SkillMesh

Final Year Project

Session 2020-2024

A project submitted in partial fulfillment of the degree of

BS in Computer Science

Department of Computer Science

Faculty of Computer Science & Information Technology

The Superior University, Lahore

Spring 2024

Type (Nature of project)	[ ] **D**evelopment [ ] **R**esearch [✓] **R&D**
Area of specialization	Blockchain
FYP ID	FYP-BCSM-F23-038

Project Group Members

Sr.#	Reg. #	Student Name	Email ID	*Signature
(i)	Bcsm-f20-079	Haseeb Ahmed	BCSM-F20-079@superior.edu.pk	
(ii)	Bcsm-f20-216	Sufyan Ahmed	BCSM-F20-216@superior.edu.pk	
(iii)	Bcsm-f20-087	Ali Hassan	BCSM-F20-087@superior.edu.pk	

*The candidates confirm that the work submitted is their own and appropriate credit has been given where reference has been made to work of others

Plagiarism Free Certificate

This is to certify that, I _____ S/D of_____, group leader of FYP under registration no _____at Computer Science Department, The Superior University, Lahore. I declare that my supervisor checks my FYP report.

Date: Name of Group Leader: _____ Signature: _____

Name of Supervisor: Mrs.Arshia Naeem

Designation: Lecturer

Signature: _____

HoD: Dr. Irfan-ud Din

Signature: _____

[SkillMesh]

Change Record

Author(s)	Version	Date	Notes	Supervisor's Signature
Haseeb Ahmed	1.0	5 JAN 2024	Initial Draft	

APPROVAL

PROJECT SUPERVISOR

Comments: _____

_____

Name:_____

Date:_____ Signature:_____

PROJECT MANAGER

Comments: _____

_____

Date:_____ Signature:_____

HEAD OF THE DEPARTMENT

Comments: _____

_____

Date:_____ Signature:_____

Dedication

This work is dedicated to Oppressed of Palestine

Acknowledgements

I am thankful to my Supervisor Arshia Naeem who has guided me in my project. The efforts of all team members in relating with the technology have highly been noted and praised. Sovereignly, we have managed to produce tangible results from the project, and that is truly a commendable achievement.

Executive Summary

SkillMesh is a **DeFi** project that aims to connect individuals who have specific skills and expertise with those who are seeking those skills, all while utilizing **blockchain** technology and **smart contracts** to facilitate secure, transparent, and efficient interactions.

The platform serves as a **marketplace** where users can offer their skills, services, and expertise to a global audience. Users who possess skills in areas such as graphic design, programming, writing, tutoring, translation, and more can create profiles and list the services they offer.

Service Providers: These are individuals with specific skills and services to offer. They create profiles highlighting their experience, portfolio, pricing, and availability. Service providers can set their rates in a preferred cryptocurrency or token.

Service Seekers: These are individuals or organizations seeking particular skills or services. They can search the platform for relevant service providers, view their profiles, and engage them for their required tasks

Table of Contents

Dedication v
Acknowledgements vi
Executive Summary vii
Table of Contents viii
List of Figures x
List of Tables x
Chapter 1 11
Introduction 11
1.1. Background 12
1.2. Motivations and Challenges 12
1.3. Goals and Objectives 13
1.4. Literature Review/Existing Solutions 14
1.5. Gap Analysis 14
1.6. Proposed Solution 15
1.7. Project Plan 15
1.7.1. Work Breakdown Structure 16
1.7.2. Roles & Responsibility Matrix 17
1.7.3. Gantt Chart 18
1.8. Report Outline 19
Chapter 2 21
Software Requirement Specifications 21
2.1. Introduction 22
2.1.1. Purpose 22
2.1.2. Document Conventions 22
2.1.3. Intended Audience and Reading Suggestions 22
2.1.4. Product Scope 24
2.2. Overall Description 25
2.2.1. Product Perspective 25
2.2.2. User Classes and Characteristics 26
2.2.3. Operating Environment 27
2.2.4. Design and Implementation Constraints 28
2.2.5. Assumptions and Dependencies 30
2.3. External Interface Requirements 30
2.3.1. User Interfaces 30
2.3.2. Hardware Interfaces 33
2.3.3. Software Interfaces 33
2.3.4. Communications Interfaces 35
2.4. System Features 36
2.4.1.1. Description and Priority 36
2.4.1.2. Stimulus/Response Sequences 37
2.4.1.3. Functional Requirements 39

2.5. Nonfunctional Requirements 39
2.5.1. Performance Requirements 40
2.5.2. Safety Requirements 40
2.5.3. Security Requirements 41
Chapter 3 45
Use Case Analysis 45
3.1. Use Case Model 47
3.2. Fully Dressed Use Cases 48
Chapter 4 51
System Design 51
4.1. Architecture Diagram 53
4.2. Domain Model 54
4.3. Entity Relationship Diagram with data dictionary 55
4.4. Sequence / Collaboration Diagram 56
4.5. Operation contracts 57
4.6. Activity Diagram 59
4.7. State Transition Diagram 61
4.8. Component Diagram 62
4.9. Deployment Diagram 63
4.10. Data Flow diagram [only if structured approach is used - Level 0 and 1] 64
Chapter 5 66
Implementation 66
5.1. Components, Libraries, Web Services and stubs 67
5.2. Deployment Environment 67
5.3. Tools and Techniques 68
5.4. Best Practices / Coding Standards 68
5.5. Version Control 69
Chapter 6 71
Testing and Evaluation 71
Chapter 7 76
Summary, Conclusion and Future Enhancements 76
7.1. Project Summary 77
7.2. Achievements and Improvements 77
Appendix A:Information / Promotional Material 3
Reference and Bibliography 5
Index 7

List of Figures

1.1	Work Breakdown Structure	16
1.2	Gantt Chart	18
2.1	Major Components	26
3.1	Use Case Model	47
4.1	System Design	52
4.2	Architecture Diagram	53
4.3	Domain Model Diagram	54
4.4	Entity Relationship Diagram with data dictionary	55
4.5	Sequence / Collaboration Diagram	56
4.6	Buyer Activity Diagram	59
4.7	Seller Activity Diagram	60
4.8	State Transition Diagram	61
4.9	Component Diagram	62
4.10	Deployment Diagram	63
4.11	Data Flow diagram Level 0	64
4.12	Data Flow diagram Level 1	64
4.13	Data Flow diagram Level 2	65
6.1	Login/Signup Testing	72
6.2	Login/Signup Testing	72
6.3	Login/Signup Testing	73
6.4	Gig Creation Testing	73
6.5	Wish List Testing	74
6.6	Wish List Testing	75
6.7	Wish List Testing	75
A1	Brochure	03
A2	Standee	04

List of Tables

1.1	Gap Analysis	14
1.2	Roles & Matrix	17
1.3	Empathy Map	20
2.1	Functional Requirements	39
2.2	Non Functional Requirements	39
5.1	Tools and Techniques	68

Chapter 1

Introduction

Chapter 1: Introduction

SkillMesh serve as digital marketplaces connecting businesses with freelance professionals globally. Their primary purpose is to enable access to a diverse talent pool, offering specialized skills for various projects and tasks. This platform promotes cost-efficiency, flexibility, and streamlined matchmaking, making it easier for both businesses and freelancers to find the right opportunities and talents, fostering a dynamic and interconnected gig economy. It is a **decentralized** platform and comes under the category of **DeFi (Decentralized Finance)**.

1.1. Background

Freelancing websites have transformed the traditional employment landscape by leveraging the power of the internet to connect freelancers with clients seeking a diverse array of skills and services. Freelancing websites continue to evolve further. After Elance and oDesk in the late 1990s, entrepreneurs created specialized platforms in the 2010s, with Fiverr and Toptal being the most popular. The 2020s also bring new challenges and opportunities. The COVID-19 pandemic has made remote working more popular, increasing the demand for freelancing platforms. They address new issues, including payment security and utilize new technologies such as blockchain. In general, the future of work may involve more freelancers and temporary employees who find their job there. Freelancing websites optimize the search for gigs for workers and give businesses access to talent worldwide.

1.2. Motivations and Challenges

Increased security and trust: The blockchain enables decentralized platforms to increase safety due to transparent and unchangeable transaction records. Users can rely on the honesty of the payment system and have their payments conducted fairly with minimal possibility of fraud or manipulation.

Lower fees and fair payment: Without intermediate brokers, the platforms could potentially decrease the cost of transaction fees. Smart contracts and decentralized systems can automate payment processes, ensuring fair and timely remuneration for freelancers.

Decrease of centralized authority: In a decentralized market, there is no center authority governing the processes. Therefore, there is no single entity to collaborate with to enforce and execute decisions.

Usability: For the platform to be adopted, the need for a user-friendly interface must be guaranteed.

Quality Control: A high standard of work quality must be set and no low-quality service should be offered.

Regulatory Considerations: There might be regulatory struggles based on the platform's jurisdiction and users' autonomous region.

Dispute Resolution: Implementing an effective and fair dispute resolution mechanism is necessary for maintaining user trust.

1.3. Goals and Objectives

A decentralized freelancing website aims to revolutionize the gig economy by leveraging blockchain technology and decentralized principles to create a platform that prioritizes transparency, security, and fairness for both freelancers and clients. The primary goals and objectives of such a platform include:

- **Blockchain Security and Transparency:** Implementing blockchain ensures secure and transparent transactions. Smart contracts, which are self-executing contracts with the terms of the agreement directly written into code, can automate payments and provide a trustworthy system for freelancers to receive compensation for their work.
- **Reduced Intermediary Dependency**: Decentralized freelancing platforms seek to minimize reliance on intermediaries, eliminating the need for traditional banking systems or third-party payment processors. This can lead to lower transaction fees and faster payment processing times, benefitting both freelancers and clients.
- **Community Governance:** Decentralized platforms often incorporate community-driven governance models, allowing users to have a say in the decision-making processes. This democratized approach ensures that the platform evolves in a way that aligns with the needs and preferences of its user base.
- **Identity Verification and Reputation Systems:** Leveraging blockchain for identity verification can enhance security and build trust among users. A decentralized reputation system can allow freelancers to showcase their skills and reliability, helping clients make informed decisions when selecting service providers.
- **Global Accessibility:** By leveraging blockchain's borderless nature, decentralized freelancing platforms can provide equal opportunities to freelancers from around the world, promoting inclusivity and diversity in the gig economy.
- **Resilience to Censorship:** Decentralized platforms aim to be resistant to censorship, ensuring that freelancers and clients have the freedom to engage in transactions without being subject to arbitrary restrictions.
- **Fair Compensation and Dispute Resolution:** Smart contracts and decentralized dispute resolution mechanisms can ensure that freelancers receive fair compensation for their work, and

conflicts are resolved impartially, fostering a more equitable and trustful freelancing environment.

1.4. Literature Review/Existing Solutions

Freelancing Platforms are charging too much profit from each order up to 15% to 25% from sellers as well as from buyers. Due to the above problem, many users find clients on the platform and deal outside the platform to avoid giving so much profit to the platform.
As the platforms are, centralized one of the negative risks is unauthorized access, which is a common problem across the centralized platforms.

1.5. Gap Analysis

There are multiple Freelancing websites in existence today but all are centralized and charging many fees from sellers as well as from buyers.

Some of the freelancing websites are mentioned in the following table along with the charging fesses.

Platform	Fees for Freelancer	Fees for Clients	Realm
Fiverr	20% on all orders	5.5% on all orders $2.50 additional charge on orders under $75	Centralized
Upwork	10% on all orders	5% 3% for eligible US clients paying via ACH (+$3 per $100)	Centralized
PeoplePerHour	20% for $0-350 7.5% for $350.01-7,000 3.5% for $7,000+	10% + $0.70	Centralized
Freelancer.com	10% or $5 (whichever is greater) 10% per milestone for hourly rate jobs	3% or $3 (whichever is greater) 3% per milestone for hourly rate jobs	Centralized
SkillMesh	5% on all orders	No charges	Decentralized

Table: 1.1 Gap Analysis

1.6. Proposed Solution

In **SkillMesh,** we will only charge 5% form the sellers and nothing from the buyers.

As our platform is **decentralized,** we use digital wallets to get access to the platform result in the elimination of negative risk of unauthorized access to the account.

1.7. Project Plan

The SkillMesh website project has been carefully designed to use modern technology in order to provide a strong and safe platform for freelancers. The frontend development will be driven by Next.js, ensuring a responsive and engaging user interface. The integration of Node.js and Express.js will establish a robust and expandable backend infrastructure. MongoDB is selected as the database option because it allows for greater flexibility and efficiency in data management. To make the site safer and more open, the use of blockchain will be seamlessly added. Smart contracts will be used for deals that are safe and can be checked. This entire technical stack seeks to provide a smooth and unique user experience for freelancers and clients in the decentralized SkillMesh ecosystem.

1.7.1. Work Breakdown Structure

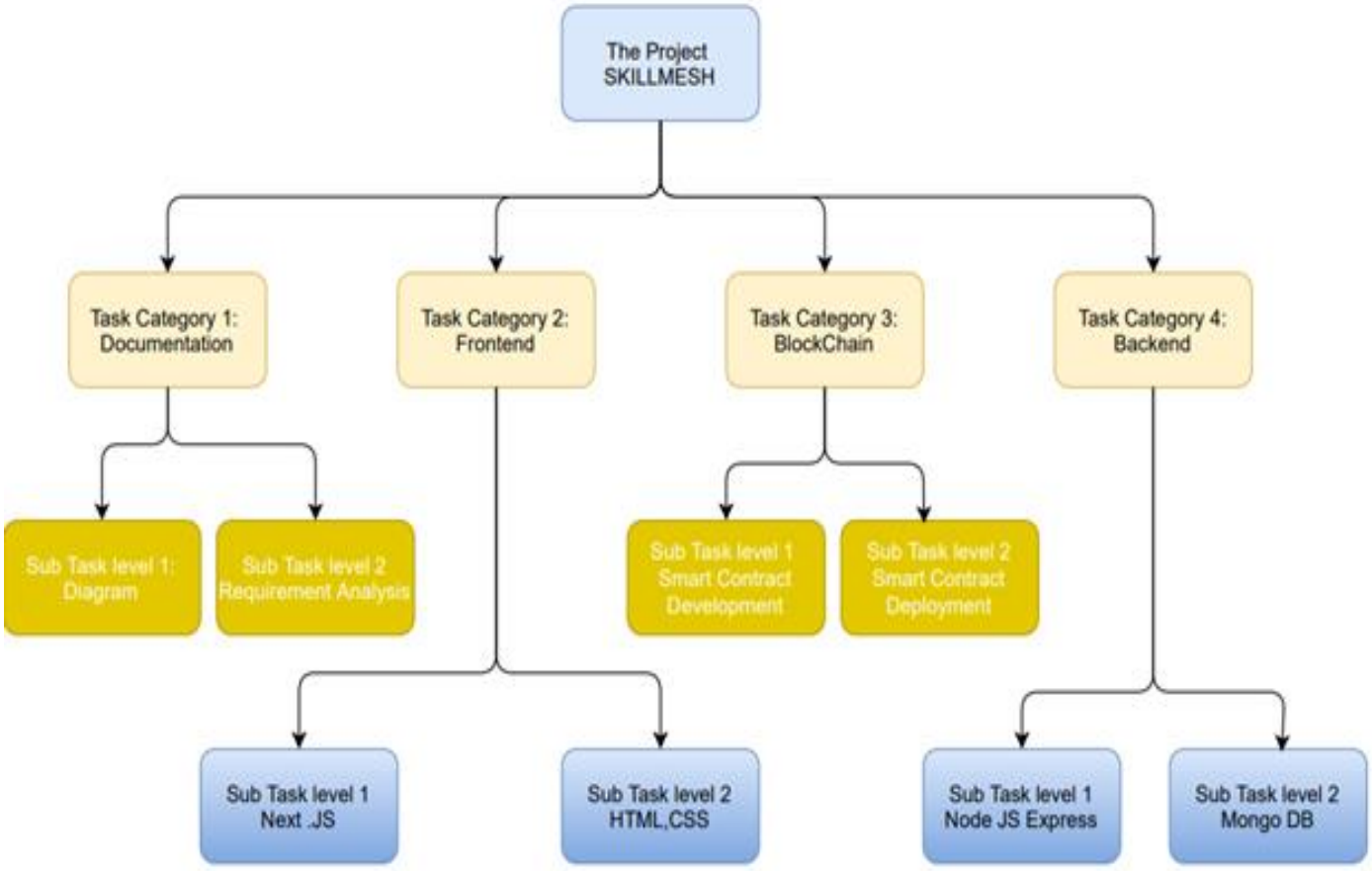


Figure: 1.1 Work Breakdown Structure

1.7.2. Roles & Responsibility Matrix

The roles and responsibilities matrix is used to specify who is responsible for what.

WBS #	WBS Deliverable	Activity #	Activity to Complete the Deliverable	Duration (# of Days)	Responsible Team Member(s) & Role(s)
01	Gather requirements	1	Gather data from the Research paper	15	Ali Hassan
02	Analysis for project	2	Understand the user's need	6	Ali Hassan Haseeb Ahmed
03	UI/UX Design	3	How does the website look like	30	Haseeb Ahmed
04	Front–End Development	4	Building a Website	15	Ali Hassan Haseeb Ahmed Sufyan Ahmed
05	Back–End Development	5	NodeJs, ExpressJs, MongoDB	55	Sufyan Ahmed
06	Blockchain	6	Solidity	15	Sufyan Ahmed Haseeb Ahmed
07	Testing and Debugging	7	Test Module	15	Ali Hassan Haseeb Ahmed
08	Deployment	8	Deploying our website	10	Ali Hassan Haseeb Ahmed Sufyan Ahmed
09	Documentation	9	Final Documentation	15	Ali Hassan Haseeb Ahmed

Table: 1.2 Roles & Matrix

1.1.1. Gantt chart

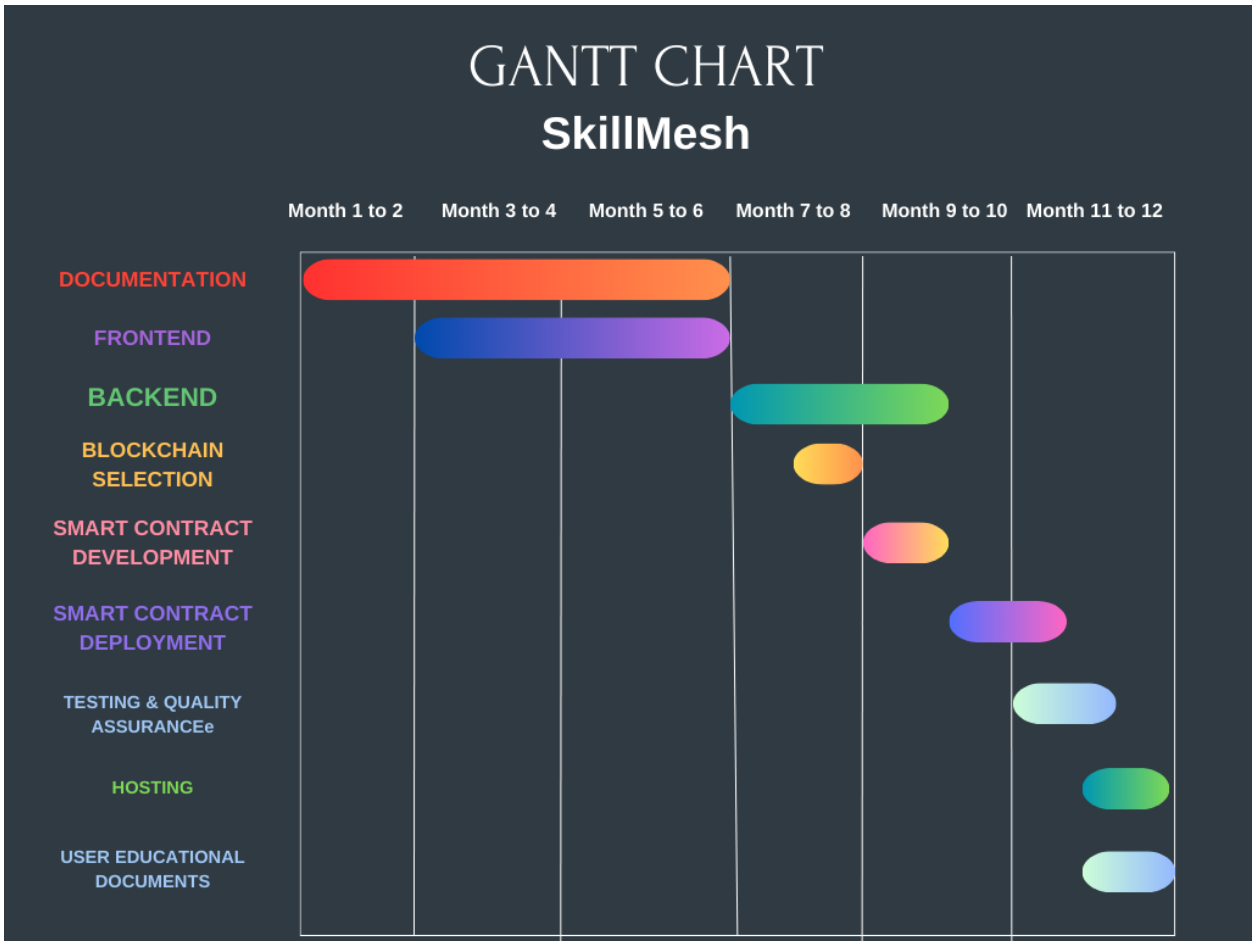


Figure: 1.2 Gantt chart

1.2. Report Outline

SkillMesh, a breakthrough decentralized program, is ready to transform the freelance environment by employing cutting-edge technology. SkillMesh assures efficient and transparent transactions through the integration of smart contract and blockchain functionalities on the Ethereum platform. The technological stack, which includes NextJs, MongoDB, Tailwind CSS, Sass, Solidity, Node.js, Express.js, and Typescript, demonstrates a dedication to robust and scalable development.

The platform's distinguishing features include simple account setup, seller account administration, gig creation, and wish-listing capabilities. Its search function enables purchasers to easily explore through categories and subcategories, picking services that meet their demands.

SkillMesh distinguishes itself by charging a minimal 5% charge to both vendors and buyers, a clear and equitable strategy for supporting the platform.

As a Software Engineer, this report outline serves as a comprehensive roadmap, encapsulating the project's architecture, design principles, database schema, smart contract implementation, testing strategies, security measures, performance optimization, deployment processes, and ongoing maintenance plans. Through this meticulous approach, SkillMesh aims to deliver a seamless and secure freelancing experience while pushing the boundaries of decentralized applications in the digital economy.

1.3. Empathy Map

Says: Users express satisfaction with SkillMesh transparent and fair fee model, acknowledging the platform's commitment to a decentralized approach. Positive comments include appreciation for the nominal 5% fee from both sellers and buyers, highlighting the appeal of SkillMesh innovative and equitable pricing structure.	**Thinks:** Users recognize the advantages of blockchain technology for secure and transparent transactions on SkillMesh. They contemplate the potential impact of decentralization on usability and governance effectiveness. Questions arise regarding how the platform compares to centralized alternatives in terms of user experience and whether the claimed resilience to censorship holds true.
Does: SkillMesh users engage confidently in transactions, exploring services with ease through the efficient search mechanism. They actively participate in the platform's community-driven governance, expressing a desire to have a say in decision-making processes. Users are motivated to verify their identity and build a reliable reputation within the decentralized system.	**Feel:** Users feel empowered by SkillMesh commitment to transparency, security, and fairness. There is a sense of trust in the platform's unique 5% fee structure for both sellers and buyers. However, concerns linger about the impact of decentralization on usability and the effectiveness of governance and dispute resolution mechanisms, indicating a need for clear communication and reassurance from the platform.

Table:1.3 Empathy Map

Chapter 2

Software Requirement Specifications

Chapter 2: Software Requirement Specifications

2.1. Introduction

2.1.1. Purpose

The SRS outlines the requirements for the entire decentralized freelancing platform. This includes all the core functionalities and features essential to the operation and performance of the platform. The system covered by this SRS represents the entire ecosystem that facilitates interactions between freelancers and clients through a decentralized, blockchain-based network.

2.1.2. Document Conventions

The First Letter of my new **paragraph** is **Capital** because it is showing up the new paragraph starting from here and the only more thing I implemented here. That is one is **Bold** the word. Where I did **bold** the word, it means that is word is more important in this paragraph.

2.1.3. Intended Audience and Reading Suggestions

Intended Audience:

- **Developers:** Developers will find detailed information on the technical architecture of the decentralized freelancing platform, including blockchain integration, smart contract implementation, and any specific coding requirements. Sections on API documentation, protocols, and data structures will be pertinent to this audience.

- **Project Managers:** Project managers will be interested in understanding the high-level goals, timelines, and resource requirements for developing and maintaining the decentralized freelancing platform. They should focus on sections outlining project milestones, key deliverables, and dependencies.

- **Marketing Staff:** Marketing personnel will focus more on how to make the platform popular and reach much larger audiences. Selling points, potential clients whose preferences are going to be targeted and marketing strategies would be of great concern to them. Besides, they will want to acquire users, as well as their consequent commitment to the platform.

- **Users:** Users for instance freelancers and clients will tend to choose options that are concerned with the user experience, the onboarding process and the security of their transactional processes. In short, onboarding of users is security of transactions. On the other hand, information about the token system wallet configuration, withdrawals and grievance redressal will be an important part of it as well.

- **Testers:** The testers will be provided with a detailed test case, scenarios, and expected outcomes while going through the product features. In addition to that, is security testing, performance testing, and user acceptance testing information will be essential for this group.
- **Documentation Writers:** The writing of documentation for the platform will concentrate on chapters that explain the user side of the platform: guides on how to use the program, FAQs, and the handbook on any probable issues and solutions. Easy-to-understand and unambiguous documentation is undoubtedly an important ingredient that sets users on the right track of using the system exclusively.

Organization of the Software Requirements Specification (SRS):

- **Introduction:** In general, the introduction serves to highlight the main objective of the platform and, as a means of achieving this, it is planned to use a decentralized architecture, the integration of blockchain and the advantages of the system for both freelancers and clients.

- **Overall Description:** Briefly explaining, this section features the essential parts such as whopping contracts, token structure, and user-friendly design to reveal the ideals of this platform that are decentralized economic model, security and transparency.

- **External Interface Requirements:** Detailing interactions, this section outlines user interfaces, third-party integrations, and blockchain interactions, emphasizing transparency, security measures, and communication channels.

- **System Features:** This part closely looks at user activities like signup process, order placing, and real-time communication, in addition to features like a decentralized reputation system, transparent feedback, and strong dispute resolutions tools by checking the distribution of different security system among user.

- **Other Non-Functional Requirements:** This section includes non-functional issues like security, performance (internet speed), user experience and regulatory frameworks that would cover critical areas such as reliability, compatibility and fault tolerance to ensure the platform performance is measured against good practice and user satisfaction.

Sequence for Reading:

- **All Readers:** Firstly, go through the Overview section in order to comprehend the main idea and objectives of the platform.
- **Developers:** Go on to the system architecture part for technical aspects.
- **Project Managers:** Move on to the Project Timeline and Milestones section for information on project planning.
- **Marketing Staff:** Focus on the Marketing and User Acquisition section to understand strategies for promoting the platform.
- **Users:** Explore the User Experience and Onboarding section for guidance on using the platform effectively.
- **All Readers:** Review the Payments section for information on the platform's economic model.
- **All Readers:** Examine the Security Features section for insights into the platform's security measures.
- **Testers:** For details of the Testing Procedure section, please refer to it.
- **Documentation Writers:** Therefore, it is important to go through the Documentation Guidelines chapter to be able to prepare user documentation that would be easy to understand.

2.1.4. Product Scope

The functionality of a decentralized freelancing website comprises of the key features that the platform requires in order to operate. It will include user registration and profile management for freelancers, clients and administrators. It involves a platform whereby clients post job and projects with having search and matchmaking tools that hook up the right users to the right job/project. Communication tools like messaging, document sharing, and real time conferencing may facilitate the process of collaboration. A secure payment system based on the blockchain technology provides decentralized, transparent, automatic and reliable transactions overall market. Furthermore, the platform has a rating system for members, a remedy system, security strong measures, an administrative back-end, a user-friendly interface, and compliance with legal and regulatory requirements in the freelancer industry.

2.2. Overall Description

2.2.1. Product Perspective

SkillMesh arises as a standalone and novel product, which is designed to alter the gig economy paradigm through decentralized architecture and blockchain-based functionalities. It should be emphasized that SkillMesh is not meant to be a replacement for existing systems but rather, as a trailblazing, standalone platform. Achieving such a vision lies at the heart of SkillMesh founding mission, which focuses on creating a marketplace with increased security and control, as well as operational efficiency for both suppliers and consumers.

SkillMesh is the standalone module that distinguishes its core functions from other fully-fledged systems. Besides, it could be compatible with the other systems and state of the art technologies, which helps to make its operations more effective.

Major Components:

Frontend (Next.js): The key component for this is the human interface that stage-manages the user-platform relationship to ensure that the end user has an advanced user interface.
Backend (Node.js with Express.js): As the crux of server-side logic, this element adeptly manages incoming requests, orchestrates data processing, and interfaces seamlessly with the database.
Database (MongoDB): It is due to this storage format that the user profiles, gig details, and transactional data are stored and retrieved securely and at the same time, scalable.
Media Management (Cloudinary): A well-planned, external server component provides a centralized approach to store and retrieve media files tightly connected to listed gigs.

Blockchain Integration (Solidity): SkillMesh is using Ethereum blockchain to create some contracts with the smart contract technology providing the secure and transparent transactions.

User Interface Styling (Tailwind CSS and Chakra UI): These tech-savvy libraries when properly deployed acts as the final changes in producing the aesthetic and responsive design pivotal in SkillMesh user interface.

Subsystem Interconnections:

SkillMesh is designed to mimic real-life systems, whereby subsystems join and collaborate to achieve a successful operation:

- The frontend communicates with the backend to send and receive user requests.
- The backend interacts with the database to store and retrieve user data, gig information, and transaction detail.
- Media management through Cloudinary ensures the efficient handling of images and other media associated with gigs.

- Blockchain integration, as it applies to smart contracts, ensures a transaction with safety and entrepreneurship.

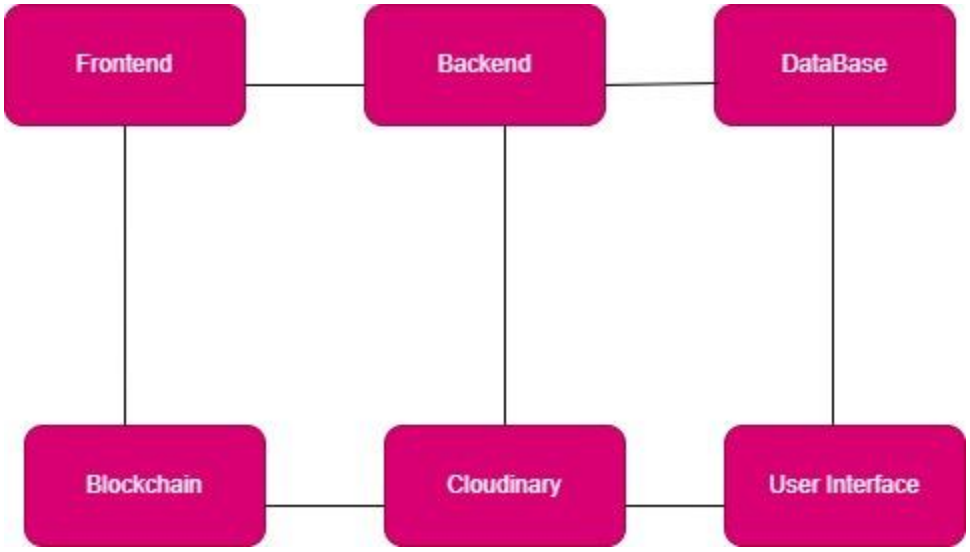


Figure: 2.1 Major Components

2.2.2. User Classes and Characteristics

1. Freelancers:

- Possess specialized skills in various domains.
- Can offer services and create gig listings.
- May include graphic designers, writers, programmers, etc.
- Let profiles showing off their skillset and work histories be made.

2. Clients/Buyers:

- Find freelancers for particular jobs or tasks.
- Create post job listings and place service orders.
- Evaluate freelancers by profiles, reviews, and ratings.
- Freelancers can be reached at any given point.

3. Administrators:

- Ensuring the viability of the platform by overseeing and running its operations.
- Have the possibility of making use of office suitable equipment and resources.
- Monitor the user activities to guarantee the platform integrity.
- Address and resolve issues or disputes if any.

4. Visitors/Unregistered Users:

- Pages Accessed without a User Signup.
- Restrictions to aspect of the applications and utilities that are not available.
- It could happen that they register as users by creating an account.

5. System Moderators

- Moderate content and interactions within the platform.
- Ensure compliance with platform guidelines and policies.
- Address reported issues and maintain a positive community.

6. Help/Support:

- The users who might be stuck or have any doubts to be helped.
- Answer the questions that are inquiry related to the platform features or transactions.
- Help in resolving disputes and creating a good user experience.

7. Blockchain Validators

- Verify and validate transactions on the blockchain.
- Ensure the security and integrity of decentralized operations.
- Play a role in the execution of smart contracts.

2.2.3. Operating Environment

1. **Web 3.0 Infrastructure:**
    - SkillMesh operates in a Web 3.0 environment, characterized by decentralized protocols, peer-to-peer interactions, and blockchain technology.
    - Utilizes decentralized web standards for increased security, transparency, and user control over data.
2. **Blockchain Network:**
    - Relies on a blockchain network, likely Ethereum or a similar decentralized platform, for smart contract execution, secure transactions, and transparent record keeping.
    - Involves nodes, validators, and smart contract interactions within the blockchain.
3. **Decentralized Identity Management:**
    - Implements decentralized identity solutions based on blockchain for secure and user-controlled identity verification.
    - Users have control over their identity information, enhancing privacy and security.

4. **Smart Contracts:**
    - Smart contracts are deployed on the blockchain to automate and execute agreements between users.
    - They govern processes like payments, order fulfillment, and dispute resolution in a trustless manner.
5. **Cryptocurrency Integration:**
    - Involves the use of cryptocurrencies as a means of payment for services on the platform.
    - Smart contracts handle cryptocurrency transactions securely and transparently.
6. **Decentralized Storage:**
    - Leverages decentralized storage solutions to store and retrieve user data in a distributed manner.
    - Enhances data security, resilience, and availability.
7. **Web 3.0 Browser Compatibility:**
    - Users access SkillMesh using Web 3.0 compatible browsers that support decentralized protocols and blockchain interactions.
    - Ensures a seamless and secure user experience in the decentralized web.

2.2.4. Design and Implementation Constraints

1. **Blockchain Network Constraints:**
    - **Constraint:** Relies on a specific blockchain network (e.g., Ethereum).
    - **Impact:** Integration challenges if the chosen blockchain faces scalability or network congestion issues.
2. **Smart Contract Limitations:**
    - **Constraint:** Smart contracts are used for platform operations.
    - **Impact:** Changes to smart contracts may require careful consideration and testing to avoid disruptions.

3. **Cryptocurrency Volatility:**
    - **Constraint:** Uses cryptocurrencies for transactions (e.g., Ether).
    - **Impact:** Users may be exposed to the volatility of cryptocurrency values, affecting transaction amounts and user behavior.
4. **Interoperability Challenges:**
    - **Constraint:** Requires interoperability with various blockchain tools and standards.
    - **Impact:** Ensuring compatibility with different blockchain networks, wallets, and decentralized applications can be complex.
5. **User Education Requirements:**
    - **Constraint:** Users need to understand blockchain concepts and decentralized operations.
    - **Impact:** Requires comprehensive educational resources and user support to navigate the decentralized environment.
6. **Scalability Considerations:**
    - **Constraint:** Blockchain networks may face scalability challenges.
    - **Impact:** Potential limitations on the number of transactions and user interactions, especially during periods of high demand.
7. **Security and Auditing:**
    - **Constraint:** Security is paramount due to the use of smart contracts and decentralized infrastructure.
    - **Impact:** Requires continuous auditing and testing to identify and address security vulnerabilities.
8. **Web 3.0 Browser Compatibility:**
    - **Constraint:** Relies on Web 3.0 compatible browsers.
    - **Impact:** Users must use browsers that support decentralized protocols, potentially limiting accessibility for some users.

2.2.5. Assumptions and Dependencies

Assumed factors that could significantly impact the requirements stated in the SRS for website include:

1. **Blockchain Technology Assumptions:** If the assumed functionality or capability of blockchain systems for transaction security, scalability, or interoperability differs from reality, it can significantly affect the integration and performance of the platform.

2. **Third-Party API Reliability:** Dependencies on third-party services, such as blockchain platforms, payment gateways, or communication tools. Any unforeseen changes, service interruptions, or reliability issues with these external components could impact the platform's functionalities.

3. **Regulatory and Compliance Changes:** Assumptions about legal and regulatory compliance may change over time, affecting how the platform operates within different jurisdictions, especially regarding financial transactions, data protection, and privacy laws.

4. **Operating Environment Stability:** Changes or constraints within the operating environment, such as browser updates, device compatibility issues, or evolving security standards, can affect the software's performance and user experience.

5. **Development Tool Suitability:** Assumptions regarding the effectiveness and compatibility of chosen development tools, programming languages, or methodologies may prove incorrect, impacting the development process and timeline.

6. **Market and User Behavior Assumptions:** Assumptions about user behavior, market trends, or the adoption of decentralized platforms could influence the acceptance and success of the platform.

2.3. External Interface Requirements

2.3.1. User Interfaces

SkillMesh aims to deliver a seamless and intuitive user experience through well-designed user interfaces. The platform's user interfaces cater to various user classes, including freelancers, clients, administrators, and visitors. Key aspects of the user interfaces include:

1. **User-Friendly Account Creation:**

    - **Objective**: Enable easy registration for freelancers and clients.
    - **Design Features:** Intuitive sign-up forms, clear instructions, and on-screen guidance.
    - **Benefits**: Streamlined onboarding, encouraging user engagement.

2. **Seller Account Management:**
    - **Objective:** Provide freelancers with tools to manage their accounts and showcase their skills.
    - **Design Features:** Intuitive profile management, portfolio uploads, and skill categorization.
    - **Benefits:** Empowers freelancers to present their expertise effectively.
3. **Gig Creation and Management:**
    - **Objective:** Allow freelancers to create and manage gig listings.
    - **Design Features:** Easy-to-use gig creation forms, multimedia uploads, and scheduling options.
    - **Benefits:** Efficient gig management and increased visibility for freelancers.
4. **Search and Matchmaking Tools:**
    - **Objective:** Facilitate clients in finding suitable freelancers for their projects.
    - **Design Features:** Robust search functionalities, filters based on skills and categories.
    - **Benefits:** Efficient and targeted talent discovery for clients.
5. **Real-Time Communication:**
    - **Objective:** Enable seamless communication between freelancers and clients.
    - **Design Features:** Messaging systems, file sharing capabilities, and potentially video conferencing.
    - **Benefits:** Enhances collaboration and project coordination.
6. **Transparent Feedback and Ratings:**
    - **Objective:** Foster a transparent feedback system.
    - **Design Features:** Ratings and review mechanisms, clear feedback forms.
    - **Benefits:** Builds trust, allows users to make informed decisions.
7. **Decentralized Reputation System:**
    - **Objective:** Showcase freelancers' reliability and skills.
    - **Design Features:** Blockchain-based reputation tracking, visible performance metrics.

- **Benefits:** Enhances user trust and credibility.

8. **Secure Payment System:**
    - **Objective:** Ensure secure and transparent financial transactions.
    - **Design Features:** Integration of blockchain for payment processing, clear transaction histories.
    - **Benefits:** Builds confidence minimizes payment-related risks.
9. **Administrative Backend:**
    - **Objective:** Provide administrators with tools for platform management.
    - **Design Features:** Dashboard with analytics, user management, and content moderation.
    - **Benefits:** Streamlines administrative tasks for platform oversight.
10. **Responsive User Interface:**
    - **Objective:** Ensure accessibility across devices and browsers.
    - **Design Features:** Responsive design, cross-browser compatibility.
    - **Benefits:** Enhanced user experience on various platforms.

These user interfaces collectively contribute to **SkillMesh** commitment to delivering a user-centric and efficient **freelancing experience**. The design prioritizes simplicity**, transparency**, and accessibility for all users interacting with the platform.

2.3.2. Hardware Interfaces

SkillMesh, being a decentralized platform built on web 3.0 principles, primarily interacts with standard hardware components that support web browsing and blockchain transactions. The hardware interfaces are designed to ensure compatibility and optimal performance for users accessing the platform. Key considerations for hardware interfaces include:

1. **Web Browsing Devices:**
    - **Compatibility:** SkillMesh is accessible through a wide range of devices, including desktop computers, laptops, tablets, and smartphones.
    - **Requirements:** Standard hardware configurations capable of running modern web browsers.
2. **Web 3.0 Compatible Browsers:**
    - **Compatibility:** SkillMesh interfaces seamlessly with browsers that support web 3.0 principles and decentralized protocols.
    - **Requirements:** Users are encouraged to use browsers like MetaMask, Brave, or others compatible with decentralized applications (DApps).
3. **Blockchain Wallets:**
    - **Compatibility:** Users need hardware that supports blockchain wallet integration.
    - **Requirements:** Compatible hardware wallets or software wallets (e.g., MetaMask) for secure cryptocurrency transactions.

2.3.3. Software Interfaces

The software interfaces for the SkillMesh platform can be categorized into various components, each serving a specific purpose:

1. **User Interface (UI):**
    - The UI will be designed using Next.js for a responsive and engaging experience.
    - Intuitive navigation, user-friendly forms, and visually appealing layouts will enhance user interaction.
    - Components for profile management, project creation, and search functionalities will be integrated.

2. **Application Programming Interface (API):**
    - RESTful APIs will be developed using Node.js and Express.js to facilitate communication between the frontend and backend.
    - Endpoints for user authentication, project management, payment processing, and other core functionalities will be exposed.
3. **Database Interface:**
    - MongoDB will serve as the database backend.
    - MongoDB drivers or Object-Document Mapping (ODM) libraries will be used to interact with the database.
    - Queries for user data, project details, and transaction records will be handled through the database interface.
4. **Blockchain Integration Interface:**
    - Smart contracts on the blockchain will be interacted with through a blockchain integration layer.
    - Web3.js or similar libraries will be used to communicate with the blockchain, executing smart contract functions related to payment processing and transaction verification.
5. **Payment Gateway Interface:**
    - Integration with a secure and reliable payment gateway will be implemented.
    - APIs from the payment service provider will be utilized for handling financial transactions securely.
6. **Notification Interface:**
    - Integration with a notification service will be implemented for real-time updates.
    - Email notifications, in-app messages, or push notifications may be used to keep users informed about project updates, messages, and other relevant activities.

7. **Admin Interface:**
    - An administrative dashboard or interface for platform administrators to manage users, projects, and resolve disputes.
    - Access controls and security measures will be implemented to ensure the integrity of administrative functions.

2.3.4. Communications Interfaces

1. **Frontend-Backend Communication:**
    - RESTful APIs for communication between frontend (Next.js) and backend (Node.js/Express.js).
    - JSON data format for request and response payloads.
    - HTTPS protocol for secure data transmission.
2. **Database Communication:**
    - MongoDB drivers or ODM (Object-Document Mapping) libraries for CRUD operations.
    - Encrypted connections for data privacy and security.
    - Asynchronous queries to optimize database interactions.
3. **Blockchain Integration:**
    - Web3.js library for interaction with smart contracts on the blockchain.
    - Ethereum as the preferred blockchain platform.
    - Secure and authenticated communication channels for transaction processing.
4. **Payment Gateway Integration:**
    - Integration with a secure payment gateway API.
    - SSL/TLS for secure data transmission during financial transactions.
    - Handling response codes and callbacks for transaction status.
5. **Authentication and Authorization:**
    - OAuth or JWT (JSON Web Token) for secure user authentication.
    - Token-based authentication for API access.
    - Implementation of OAuth/Open ID Connect flows for user authorization.

2.4. System Features

1. **User Registration:** Enable users to register on the SkillMesh platform.
2. **User Authentication:** Provide secure login functionality.
3. **Profile Management:** Users will be able to register, update and manage their profiles.
4. **Real-time Communication:** Implement in-platform communication, file sharing.
5. **Peer-to-Peer Transaction:** Introducing a blockchain model for the safe and secure record keeping.
6. **Transaction History:** The platform should display users' transactions or history of transactions.
7. **Rating and Review System:** Give users the ability to give feedback and overall ratings for each other.

2.4.1.1. Description and Priority

1. **User Registration:**

   Description: SkillMesh platform functions this way: for people who have predetermined what skill they want to work on can do so by opening a user account on the platform after they have logged in.

   Priority: High

2. **User Authentication:**

   Description: The SkillMesh solution brings secure personal identification with high encryption tools that keep user's identity hidden.

   Priority: High

3. **Profile Management:**

   Description: Users can build their own profiles, which can then be modified and managed according to personal information (skills and experience), and other related information.

   Priority: Medium

4. **Real-time Communication:**

   Description: SkillMesh is a software platform that provides a real-time communication through in-platform messaging, file sharing which are much more effective, compared to the existing systems.

Priority: High

5. **Peer-to-Peer Transaction:**

   Description: The system combines block chain technology for safe and transparent peer-to-peer transactions, which will in turn form trust and reliability among users.

   Priority: High

6. **Transaction History:**

   Description: Users will see all transaction history when they log in, this will offer a complete and easy to read transaction history.

   Priority: Medium

7. **Rating and Review System:**

   Description: A rating and a comments system is another tool belongs to SkillMesh that allows users to express their opinion about the services to improve the transparency and credibility of the platform.

   Priority: Medium

2.4.1.2. Stimulus/Response Sequences

1. **User Registration:**

   Stimulus: The user presses the "Register" button.

   Response: SkillMesh consists in a form that must be filled by the user with necessary data like Username, MetaMask. Upon uploading, the system runs the data validation and saves it as a new user's account.

2. **User Authentication:**

   Stimulus: Enter an identification, click on the "Login" button.

   Response: In SkillMesh, the data provided by the user is verified against the stored information. Logged on the system, the user makes their way to their dashboard as well as all the other available functions.

3. **Profile Management:**

   Stimulus: User tap on "Profile" section.

   Response: On SkillMesh, users can choose create, edit, or maintain their profiles. The platform updates the user's info in the system.

4. **Real-time Communication:**

   Stimulus: To ensure starts up a chat messaging.

   Response: SkillMesh will have a direct and real communication channel that enables them to forward messages to one another, share files.

5. **Peer-to-Peer Transaction:**

   Stimulus: User can confirm project completion.

   Response: SkillMesh performs smart contract laying down peer-to-peer transaction. The system sends a progress report to the client and if the status is complete, it initiates a payment to the freelancer.

6. **Transaction History:**

   Stimulus: User can click on the "Transaction History" tab.

   Response: On SkillMesh, payments for projects you are part of, your earnings from the work done, and financial activities that are relevant to you and here clearly visible.

7. **Rating and Review System:**

   Stimulus: User complete projects.

   Response: SkillMesh encourages the user to evaluate the freelancer's open these up in a 5-star rating and review system. The system saves feedback of clients to the freelancer' rating.

2.4.1.3. Functional Requirements

No.	Name	Functional	Description	Actor
1.	User Registration	Yes	Enable users to register on the SkillMesh platform	User
2.	User Authentication	Yes	Provide secure login functionality	User
3.	Profile Management	Yes	Allow users to create, edit, and manage profiles	User
4.	Real-time Communication	Yes	In-platform messaging, file sharing, and video conferencing	User
5.	peer to peer transaction	Yes	Implement blockchain for secure and transparent transactions	System
6.	Transaction History	Yes	Provide users with a history of transactions	User
7.	Rating and Review System	Yes	Allow users to rate and review each other	User
8.	Identity Verification	Yes	Use blockchain for user identity verification	System
9.	Global Accessibility	Yes	Ensure the platform is accessible worldwide	User
10.	Resilience to Censorship	Yes	Design the platform to resist censorship	User
11.	Dispute Resolution	Yes	Admin will resolve Disputes	Admin
12.	Usability and User Experience	Yes	Ensure a user-friendly interface	User
13.	Quality Control	Yes	Implement mechanisms to maintain high work quality	System
14.	Regulatory Compliance	Yes	Address regulatory challenges based on jurisdictions	System

Table: 2.1 Functional Requirements

2.5. Nonfunctional Requirements

No.	Name	Non-Functional	Description	Actor
1.	Privacy	Yes	Ensure user privacy is maintained securely	System
2.	Validity	Yes	Handle errors effectively for a robust system	System
3.	Performance	Yes	Optimize application performance for efficiency	System
4.	Usability	Yes	Ensure the platform is user-friendly for everyone	System
5.	Reliability	Yes	Build trustworthiness in the system for users	System
6.	Security	Yes	Implement robust security measures for data protection	System
7.	Scalability	Yes	Design the platform to scale with increasing users and data	System
8.	Availability	Yes	Ensure high availability for uninterrupted service	System
9.	Compatibility	Yes	Make the platform compatible with various devices and browsers	System
10.	Compliance	Yes	Adhere to legal and regulatory compliance standards	System

Table: 2.2 N Nonfunctional Requirements

2.5.1. Performance Requirements

SkillMesh will give top priority on performance, as a good delivery of a freelancing service requires seamless and effective operations. The system shall be optimized for a quick response client, which guarantees that user interactions, like clicks and queries, is fulfilled in 1-3 seconds. Concurrency has been also acknowledged with designing the platform that will support many users in processes like browsing projects, real-time communication and concurrent transactions occurrence without any performance degradation. It is possible to link Scalability with SkillMesh - a system that is ready to serve users, who are growing, the volume of projects is also increasing. The infrastructure is designed to handle 1000 transactions per minute, this high throughput we make made possible with robust performance characteristics. Data looking up from the system is designed to take approximately 2 seconds for contacts of a user and projects data. SkillMesh emphasizes that it targets the settlement of blockchain transactions within 15 minutes indicating its confirmation times in the mission. At the target, Availability level set of a minimum 99.9% and with scheduled maintenance while unplanned outages are minimized effectively. Error handling mechanisms pace, secure data methods and cross-browser (cross-device) compatibility are the key SkillMesh values that emphasize the quality of freelancing. The final aim is to make users not only have responsive, scalable and reliable platform but also help them to achieve highly effective and satisfactory experience of freelancing.

2.5.2. Safety Requirements

The SkillMesh platform is well conscious about the security of the users and ensures that they are well protected leaving no room for data breaches. The system is compliant of strict data safety procedures where it utilizes advanced encryption methods for the protection of the highly sensitive data like user credentials, transaction details, and communication log. Extensive measures of error reporting are implemented to ensure that discrepancies affecting the smooth functionality of the service are addressed on time, thus providing end users with useful error messages. Besides, SkillMesh has adopted the identity safety paradigm via blockchain technology for identity verification, meaning that it has expanded the overall security and the trustworthiness of the platform. The safety of user interactions is maintained via the convenient mechanism of the comprehensive rating and review system, where each user provides feedback to others on their behavior and deeds, which in turn helps to ensure the trust of the whole SkillMesh community. In addition, the platform is compliable with the standards of the industry when it comes to safety and security as well as it is adaptable for multiple browsers and devices. SkillMesh upholds its safety principles, regularly monitoring and improving the measures applied to match the existing challenges and recommendations yielding safe working environments in the dynamic freelancing set up.

2.5.3. Security Requirements

SkillMesh is fully secured as we advocate effective techniques and adopted different protocols we ensure the integrity and confidentiality of user data, transactions, and other actions on our platform. The platform it uses the tough entry restrictions and login protocols to make certain that only those users who are authorized to obtain access venture into the system, which helps to minimize the risk of unauthorized access. The hackers used innovative encryption algorithm to cover their tracks and protect their identity that includes users' credentials and transaction details during transmission and storing of data.

Blockchain technology is used for identity affirmation. This tracks digitally ones identity whereby the information is stored in a safe, not editable and it does not belong to any entity, but the digital user. Furthermore, SkillMesh adopts secure coding methods as an effort to prevent the common vulnerabilities (for instance, SQL injection and cross-site scripting and as a result the chance of malicious attacks would be greatly, reduced.

To take care of the potential danger of cyber-attacks, the platform offers intrusion detection and prevention systems that are being actively monitors for any unusual activities and responding in an occurrence of anomalies. Performing regular audits and assessments of the security, wherein the identification and corrections of the system vulnerabilities are done.

In case of an information breach or a security threat, SkillMesh has a proper event response plan, which has been approved and defined. This scheme would make provision for rapid detection and isolation, thorough cleansing and total eradication process, responsible communication and appropriate recovery for concerned parties to enjoy a quick and responsive solution to security incidences.

SkillMesh is to a legally approved procedure sorted reliably and constantly attains change in security prerequisites because of legal requirements based on different jurisdictions. Furthermore, the system performs continuous maintenance of all user accounts and keeps each account updated while upholding a detailed security policy aimed toward educating users on ways to protect their accounts and data.

According to SkillMesh, ensuring security is a lifetime process and making deliberate acts of monitoring and updating the security measures are important to achieve a secure freelancing environment. That platform will be a constant figure, which is prior to the activities of the constantly evolving cyber-crimes, putting forward the proactive protective measures to keep an entire community of the users confident and assured.

2.5.4. Usability Requirements

SkillMesh is the chosen brand name of the User Experience, which is defined through a list of Usability Requirements that will make the platform easy to use and effective. The UI design targets clear, instantaneous and user intuitive experience that guarantees smoother navigation for clients and contractors. SkillMesh is committed to a smooth and quick system registration; you can sign up within just 2minutes. Real-time communication features such as messaging and video conferencing that are efficient for group work is now an integral part of the online learning environment. The platform comprising of file sharing saves the time by improving the speed of uploads and downloads to upscale the project communication. User profiles are editable directly, which eliminates the need for editing and thus makes it easy to enlist your skills neatly. For an overwhelmingly wide-ranging user, SkillMesh makes sure compatibility with every web browser and even across various devices. The company pledges to have all freelancers having had a great and easy time using the portal that is full of a wide variety of features.

2.5.5. Reliability Requirements

SkillMesh sets up a pillar of reliability standards to make sure that there are efficient measures in place to stabilize and secure the platform. Such system has high level of performance, guaranteed 99.9% uptime, and provides scheduled maintenance while effectively eliminate unplanned downtime. The platform SkillMesh uses advanced error handling features, incorporates them and they take care of the issues in a timely fashion thus ensuring a seamless user experience.

The instrument adopts a rigorous test procedure as part of the system, including stress testing, load testing and simulation of different conditions. It ensures the scalability and performance of the system under varying situations. Being able to manage higher traffic involving more users, projects or simultaneous activities, SkillMesh is the best-adapted solution to preserve reliability.

Data integrity is a top ranking priority of SkillMesh and is ensured by the secure data storage technique, which minimizes data corruption and loss. We perform regular backups to ensure a system, which can store the information safely in event of disaster. Freelancers will always be aware that they do not need to worry.

SkillMesh one of the best is reliable monitoring thanks to using designed tools and methods that ensure agile detection of deviations and other potential issues in real-time. It virtually implies that actions can be taken quickly to newly arising matters making the platform more and more trustworthy among users. Besides that, the platform complies with all the network security norms that can prevent potential cyber threats and unauthorized access.

Should a SkillMesh become a victim to an incident, the organization respects the role of a detailed incident response plan, which is aimed at the stabilization and recovery of the disruption of the normal

operations. SkillMesh regularly updates maintenance routines between revisions and conducts all the maintenance activities with precision making the SkillMesh platform more reliable and thus it has long-term stability.

Achieving these aims of SkillMesh like availability, error resilience, data integrity, and proactive monitoring, empowers the system to become dependable, trusted, which in turn, this build up users' confidence, and creates a reliable environment for freelancers and clients.

2.5.6. Maintainability/Supportability Requirements

SkillMesh is composed of a solid maintenance and supportability paradigms. It uses plug and play design approach, which provides upgradability and maintenance ease. Codebase is documented carefully and versioned properly enabling new developers or contributors to understand the solutions easily and allow modifications smoothly. In order for the complexity of software implementation to be minimized, SkillMesh puts high user support importance and offers clear user guides and responsive customer service channels. The system incorporates self-diagnosing apps and error logging as well as user-friendly interfaces that developers can use to quickly identify and resolve problems. Continuous system audits and updates will be performed to ensure stability and fulfil the varied needs of a growing community and adjust to technology changes. Besides it, SkillMesh develops the principles of sustainable and supported framework forming a platform with provided freelancing services but at the same time, which is reliable and effective.

2.5.7. Portability Requirements

In SkillMesh, portability is stressed to ensure that users have a seamless flow of the experience and can access the services at their own convenience. The platform is developed to be functional in simultaneous with different web browsers and gadgets, thus users are allowed to conveniently use its features via desktops, laptops, tablets, and mobile devices. SkillMesh attaches great importance to multiplatform ability, so the software enables the user to get the device he/she prefers without any lack of functionality. The portability target in this goal focuses on ensuring the SkillMesh platform operates smoothly across different environments whether you are in the office or on the go thus making the entire freelancing experience easier and more inclusive.

2.5.8. Efficiency Requirements

In SkillMesh, we make a priority on efficiency in order to run outstanding freelancing platform. The system is planned to respond within a second-range time period for various user interactions to make sure of unparalleled and uninterrupted user-experience. SkillMesh is designed to increase data retrieval speed by placing search velocity that is less than 2-second to user profiles and the project details. The platform merges blockchain with instant transactions with a rough confirmation time of 15 minutes and this empowers both peer-to-peer transaction and smart contract accomplishment.

SkillMesh is tolerant and is able to handle not less than 1,000 parallel deals per minute for added scalability as user activity grows. Such efficiency requirements put together are a foundation for a platform that is flexible and reactive as well as able to respond to the to changing conditions of freelancing community.

Chapter 3

Use Case Analysis

Chapter 3: Use Case Analysis

In a SkillMesh scenario, at a minimum, the following cases can be considered: a freelancer finally generates and optimizes the Profile to attract clients, clients make an order, freelancers accept or reject a project, a secure transaction with cryptocurrency, dispute resolution mechanism through decentralization, and tracking the transparent reputation of a user through a blockchain-based support system. Such use cases end up making the usage of the platform more intuitive by focusing on the decentralization aspects, which are features like smart contracts, identity verification, and borderless secure freelancing.

3.1. Use Case Model

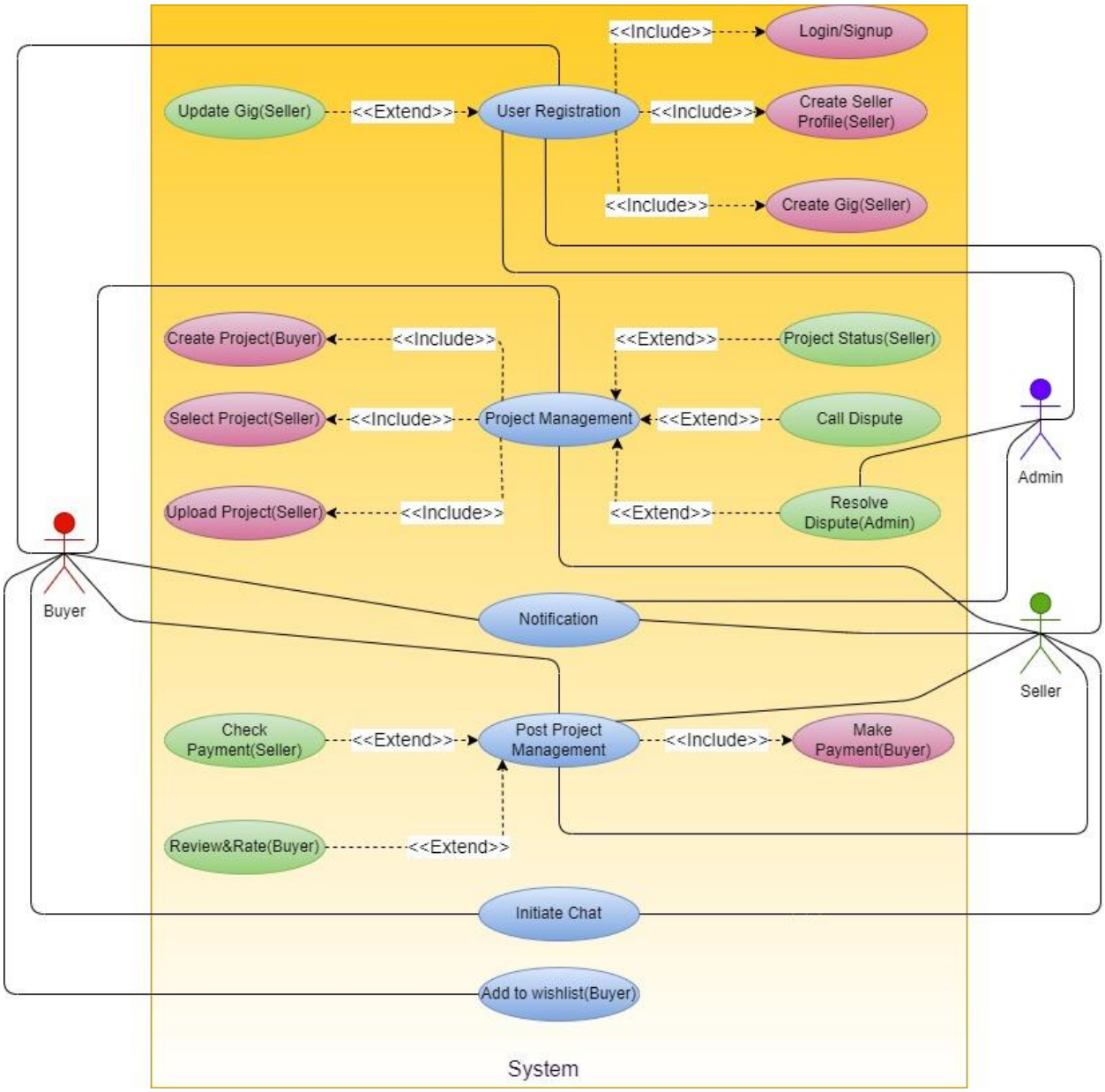


Figure: 3.1 Use Case Model

3.2. Use Cases Description

Actors:

- **Buyer**
- **Seller**

Seller Use Case:

- **Login/Sign Up:**
  Description: User will have a choice of logging in or signing up by using MetaMask wallet.
  Pre-condition: A seller had not logged in yet.
  Post-condition: The seller is granted access to the Platform features upon a successful login or registration.
- **Create Seller Profile:**
  Description: Sellers can create their online presence by using their own profile, the capabilities they have developed, and the portfolio of work they have delivered.
  Pre-condition: Seller logged in.
  Post-condition: Seller's page is now visible to prospective clients.
- **Create Gig:**
  Description: Sellers can define, publish their individual service offerings.
  Pre-condition: Seller has the fully functioning and active profile.
  Post-condition: Gig is designed, which displays seller's expertise and individuality.
- **Update Gig:**
  Description: Allows sellers to select and customize the offerings of their existing gig.
  Pre-condition: Seller is currently selling in a gig.
  Post-condition: Gig data is changed to represent modifications, concerning offers or conditions.
- **Check Payment:**
  Description: Buyers can check payment confirmation for completed task.
  Pre-condition: Task listed is checked as completed.
  Post-condition: Seller, transaction details and bill view.
- **Notification:**
  Description: If seller have any new notification this app will inform about any new job opportunity or a new gig requests. They will also receive Alert as well.
  Pre-condition: Seller logged in.
  Post-condition: Seller receives real-time notifications.
- **Upload Project:**
  Description: Enables the sellers to deliver a complete project to the clients.
  Pre-condition: Mark the task as done.
  Post-condition: As a project, my paper is uploaded; it is now undergoing the client's approval process.

- **Call Dispute:**
  Description: Offers a seller the chance of initiating a dispute resolution process in case of disputes with clients.
  Pre-condition: Disputes arise between client and sellers.
  Post-condition: Dispute resolution is initiated, enabling communication and settlement.

Buyer Use Case:

- **Buyer Login/Signup:**
  Description: Provides buyers with the opportunity to access the platform through login or registration.
  Pre-condition: Buyer not logged in.
  Post-condition: Once a buyer registers and logs in, he or she will have access to the entire platform.
- **Order Place:**
  Description: Enables buyers to define and Place new Orders/projects for freelancers.
  Pre-condition: Buyer logged in.
  Post-condition: A task being presented, visible to freelance workers and there for selection.

- **Review Rate Project:**
  Description: Provides a platform for clients to evaluate completed projects, leave reviews, and rate workers.
  Pre-condition: The project is marked as finished in the last step.
  Post-condition: The buyer provides feedback and rating of the work of the freelancer.
- **Make Payment:**
  Description: Allows purchasers to free Payment after tasks are completed.
  Pre-condition: Once you have finished the task or project, indicate that it is done.
  Post-condition: Payment is done to the freelancer. Thereafter, transaction detail is updated.
- **Call Dispute:**
  Description: Let buyers start a dispute resolution process in case of conflicts with freelancers.
  Pre-condition: Disagreement between buyer and freelancer.
  Post-condition: Disputation process is started and this helps interaction and understanding.
- **Initiate Chat:**
  Description: Provides potential consumers with instant chat functionality for project engagements.
  Pre-condition: Buyer signs in and the project is created.
  Post-condition: Chat session is commenced between a buyer and invited freelancer.
- **Add to Wish List:**
  Description: It allows customers to save favorite gigs to a saved wish list for the later use.

Pre-condition: Buyer logged in.
Post-condition: This is added to the buyer's wish list.

- **Gig Searching:**
  Description: Lets users pick among sellers who can be searched for by keywords or criteria.
  Pre-condition: Buyer logged in.
  Post-condition: Buyer sees competent freelancers or gigs in line with the search keywords.
- **Notification:**
  Description: Communicate all relevant updates: project completion/ upgrade, chat messages.
  Pre-condition: Buyer logged in.
  Post-condition: Buyer receives all real-time notifications.

Chapter 4

System Design

Chapter 4: System Design

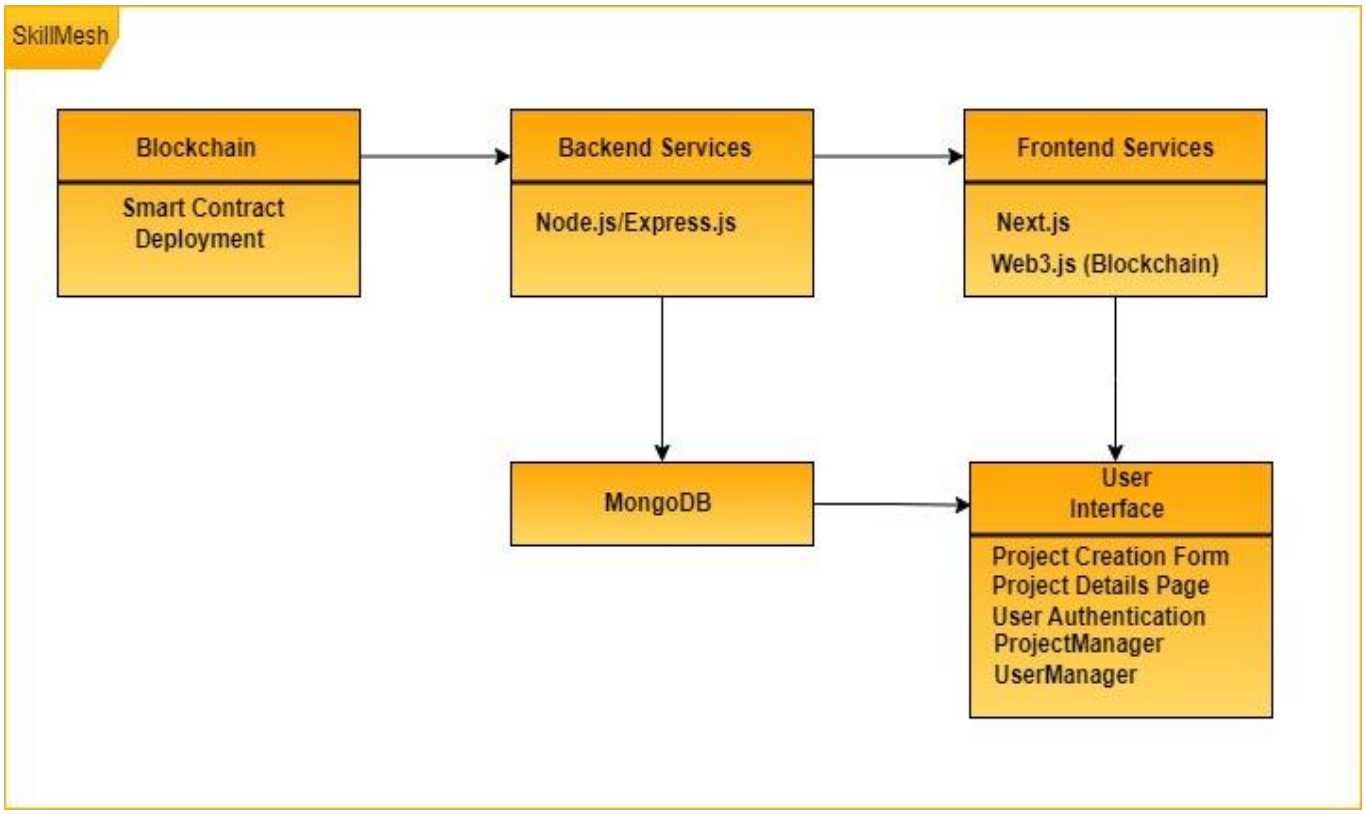


Figure: 4.1 System Design

4.1. Architecture Diagram

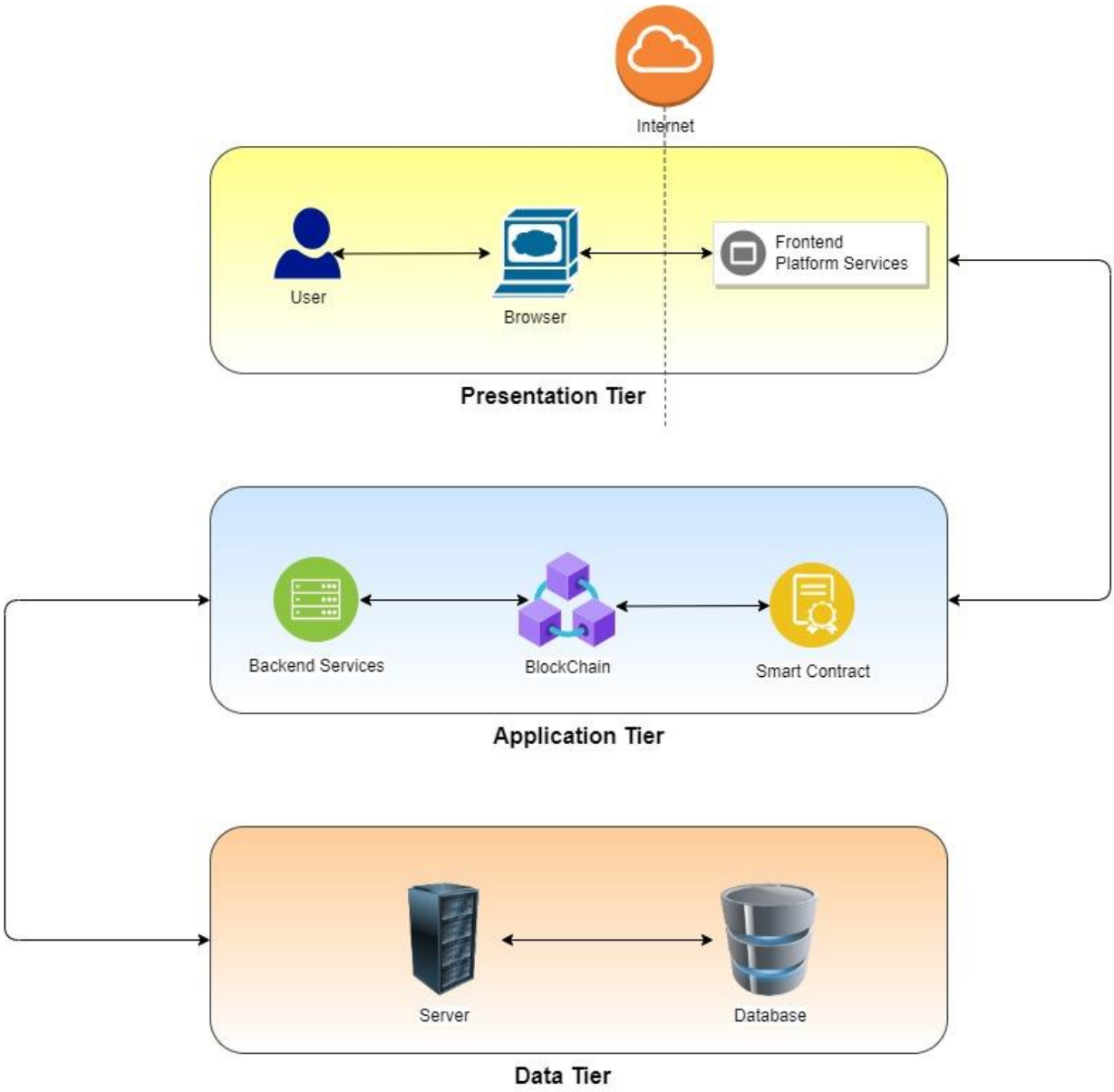


Figure: 4.2 Architecture Diagram

4.2. Domain Model

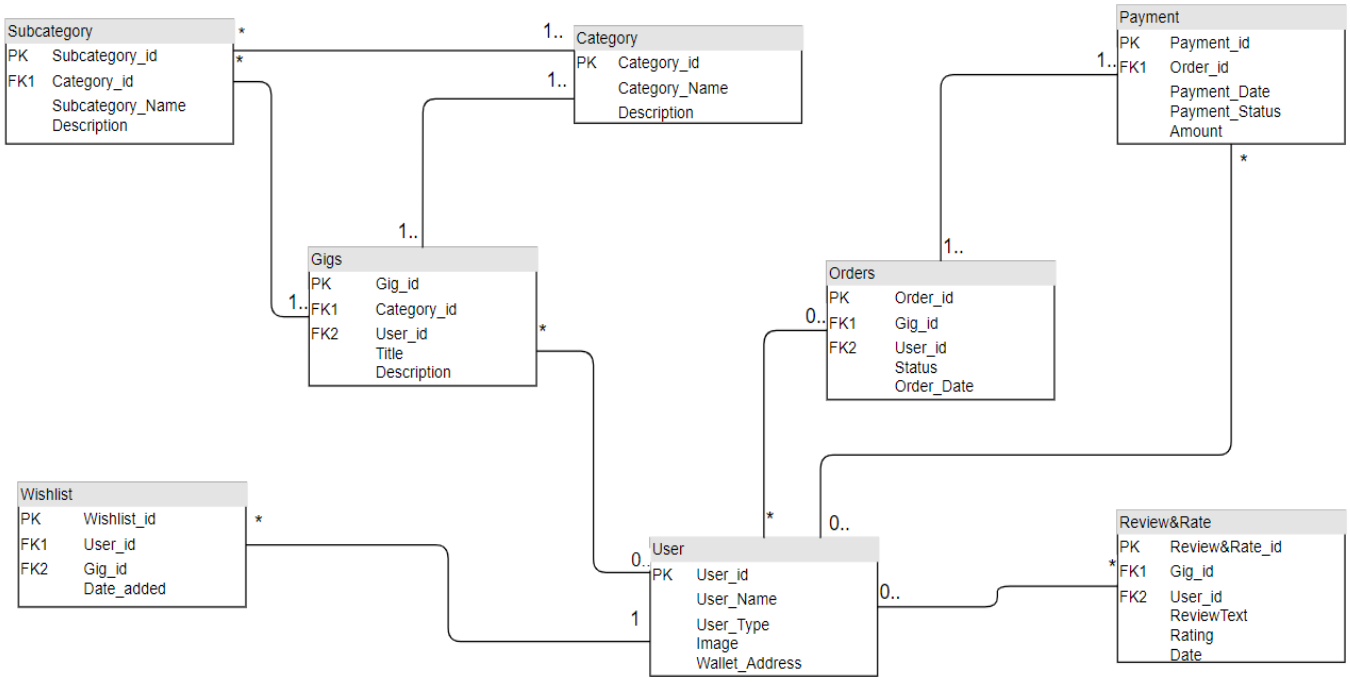


Figure: 4.3 Domain Model Diagram

4.3. Entity Relationship Diagram with data dictionary

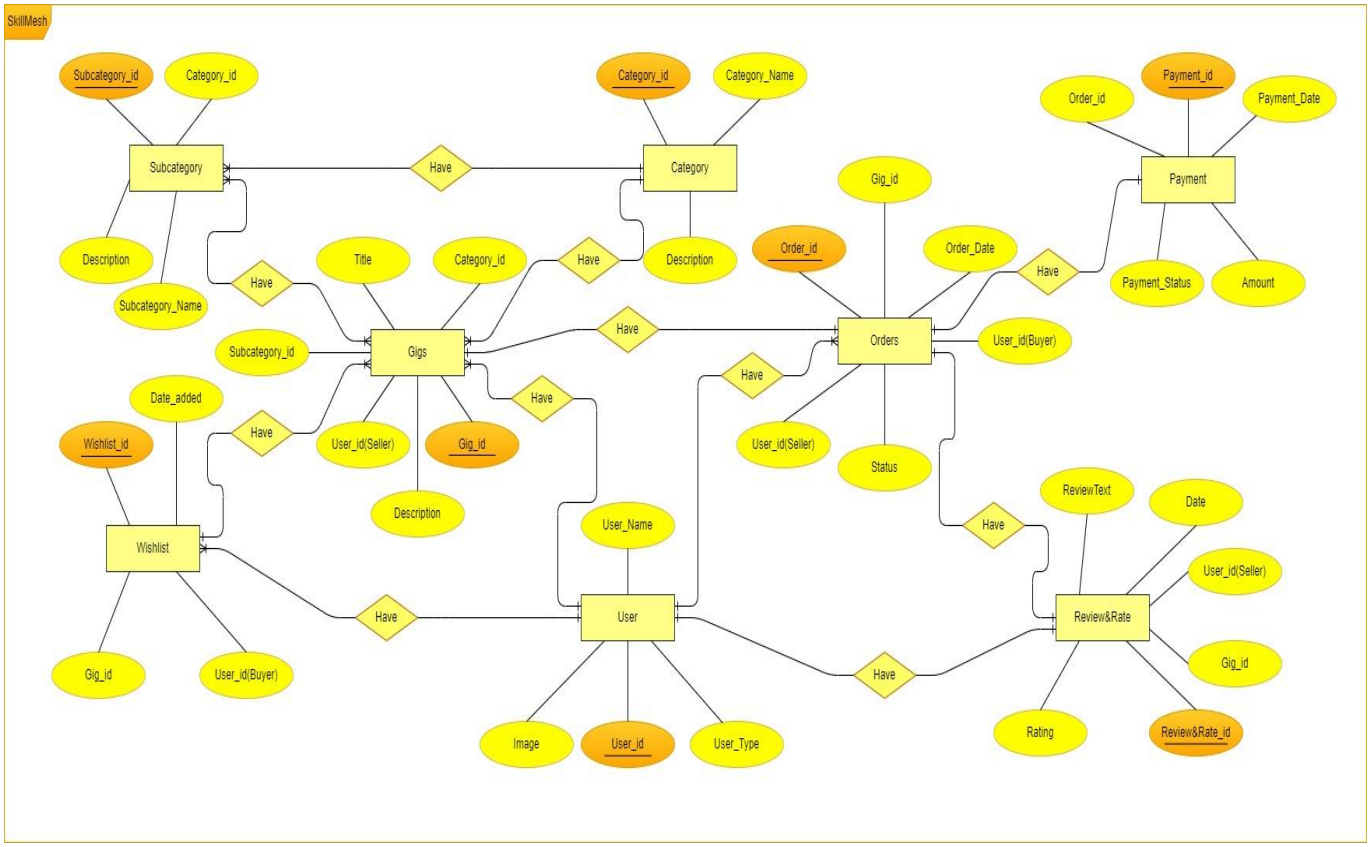


Figure: 4.4 Entity Relationship Diagram with data dictionary

4.4. Sequence / Collaboration Diagram

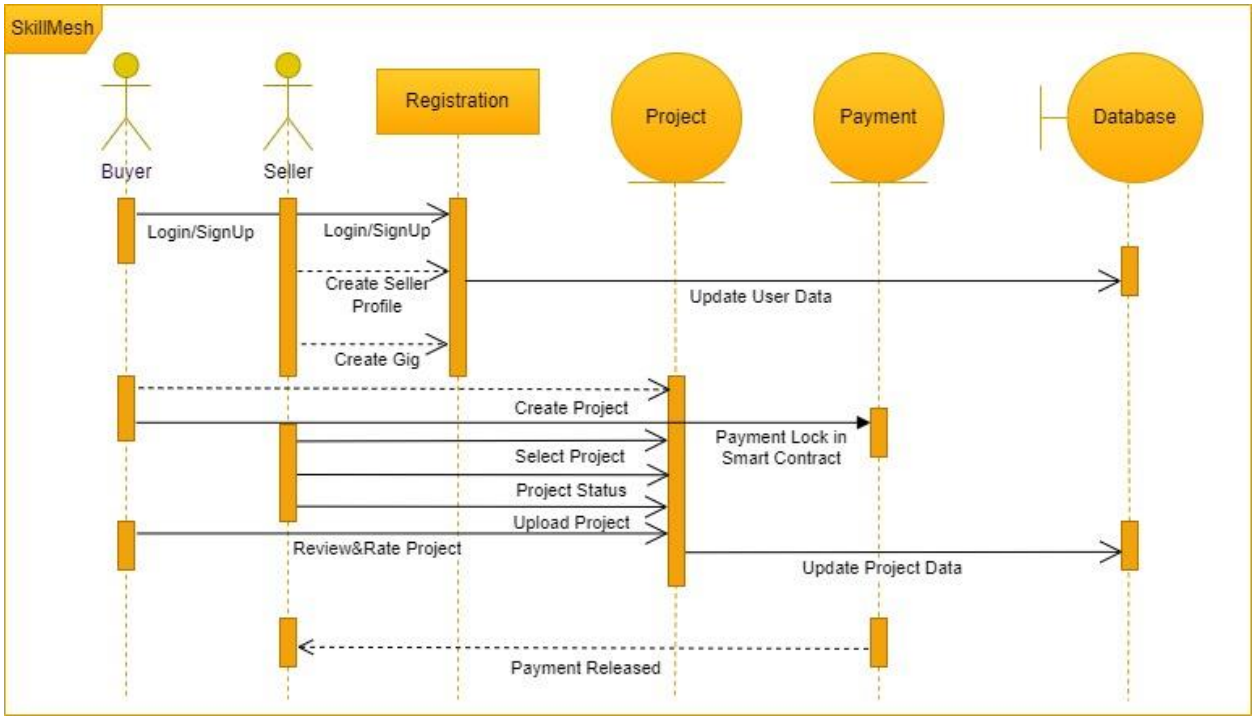


Figure: 4.5 Sequence / Collaboration Diagram

4.5. Operation contracts

1. **User Registration:**
    - **Operation Contract: register User(username, email, password)**
    - **Description:** Registers a new user on SkillMesh.
    - **Preconditions:** The provided username and email must be unique. The password must meet security requirements.
    - **Postconditions:** A new user account is created, and the user is logged into the system.

2. **Create Project:**
    - **Operation Contract: create Project(title, description, skillsRequired, budget)**
    - **Description:** Allows a client to create a new project on SkillMesh without a bidding mechanism.
    - **Preconditions:** The user must be authenticated. The project title, description, and skills required must be provided.
    - **Postconditions:** A new project is created and added to the platform.
3. **Place Order:**
    - **Operation Contract: place Order(projectId)**
    - **Description:** Enables a client to place an order for a project without a bidding process.
    - **Preconditions:** The user must be authenticated as a client. The project must be available for ordering.
    - **Postconditions:** An order is placed for the specified project.
4. **Accept Order:**
    - **Operation Contract: acceptOrder(projectId)**
    - **Description:** Allows a freelancer to accept an order placed by a client.
    - **Preconditions:** The user must be authenticated as the freelancer. The project must be available for accepting orders.
    - **Post conditions:** The freelancer is notified, and the project status is updated.

5. **Process Payment:**
    - **Operation Contract: processPayment(projectId)**
    - **Description:** Initiates the payment process for a completed project.
    - **Preconditions:** The project status must be marked as completed by both the client and freelancer.
    - **Post conditions:** Payment is processed, and the freelancer is compensated according to the agreed terms.
6. **Resolve Dispute:**
    - **Operation Contract: resolve Dispute(projectId, resolution Details)**
    - **Description:** Allows users to resolve disputes related to a project.
    - **Preconditions:** A dispute must be initiated by either the client or freelancer.
    - **Post conditions:** The dispute is resolved, and the project status is updated accordingly.
7. **Update User Profile:**
    - **Operation Contract: updateUserProfile(user ID, new Details)**
    - **Description:** Allows users to update their profile information.
    - **Preconditions:** The user must be authenticated. Valid updated details must be provided.
    - **Post conditions:** The user's profile information is updated

4.6. Activity Diagram

Buyer Activity

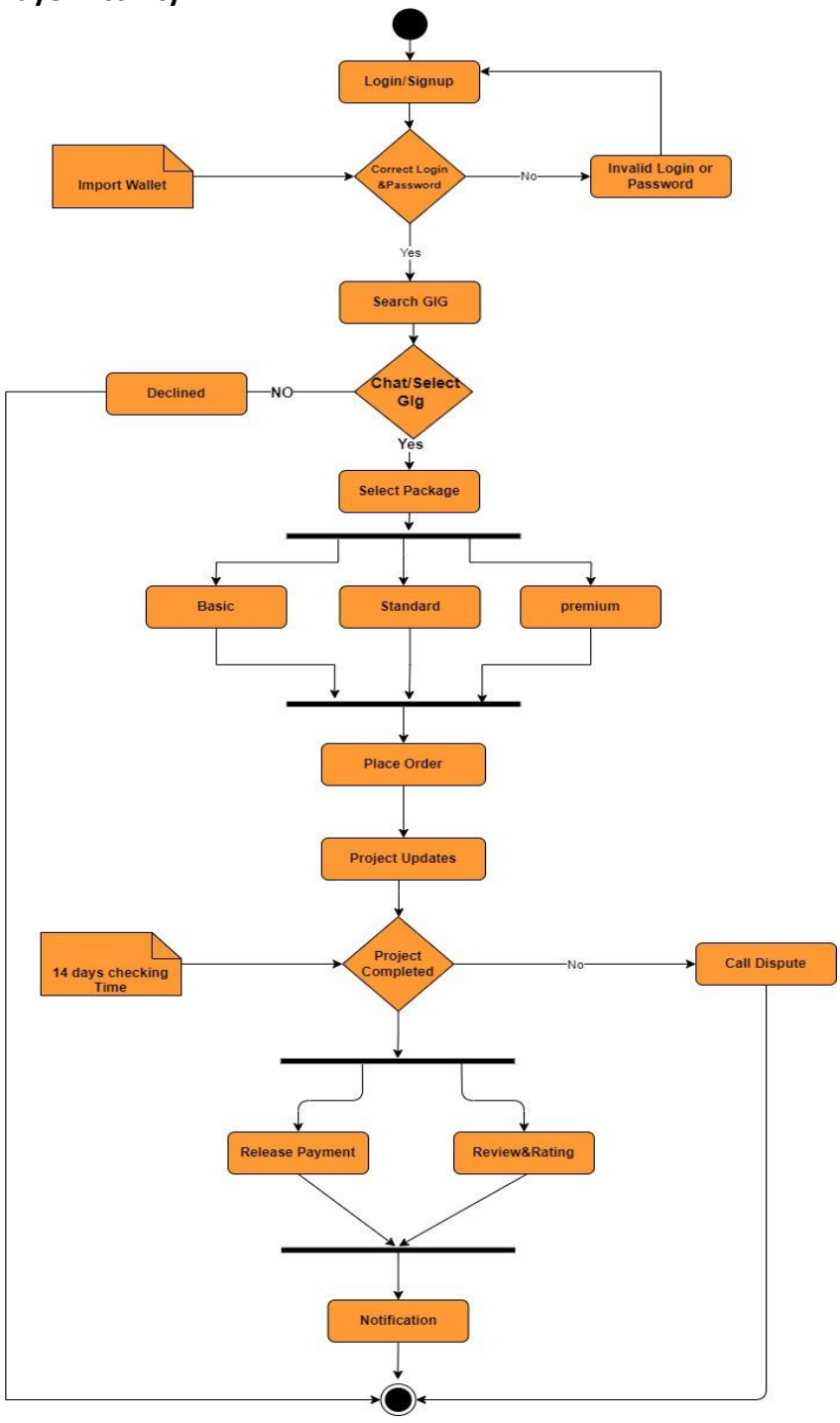


Figure: 4.6 Buyer Activity Diagram

Seller Activity

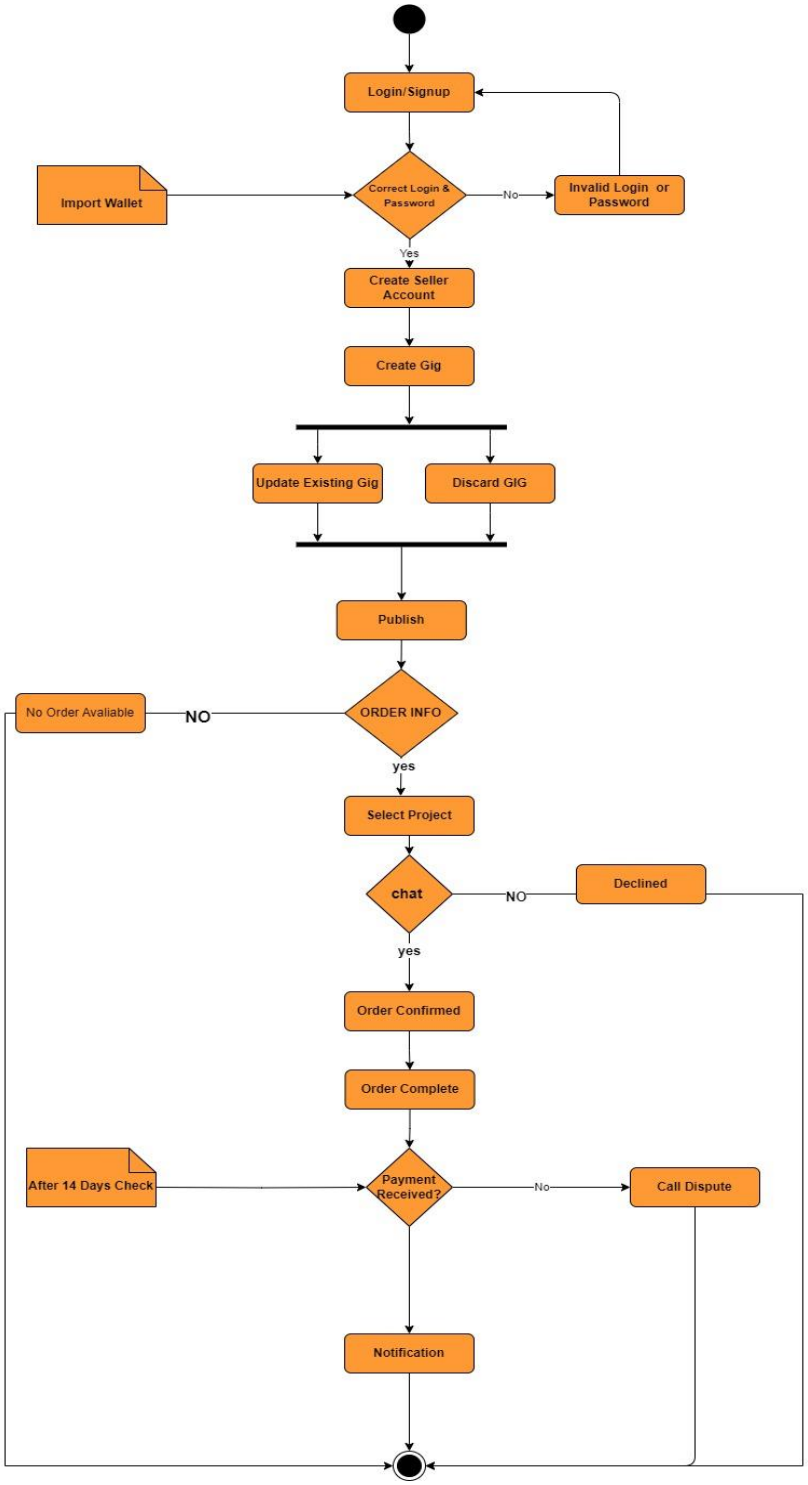


Figure: 4.7 Seller Activity Diagram

4.7. State Transition Diagram

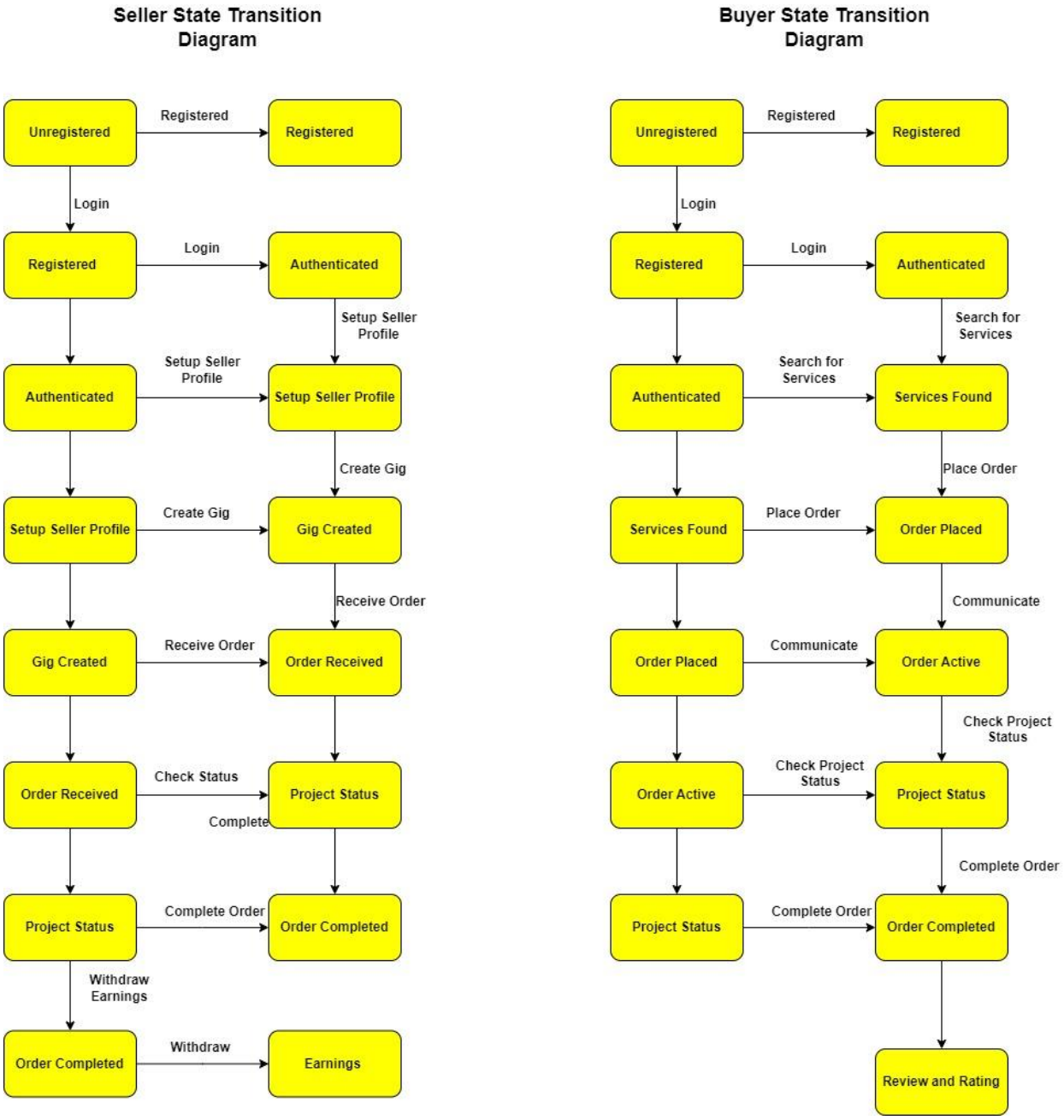


Figure: 4.8 State Transition Diagram

4.8. Component Diagram

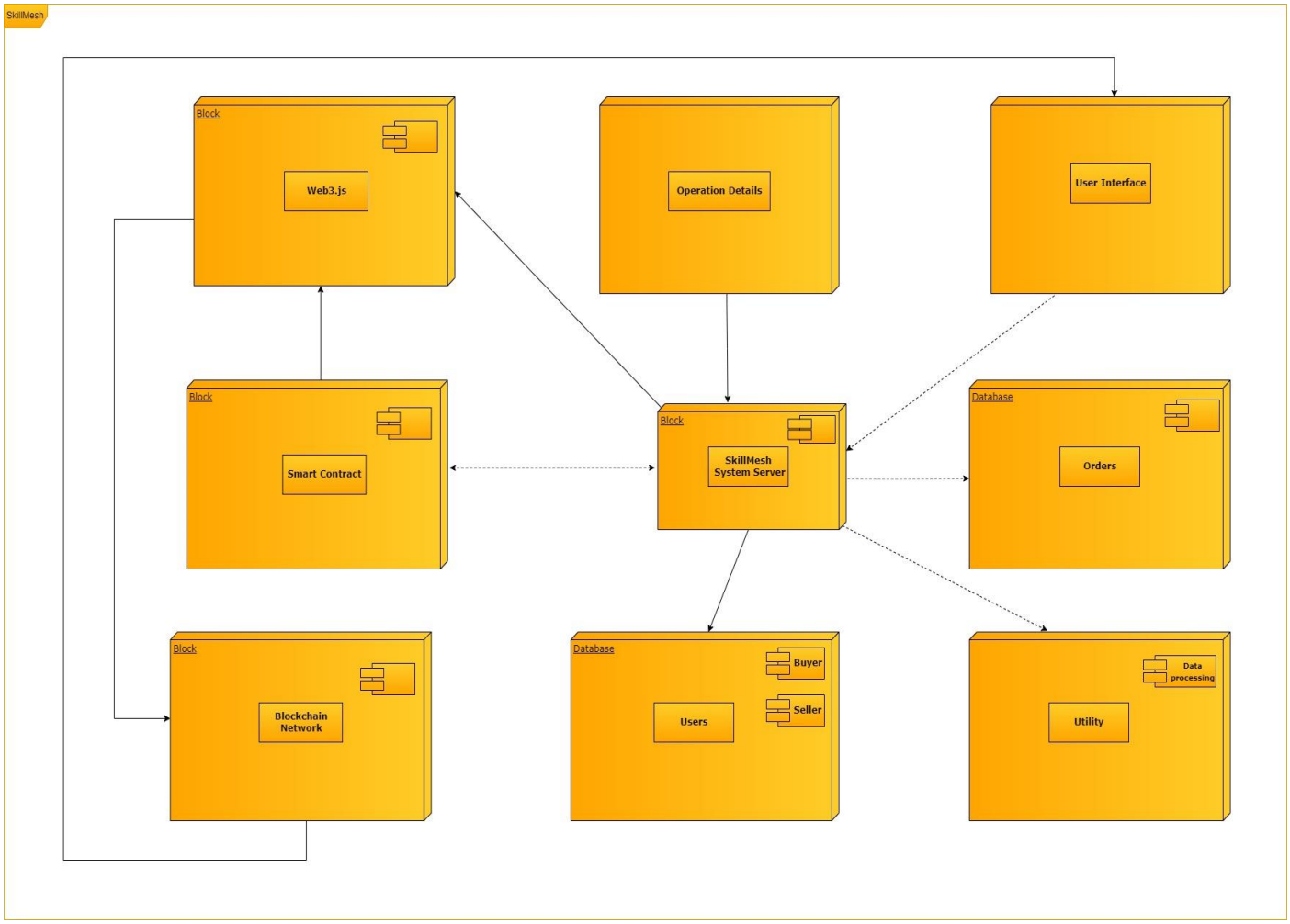


Figure: 4.9 Component Diagram

4.9. Deployment Diagram

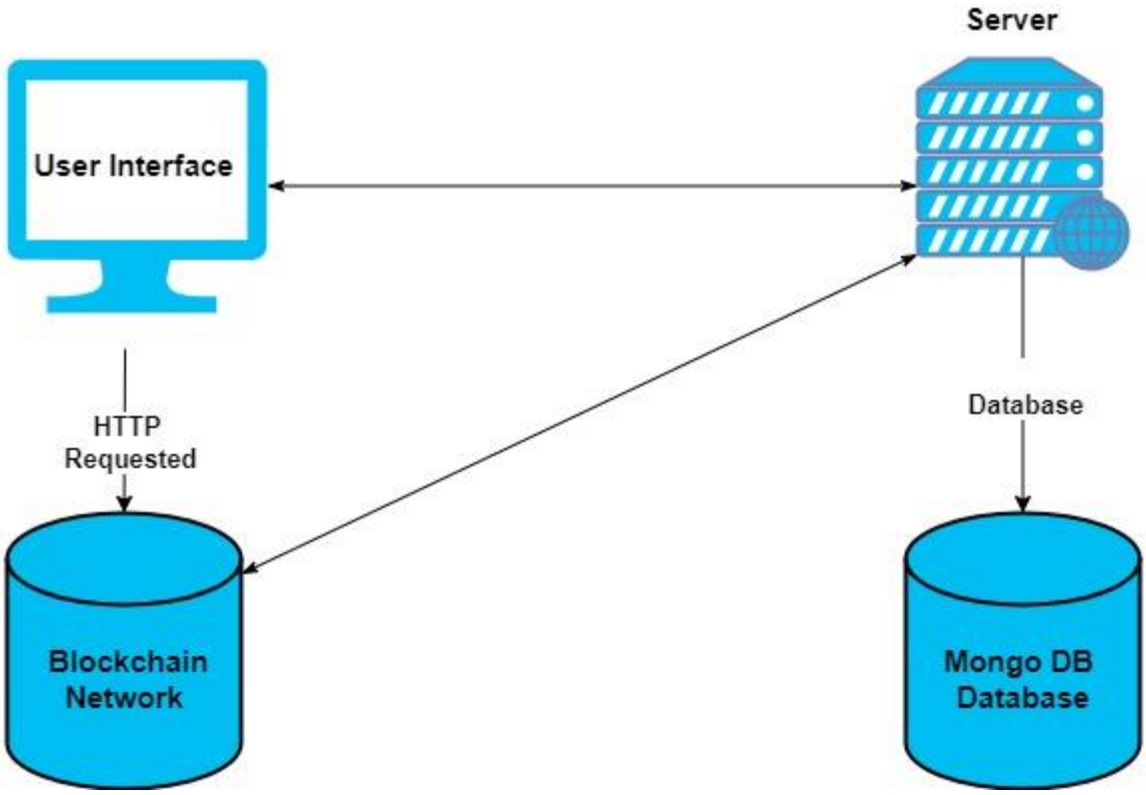


Figure: 4.10 Deployment Diagram

4.10. Data Flow diagram [*only if structured approach is used - Level 0 and 1*]

Level 0

Figure: 4.11 Data Flow diagram Level 0

Level 1

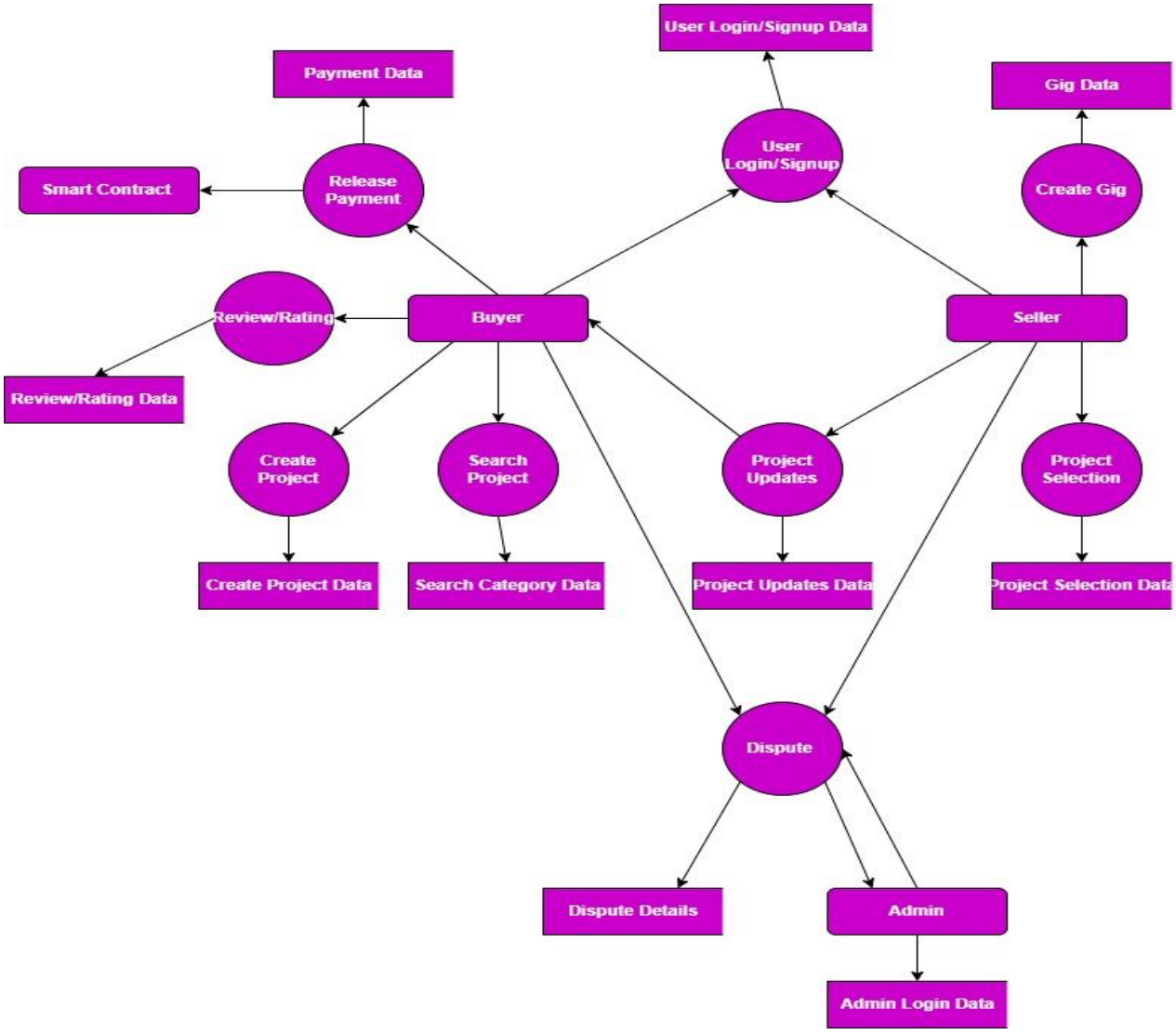


Figure: 4.12 Data Flow diagram Level 1

Level 2

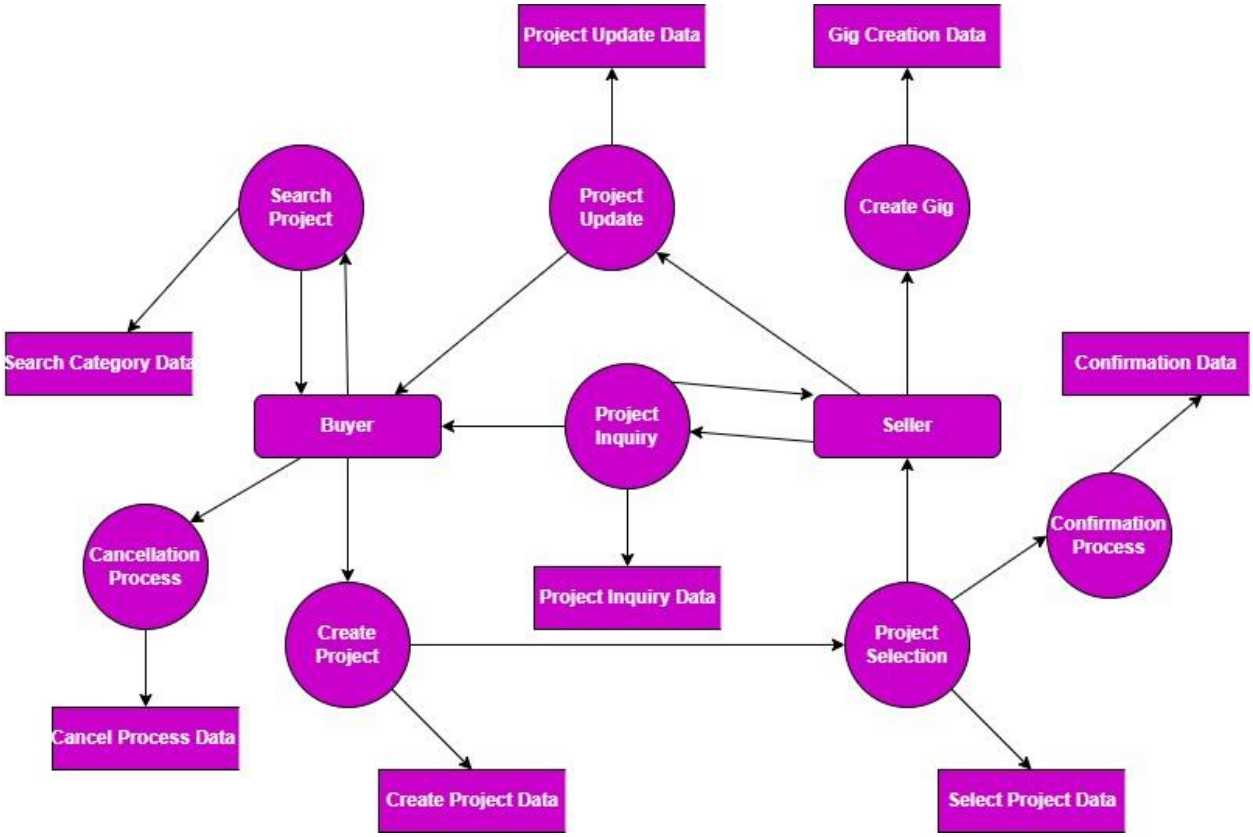


Figure: 4.13 Data Flow diagram Level 2

Chapter 5

Implementation

Chapter 5: Implementation

The Implementation of SkillMesh starts with the frontend designing using **Next.js, Tailwind css and Chakra UI**. User-friendly interfaces are designed for a seamless experience. Backend is powered by **Node.js, Express.js. Mongo DB** is used for storing data. It is a No SQL database. For smart contract, we will use **Solidity** and **MetaMask** will be used for peer-to-peer transactions. **Polygon (MATIC)** will be used as a blockchain network.

5.1. Components, Libraries, Web Services and stubs

Library: web3.js, uPort, Self-Key, MongoDB, Solidity, React Router
Framework: Next.js or Node.js with Express.js

```
dependencies": {
   "@chakra-ui/react": "^2.8.1",
   "@mui/icons-material": "^5.14.19",
   "@react-spring/parallax": "^9.7.3",
   "@reduxjs/toolkit": "^1.9.7",
   "emoji-picker-react": "^4.5.6",
   "framer-motion": "^10.16.4",
   "infinite-react-carousel": "^1.2.11",
   "next": "^14.0.3",
   "react": "latest",
   "react-dom": "latest",
   "react-icons": "^4.11.0",
   "react-redux": "^8.1.3",
   "react-toastify": "^9.1.3"
 }
```

5.2. Deployment Environment

For deploying **SkillMesh Polygon (MATIC),** a side chain of Ethereum will be used. Polygon is the perfect substitute for Ethereum due to the expensive gas fees and deployment charge from Ethereum. Polygon is approximately **10000x** cheaper than Ethereum and provide security of Ethereum. **MetaMask** will be used as web3.0 wallet and will be used for peer-to-peer transactions. MetaMask is the most popular wallet used at the moment. **Polygon (MATIC)** will be used for payments of the projects.

5.3. Tools and Techniques

Tools	Description
Next.js	React framework use to create frontend of the website.
Node.js	Node.js is an open source server environment
Express.js	Express.js is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications.
Mongo DB	No SQL Database use for storing data.
Solidity	Use to write smart contract.
Tailwind css	Let's you build any design, directly in your HTML
Chakra UI	Chakra UI is a component library that lets you create accessible and beautiful React apps with less code and more speed.

Table: 5.1 Tools and Techniques

5.4. Best Practices / Coding Standards

SkillMesh adheres to industry-best practices and coding standards to ensure maintainability, scalability, and readability of codebase. Following the Model-View-Controller (MVC) architectural pattern, the code is organized into distinct components for efficient development and future enhancements.

Best Practices and Coding Standards:

1. Solidity Smart Contracts:

- Contracts follow the Solidity style guide and are well-commented for clarity.
- Code is modular, promoting reusability and separation of concerns.
- Security considerations, such as avoiding reentrancy vulnerabilities, are addressed.

2. Node.js and Express.js Backend:

- Code adheres to the JavaScript Standard Style, ensuring consistency.
- Express.js middleware is used for tasks such as request parsing and error handling.
- RESTful API endpoints follow naming conventions and are designed with simplicity and clarity in mind.

3. Next.js Frontend:

- Components follow the JSX and ES6 syntax, enhancing code readability.
- State management is handled efficiently using Next.js state and props.
- Code is organized into reusable and maintainable components

4. Database Interaction (MongoDB):

- MongoDB queries are written in a concise and readable manner.
- Connection handling follows best practices, including connection pooling and error handling.

5. Blockchain Integration (Web3.js):

- Web3.js code is structured for easy interaction with the Ethereum blockchain.

- Smart contract interactions are encapsulated in well-defined functions, enhancing code maintainability.

6. MVC Model Pattern:

1. **Model (Smart Contracts):**
    - Represents the data and logic of SkillMesh projects.
    - Manages the state and rules for creating and completing projects.
2. **Next.js (Frontend):**
    - Renders the user interface, presenting data from the backend to users.
    - React components are organized hierarchically to encapsulate views.
3. **Controller (Express.js Backend):**
    - Manages the flow of data between the model and the view.
    - Handles business logic, including project creation and completion.

5.5. Version Control

Version control is a critical aspect of SkillMesh development lifecycle, providing a systematic approach to managing code changes, collaborating among developers, and ensuring the stability of the codebase. SkillMesh uses Git as its version control system, employing best practices to streamline development workflows and facilitate collaboration.

Key Version Control Practices:

1. Git Repository:

- The SkillMesh codebase is hosted in a central Git repository, providing a single source of truth for the project.

2. Branching Strategy:

- **A branching strategy** is in place to manage different aspects of development.
- **Main Branch:** Represents the production-ready code.
- **Development Branch:** Integrates feature branches for ongoing development.
- **Feature Branches:** Created for each new feature or enhancement, ensuring isolation and easy integration.

3. Commit Messages:

- Descriptive and meaningful commit messages are used to explain the purpose of each commit clearly.
- Follows the conventional commit format for consistency.

4. Pull Requests (PRs):

- Feature branches are merged into the **development** branch through pull requests.
- Pull requests include a summary of changes, any related issues, and are reviewed by peers before merging.

5. Continuous Integration (CI):

- CI pipelines are configured to run automated tests on each pull request.
- Ensures that new code changes do not introduce regressions or errors.

6. Tagging Releases:

- Releases are tagged in Git to mark specific points in the project's history.
- Follows semantic versioning to convey the nature of changes.

7. Git Hooks:

- Pre-commit and pre-push Git hooks are utilized to enforce code quality checks, ensuring only high-quality code is committed.

8. Code Review:

- Code review is, in fact, a significant step in any version control procedure.
- The code-writing process shall involve the error catching, compliance with code-writing standards, and knowledge sharing processes also, in which a peer review the code produced by others.

9. Gitignore:

- A good .gitignore file can help you clean up redundant files and directories in your repository, ensuring the repository size is useful.

10. Git Flow Documentation:

- The project will have a description about the Git Flow used and will include steps for the branching model and handling releases and hotfixes.

Chapter 6

Testing and Evaluation

Chapter 6: Testing and Evaluation

Testing and evaluation are crucial phases in development lifecycle of **SkillMesh** application, ensuring its functionality, usability and effectiveness in addressing the targeting objective. This phase involves rigorous examination of the application features, user interaction and performance to identify and rectifying any issues or shortcomings before deployment.

a. Login/Signup Testing:

If there is no MetaMask, extension installed

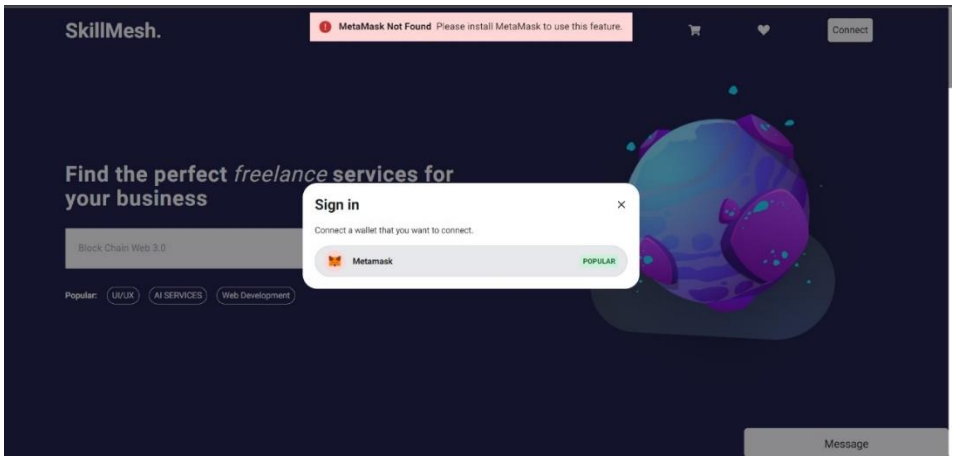


Figure: 6.1 Login/Signup Testing

If the password is wrong

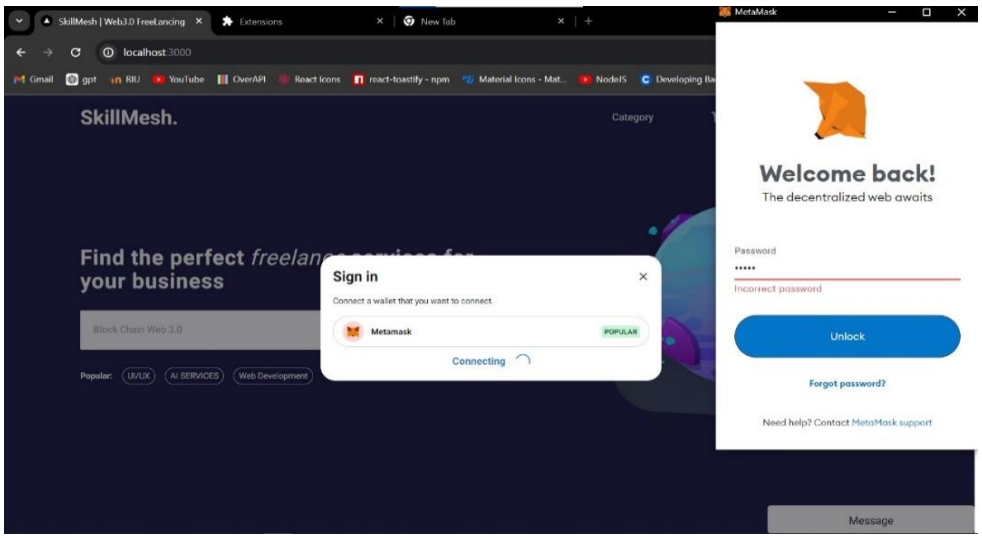


Figure: 6.2 Login/Signup Testing

With correct Password

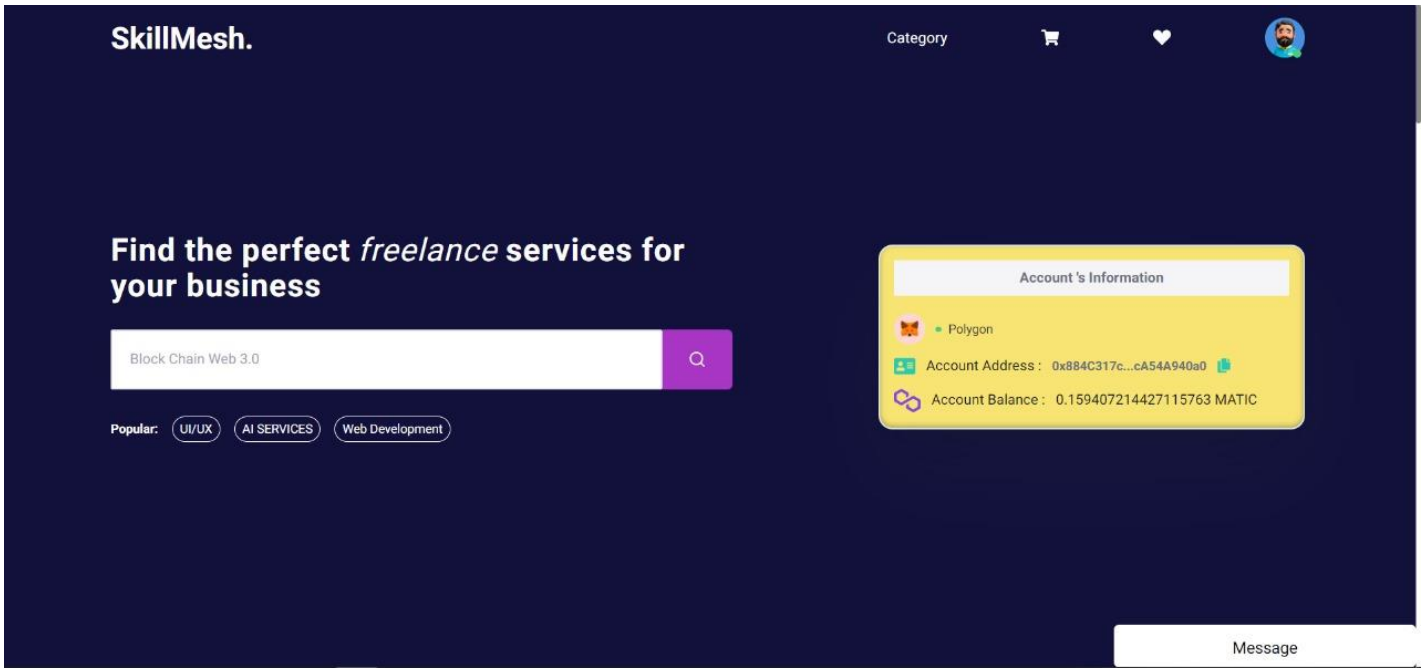


Figure: 6.3 Login/Signup Testing

b. Gig Creation Testing

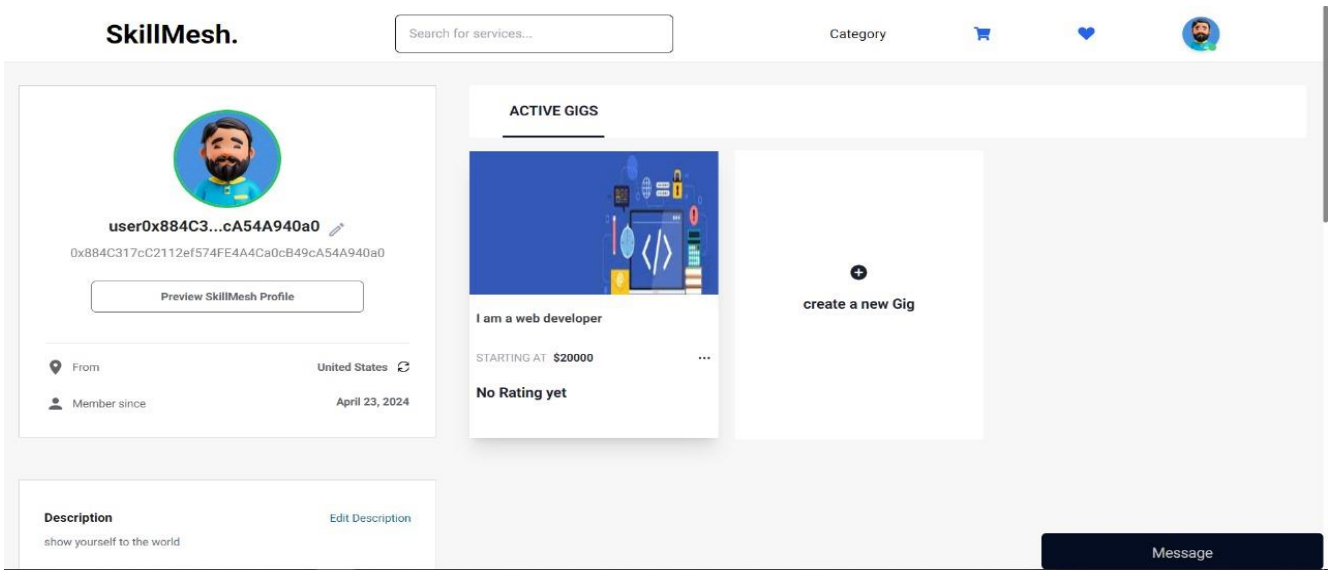


Figure: 6.4 Gig CreationTesting

c. Wish list testing

Adding gig in the wish list

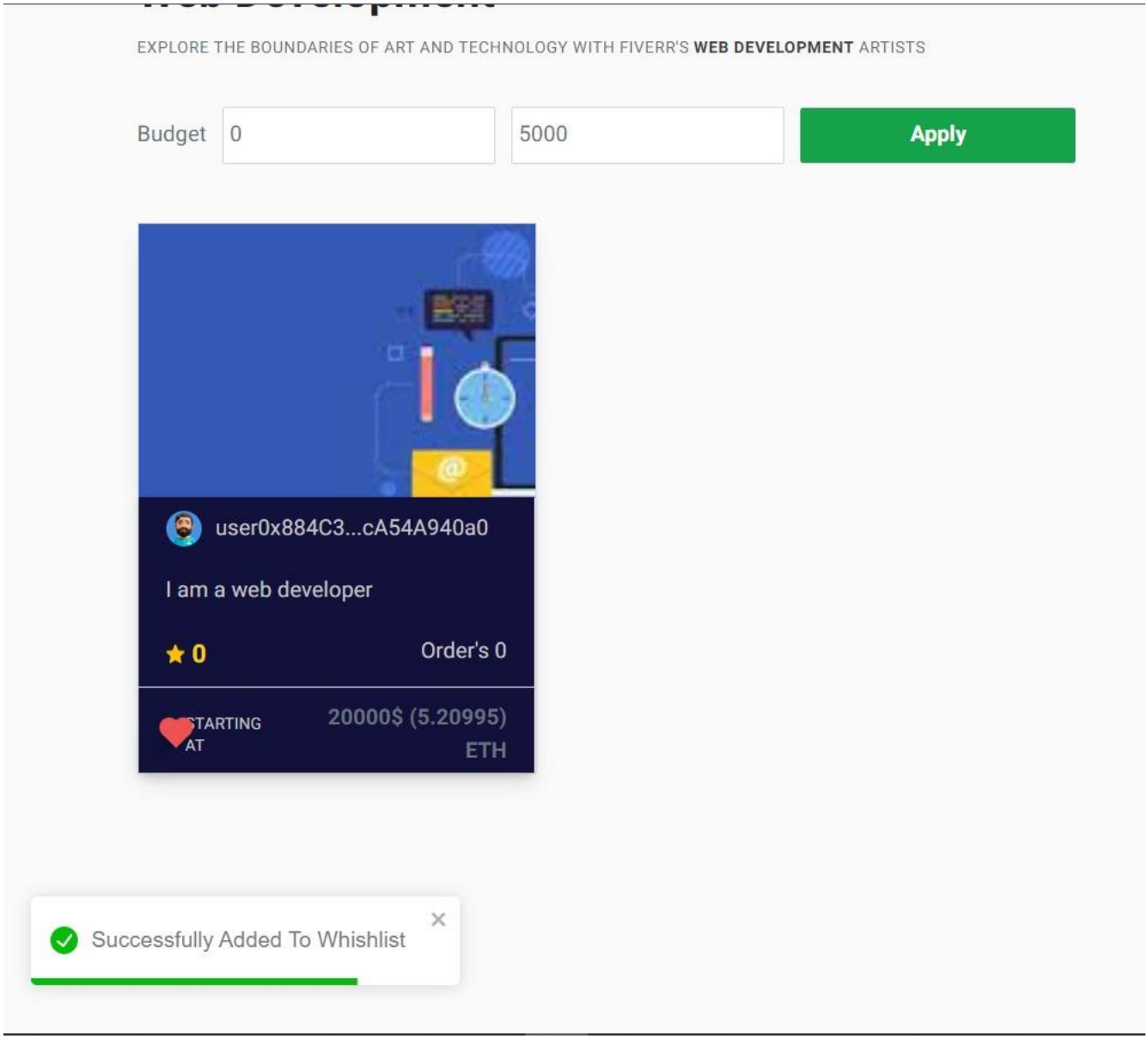


Figure: 6.5 Wish List Testing

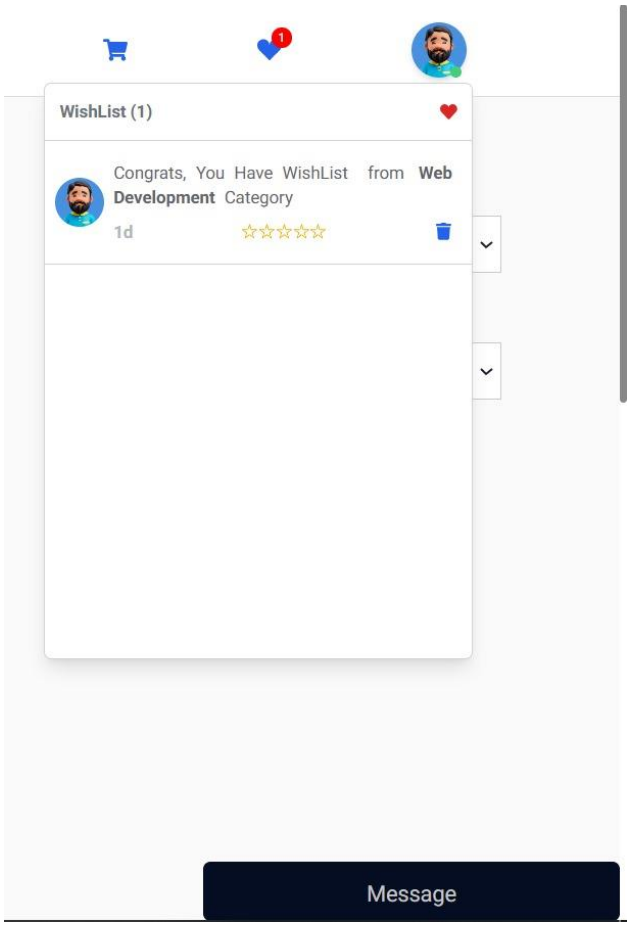


Figure: 6.6 Wish List Testing

Removing from the Wishlist

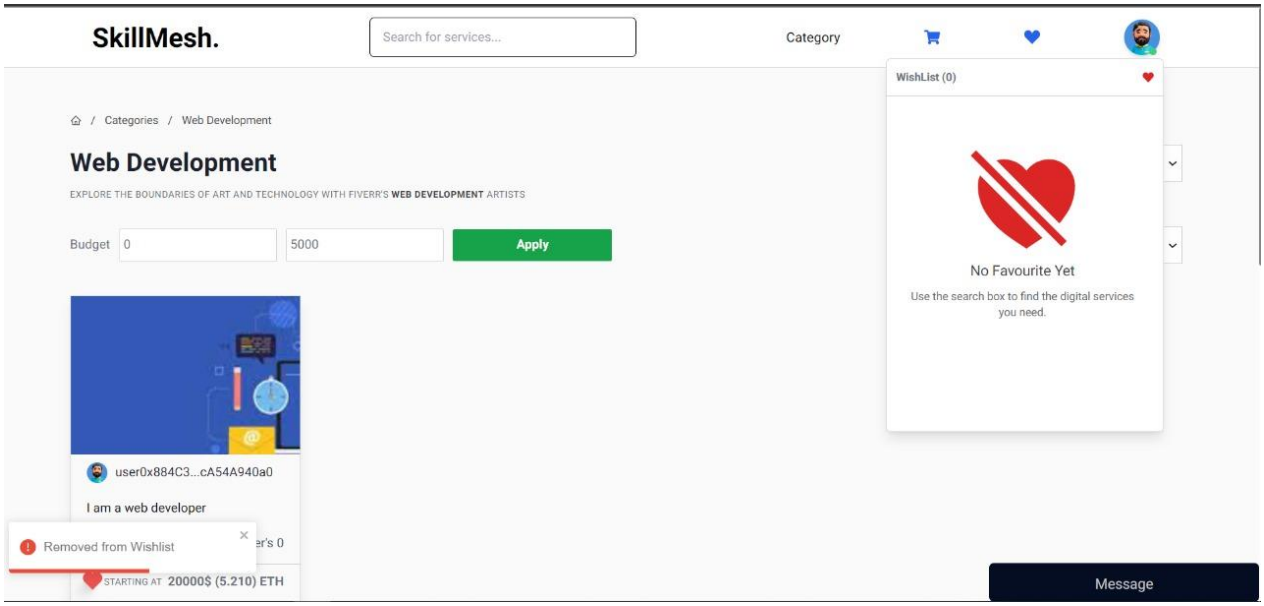


Figure: 6.7 Wish List Testing

Chapter 7

Summary, Conclusion and Future Enhancements

Chapter 7: Summary, Conclusion & Future Enhancements

7.1. Project Summary

SkillMesh is a **DeFi** project that aims to connect individuals who have specific skills and expertise with those who are seeking those skills, all while utilizing **blockchain** technology and **smart contracts** to facilitate secure, transparent, and efficient interactions.

The platform serves as a **marketplace** where users can offer their skills, services, and expertise to a global audience. Users who possess skills in areas such as graphic design, programming, writing, tutoring, translation, and more can create profiles and list the services they offer.

7.2. Achievements and Improvements

- Smart Contract Successful deployed
- Login/Signup using MetaMask is implemented
- Transactions using smart contract implemented
- User friendly GUI
- Chat integrated
- Filters integrated for better gig searching
- Wish list for saving favorite gigs
- Notification features integrated
- Data retrieves form database
- Seller Account Creation & Gig Creation

Improvements:

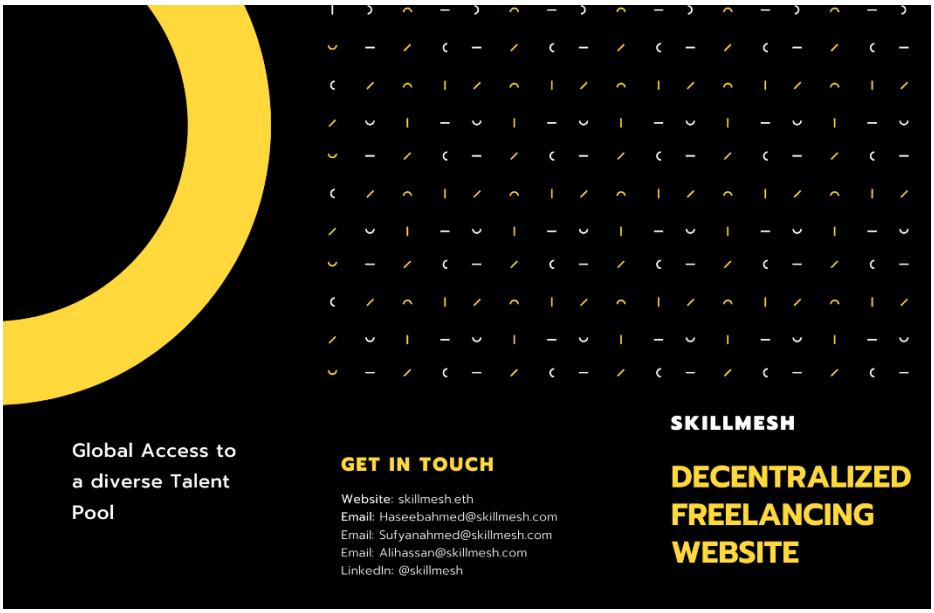
- Launching Token for Transaction
- Implementing Bidding System

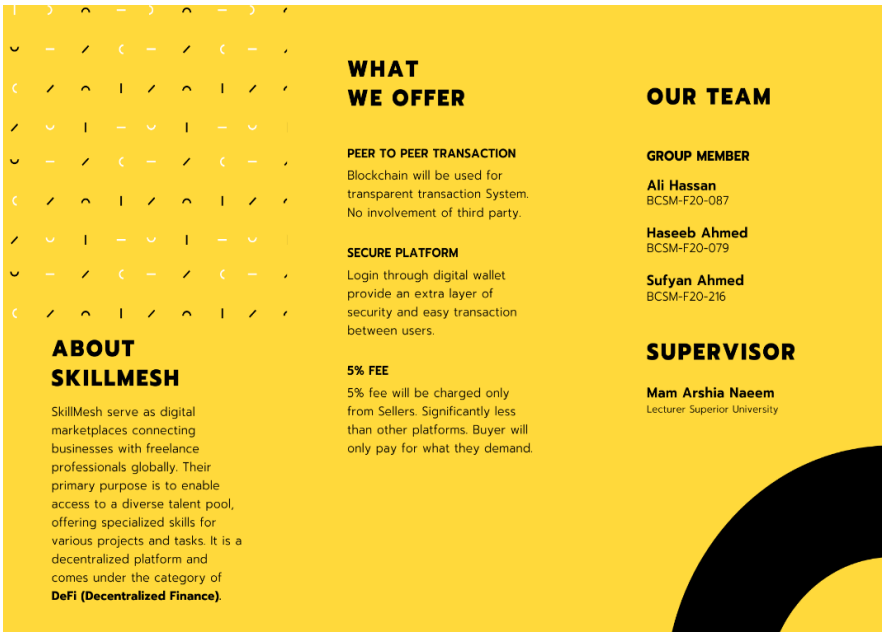
Appendices

Appendix A: Information / Promotional Material

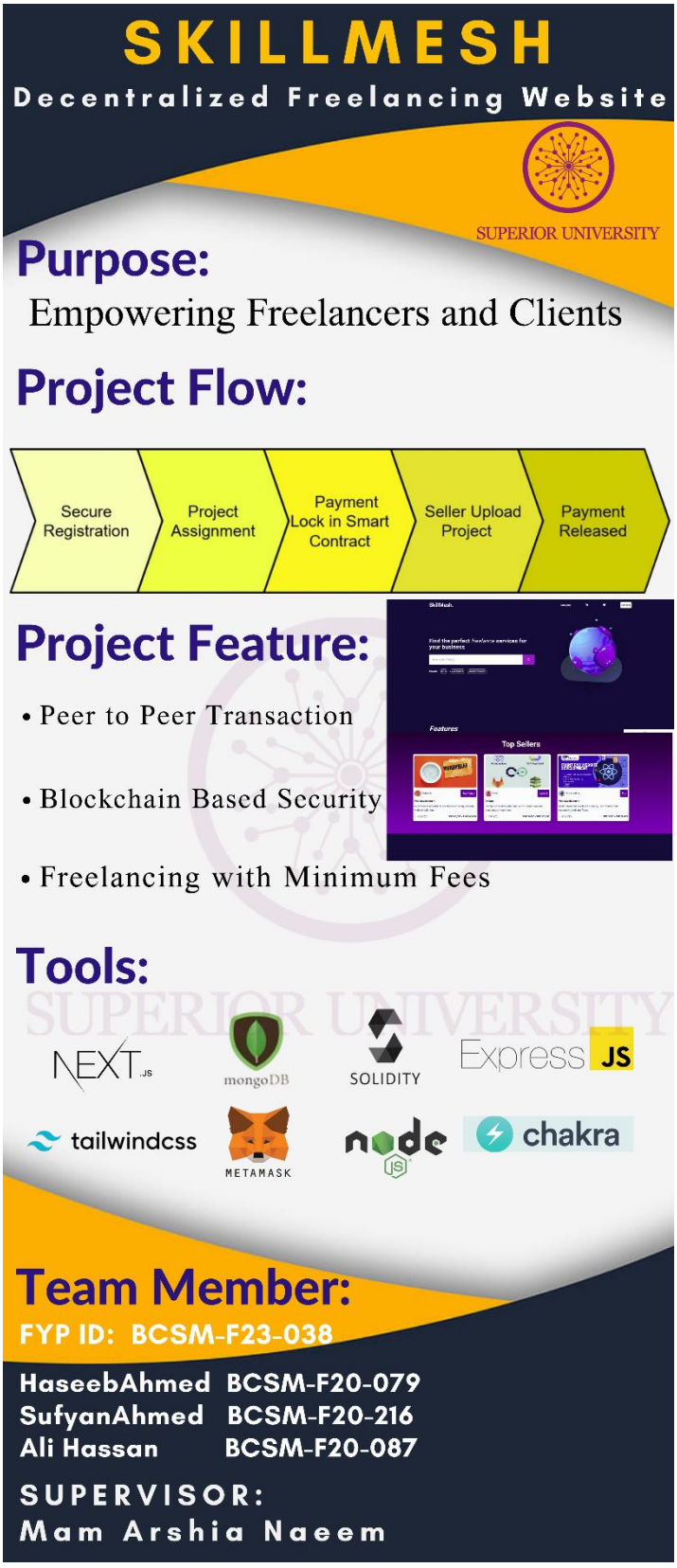
On applying Best Practice regarding Marketing Strategies, we qualified it with those such as Broacher and Standee. Some other things to post the design and running state of our work post on social media like Facebook Instagram etc.

A.1. Broacher





A.2. Standee


SKILLMESH
Decentralized Freelancing Website
SUPERIOR UNIVERSITY
Purpose:
Empowering Freelancers and Clients
Project Flow:
Secure Registration
Project Assignment
Payment Lock in Smart Contract
Seller Upload Project
Payment Released
Project Feature:
• Peer to Peer Transaction
• Blockchain Based Security
• Freelancing with Minimum Fees
Tools:
NEXT.JS
mongoDB
SOLIDITY
Express JS
tailwindcss
METAMASK
node JS
chakra
Team Member:
FYP ID: BCSM-F23-038
HaseebAhmed BCSM-F20-079
SufyanAhmed BCSM-F20-216
Ali Hassan BCSM-F20-087
SUPERVISOR:
Mam Arshia Naeem

Reference and Bibliography

Reference and Bibliography

[1] Miss. Kate Stacy, "*How Much Does Fiverr Take? (Fees Explained for Beginners)*" Website Sophical Content, 9th June 2023. [**Error! Reference source not found. Error! Reference source not found.**]

[2] Mr. Avneesh Agarwal, "Deploy a Smart Contract on Polygon (MATIC) " Website third web, 10th March 2023 [**Error! Reference source not found. Error! Reference source not found.**]

Index

Index

[A]

[B]

[C]