- Use case: Use case basically represents high level functionality of the system. Use case is represented by an ellipse.
- System boundary: Use cases are packed inside a box called the system boundary. Actors are outside the boundary. System boundary is used to show the enclosure of system functionality.

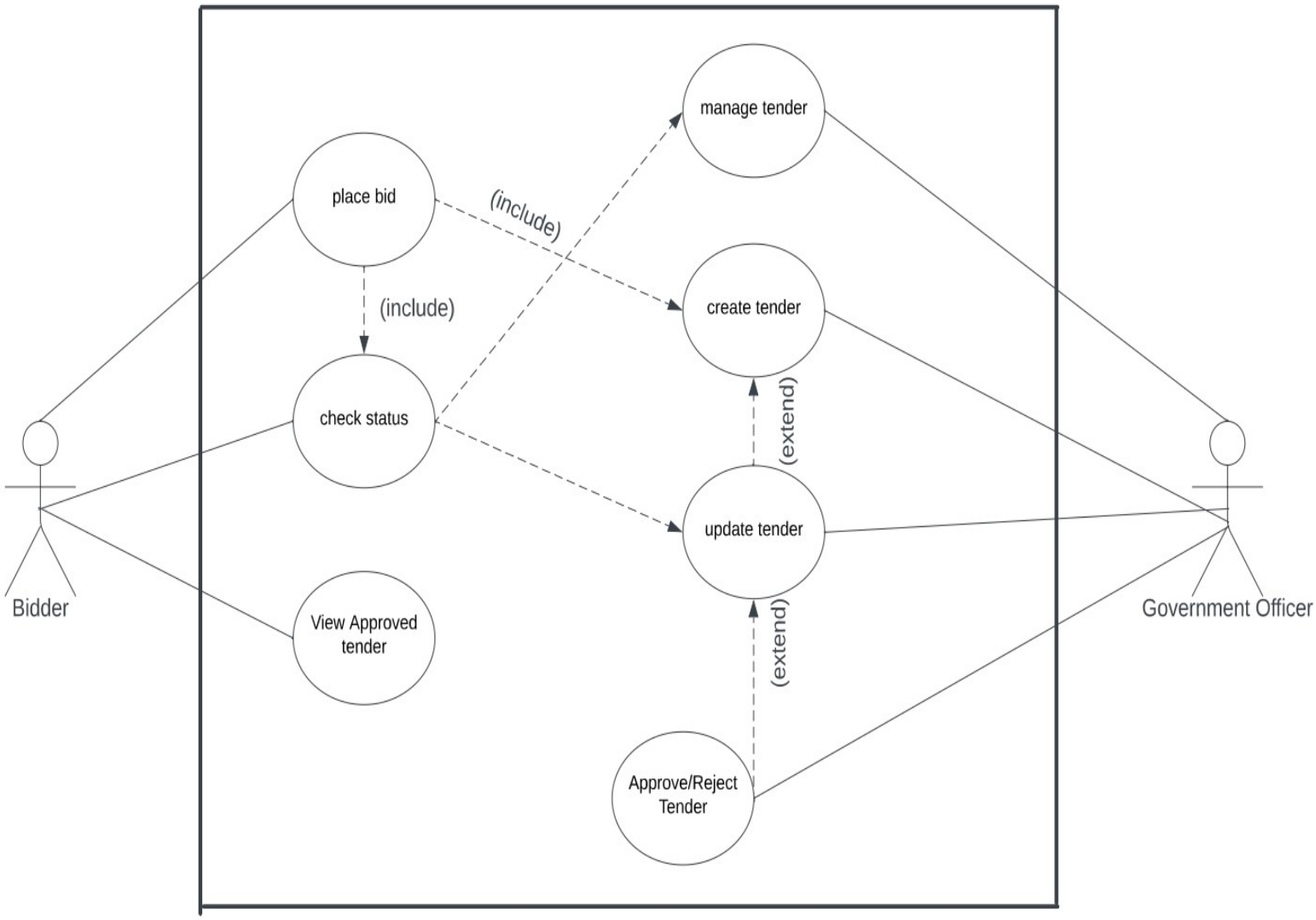

Figure 7: Use Case Diagram

4.3.3 Deployment Diagram

Deployment diagrams are used to visualize the topology of the physical components of a system, where the software components are deployed.

Deployment diagrams are used to describe the static deployment view of a system. Deployment diagrams consist of nodes and their relationships.

The purpose of deployment diagrams can be described as :

- Visualize the hardware topology of a system.
- Describe the hardware components used to deploy software components.
- Describe the runtime processing nodes.

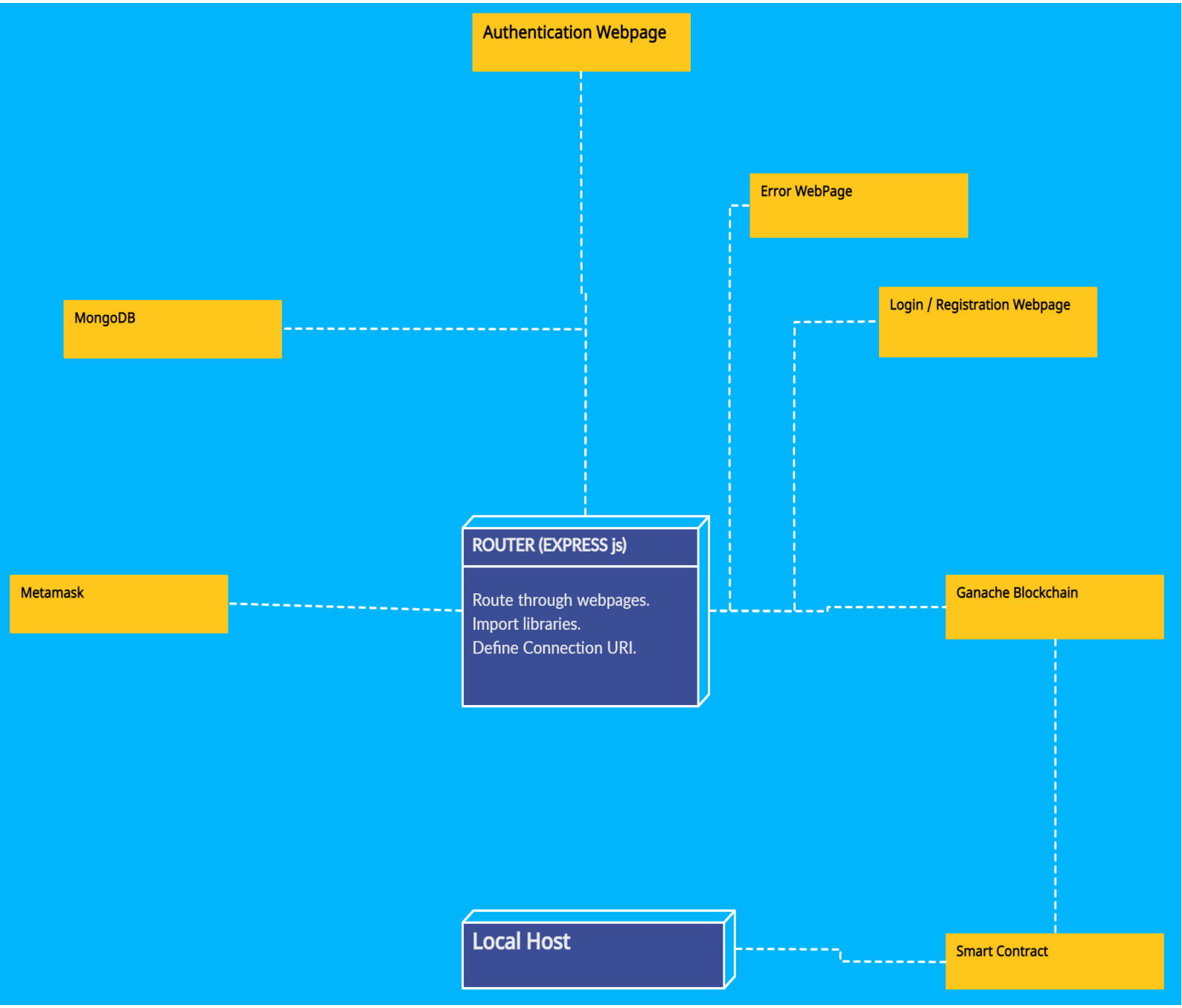

Figure 8: Deployment Diagram

4.3.4 Sequence Diagram

A UML Sequence diagram shows object interactions arranged in time sequence, which means it depicts the order of operations in a system. When it comes to the Decentralized Tender Management System for Road Development, here are potential components of a sequence diagram:

Actors: These are the external entities that interact with the system. They can be a person, a system, or an external device. In this case, the actors might be the Government Agency (or Officer), Contractor (or Bidder), and possibly other entities like an Audit or Oversight Committee.

Objects: These are the instances of classes within the system. Objects are depicted as rectangles with the object name (and optionally the class name) inside. The objects could represent the software modules or interfaces that help the actors interact with the system, such as a Tender Management Interface, Bid Evaluation Module, or Notification System.

Lifelines: These are vertical dashed lines that indicate the life of an object during the interaction. Each actor and object will have a lifeline.

Messages: These are horizontal arrows from one lifeline (object or actor) to another. They represent communication between objects. Examples of messages in this case could be ”CreateTender”, ”SubmitBid”, ”EvaluateBid”, ”UpdateBidStatus”, etc.

Activation boxes: These are thin rectangles on a lifeline representing the period during which an object is performing an action, i.e., when a method of the object is executing.

Return messages: These are dashed lines with an arrowhead pointing towards the actor or object that initiated the message. It represents the response of a message.

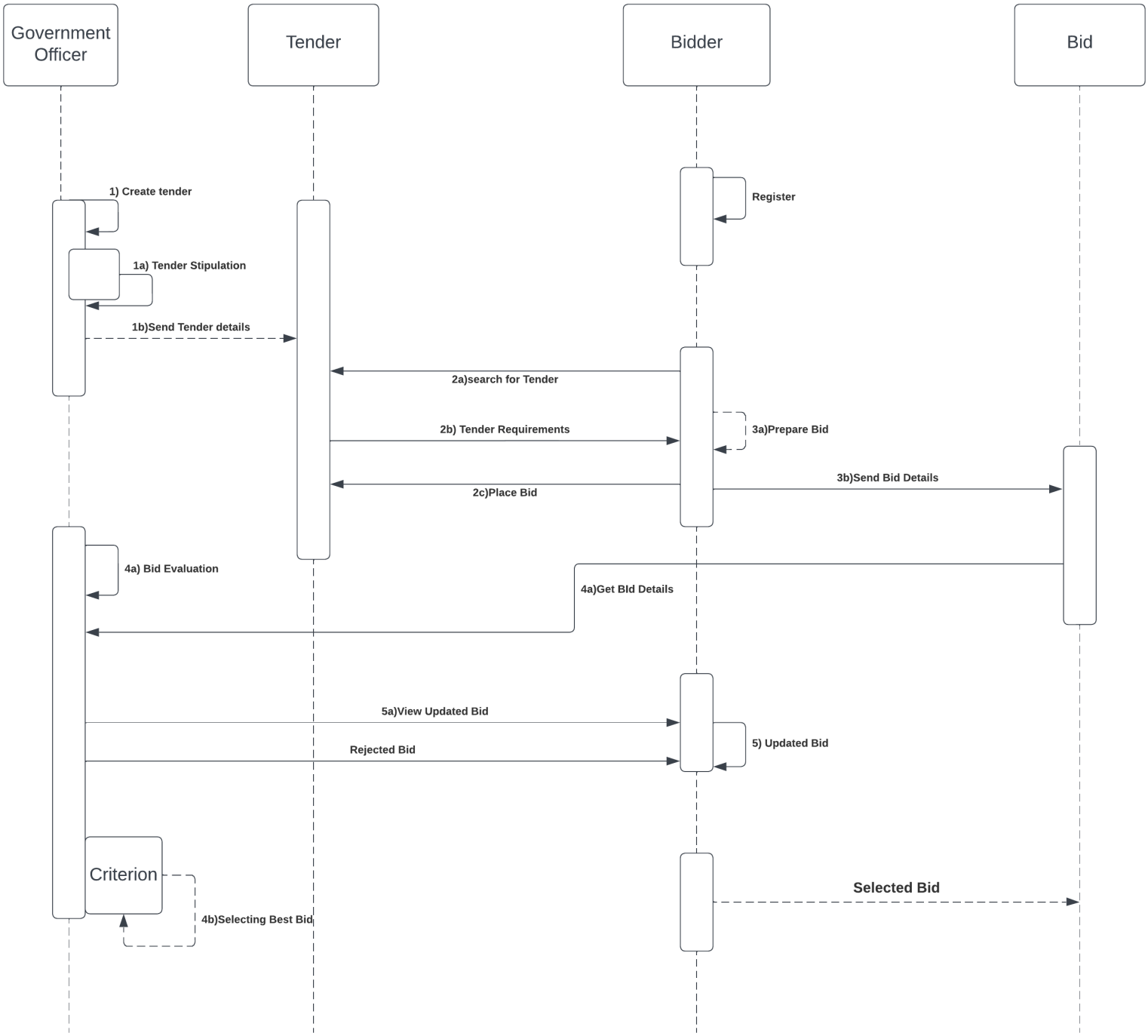

Figure 9: Sequence Diagram

5 Project Plan

5.1 Project Estimate

Cost estimation for a Decentralized Tender Management System project involves calculating the total costs associated with all aspects of the project, including planning, design, development, testing, deployment, and maintenance. Several factors can influence the cost of such a project.

5.1.1 Reconciled Estimates

The cost structure for the complete project as above

Decentralized Tender Management System for Road Development

INCUGLOBE

Group Members
1. Shankar Avad
2. Atharv Vaspate
3. Aniket Datar
4. Niranjan Patankar

Project Budget
Project Lead: Mr. Rohan Patel
Start Date: 1/6/2022

Budget Summary

Budget	Actual	Under(Over)
$ 3,740	$ 2,070	$ 1,670

Tasks	Hrs	Internet Nil	Nil	Nil	Resources	Other	Fixed Costs Operations	Budget	Actual	Under(Over)
Initial Discovery								$ 2,950	$ 1,220	$ 1,730
Topic Finalization	180	$0.00					-	200.00	-	200.00
Confirmation						-	-	400.00	-	400.00
Literature Survey					-	-	250.00	1,200.00	250.00	950.00
Meeting with Officials	2	$0.00			-	20.00	100.00	200.00	120.00	80.00
Surveying Existing Technologies	24					-	50.00	100.00	50.00	50.00
Identifying problems		$0.00			500.00	-	-	550.00	500.00	50.00
Finalizing Scope of Work						-	300.00	300.00	300.00	-
Design and Visualization								$ 790	$ 850	$ (60)
Lucid Chart					250.00			240.00	250.00	(10.00)
Figma					100.00			150.00	100.00	50.00
Design review and finalization							500.00	400.00	500.00	(100.00)
Development and Reporting								$ -	$ -	$ -
Development Enviroment									-	-
Integrations									-	-
Drafting and Presentations									-	-
Feature reviews									-	-
Analytics & Feedback									-	-

Figure 10: Budget

The following assumptions are made:

1. The site would be hosted on company servers should our contribution be considered for further integration. 2. Depending on the client requirements, the capacity and tenacity of the application can be scaled.

5.1.2 Project Resources

- Stake holders: 4 Team Members, Internal, External Guide, Company.
- Database and Authentication: MongoDB, ExpressJS
- Development tools and frameworks: Truffle, Ganache, Metamask
- Technologies: Solidity, MongoDB, JavaScript, Json, jQuery, Html/CSS.

5.2 Risk Management

5.2.1 Risk Identification

Entailing numerous discussions with our project guide, we identified various risks involved in executing various tasks in our project. Answers to certain questions revealed some risks. Each risk is categorized as per the categories mentioned in the table in the following section. Few fundamentally crucial questions have been duly answered throughout the process :

Has the client formally committed to supporting the project?

Yes.

2. Would end-users be enthusiastically committed to the project and the system/product to be built?

Yes.

3. Are requirements fully understood by the software engineering team?

Yes.

4. Will the system have a real-time application?

Yes.

5.2.2 Risk Analysis

In terms of the project that we have completed, risk analysis involves the identification and prioritization of potential risks within applications for the purpose of testing. A risk, in this context, refers to the possibility of detrimental consequences or harm that an organization may face due to various threats.

We conducted risk analysis as part of our software testing process to gain a comprehensive understanding of potential issues that could arise in an application before its deployment into production. Below are some of the risks identified for our project:

Smart Contract Bugs: Smart contracts automate transactions within Decentralized Applications. However, if there's a bug in the smart contract code, it can lead to significant losses. To eliminate any potential risks in this regard, we tested smart contracts in our project with industry-graded tools, which is illustrated in the following sections of this report.

Scalability Issues: Blockchain networks often suffer from scalability issues. This means that as the DApp gains more users, the transaction speed could become slower, resulting in a poor user experience.

Data Privacy and Security Risks: Despite the enhanced security due to decentralization, DApps are not immune to security risks. User data could be exposed if there are vulnerabilities in the application, and transactions on the blockchain are public, which could be a privacy concern for some users.

Adoption and User Experience: Decentralized Applications often require users to have a certain level of understanding about blockchain and cryptocurrencies, which might pose an adoption challenge. Furthermore, DApps can sometimes offer a less fluid user experience than traditional apps due to things like transaction times and wallet management.

Interoperability: Almost all of the DApps are often built on specific blockchain platforms (like Ethereum, Binance Smart Chain, etc.), and these platforms may

not be compatible with one another. This could limit the potential user base for the DApp and cause issues if you want to switch platforms later.

Network Dependency: DApps depend on the underlying blockchain network. If the network experiences downtime, the DApp will also be affected. Also, changes or upgrades in the blockchain protocol might affect the DApp's performance or functionality.

Gas Fees: Transaction costs, or "gas fees," on certain networks can be highly variable and, in some cases, quite high. This unpredictability can create a barrier to entry for users and a challenge for developers trying to manage these costs within their DApp's economic model.

Smart Contract Upgradability: Once deployed, smart contracts on the blockchain cannot be modified. This presents a challenge when it comes to fixing bugs or upgrading the contract's functionality. Design patterns like the proxy contract pattern can be used to address this, but they come with their own complexities and potential vulnerabilities.

Token Economics: if DApp includes a native token, it is needed to carefully design the token economics to incentivize the desired behavior, maintain the token's value, and avoid scenarios that could lead to rapid inflation or other economic issues. Missteps here can lead to drastic consequences for the DApp and its users. However, the scope of our project does not include payments and transactions, hence we restricted risk analysis to core risks strictly.

In virtue of our project being financially underwritten, the corporate entity possessed the jurisdiction to discern, manage, and mollify the above-mentioned risks. Given that our initiative is envisaged as an integral component of a more extensive endeavor, it is anticipated to be ultimately tendered to an eminent client in subsequent phases.

ID	Risk Description	Likelihood of Risk Occurring	Impact if the risk occurs
1	Project Purpose and need is no well-defined	Low	Medium
2	Project Schedule is not clearly defined or understood	Low	Low
3	Unplanned work that must be accommodated	Medium	Medium
4	Smart Contract defects	Low	High
5	Database de-sanitization	Low	Low
6	UI bugs – Improper/invalid actions upon Interface interactions	Low	medium

Figure 11: Risks and Analysis

5.2.3 Overview of Risk Mitigation, Monitoring & Management

General specifications for Risk Monitoring

Risk monitoring in a Decentralized Tender Management project for road development involves tracking identified risks, identifying new risks, and evaluating risk process effectiveness throughout the project. Effective risk monitoring can help detect potential problems early and allow for timely corrective action.

Probability	Description	
High	Probability of Occurrence is	> 75%
Medium	Probability of Occurrence is	26-75%
Low	Probability of Occurrence is	<25%

Figure 12: Risks and Analysis

5.3 Project Schedule

5.3.1 Project Task Set

1.Research: -

Searching for IEEE and other publication papers related to Decentralized Systems based on blockchain technology.

2. Software Requirement Specification: - For understanding customer requirements and analyzing objectives to be achieved.

3. Synopsis Formation:

It sets the stage for what the project aims to accomplish, the challenges it will address, and the approach it will take.

4. Designing of system architecture.

We finalized the design for our project and drafted various UML as well as architecture diagrams, which are illustrated in the diagram section of the report.

5. Implementation of system.

With the guidance of our external mentor, we initiated the development of our project, starting with the Front-End part. We have exemplified the process in the project implementation section.

6. Testing of system by using different test cases.

We tried to isolate various issues which needed to be addressed either via extensive research or simple test scenarios:

Decentralized Systems: The decentralized nature of blockchain applications adds another layer of complexity for testing. Unlike traditional applications where you're mainly testing the central server's behavior, in a decentralized system you need to test all the nodes and their interactions. This requires a different testing approach and more thorough testing to ensure system integrity and consistency.

Security: Blockchain applications are often targets for hackers due to the value of the assets they manage. While the blockchain itself is generally secure, vulnerabilities can exist in the application layer, especially in smart contracts. Comprehensive security testing can help identify and fix these vulnerabilities before they can be exploited.

Performance and Scalability: Blockchain applications often face performance and scalability issues due to the inherent limitations of blockchain technology. Performance testing can help identify bottlenecks and optimize the system for better performance. It also helps in understanding how the system behaves under heavy load or large transaction volumes, which is critical for scalability.

User Experience: Despite the technical nature of blockchain applications, they still need to deliver a positive user experience. Usability testing can help ensure that the application is easy to use and meets user expectations.

Interoperability: With various blockchain platforms available (like Ethereum, Hyperledger, etc.), it's crucial to ensure that your application is inter-operable where it needs to be. This is particularly important for Decentralized Finance (DeFi) applications that need to interact with various other protocols.

7. Report.

Schedule of the project:

Start Date: August 2022

End Date: May 2023

Duration: Approx 10 Months

5.3.2 Task Network

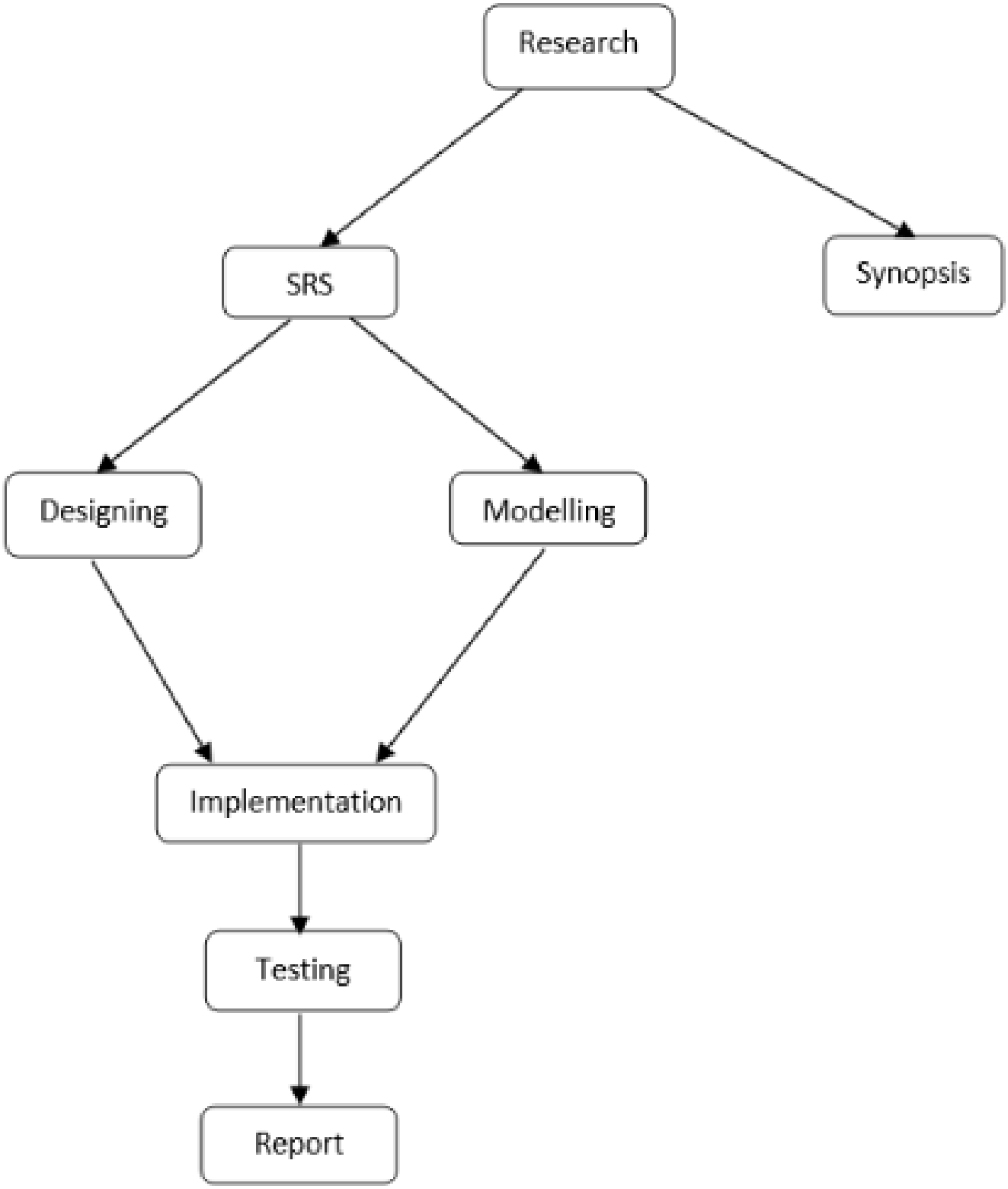

Figure 13: Network Diagram

5.3.3 Timeline Chart

Figure 14: Gannt Timeline

5.4 Team Organization

5.4.1 Team Structure

1. Shankar Avad: Front-end, Design and Documentation.

2. Atharv Vaspate: Back-end, Debugging and Reporting

3. Aniket Datar: Back-end; Smart Contract Development and Reporting

4. Niranjan Patankar: Design and Reporting

5.4.2 Management Reporting and Communication

- Fortnightly reporting to respective guides.
- Continuous updating and reviewing of SRS and Development Process.
- Monthly reporting to external guide.
- Expert guidance if required.
- Use of Agile SDLC workflow model to collaborate and work efficiently.

6 PROJECT IMPLEMENTATION

6.1 Overview of Project Modules

Tender Creation and Publication:

Enable project creators to publish tenders for specific road development projects. Include detailed tender requirements, specifications, submission deadlines, and evaluation criteria. Securely store tender information on the blockchain.

Bid Submission:

Allow contractors or interested parties to submit bids for specific tenders. Validate bid submissions to ensure compliance with the specified requirements [5] . Assign unique identifiers (e.g., cryptographic hashes) to each bid and store them securely on the blockchain.

Bid Evaluation and Selection:

Implement a bid evaluation process, which can be automated or manual, based on predefined criteria. Enable authorized evaluators to review and compare bids. Transparently record the bid evaluation process on the blockchain to ensure transparency, fairness, and accountability.

User Registration and Authentication:

Provide user registration functionality, allowing individuals to create accounts. Implement robust authentication mechanisms to ensure secure access to the system. Safeguard user credentials and sensitive information using encryption techniques.

6.2 Tools and Technologies used

Blockchain Platforms:

Ethereum is a widely used blockchain platform known for its smart contract functionality, enabling the development of decentralized applications (dApps).

Solidity is a programming language specifically designed for writing smart con-

tracts on the Ethereum platform.

Blockchain Development Frameworks and Libraries:
Truffle is a development framework for Ethereum that simplifies the creation, testing, and deployment of smart contracts.
Web3.js is a JavaScript library that provides an interface for interacting with Ethereum smart contracts and the Ethereum network.

Database:
MongoDB is a NoSQL database known for its flexibility, scalability, and ease of use. It can be used to store user registration details, authentication credentials, and other relevant user information in a blockchain-based system.
The Truffle Suite is a collection of development tools and frameworks specifically designed for Ethereum blockchain development. It provides a comprehensive set of tools that simplify the process of creating, testing, and deploying smart contracts and decentralized applications (dApps) on the Ethereum platform.

Web Development:
Front-End: HTML, CSS, and jQuery are commonly used for building the user interface (UI) of web applications.
Back-End: Node.js and Express.js are popular technologies for server-side logic and hosting web applications.

Additional Tools and Technologies:
Git is a widely used version control system for managing source code and collaboration.
Tools like Visual Studio Code and Remix IDE can assist with smart contract development, testing, and deployment.

Testing and Deployment:
Remix is a web-based IDE specifically designed for Ethereum smart contract development, offering features for writing, testing, debugging, and deploying smart

contracts.
Mythril is an open-source security analysis tool used to identify potential security vulnerabilities in Solidity smart contracts.
Ganache, part of the Truffle Suite, provides a local test network for deploying and testing smart contracts locally without interacting with the live Ethereum network.

Authentication and Security:
Algorithms like AES (Advanced Encryption Standard) and SHA-256 are commonly used for encrypting sensitive data and hashing passwords.
Secure authentication protocols and mechanisms are implemented to ensure secure access to the decentralized tender management system.

6.3 Algorithm Details

6.3.1 Algorithm 1: Bidder Dashboard - List All Tenders

Step 1 : Retrieve the total number of tenders using the tenderCount() function from the App.TenderAuction contract.
Step 2 : Initialize a loop variable i to 1 and iterate while i is less than or equal to the tender count.
Inside the loop:
Step 3 : Retrieve the details of the tender at index i using the tenders() function from the App.TenderAuction contract.
Step 4 : Generate a tenderTemplate string that represents a table row with the tender details.
Step 5 : Generate a tenderPopupTemplate string that represents the HTML structure for a tender popup with additional details.
Step 6 : Append the tenderTemplate to the allTenders element in the DOM.
Step 7 : Append the tenderPopupTemplate to the tenderPopup element in the DOM.
Step 8 : Increment the loop variable i by 1.
Step 9 : End the loop.
Step 10 : End the algorithm.

Time complexity O(n)

where n is the number of tenders

6.3.2 Algorithm 2: Bidder Dashboard - List My Bids

:

Step 1 : Get the total number of bids by calling the bidCount() function from the App.TenderAuction contract and store it in the bidCount variable.

Step 2 : Iterate over each bid from 1 to bidCount.

Step 3 : Inside the loop, retrieve the bid details by calling the bids() function from the App.TenderAuction contract, passing the bid index i, and store the result in the bid variable.

Step 4 : Check if the bidder address (bid[4]) matches the current user's account (App.account).

Step 5 : If the bidder addresses match:

Log the user's account to the console using console.log(App.account)

Step 6 : Store the bidder's name in the checkbiddername variable (bid[2]).

Step 7 : Initialize the checkbidvalue variable to null.

Step 8 : Update checkbidvalue with the bid value (bid[3]).

Step 9 : Generate an HTML template for displaying the bid information using the bidTemplate string.

Step 10 : Append the bid template to the HTML element with the ID myBids.

Step 11 : End the loop.

Time complexity O(n)

where n is the number of bids

6.3.3 Algorithm 3: Uploader Dashboard - Show All Bids

:

Step 1 : Get the total number of bids by calling the bidCount() function from the App.TenderAuction contract and store the result in the bidCount variable.

Step 2 : Initialize a loop variable i to 1 and iterate while i is less than or equal to bidCount.
Step 3 : Inside the loop:
Retrieve the tender details by calling the tenders() function from the App.TenderAuction contract, passing the tender index i, and store the result in the tender variable.
Step 4 : Retrieve the bid details by calling the bids() function from the App.TenderAuction contract, passing the bid index i, and store the result in the bid variable.
Step 5 : Extract the necessary information from the tender and bid arrays, such as tender name (tender[1]), bid value (bid[3]), and any other relevant fields.
Step 6 : Store the tender name (tender[1]) in the local storage with the key ”Bidaccepted”.
Step 7 : Retrieve the accepted tender name from the local storage using the key ”Bidaccepted” and store it in the acceptedtendername variable.
Step 8 : Generate an HTML template for displaying the bid information using the bidTemplate string, where each field is filled with the corresponding tender and bid array elements.
Step 9 : Append the bid template to the HTML element with the ID bidList.
Step 10 : End the loop.
Time complexity O(n)
where n is the number of tenders or bids

6.3.4 Algorithm 4: Smart Contract (Tender.sol)

To check if address is already present or not in the uploader or bidder mappings

Step 1 : Start the modifier alreadyPresent.
Iterate over the uploaderCount from i = 1 to uploaderCount.
Step 2 : Check if the uploader address at index i in the whoIsUploader mapping is equal to the address parameter.
Step 3 : If it is equal: Throw an error with the message ”Address already present” using the require statement.

Step 4 : Iterate over the bidderCount from i = 1 to bidderCount.

Step 5 : Check if the bidder address at index i in the whoIsBidder mapping is equal to the address parameter.

Step 6 : If it is equal: Throw an error with the message ”Address already present” using the require statement.

Step 7 : Continue with the execution of the modified function.

Step 8 : End the modifier.

Time complexity O(n)

where n is the maximum value between uploaderCount and bidderCount.

7 SOFTWARE TESTING

7.1 Type of Testing

When developing a decentralized tender management system for road development using blockchain technology, there are several types of testing that can be conducted to ensure the system's functionality, security, and performance. Here are some relevant types of testing:

Ui testing:
UI testing, also known as User Interface testing or Front-end testing, focuses on evaluating the graphical user interface of the decentralized tender management system.
The primary goal is to ensure that the user interface is intuitive, visually appealing, and functions correctly.
Here are some key aspects of UI testing that we used for our decentralized tender management system:
1. We Verified that the UI elements, such as buttons, forms, tables, and navigation menus, are displayed correctly and aligned properly on different screen sizes and resolutions.
2. We checked for bugs in the UI related to whether any styling was blocking the required text fields , text boxes , tables etc.
3. We manually checked whether by clicking on the specific button it is getting redirected to the correct page.

Functionality Testing:
We Validated that all the UI components and interactive elements, such as buttons, links, input fields, and dropdown menus, are functioning as expected.
Test the user flows and interactions to perform tasks like creating tenders, submitting bids, and viewing tender details.

Input Validation Testing :

We Tested the validation of user input, such as enforcing required fields, validating data formats (e.g., email addresses or numeric values), and displaying appropriate error messages for invalid inputs.
We checked for various login conditions such as entering wrong usernames , passwords , empty fields etc.

Smart Contract Testing (Remix and Mythril) :
We used remix in order to carry out testing of our smart contracts.
Manual Testing:
We Used Remix's built-in Solidity compiler to deploy our smart contracts to the test network.
We then interacted with the contract using remix's UI and provided various inputs to verify the specific outputs.
We also manually executed different contract functions and verified that the contract behaves as expected.
We then Used Remix's console to log and inspect contract state changes and used it debug the issues.

Database Testing (MongoDB) :
We have used MongoDB in order to store user data such as username , email , password(hashed) , gender , region etc.
We have carried out several tests on the MongoDB database :
1. The main test we carried out was to check whether data is getting stored int the database or not .
2. We also checked by entering wrong formats of email or wrong contact numbers to check whether specific error is shown or not .
3. We checked for storage of gender and region which gets input through a dropdown menu.
4. the most important test we carried out was to check that when the password is being stored in the database whether it is being stored by salting and hashing it so that the password cannot be hacked.

7.2 Test Cases and Test Results

Test case 1 :

Testing for the login password :

Entered correct username password which should redirect to the next page .

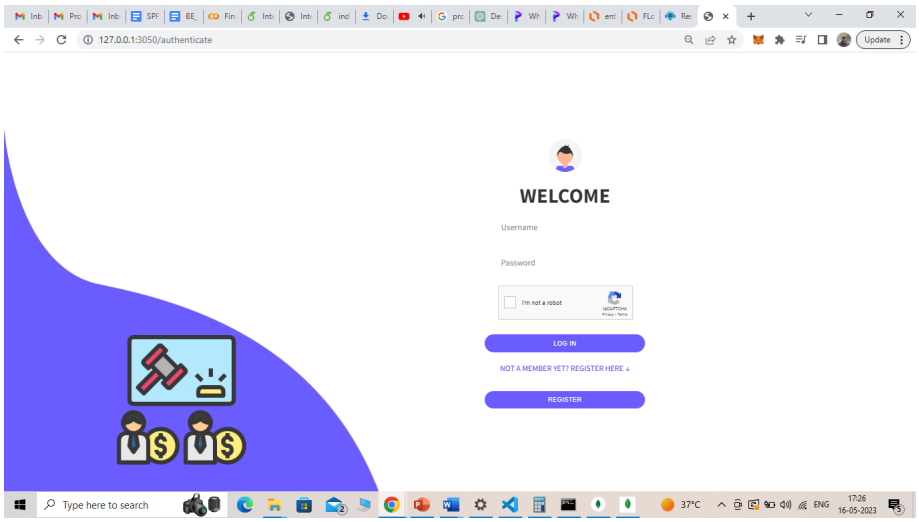

Figure 15: T.C. 1

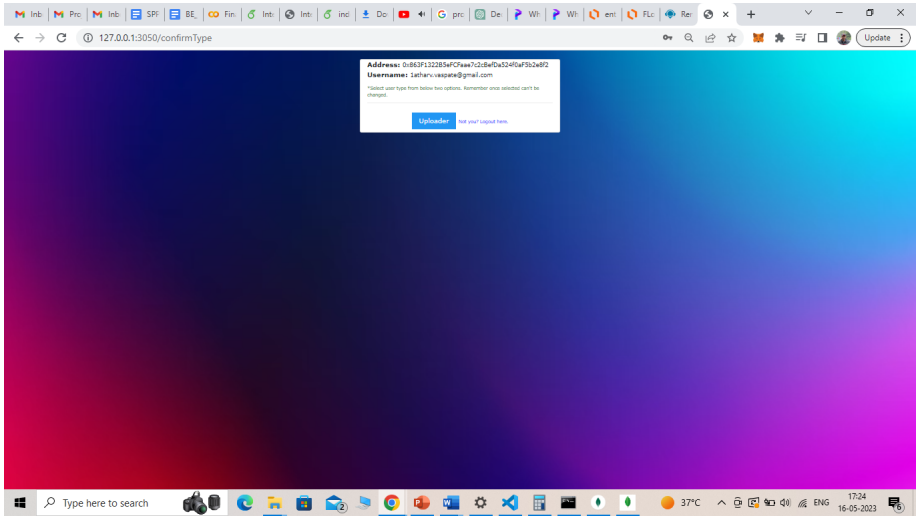

Figure 16: T.C. 1 Passed

Entered wrong username and password which should redirect to the error page.

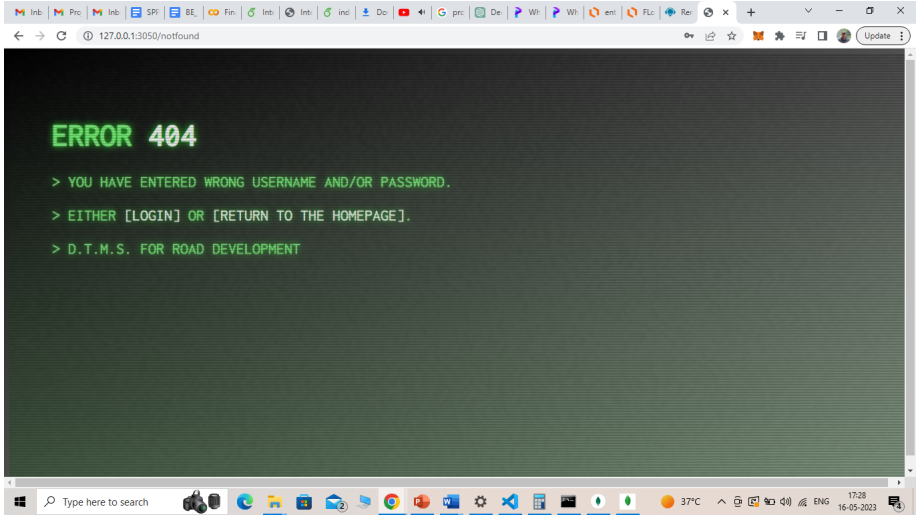

Figure 17: T.C. 1 Unsuccessful

Test case 2 :

Checking for the functionality of the ui components whether the buttons , sliders etc are working correctly.

Test case 3 :

Testing of the smart contracts using remix :

We deployed the smart contracts on remix to check for their working.

The functions were carried out to check whether the specific output is being obtained or not.

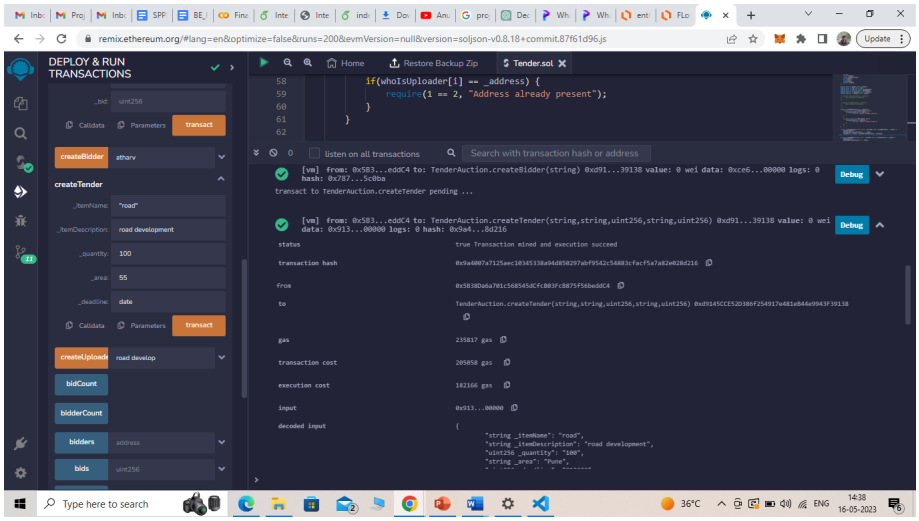

Figure 18: T.C. 3 Passed

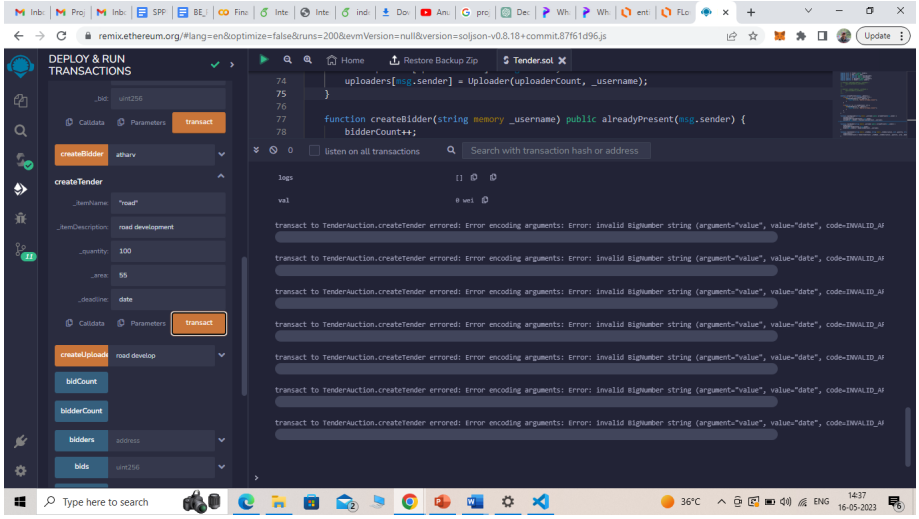

Figure 19: T.C. 3 Unsuccessful

Test case 4 :

Metamask Testing :

The most important aspect of the project was to check whether the transactions are carried out successfully.

We carried out two cases :

a. In the first case we accepted the transaction which should lead to the transaction being successfully accepted which can be seen in the logs .

b. In the second case we denied the transaction which shows the error ibn c=vs code and can also be seen in the logs.

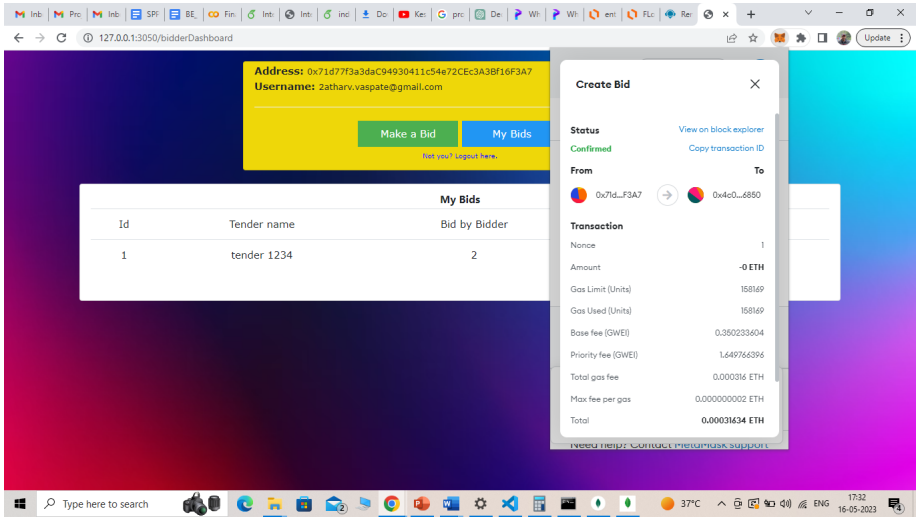

Figure 20: T.C. 4 Passed

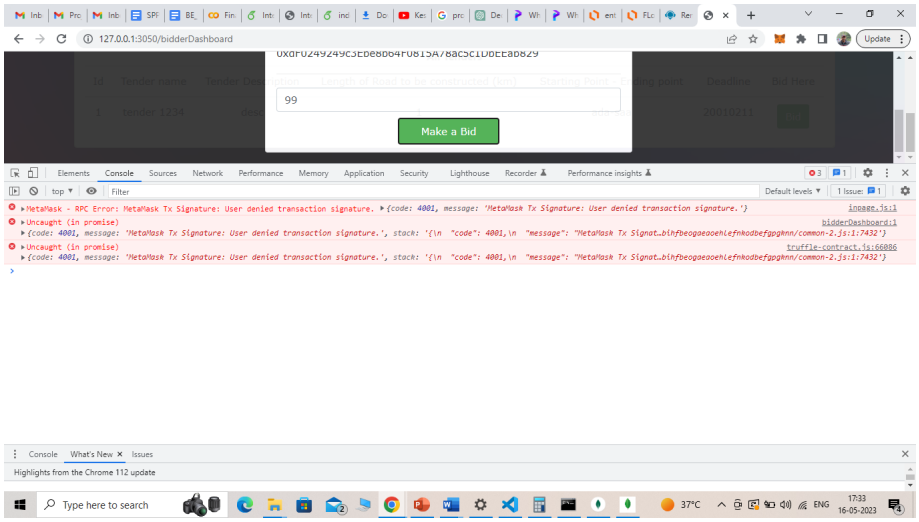

Figure 21: T.C. 4 Unsuccessful

Test case 5 :

MongoDB testing :

We tested various conditions of storing the data in the MongoDB database .

The main condition to be checked was whether the password was being hashed during storage.

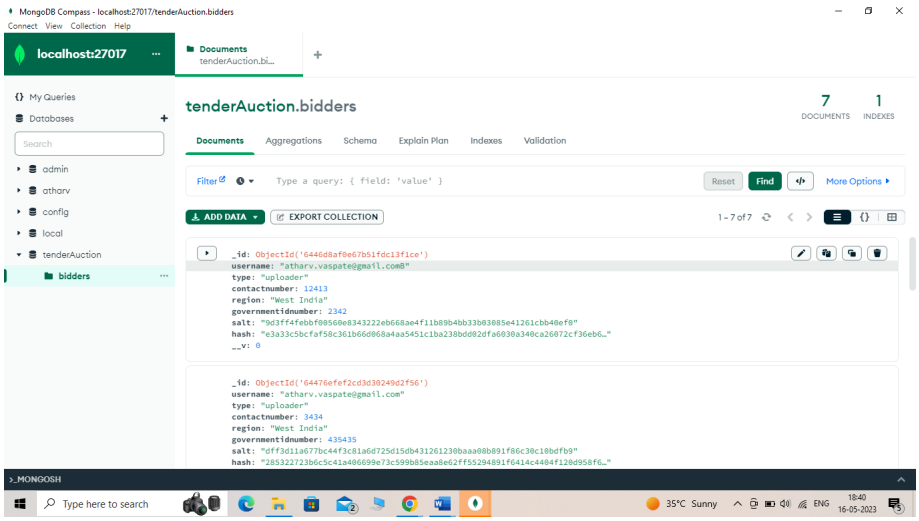

Figure 22: T.C. 5 Passed

We also checked the condition when a user used same email to register was not allowed to register.

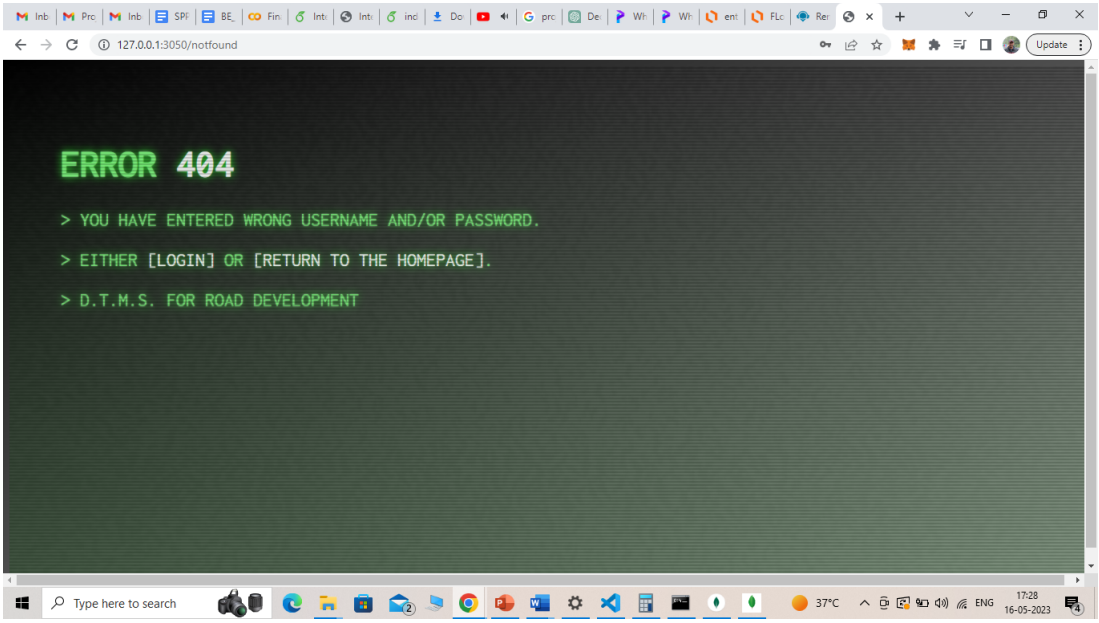

Figure 23: T.C. 5.1 Unsuccessful

8 RESULTS

8.1 Outcomes

Thus by implementing a decentralized tender management system for road development we have come up with several benefits over the existing system.
The various outcomes are :

Enhanced Transparency: The implementation of a decentralized tender management system using blockchain technology ensures increased transparency in the entire tender process [6] .
The use of a transparent and decentralized ledger enables all stakeholders, including government entities, contractors, and bidders, to access auditable records of tender processes and bids.
This transparency reduces the chances of corruption, favoritism, or biased decision-making, fostering a fair and accountable environment.

Improved Efficiency : The automation of tender processes through smart contracts on the blockchain significantly improves efficiency.
Manual paperwork and intermediaries are eliminated, resulting in streamlined tender issuance, bid submission, and evaluation.
The use of digital processes and real-time updates reduces delays and administrative overhead, leading to faster project delivery and cost savings.

Enhanced Security: Blockchain technology provides enhanced security for tender-related data.
The use of cryptographic techniques ensures data integrity and prevents unauthorized modifications or tampering.
Tender details, bid information, and contract terms are stored securely on the blockchain, reducing the risk of data breaches and fraud.
This enhanced security generates trust among stakeholders and safeguards sen-

sitive information.

Increased Participation: Decentralized tender management systems facilitate increased participation from a diverse range of bidders, including smaller contractors and suppliers.

Traditionally, centralized tender processes may exclude these smaller players, limiting competition and innovation.

With a decentralized system, barriers to entry are lowered, fostering an inclusive environment that promotes competition and drives innovation in the road development sector.

8.2 Screenshots

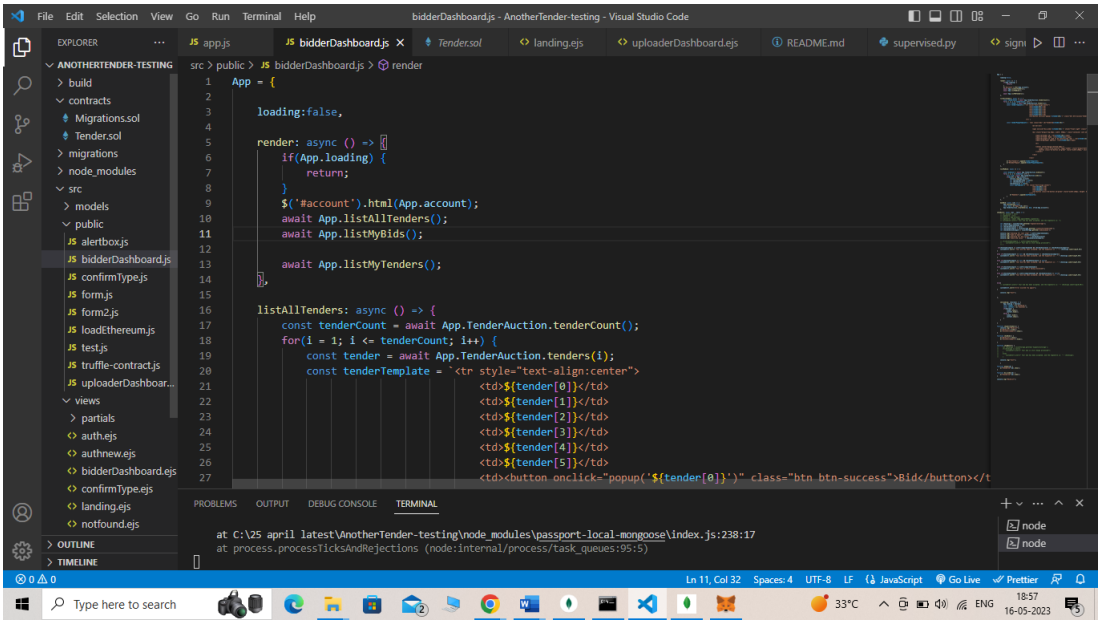

Figure 24: Code Image

Figure 25: Landing Page

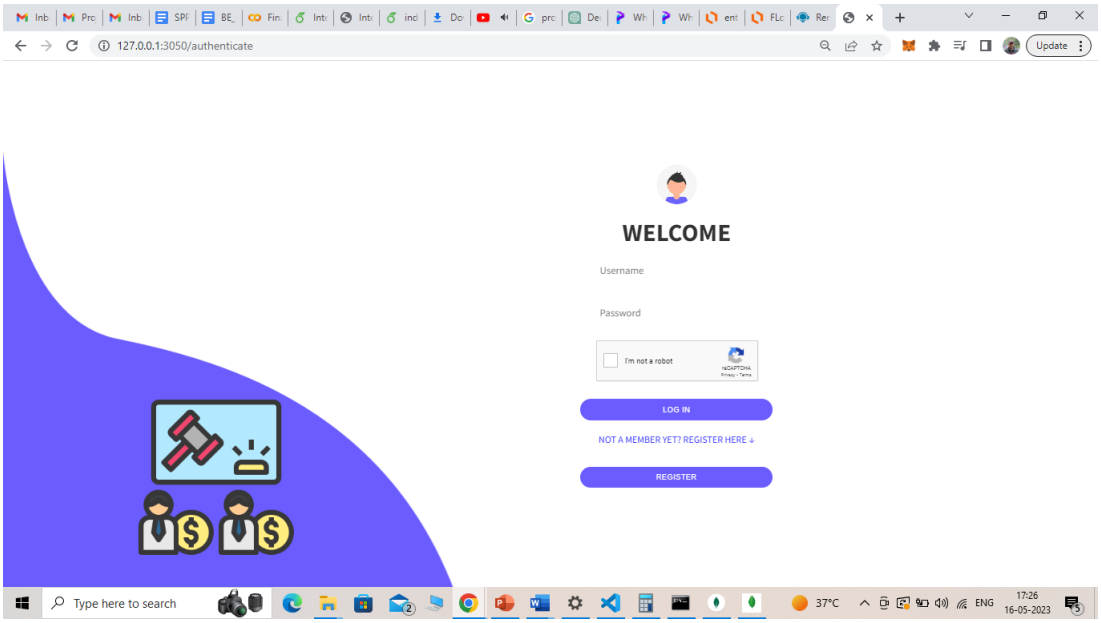

Figure 26: Login Page

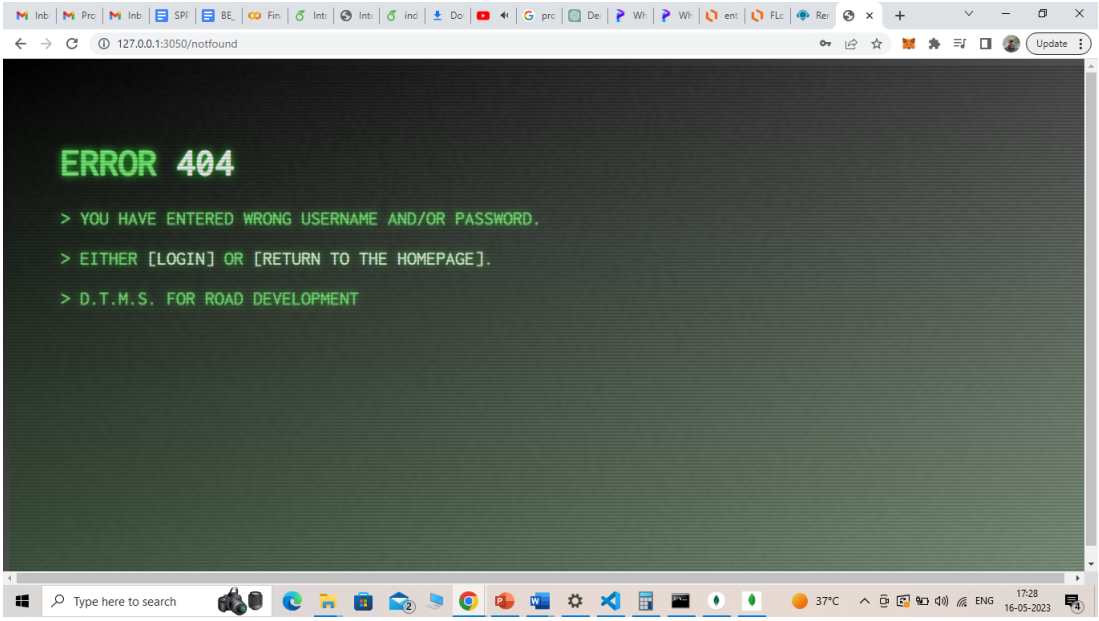

Figure 27: Error Page

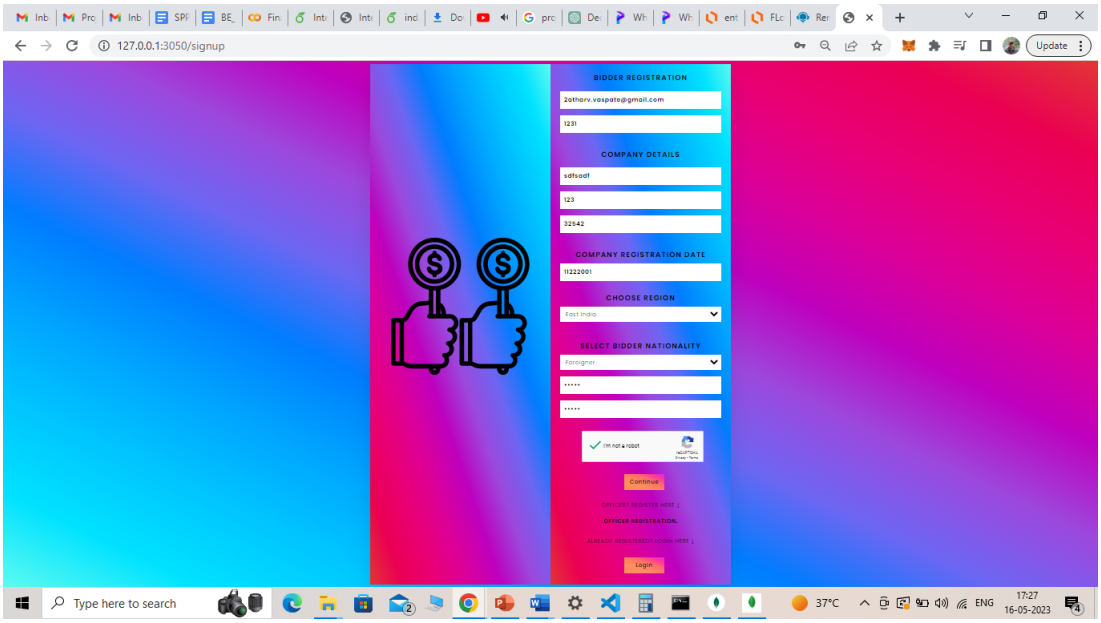

Figure 28: Bidder Registration Page

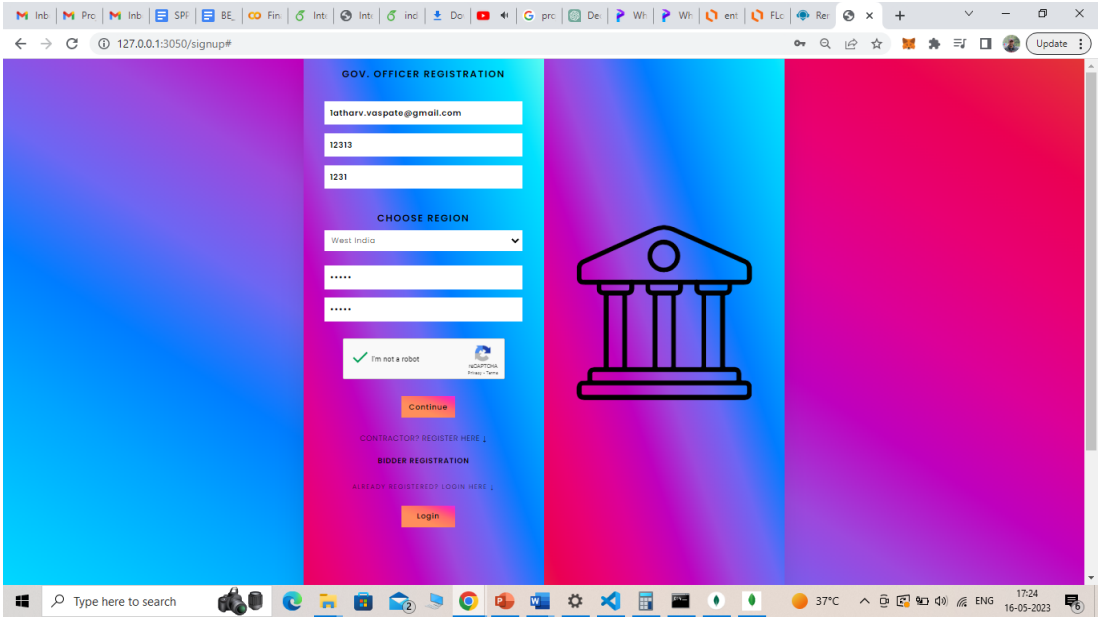

Figure 29: Government Officer Registration Page

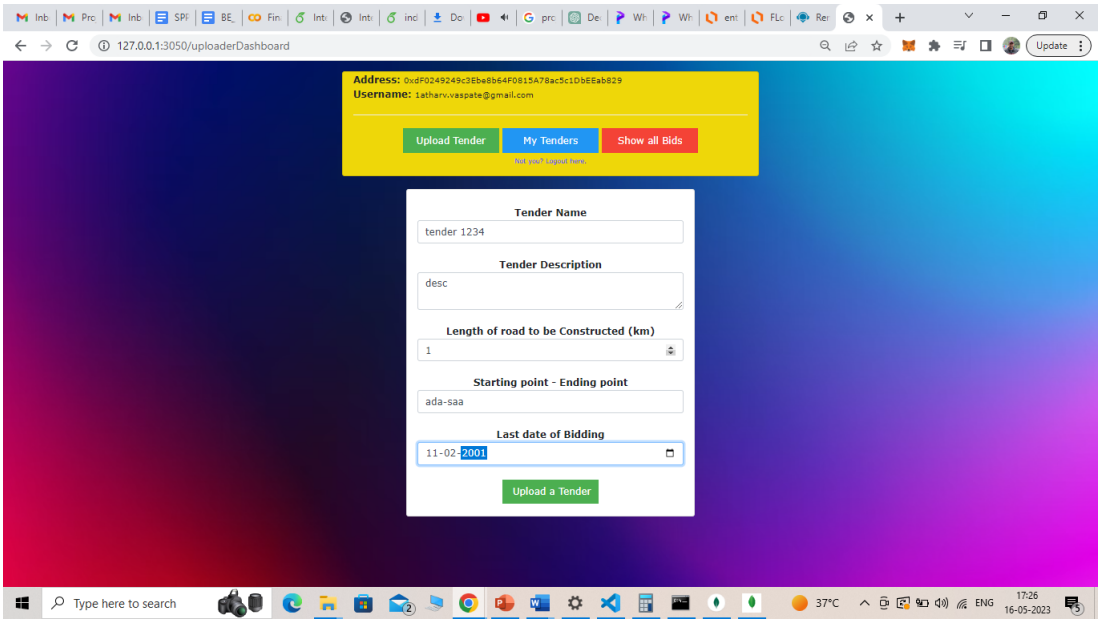

Figure 30: Create Tender

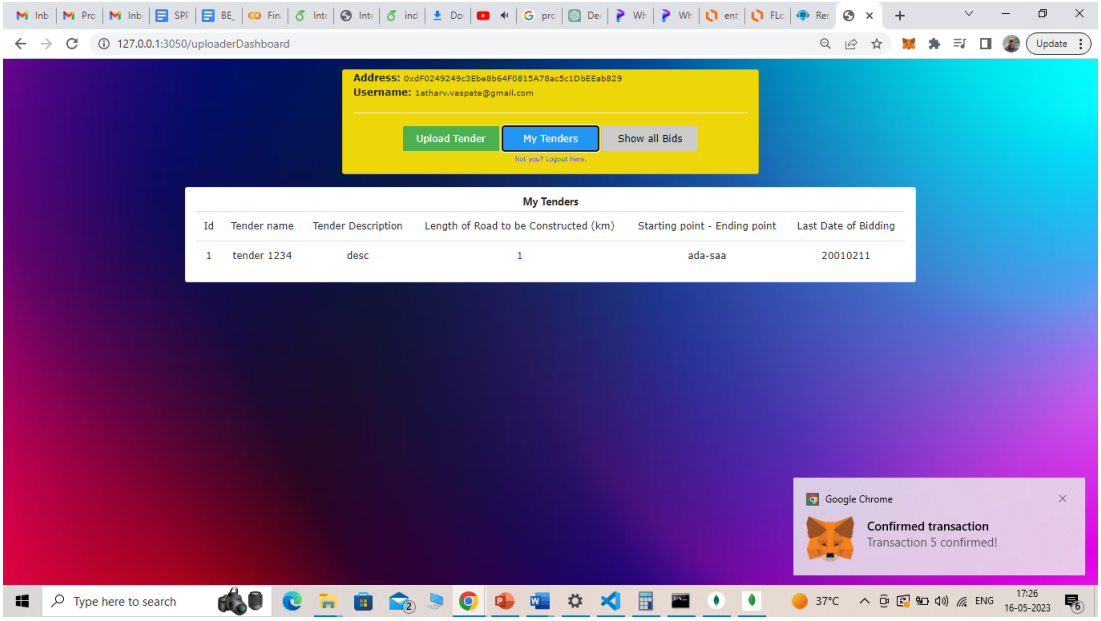

Figure 31: My Tenders

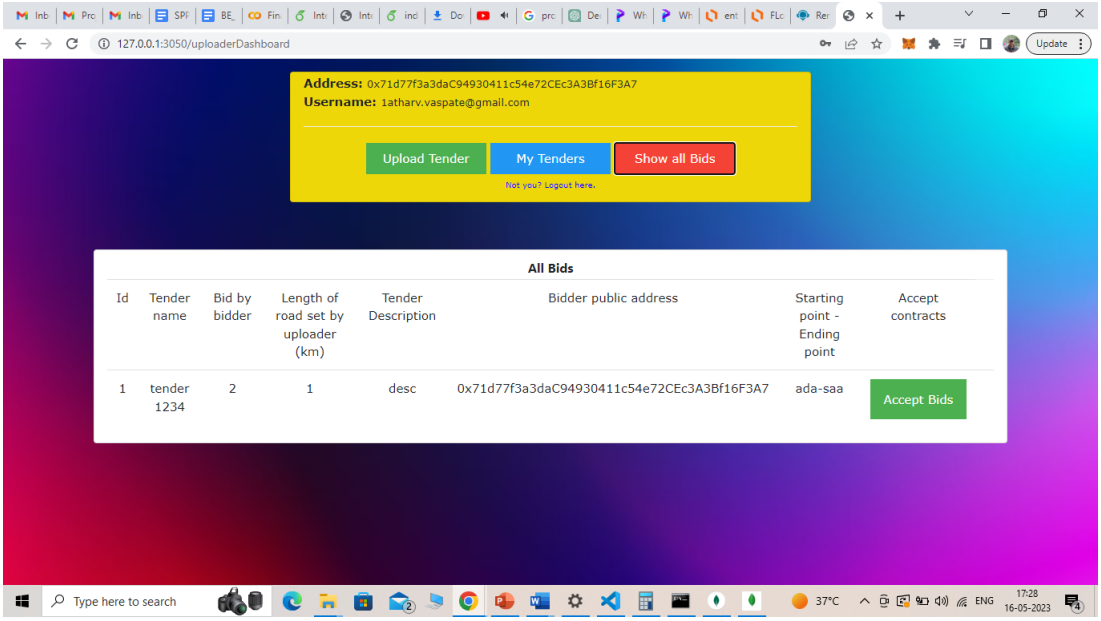

Figure 32: Show all Bids

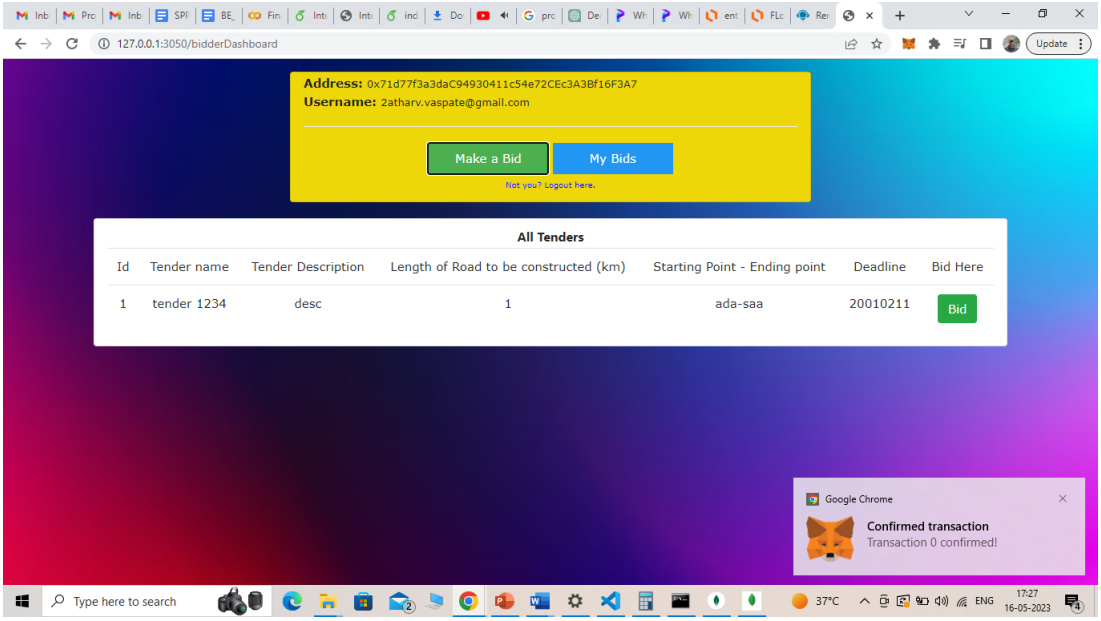

Figure 33: Make a bid

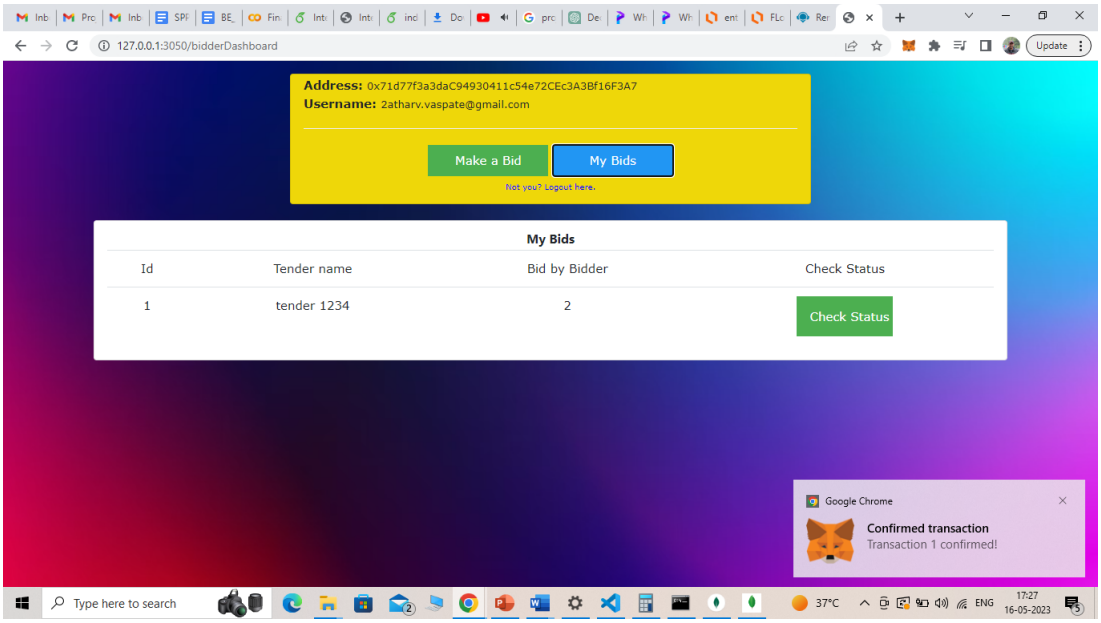

Figure 34: My Bids

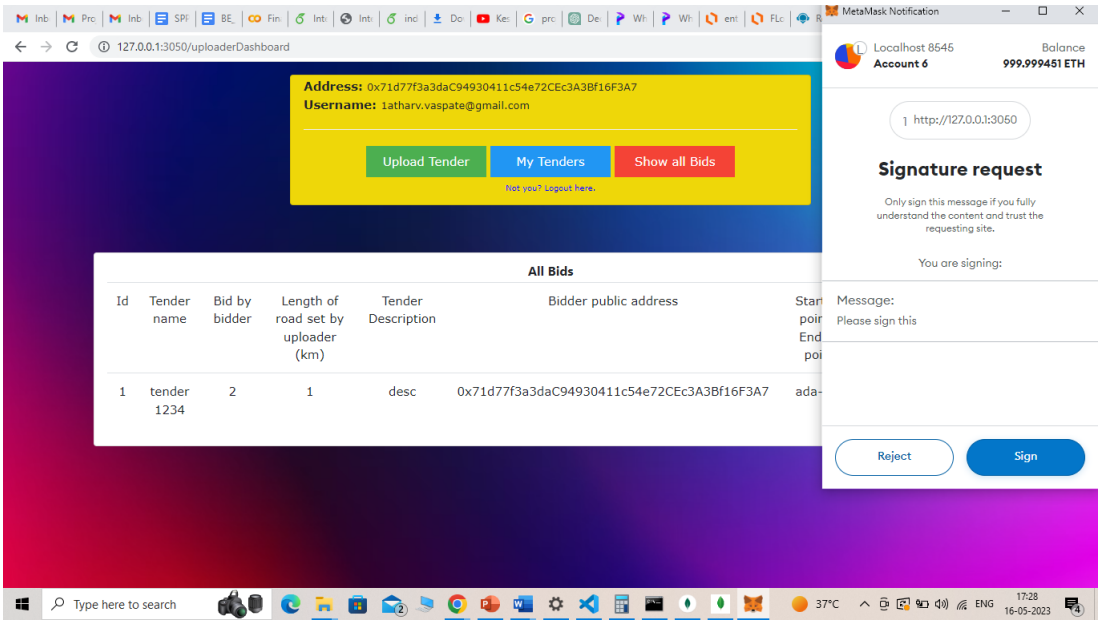

Figure 35: Sign Transaction

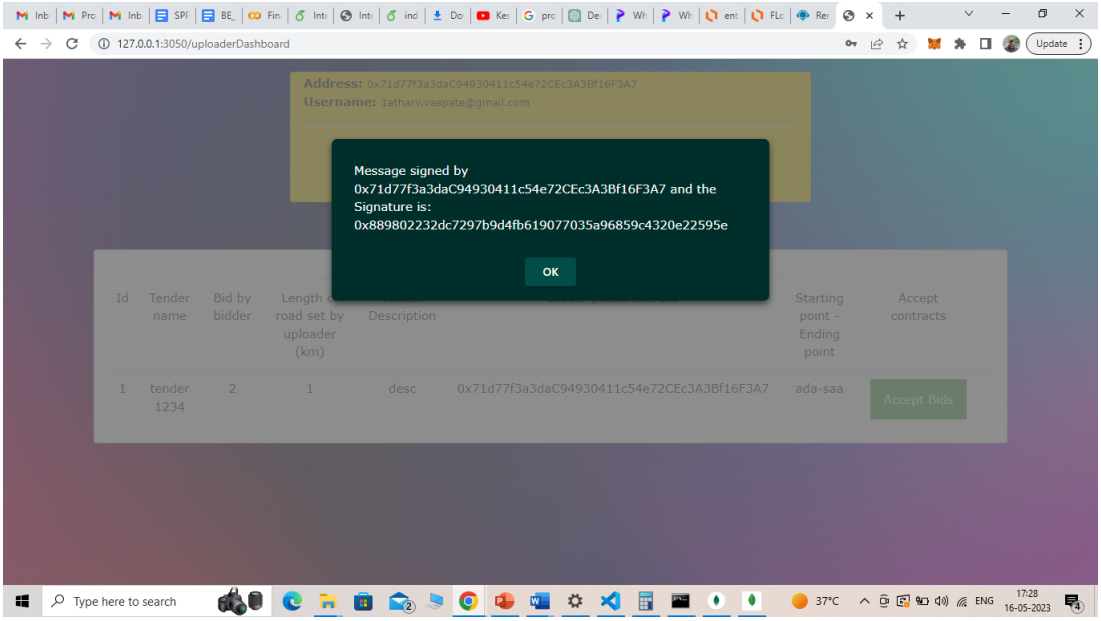

Figure 36: Transaction Signed Successfully

9 CONCLUSIONS

9.1 Conclusions

The implementation of a decentralized tender management system for road development using blockchain technology offers numerous benefits to the overall tender process. It brings increased transparency, improved efficiency, enhanced security, and greater participation. By leveraging the unique features of blockchain, the road development sector can revolutionize its tender processes, promote fairness and accountability, and achieve more successful infrastructure projects.

9.2 Future Work

The decentralized tender management system using blockchain technology has several possibilities for future work and enhancements [7] .
Some of them include :

Integration with IoT and Sensors:
The integration of Internet of Things (IoT) devices and sensors can provide real-time monitoring and data collection during the road development process.
This integration allows for automated data capture of road conditions, traffic flow, and maintenance needs, which can be recorded on the blockchain.
Such data can enhance decision-making, facilitate proactive maintenance, and enable efficient planning for road development projects.

Smart Contract Enhancements:
Smart contracts can be further improved to include more complex conditions and automated processes.
For example, the system can automatically trigger payments based on predefined milestones or conditions, ensuring prompt and accurate compensation for contractors.

Additionally, incorporating penalties or incentives into smart contracts can encourage timely project completion and ensure high-quality outcomes.

Expansion to Other Infrastructure Projects:
While the focus is initially on road development, the decentralized tender management system can be expanded to cover other infrastructure projects such as bridges, tunnels, or public buildings.
Adapting the system to accommodate the specific requirements and regulations of different projects would broaden its applicability and impact, offering a comprehensive solution for managing various types of infrastructure development.

Scalability and Interoperability :
As the system grows and handles an increasing volume of tenders and bids, ensuring scalability and interoperability becomes crucial.
Developing solutions that can handle higher transaction volumes, integrate with existing systems, and inter-operate with other blockchain networks will be essential for the long-term success and widespread adoption of the decentralized tender management system.

9.3 Applications

The decentralized tender management system for road development using blockchain technology has various applications within the road development industry [7] .
Some key applications include:

Tender Issuance and Management: The system facilitates the issuance and management of tenders for road development projects.
It provides a transparent and secure platform for government entities or project owners to publish tender details, specifications, and requirements.
Contractors and suppliers can easily access and participate in the tender process, submit bids, and track the progress of their submissions.

Bid Evaluation and Selection:

The system automates the bid evaluation and selection process, ensuring fairness, transparency, and accuracy.

Smart contracts can be utilized to define evaluation criteria, automate scoring, and eliminate the need for manual intervention.

This streamlines the evaluation process, reduces human bias, and enhances the efficiency of selecting the winning bid.

Auditing and Accountability:

The blockchain-based system provides a reliable audit trail for tender-related activities.

All transactions and interactions recorded on the blockchain are immutable, allowing for transparent auditing and ensuring accountability among all involved parties.

This promotes trust and reduces the risk of corruption or malpractice.

10 REFERENCES

[1] Vivek Kharpude . ”Tendering process in India and how it is its own Achilles heel” , ESDS Software Solutions Limited.

[2] Rafael Rossi Buzzeto , Mariana Rodrigues Bauli and Marly Monteiro de Carvalho. ”The key aspects of procurement in project management: investigating the effects of selection criteria, supplier integration and dynamics of acquisitions” , January 2020 University of Sao Paulo.

[3] Yashita Goswami, Ankit Agrawal and Ashutosh Bhatia . ”E-Governance : A Tendering Framework Using Blockchain With Active Participation of Citizens” , 08 February 2021 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS).

[4] Sheetal Sharma , Darothi Sarkar and Divya Gupta . ”Agile Processes and Methodologies: A Conceptual Study” , May 2018 International Journal on Computer Science and Engineering.

[5] Maher Alharby , Amjad Aldweesh and Aad van Moorsel . ”Blockchain-based Smart Contracts: A Systematic Mapping Study of Academic Research ” , 11 July 2019 International Conference on Cloud Computing, Big Data and Blockchain (ICCBB).

[6] Thilak K , Prof. Priya . ”Smart Tender/Contract Management System Using Blockchain ” , 28 April 2018 International Journal of Research Publication and Reviews.

[7] Shafaq Naheed Khan, Faiza Loukil, Chirine Ghedira-Guegan, Elhadj Benkhelifa Anoud Bani-Hani . ”Blockchain smart contracts: Applications, challenges, and future trends” , 18 April 2021 Springer (Peer-to-Peer Networking and Applications).

11 APPENDIX A

11.1 Feasibility Study

11.1.1 Operational Feasibility

The system is easy to use, portable, and has more features related to the other systems that are already available in the market.

11.1.2 Technical Feasibility

1) The system will run efficiently if provided with good internet connection.
2) No issues regarding legality will be faced as long the project remain within the legal rights of the sponsoring institution, as some of the libraries and software used are proprietary.

11.2 Problem Analysis

11.3 Problem Type

This problem gives a solution in polynomial time. Hence, this problem statement is P type problem.

12 APPENDIX C

List of contents :

- Plagiarism Report Overall Similarity
- Expanded Content
- Sources/Links

Plagiarism Checker X - Report

Originality Assessment

8%

Overall Similarity

Date: May 19, 2023
Matches: 1076 / 14347 words
Sources: 42

Remarks: Low similarity detected, check with your supervisor if changes are required.

Verify Report:
Scan this QR Code

Sources

1	https://www.pvgcoet.ac.in/academics/departments/computer-engineering/ INTERNET 2%
2	https://ieeexplore.ieee.org/document/8756390 INTERNET 1%
3	https://teachingmaterial.github.io/ELEE1149_Exercises/UML-Class-Diagrams/UML-Class-Diagrams.html INTERNET 1%
4	https://www.simplilearn.com/tutorials/blockchain-tutorial/how-to-become-a-blockchain-developer#:~:text=Blockchain development is the process of developing and,as digital currencies, smart contracts, and distributed applications. INTERNET <1%
5	https://medium.com/techskill-brew/proof-of-work-in-blockchain-part-7-blockchain-basics-d96e38cb85af INTERNET <1%
6	https://www.geeksforgeeks.org/test-execution-for-software-testing/ INTERNET <1%
7	https://www.tutorialspoint.com/system_analysis_and_design/system_analysis_and_design_testing_quality_assurance.htm INTERNET <1%
8	https://www.tutorialspoint.com/uml/uml_deployment_diagram.htm INTERNET <1%
9	https://en.wikipedia.org/wiki/systems_architecture INTERNET <1%
10	https://www.researchgate.net/profile/A-Ospanova-3/publication/360034729_Social_Distance_Detection_and_Alert_System_using_Deep_Learning_and_Computer_Vision/links/625ea25da279ec5dd702b2af/Social-Distance-Detection-and-Alert-System-using-Deep-Learning-and-Computer-Vision.pdf INTERNET <1%
11	https://www.researchgate.net/publication/339208127_The_key_aspects_of_procurement_in_project_management_investigating_the_effects_of_selection_criteria_supplier_integration_and_dynamics_of_acquisitions INTERNET <1%
12	https://beincrypto.com/blockchain-sustainable-development-technology-global-challenges/#:~:text=Blockchain can help to address these challenges by,recording each transaction in a secure, decentralized ledger. INTERNET <1%

13	https://ieeexplore.ieee.org/document/8756390#:~:text=Our interest is twofold, namely to provide a,up smart contract technologies and established scientific outputs. INTERNET <1%
14	https://www.researchgate.net/publication/333748177_Blockchain-based_Smart_Contracts_A_Systematic_Mapping_Study_of_Academic_Research_2018 INTERNET <1%
15	https://www.codementor.io/blog/blockchain-scalability-5rs5ra8eej#:~:text=As the number of transactions grows, these systems,factor in a wider acceptance of blockchain systems. INTERNET <1%
16	https://www.mondaq.com/india/government-contracts-procurement-ppp/1161036/public-procurement INTERNET <1%
17	https://www.sciencedirect.com/science/article/pii/S2452414X21000169 INTERNET <1%
18	https://101blockchains.com/best-smart-contract-development-tools/ INTERNET <1%
19	https://www.conceptdraw.com/examples/what-is-boundary#:~:text=Use case is usually shown as an ellipse,is visually located inside the system boundary rectangle. INTERNET <1%
20	https://en.wikipedia.org/wiki/Unified_Modeling_Language INTERNET <1%
21	https://blog.netwrix.com/2021/07/26/data-security/ INTERNET <1%
22	https://opo-boa.gc.ca/praapp-prorev/2016-2017/pes-bep-eng.html#:~:text=As this principle applies to the bid evaluation,conducted in a fair, open and transparent manner. INTERNET <1%
23	https://www.geeksforgeeks.org/solidity/ INTERNET <1%
24	https://www.sciencedirect.com/science/article/pii/S0268401219303329 INTERNET <1%
25	https://www.smartdraw.com/sequence-diagram/#:~:text=Lifelines are vertical dashed lines that,indicate the object's presence over time. INTERNET <1%
26	https://www.mikevestil.com/risk-monitoring/#:~:text=Effective risk monitoring can help organizations minimize risk,,actions to reduce or prevent them from occurring. INTERNET <1%

27	https://sci-hub.se/10.1007/S12083-021-01127-0 INTERNET <1%
28	https://www.projectmanager.com/blog/product-backlog-sprint-backlog INTERNET <1%
29	https://support.microsoft.com/en-us/office/activation-shape-98809fc5-72c1-40b7-98a3-519df518c8ea INTERNET <1%
30	https://www.guru99.com/best-operating-systems.html#:~:text=Supported Platforms: 64-bit ARM, x86-64,,unofficial RISC-V support, and 32-bit INTERNET <1%
31	https://careerfoundry.com/en/blog/ui-design/what-is-a-user-interface/ INTERNET <1%
32	https://ieeexplore.ieee.org/document/9342816 INTERNET <1%
33	https://www.h2kinfosys.com/blog/three-approaches-for-the-requirement-analysis/ INTERNET <1%
34	https://www.softermii.com/blog/top-programming-languages-and-frameworks-for-software-development INTERNET <1%
35	https://projectsly.com/sprint-planning-in-project-management INTERNET <1%
36	https://dev.to/0xfedev/5-tools-for-analysing-the-security-of-solidity-smart-contracts-in-2022-39e1 INTERNET <1%
37	https://www.orientsoftware.com/blog/software-development-frameworks/#:~:text=A development framework is a reusable set of,so you don't have to write it yourself. INTERNET <1%
38	https://www.indusnet.co.in/leveraging-blockchain-for-secure-supply-chain-management/#:~:text=Blockchain technology is a distributed ledger technology where,higher transparency, better information flow, and enhanced safety. INTERNET <1%
39	https://blog.logrocket.com/truffle-suite-tutorial-develop-ethereum-smart-contracts/ INTERNET <1%
40	https://www.knowledgehut.com/blog/project-management/project-management-sprints#:~:text=In Scrum, a sprint is a set of,period in which minor targets are successfully achieved. INTERNET

<1%

41	https://www.mongodb.com/developer/products/atlas/unit-test-atlas-serverless-functions/ INTERNET <1%
42	https://ijrpr.com/uploads/V3ISSUE4/IJRPR3284.pdf INTERNET <1%

EXCLUDE CUSTOM MATCHES	ON
EXCLUDE QUOTES	OFF
EXCLUDE BIBLIOGRAPHY	OFF

Report Code: CS-BE Project 2022-23	**Report Number:** 02
Report Title: Decentralized Tender Management System for Road Development using Blockchain.	
Address Details: PVG's College of Engineering & Technology and GKPIM, Pune Pin- 411009, State: Maharashtra, INDIA.	

Atharv Vaspate Address: Karvenagar, Pune Mob: 7038884887 Email: 19100015@pvgcoet.ac.in Roll No: 9015	Niranjan Patankar Address: Sahakar Nagar-2, Pune Mob: 8605206788 Email: 19100066@pvgcoet.ac.in Roll No: 9066	Shankar Avad Address: Satara Road, Pune Mob: 8830592282 Email: 19100055@pvgcoet.ac.in Roll No: 9055	Aniket Datar Address: Karve Nagar, Pune Mob: 8380009808 Email: 19110065@pvgcoet.ac.in Roll No: 9065

Year: 2022-23

Branch: Computer Engineering

Keyword:

DTM - Decentralized Tender Management , RD - Road Development , BC – Blockchain , IoT - Internet of Things

Type of Report:	**Report Checked By:**	**Report Check Date:**	**Guide's Complete Name:**	**Total Copies**
Final			Prof. Manisha Marathe	5

Abstract:

The tender management process is a critical aspect of procurement and plays a vital role in ensuring fairness, transparency and efficiency in awarding contracts. However, traditional tender management systems often face challenges such as lack of trust, transparency, and vulnerability to corruption. This abstract introduces a decentralized tender management system that leverages blockchain technology to address these challenges.