

Blockchain Freelancing Platform for Secure Payments and Skill-Based Project Matching Using NLP

***A Project Report submitted
in partial fulfillment of the requirements
for the award of the degree of***

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE & ENGINEERING

By

1. R. Nikhila – 21B01A05F1
2. R. Pavani Durga – 21B01A05F2
3. R. Bhavana – 21B01A05F9

Under the esteemed guidance of
Mr. P. Nagaraju Assistant Professor



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SHRI VISHNU ENGINEERING COLLEGE FOR WOMEN(A)**
(Approved by AICTE, Accredited by NBA & NAAC, Affiliated to JNTU Kakinada)
BHIMAVARAM – 534 2022024 – 2025

SHRI VISHNU ENGINEERING COLLEGE FOR WOMEN(A)
(Approved by AICTE, Accredited by NBA & NAAC, Affiliated to JNTU Kakinada)
BHIMAVARAM – 534 202

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING



CERTIFICATE

*This is to certify that the project entitled "**Blockchain Freelancing Platform for Secure Payments and Skill-Based Project Matching Using NLP**", is being submitted by **R. Nikhila, R. Pavani Durga, R. Bhavana** bearing the **Regd. No. 21B01A05F1, 21B0A05F2, 21B01A05F9** in partial fulfillment of the requirements for the award of the degree of "**Bachelor of Technology in Computer Science & Engineering**" is a record of bonafide work carried out by her under my guidance and supervision during the academic year 2024–2025 and it has been found worthy of acceptance according to the requirements of the university.*

Internal Guide

Head of the Department

External Examiner

ACKNOWLEDGMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose constant encouragement and guidance has been a source of inspiration throughout the course of this seminar. We take this opportunity to express our gratitude to all those who have helped us in this seminar.

We wish to place our deep sense of gratitude to **Sri. K. V. Vishnu Raju, Chairman of SVES**, for his constant support on each and every progressive work of mine.

We wish to express our sincere thanks to **Dr. P. Srinivasa Raju, Director Students Affairs & Admin for SVES Group of Institutions**, for being a source of inspiration and constant encouragement.

We wish to express our sincere thanks to **Dr. G. Srinivasa Rao, Principal of SVECW** for being a source of inspiration and constant encouragement.

We wish to express our sincere thanks to **Prof. P. Venkata Rama Raju, Vice-Principal of SVECW** for being a source of inspiration and constant encouragement.

We wish to place our deep sense of gratitude to **Dr. P. Kiran Sree, Head of the Department of Computer Science & Engineering** for his valuable pieces of advice in completing this seminar successfully.

We are deeply indebted and sincere thanks to our PRC members **Dr. K. Ramachandra Rao, Dr. P. R. Sudha Rani, Dr. R. Anuj, Dr. N. Silpa** for their valuable advice in completing this project successfully.

We are deeply thankful to our project coordinator **Dr. P. R. Sudha Rani, Professor** for her indispensable guidance and unwavering support throughout our project's completion. Her expertise and dedication have been invaluable to our success.

Our deep sense of gratitude and sincere thanks to **Mr. P. Nagaraju, Assistant Professor** for his unflinching devotion and valuable suggestions throughout my project work.

Project Associates:

1. 21B01A05F1 - R. Nikhila
2. 21B01A05F2 - R. Pavani Durga
3. 21B01A05F9 - R. Bhavana

ABSTRACT

The Blockchain Freelancing Platform for Secure Payments and Skill-Based Project Matching using NLP aims to revolutionize the freelance marketplace by combining the power of blockchain technology and Natural Language Processing (NLP). In this platform, blockchain ensures secure, transparent, and fraud-free transactions between clients and freelancers. By leveraging smart contracts, payments are processed automatically and only released upon the successful completion and verification of a project, eliminating the risks of disputes and delayed payments. This system guarantees both clients and freelancers can trust the process, fostering a more reliable and secure environment for remote work engagements.

The platform also incorporates NLP-based algorithms to enhance the hiring experience by accurately matching freelancers to projects based on their skills, experience, and the requirements of employers. This intelligent matching system minimizes the time and effort involved in finding the right talent, ensuring that clients are connected with freelancers who meet their precise needs. By analyzing detailed project descriptions and freelancer profiles, the NLP engine optimizes recommendations, enhancing the efficiency of the hiring process and improving project outcomes.

To facilitate seamless project creation and management, clients are provided with an intuitive interface to post projects with clear requirements, milestones, and deadlines. Freelancers, in turn, can create detailed profiles that showcase their skills, experience, hourly rates, and personal bio, making it easier for clients to identify suitable candidates for their projects. The profile details are essential in the platform's ranking system, which evaluates freelancers based on their qualifications, performance, and feedback from past clients.

Moreover, the platform's transparent system, powered by blockchain, guarantees that all transactions, including payments, are traceable, secure, and free from fraudulent activities. The smart contract feature automatically releases payments only when the client verifies the completion of the project, ensuring a fair and hassle-free process for both parties. The project aims to bridge the gap between freelancers and clients, offering a trustworthy, efficient, and streamlined solution to the freelance economy. By addressing the most pressing issues in the freelancing industry, such as delayed payments, trust

concerns, and inefficient hiring processes, this platform provides a much-needed upgrade to traditional freelancing systems.

Table Of Contents

S.No	Topic	Page No
1.	Introduction	7 – 10
2.	System Analysis	11 - 14
	2.1 Existing System	
	2.2 Proposed System	
	2.3 Feasibility Study	
3.	System Requirements Specification	15 - 17
	3.1 Software Requirements	
	3.2 Hardware Requirements	
	3.3 Functional Requirements	
4.	System Design	18 - 28
	4.1 Introduction	
	4.2 Data flow diagrams (or) UML Diagrams	
5.	System Implementation	29 - 40
	5.1 Introduction	
	5.2 Project Modules	
	5.3 Algorithms	
	5.4 Libraries and Their Usage	
	5.5 Screens	
6.	System Testing	41 – 53
	6.1 Introduction	
	6.2 Testing Methods	
	6.3 Test Cases	
7.	Conclusion	54 – 56
8.	Bibliography	57 - 58
9.	Appendix	59 - 67

1. INTRODUCTION

The Blockchain Freelancing Platform for Secure Payments and Skill-Based Project Matching using NLP aims to revolutionize the freelancing industry by addressing the critical challenges faced by both freelancers and clients. As the freelance workforce expands globally, traditional platforms often struggle with issues like payment delays, fraud, and inefficient project matching. These challenges hinder the growth of the freelance economy and reduce the trust and reliability of online freelance platforms. Our project seeks to eliminate these issues through the integration of blockchain technology and Natural Language Processing (NLP), ensuring a more efficient, secure, and transparent environment for freelancers and employers.

At the core of our platform lies blockchain technology, which ensures that all transactions between freelancers and clients are secure, traceable, and immutable. By utilizing smart contracts, the platform automates and enforces agreed-upon terms for both payments and work delivery, providing a fail-safe mechanism to guarantee that funds are only released when the work is completed and verified. This eliminates the potential for fraud and ensures that clients and freelancers can rely on a fair and efficient process.

Additionally, our platform integrates NLP-based algorithms to improve the hiring process. These algorithms analyze and match freelancer profiles with project requirements based on key attributes such as skills, experience, and specific project needs. By using advanced NLP techniques, the system reduces the complexity of finding the ideal match, making the hiring process faster and more accurate for both freelancers and employers. This automation leads to better outcomes, as clients can be assured that they are working with the right talent for the job, while freelancers are more likely to find projects that align with their expertise.

The project offers a comprehensive solution by allowing clients to easily create and manage projects with clear requirements, while freelancers can build detailed profiles showcasing their work experience, skills, and hourly rates. The platform also includes a ranking system that evaluates freelancers based on feedback, performance, and qualifications, helping clients make more informed decisions when selecting talent.

In conclusion, the Blockchain Freelancing Platform offers a secure, transparent, and efficient solution to the issues plaguing the freelance industry. By merging blockchain's reliability with the power of NLP, this platform will help create a better, more trustworthy freelancing environment for both employers and freelancers, fostering growth and improving the overall freelancing experience.

By offering a transparent system powered by blockchain, our platform ensures that both freelancers and clients can trust the entire process from start to finish. The immutable nature of blockchain guarantees that all transactions are recorded and cannot be altered, providing an additional layer of security that traditional freelancing platforms lack. This not only enhances the trust between the parties involved but also creates a sense of accountability and peace of mind, knowing that all data and payments are securely handled.

Furthermore, the integration of smart contracts serves as a cornerstone of the platform's operational efficiency. These self-executing contracts automatically execute the terms agreed upon by both parties once certain conditions are met. For example, once a freelancer completes the deliverables according to the client's requirements, the smart contract ensures that payment is made immediately and without the need for intermediaries. This reduces the time typically spent in payment verification and minimizes administrative overheads for both freelancers and clients.

On the other hand, the NLP-powered matching system ensures that the process of finding the right freelancer or project is streamlined and efficient. The platform uses sophisticated language processing techniques to analyze the text in project descriptions and freelancer profiles, ensuring that only the most relevant matches are suggested. This eliminates the traditional inefficiencies associated with keyword-based matching, where freelancers might be overlooked or incorrectly matched to projects that don't align with their skill sets. By accurately understanding the context and nuances of both job descriptions and profiles, NLP empowers clients to quickly identify freelancers who can meet their project needs with the required skill and experience.

Moreover, the platform also focuses on providing freelancers with the tools they need to create comprehensive profiles that reflect their expertise, work history, and skill levels. Through these detailed profiles, freelancers can build their online presence and credibility, which is essential for securing high-quality projects. Clients, in turn, have access to a well-organized pool of talent, making it easier to evaluate potential candidates based on objective criteria such as previous work, client feedback, and ranking scores.

Incorporating these advanced technologies into a single, unified platform addresses critical gaps in the freelancing ecosystem. Blockchain technology, combined with smart contracts and NLP, offers a powerful framework to enhance transparency, security, and efficiency. As a result, the platform not only benefits clients and freelancers but also contributes to the broader freelancing economy by setting a new standard for how online work and payments should be managed.

In conclusion, this project is poised to create a more secure, efficient, and trustworthy freelancing experience for all parties involved. By combining the latest in blockchain and NLP technology, we aim to improve the way projects are matched, executed, and paid for, thereby addressing many of the pain points that have plagued the freelancing industry for years. Through this platform, we envision a future where freelancers can focus on delivering their best work, and clients can confidently engage with skilled professionals, knowing that the process will be seamless and secure.

2. SYSTEM ANALYSIS

2.1 Existing System:

In the current freelancing platforms, payments are processed through third-party services, which can often result in delays and high service fees. These intermediaries create a bottleneck in the payment process and may increase costs for both freelancers and clients. Furthermore, the matching process between freelancers and projects relies on simple keywords and basic search algorithms, which may overlook skilled professionals, leading to missed opportunities for both clients and freelancers.

Additionally, the platforms are largely centralized, meaning that a single authority controls the platform, making it vulnerable to hacking and downtime. In these systems, centralized control enforces rules and processes, but the reliance on a single authority can cause issues, such as platform outages or unfair practices. Clients and freelancers are required to trust the platform completely for ensuring fairness, which can often lead to a lack of transparency and trust, especially in cases of disputes or delayed payments.

2.2 Proposed System:

In contrast to the existing model, the proposed system offers significant improvements by utilizing blockchain technology for secure payments. By adopting smart contracts, the platform eliminates the need for third-party payment processors, enabling direct, automated transactions that are processed only when pre-defined conditions are met. This ensures that freelancers are paid promptly once they complete the agreed-upon work, reducing the potential for fraud and eliminating delays and additional fees that are typical of traditional freelancing platforms. With the blockchain's immutable ledger, every transaction is transparently recorded, allowing both freelancers and clients to have confidence in the platform's reliability.

Additionally, the platform will incorporate Natural Language Processing (NLP) to create a more intelligent matching system. Unlike simple keyword-based approaches, NLP will allow the system to understand the meaning and context behind both job descriptions and freelancer profiles. This will result in more accurate and effective matches, ensuring that clients find freelancers with the precise skills and experience required for their projects. By analyzing skills, experience, and project requirements, the NLP algorithms will drastically reduce mismatches and ensure that freelancers are only proposed for projects they are truly qualified for, improving the hiring efficiency and overall satisfaction for both parties.

The proposed platform will operate under a decentralized model, removing the need for a central authority to manage the system. This decentralized approach offers enhanced security, as there is no single point of vulnerability that could be targeted by malicious actors. Furthermore, the decentralization of control ensures that there is no risk of downtime caused by a central entity's failure. With blockchain-based automation and smart contracts, all processes, from project initiation to payment release, will be carried out seamlessly, making the system highly reliable and efficient. Clients and freelancers can trust that their transactions and interactions are secure, transparent, and fairly executed without the interference of intermediaries.

Finally, the platform will feature a ranking and reputation system for freelancers. Based on factors like completed projects, client feedback, skill proficiency, and profile completeness, this ranking system will help clients identify the best freelancers for their specific needs. With a transparent ranking mechanism, freelancers can build a strong reputation, and clients can confidently hire the most qualified professionals for their projects.

2.3 Feasibility Study:

A feasibility study, sometimes called a feasibility analysis or feasibility report, is a way to evaluate whether or not a project plan could be successful. A feasibility study evaluates the practicality of your project plan in order to judge whether or not you're able to move forward with the project.

Uses Of Feasibility studies:

Feasibility studies can be very helpful for guiding a community's decision-making process. They can provide neutral, third party professional analysis including cost-benefit analysis, alternative options and verification

This is possible only if it is feasible within limited resources and time. The different feasibilities that have to be analyzed are:

- Technical Feasibility
- Operational Feasibility
- Economic Feasibility

Technical Feasibility: The proposed system is technically feasible, given the advancements in blockchain technology and Natural Language Processing (NLP). Blockchain platforms like Ethereum or Hyperledger can be used to manage secure payments and smart contracts, ensuring secure and reliable transactions. NLP algorithms can be employed to analyze and match freelancers to projects based on natural language understanding, enhancing the accuracy and efficiency of the matching process. The development of this system is supported by the availability of cloud computing infrastructure, ensuring scalability, reliability, and performance.

Operational Feasibility: The decentralized nature of the proposed system makes it highly operationally feasible. The removal of intermediaries such as payment processors and project managers reduces operational overheads and increases efficiency. Both clients and freelancers will benefit from the ease of use and automation provided by blockchain and NLP, allowing them to focus on delivering value rather than managing administrative tasks. Additionally, the system's transparent nature ensures that clients and freelancers have access to the information they need, promoting accountability and trust.

Economic

Feasibility: From an economic standpoint, the platform's lower operational costs are one of its most significant advantages. By removing third-party intermediaries and reducing service fees, the platform offers a more cost-effective solution for both clients and freelancers. Moreover, the enhanced matching efficiency enabled by NLP will save time and increase the success rate of job placements, improving the overall experience for all users. The blockchain's ability to automatically handle payments will also streamline financial transactions, reducing administrative burdens and further lowering costs.

3.SYSTEM REQUIREMENTS SPECIFICATION

3.1 Software Requirements:

The proposed blockchain freelancing platform will require several software components to function optimally. These components are as follows:

Blockchain Platform: The platform will utilize blockchain technology, such as Ethereum or Hyperledger, for secure payments and smart contract management. The choice of blockchain will be determined based on transaction speed, scalability, and cost efficiency.

Smart Contract Framework: Smart contracts will be implemented to automate the payment and transaction process between clients and freelancers. These contracts will ensure that payments are only released when predefined conditions are met, such as completion of work or approval by the client.

Natural Language Processing (NLP) Engine: NLP algorithms will be used to analyze job descriptions and freelancer profiles, ensuring accurate matching based on skills and experience. These algorithms will be developed and integrated into the platform to process and understand natural language data from job postings and profiles.

Database Management System: A distributed database will be used to store freelancer profiles, client projects, transaction histories, and smart contract data. The database will ensure data integrity and accessibility across the decentralized system.

3.2 Hardware Requirements:

Hardware requires software to run correctly. Without the correct hardware, your software may not run efficiently or at all. It is important to consider both when making decisions about your IT systems, as this can affect the way you work, your productivity and your business bottom line.

Hardware requirements are:

- Processor - i3 and above
- Hard Disk - 160GB
- RAM - 8Gb

3.3 Functional Requirements:

Functional Requirements:

☐ **User Registration and Profile Management:**

- Clients and freelancers must be able to create accounts and complete detailed profiles, including skills, experience, hourly rates, and bios for freelancers.
- Clients must be able to post projects, specifying job descriptions, required skills, and budget.
- Freelancers should be able to browse and apply for available jobs, based on skillset and availability.

☐ **Blockchain-Based Payment System:**

- The platform must facilitate **secure transactions** via blockchain. Payments will be automatically released based on the completion of tasks defined in the smart contract.

☐ **Smart Contract Automation:**

- Smart contracts automate processes, eliminating the need for intermediaries.
- The contract executes the same logic every time under identical conditions.

☐ **Natural Language Processing (NLP) for Project Matching:**

- The system must analyze **job descriptions** and **freelancer profiles** to accurately match available freelancers to the projects that best fit their skills and experience.
- NLP algorithms will interpret the language used in job postings and profiles, considering not just keywords but the context of skills, experience, and project needs.

Non-functional Requirements:

1. Performance: The system should demonstrate high performance with minimal latency in processing freelancer-project matching and payment transactions.
2. Scalability: The system should be scalable to accommodate an increasing number of freelancers, clients, and transactions without compromising efficiency.
3. Security: The system should ensure the security and privacy of user data.
4. Reliability: The system should function seamlessly without downtime, ensuring continuous access for users.

5. Transparency: All transactions and project agreements should be recorded on the blockchain, ensuring clarity and trust between freelancers and clients.

4.SYSTEM DESIGN

4.1 Introduction:

The software design for the freelance platform integrates modern web technologies and intelligent automation to streamline the hiring process between employers and freelancers. The system employs a structured approach to user registration, project management, and contract execution while incorporating NLP-based project matching to enhance efficiency.

The platform consists of several key modules, including user authentication, project posting, freelancer recommendation, smart contract creation, and payment processing. Employers can register, post projects, and search for freelancers, while freelancers can create profiles, browse job opportunities, and apply for suitable projects.

The NLP-based project matching module leverages machine learning techniques to analyze project descriptions and freelancer profiles, ensuring optimal recommendations. This enhances the hiring process by identifying the most relevant candidates based on skills, experience, and project requirements.

The smart contract module automates agreement creation between employers and freelancers, ensuring transparency and security. Once a freelancer is hired, the contract is executed, and project tracking mechanisms ensure smooth progress. Upon completion, the system facilitates secure payment processing, ensuring that freelancers receive timely payments.

By integrating NLP-driven matching and smart contracts, the platform optimizes the freelancing workflow, reducing manual effort and improving hiring accuracy. This design ensures a seamless and efficient experience for both employers and freelancers, making the platform a reliable solution for modern freelance collaborations.

4.2 Data flow diagrams (or) UML Diagrams:

Unified Modeling Language (UML) diagrams are visual representations used in software engineering to illustrate system architecture, design, and behavior. These diagrams provide a standardized way to depict complex systems, enhancing communication among stakeholders. UML encompasses various diagram types, including class diagrams for illustrating classes and relationships, sequence diagrams for depicting interactions over time, and use case diagrams for outlining system functionalities. Widely adopted in the software development lifecycle, UML diagrams offer a visual means to document, analyze, and design systems, fostering a common understanding among developers, designers, and project stakeholders.

1. Use Case Diagram:

A **Use Case Diagram** represents the **functionalities** of a system by showing how different **actors (users or external systems)** interact with various **use cases (features)** of the system. It defines the **scope of the system** and the **user roles**.

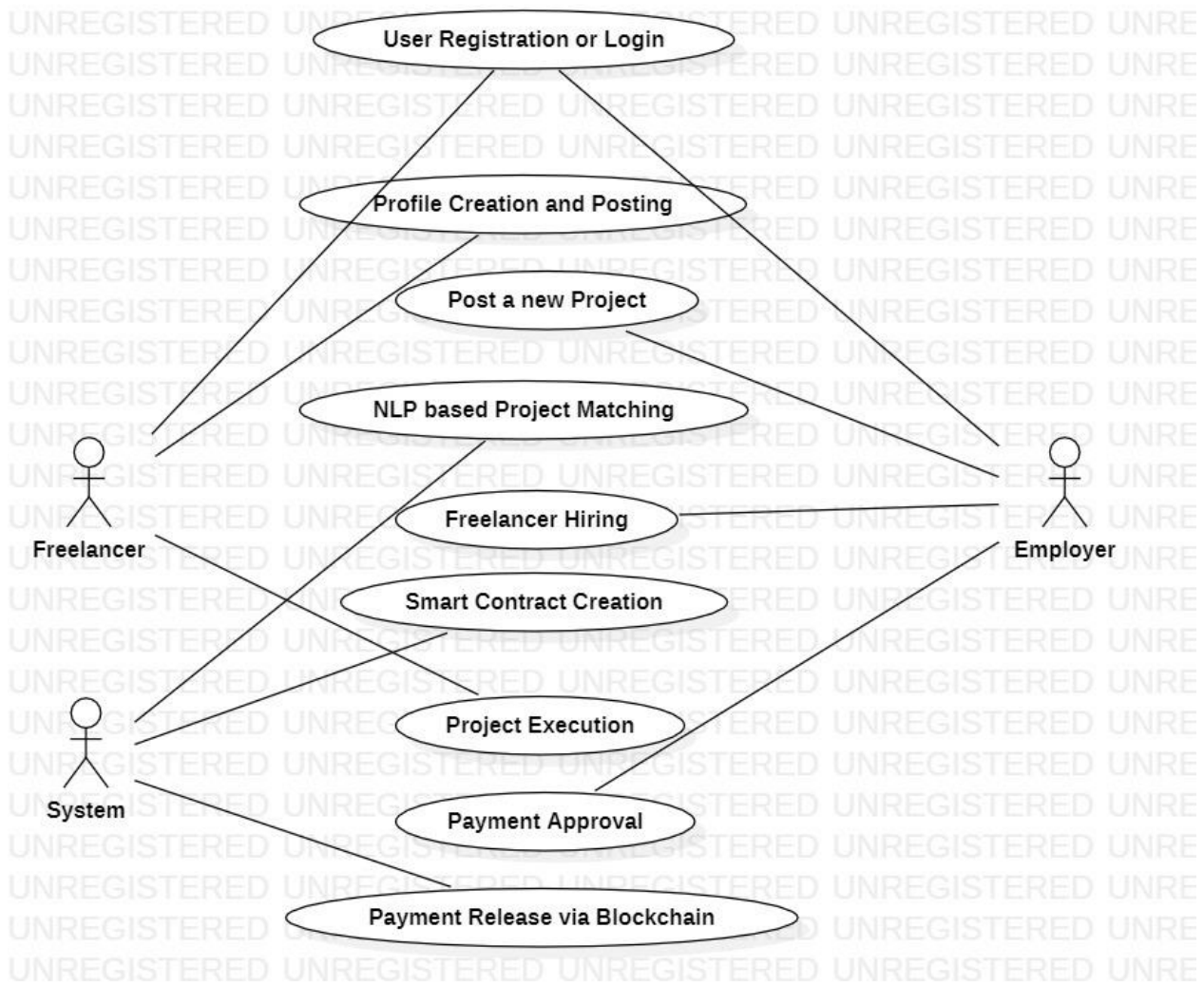
Use Case Diagram in our Freelance Platform:

Actors:

- **Employer** – Posts projects, hires freelancers, manages payments.
- **Freelancer** – Applies for projects, completes work, receives payments.
- **Admin (Optional)** – Manages users, reviews disputes.

Use Cases:

- User Registration/Login
- Post Project (Employer)
- Apply for Project (Freelancer)
- NLP-Based Freelancer Matching
- Hire Freelancer
- Smart Contract Creation
- Project Execution & Submission
- Payment Release



2. Activity Diagram:

An **Activity Diagram** represents the **workflow of a system**, showing the **step-by-step execution of processes**. It includes **decision points, actions, and control flows** to visualize how a process progresses.

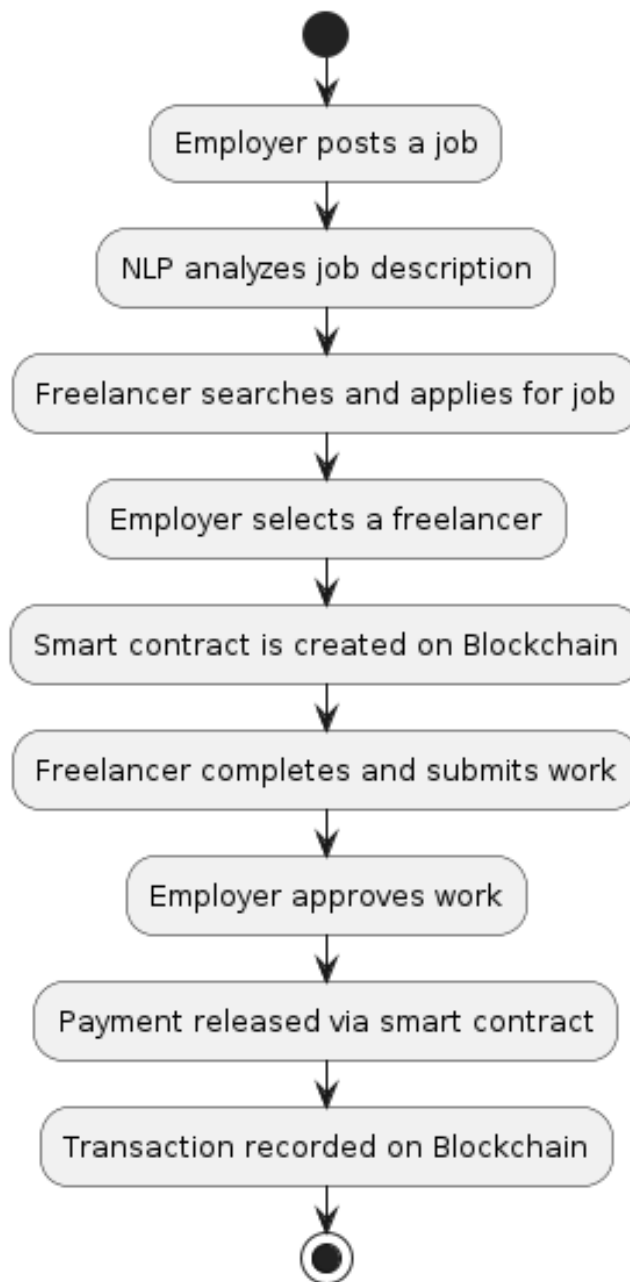
Activity Diagram in Your Freelance Platform:

Your system follows a structured workflow from project creation to completion.

Activity Flow:

1. User **Registers or Logs In**.
2. Employer **Posts a Project**.
3. **NLP-Based Matching** recommends freelancers.
4. Freelancer **Applies for the Project**.
5. Employer **Hires the Freelancer**.
6. **Smart Contract is Created** for security.
7. Freelancer **Executes the Work**.

8. Employer **Reviews & Approves** the work.
9. Payment is **Released** to the freelancer.



3. Sequence Diagram:

A **Sequence Diagram** is a type of **UML (Unified Modeling Language) diagram** used to represent the **interaction between objects** in a system over time. It specifically focuses on **how objects communicate** with each other by **exchanging messages** in a sequential order.

Key Elements of a Sequence Diagram:

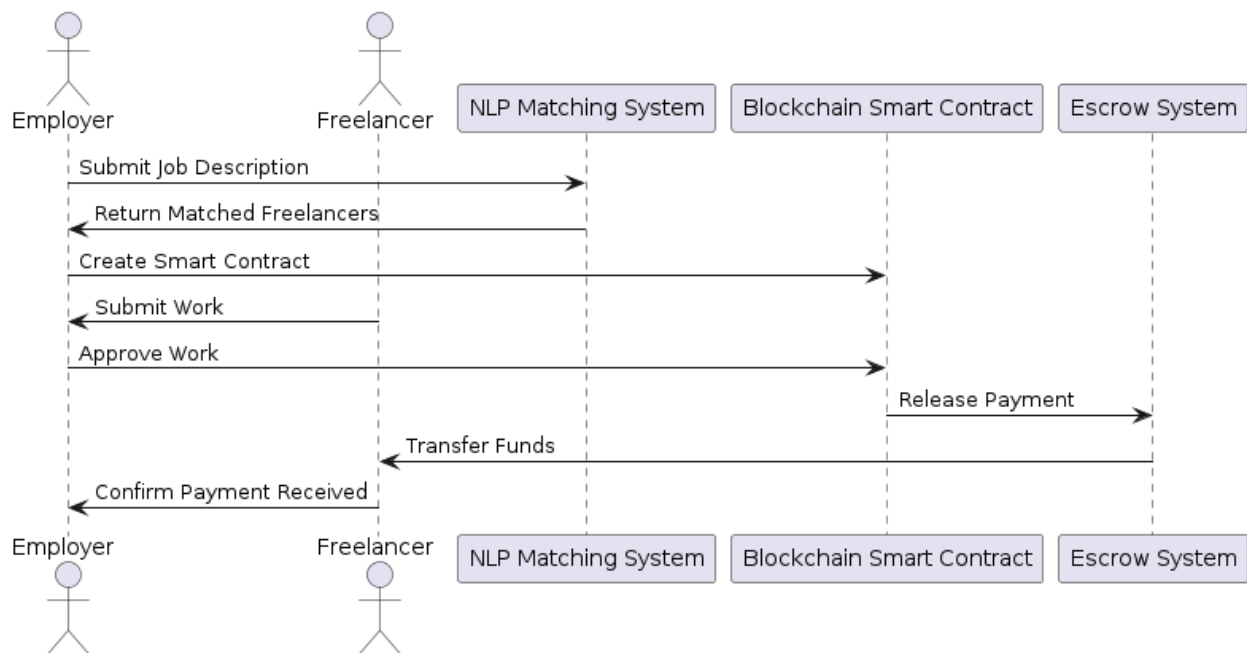
1. **Actors** – Represent users or external systems (e.g., Employer, Freelancer).
2. **Objects** – Represent system components (e.g., Project Management, Payment System).
3. **Lifelines** – Show the **existence of objects** over time.
4. **Messages** – Represent interactions between objects (e.g., Login request, Hire Freelancer).
5. **Activation Bars** – Indicate when an object is actively performing a task.

A **Sequence Diagram** for your freelance platform would represent the step-by-step interaction between **employers, freelancers, and the system**. It follows the logical sequence from project posting to completion and payment.

Sequence Flow in Your Project

1. **User Registration/Login**
 - Employer/Freelancer sends login credentials.
 - System verifies and grants access.
2. **Project Posting & Freelancer Search (Employer)**
 - Employer posts a project.
 - System processes and stores project details.
 - NLP-based recommendation system finds suitable freelancers.
3. **Freelancer Applies for Project**
 - Freelancer views and applies for a project.
 - Employer reviews applications and selects a freelancer.
4. **Hiring & Smart Contract Creation**
 - Employer sends a hiring request.
 - System creates a smart contract binding both parties.
5. **Project Execution & Completion**
 - Freelancer completes work and submits results.
 - Employer reviews the work.
6. **Payment Processing**

- If approved, payment is released to the freelancer.



4. Class Diagram:

A **Class Diagram** is a **structural UML diagram** that represents the **classes, attributes, and relationships** within the system. It defines how data is organized and how objects interact.

Class Diagram in Your Freelance Platform:

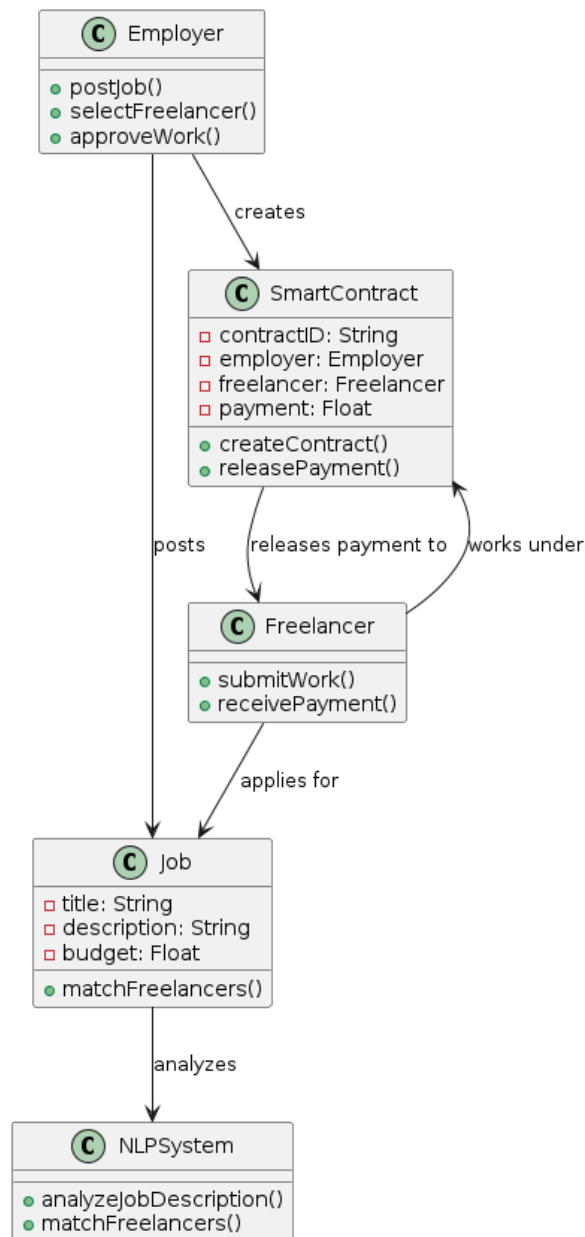
Your system includes **users, projects, contracts, and payments**.

Main Classes & Attributes:

- User (id, name, email, role [Employer/Freelancer])
- Employer (id, companyName, postedProjects)
- Freelancer (id, skills, rating, appliedProjects)
- Project (id, title, description, budget, employer_id, status)
- Contract (id, project_id, employer_id, freelancer_id, status, payment_details)
- Payment (id, contract_id, amount, status, method)

Relationships:

- **Employer** posts **Projects** (1-to-Many).
- **Freelancer** applies to **Projects** (Many-to-Many).
- **Employer and Freelancer** have a **Contract** (1-to-1).
- **Contracts** are linked to **Payments** (1-to-1).



5.Component Diagram:

A **Component Diagram** represents the **physical components** of the system, such as **databases, APIs, front-end, and back-end services**. It shows how different parts of the system **interact**.

Component Diagram in Your Freelance Platform:

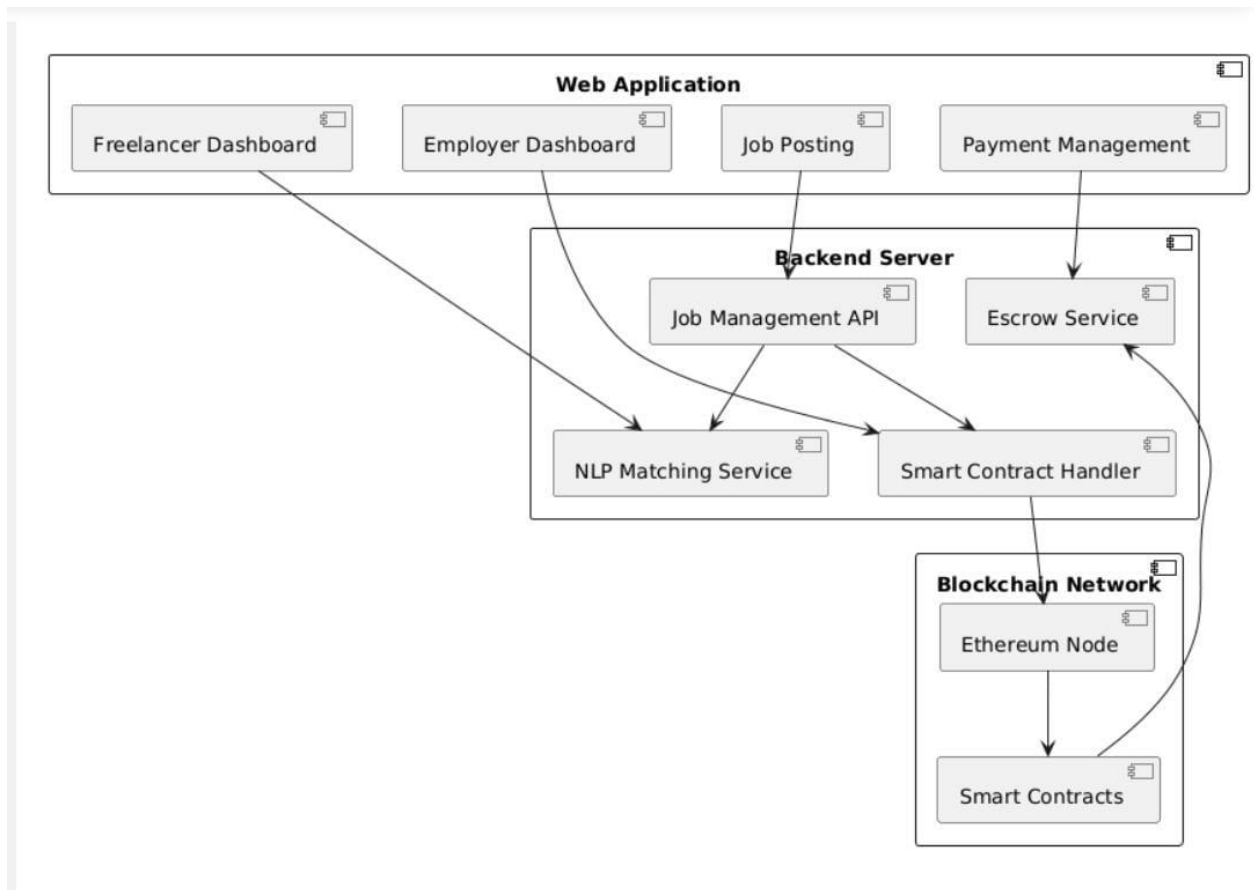
Your platform consists of multiple components, including the **frontend, backend, database, and external services**.

Main Components:

- **Frontend (React, Angular, etc.)** – Handles user interactions.
- **Backend (Django, Flask, Node.js, etc.)** – Processes business logic.
- **Database (PostgreSQL, MySQL, MongoDB)** – Stores user, project, and transaction data.
- **NLP Matching Engine (AI/ML Service)** – Analyzes job descriptions & freelancer profiles.
- **Payment Gateway (Stripe, PayPal, etc.)** – Handles transactions.

Why Use a Component Diagram?

- **Shows system architecture and dependencies.**
- **Helps developers understand how modules interact.**
- **Useful for API and microservices design.**



6. Deployment:

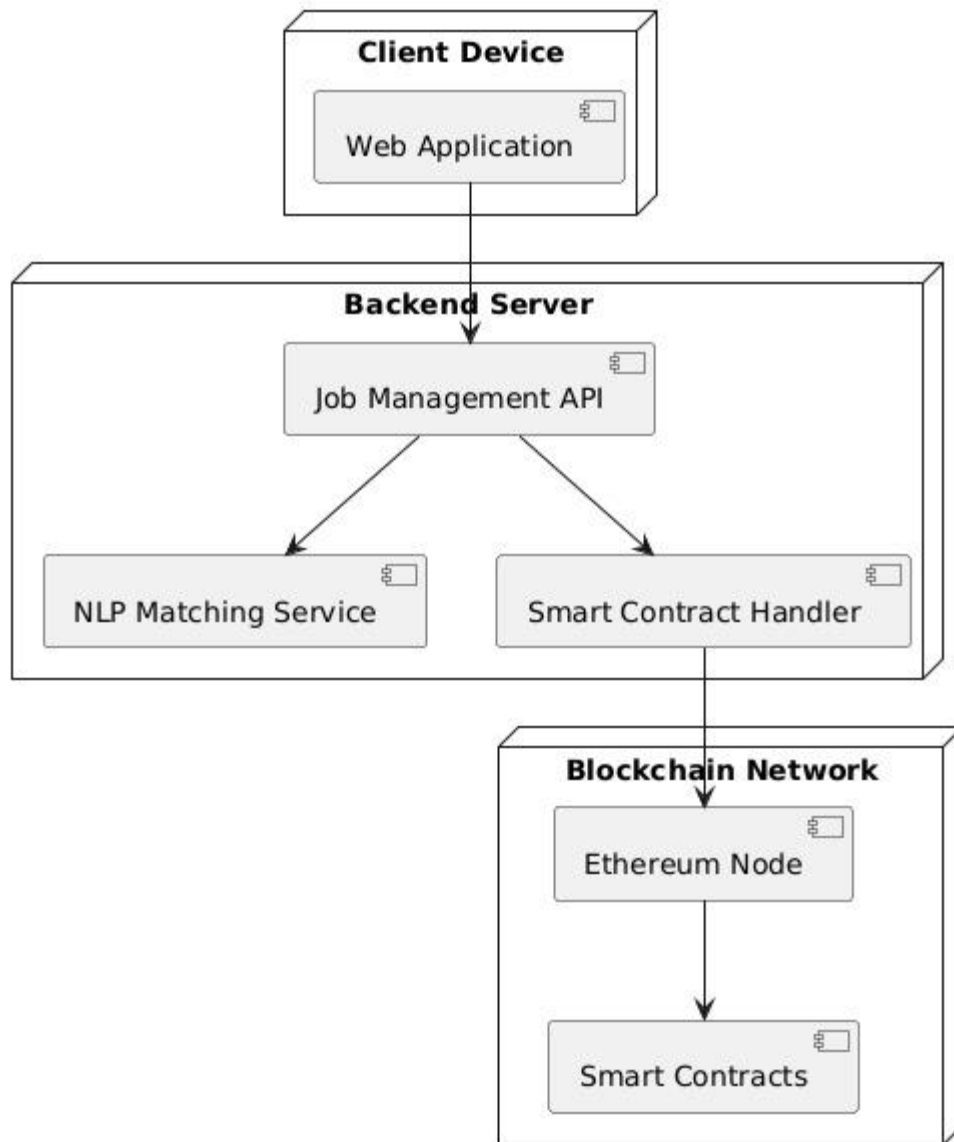
A **Deployment Diagram** represents how software components are **physically deployed** on hardware or cloud infrastructure. It shows **servers, databases, and network interactions**.

Deployment Diagram in Your Freelance Platform:

Your platform may be deployed on **cloud-based infrastructure** like AWS, Azure, or Google Cloud.

Main Deployment Components:

1. **Client Devices** (Web Browser, Mobile App)
2. **Web Server** (Handles HTTP requests)
3. **Application Server** (Runs business logic)
4. **Database Server** (Stores user, project, and contract data)
5. **NLP Server** (Handles NLP-based matching)
6. **Payment Server** (Processes payments securely)



5.SYSTEM IMPLEMENTATION

5.1 Introduction:

The implementation of the Blockchain Freelancing Platform for Secure Payments and Skill-Based Project Matching Using NLP involves a step-by-step integration of blockchain technology and Natural Language Processing (NLP) techniques to create a secure, efficient, and reliable freelancing ecosystem. The system is designed to address common issues in traditional freelancing platforms, such as delayed payments, high service fees, inefficient freelancer-job matching, and security vulnerabilities. By leveraging blockchain for transactions and smart contracts, the platform eliminates the need for intermediaries, ensuring trustless, tamper-proof transactions. Additionally, NLP enhances freelancer-project matching by analyzing job descriptions and freelancer profiles, leading to more accurate and efficient hiring processes. The system follows a modular implementation strategy, allowing seamless functionality and easy maintenance. Each module is developed to handle a specific task, ensuring high performance, security, and scalability.

5.2 Project Modules:

The system consists of several key modules, each performing essential functions to ensure smooth platform operation. These modules work together to deliver a secure and efficient freelancing experience.

1. User Authentication Module

- Provides secure user registration and login.
- Implements multi-factor authentication for enhanced security.
- Maintains user profiles, including skills, experience, and payment details.

2. Project Creation and Management Module

- Enables clients to create, edit, and manage projects.
- Provides a structured form for defining job requirements, budgets, and deadlines.
- Supports file attachments and messaging between freelancers and clients.

3. Freelancer Matching Module

- Utilizes NLP algorithms to extract key skills and experience from freelancer profiles.
- Analyzes job descriptions and ranks freelancers based on relevance.
- Displays the best-matching candidates for clients to review and hire.

4. Smart Contract & Payment Module

- Implements blockchain-based smart contracts for secure and automated payments.
- Holds funds in escrow until project completion and verification.
- Releases payments automatically based on predefined conditions, reducing disputes.

5. Ranking and Recommendation Module

- Evaluates freelancers based on skills, experience, project completion rate, and feedback.
- Uses weighted scoring techniques to rank freelancers for each project.
- Suggests top freelancers to clients for improved hiring efficiency.

5.3 Algorithm:

Several algorithms drive the efficiency and reliability of the platform. These algorithms ensure secure transactions, accurate freelancer matching, and a robust ranking system.

1. NLP-Based Freelancer Matching Algorithm

- Extracts keywords from freelancer profiles and job descriptions.
- Uses cosine similarity and TF-IDF techniques to find the best matches.
- Ranks freelancers based on experience, skill set, and job requirements.

2. Smart Contract Execution Algorithm

- Deploys a smart contract on the blockchain when a project starts.
- Stores payment securely in escrow until project completion.
- Releases payment once the client verifies the delivered work.
- Provides a dispute resolution mechanism if disagreements arise.

3. Ranking and Recommendation Algorithm

- Assigns scores to freelancers based on experience, reviews, and successful project completions.
- Uses machine learning techniques to improve ranking accuracy over time.
- Ensures fair and unbiased ranking by analyzing multiple performance indicators.

5.4 Libraries and Their Usage:

The platform utilizes several key libraries to implement its functionalities efficiently:

- **Streamlit:** Used to develop an interactive and user-friendly web interface for clients and freelancers.
- **re:** Helps in processing and extracting specific text patterns for freelancer matching.
- **sqlite3:** Manages local database storage for user credentials and project details.
- **hashlib:** Ensures secure hashing of sensitive information like passwords.
- **BlockchainInterface:** A custom module handling blockchain interactions for smart contract execution and secure transactions.
- **scikit-learn (TfidfVectorizer & cosine_similarity):** Implements NLP-based ranking and freelancer-job matching.
- **NumPy:** Optimizes mathematical computations for machine learning algorithms.
- **Web3.py:** Enables blockchain connectivity and execution of Ethereum smart contracts.
- **eth-account:** Manages cryptographic wallet keys and signatures for blockchain transactions.
- **JSON:** Handles structured data exchange between different modules and blockchain transactions.
- **Solcx:** Compiles and deploys Solidity-based smart contracts on the Ethereum blockchain.

These libraries work together to ensure secure payments, accurate freelancer-job matching, and seamless blockchain integration.

5.5 Screens:

The user interface of the system includes various screens designed to enhance user experience and accessibility. These screens ensure smooth navigation and efficient interaction between freelancers and clients.

1. User Dashboard

- Displays active projects, earnings, and notifications.
- Shows pending payments and contract statuses.
- Provides a quick overview of freelancer rankings and recommendations.

2. Freelancer Profile Page

- Showcases skills, experience, and past projects.
- Displays ratings, reviews, and hourly rates.
- Allows freelancers to update their profiles and portfolios.

3. Project Posting Screen

- Enables clients to create new job listings.
- Includes sections for job descriptions, budgets, and deadlines.
- Allows clients to browse and select recommended freelancers.

4. Matching Results Screen

- Displays top-ranked freelancers for a given project.
- Includes freelancer profiles, experience, and expected rates.
- Allows clients to shortlist or directly hire freelancers.

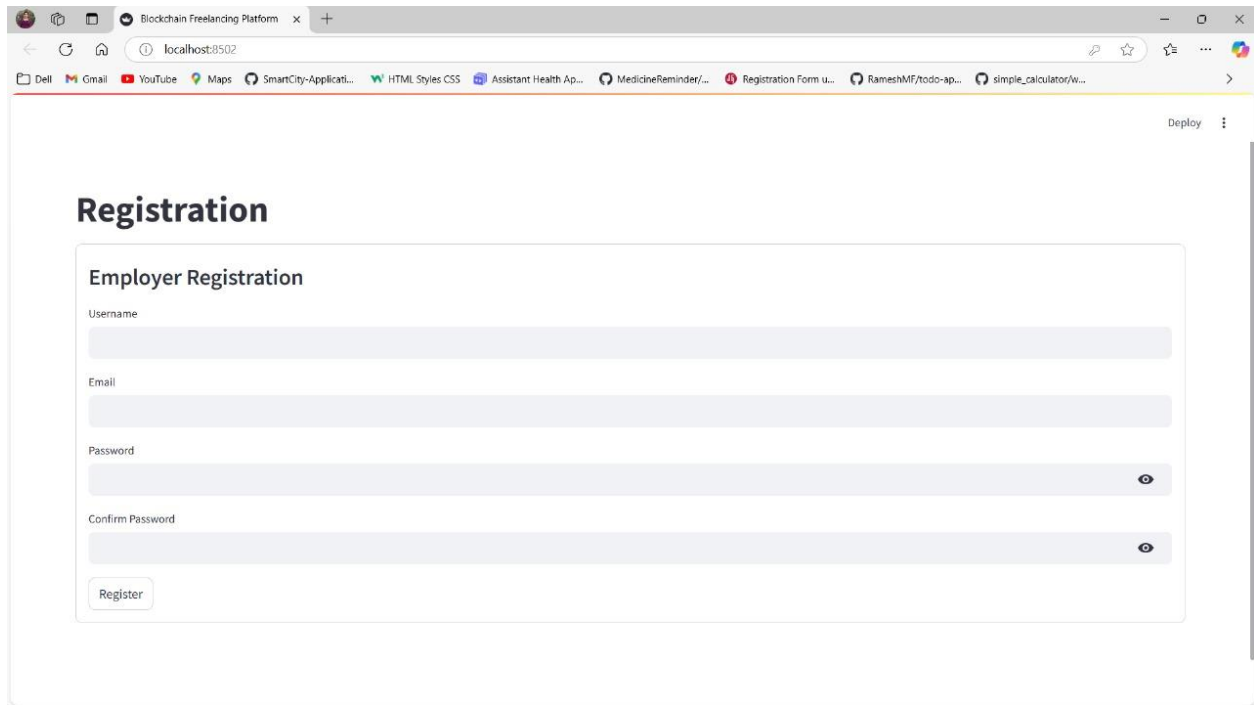
5. Smart Contract and Payment Screen

- Shows contract details, payment terms, and escrow status.
- Provides an option to approve work and release payment.
- Ensures transaction transparency and security through blockchain.

6. Feedback and Review Screen

- Allows clients to rate and review freelancers upon project completion.
- Displays a freelancer's overall rating and feedback history.
- Helps maintain credibility and trust within the platform.

Home Page, Employer Registration Page, Freelancer Registration Page, Login Page



The screenshot shows a web browser window with the title "Blockchain Freelancing Platform". The address bar shows "localhost:8502". The page content is titled "Registration" and contains a form for "Employer Registration". The form has four input fields: "Username", "Email", "Password", and "Confirm Password". Each input field has a light blue placeholder. The "Password" and "Confirm Password" fields have a small eye icon to the right, indicating a toggle for password visibility. Below the input fields is a "Register" button. The browser's top bar shows several open tabs, including "Dell", "Gmail", "YouTube", "Maps", "SmartCity-Applicati...", "HTML Styles CSS", "Assistant Health Ap...", "MedicineReminder/...", "Registration Form u...", "RameshMF/todo-ap...", and "simple_calculator/w...". A "Deploy" button is visible in the top right corner of the browser window.

Blockchain Freelancing Platform

localhost:8502

Deploy

Registration

Employer Registration

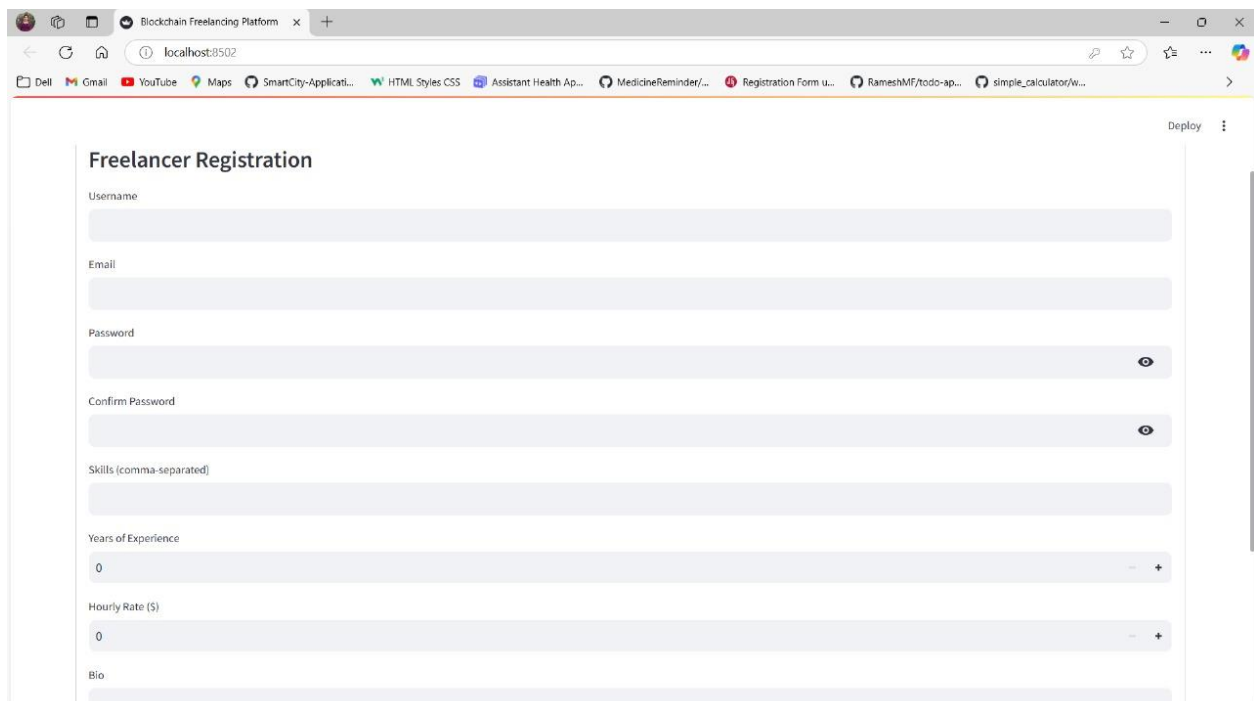
Username

Email

Password

Confirm Password

Register



The screenshot shows a web browser window with the title "Blockchain Freelancing Platform". The address bar shows "localhost:8502". The page content is titled "Freelancer Registration". The form has seven input fields: "Username", "Email", "Password", "Confirm Password", "Skills (comma-separated)", "Years of Experience", and "Hourly Rate (\$)". Each input field has a light blue placeholder. The "Password" and "Confirm Password" fields have a small eye icon to the right, indicating a toggle for password visibility. The "Years of Experience" and "Hourly Rate (\$)" fields have a small minus and plus icon to the right, indicating a range or increment. Below the input fields is a "Bio" field. The browser's top bar shows several open tabs, including "Dell", "Gmail", "YouTube", "Maps", "SmartCity-Applicati...", "HTML Styles CSS", "Assistant Health Ap...", "MedicineReminder/...", "Registration Form u...", "RameshMF/todo-ap...", and "simple_calculator/w...". A "Deploy" button is visible in the top right corner of the browser window.

Blockchain Freelancing Platform

localhost:8502

Deploy

Freelancer Registration

Username

Email

Password

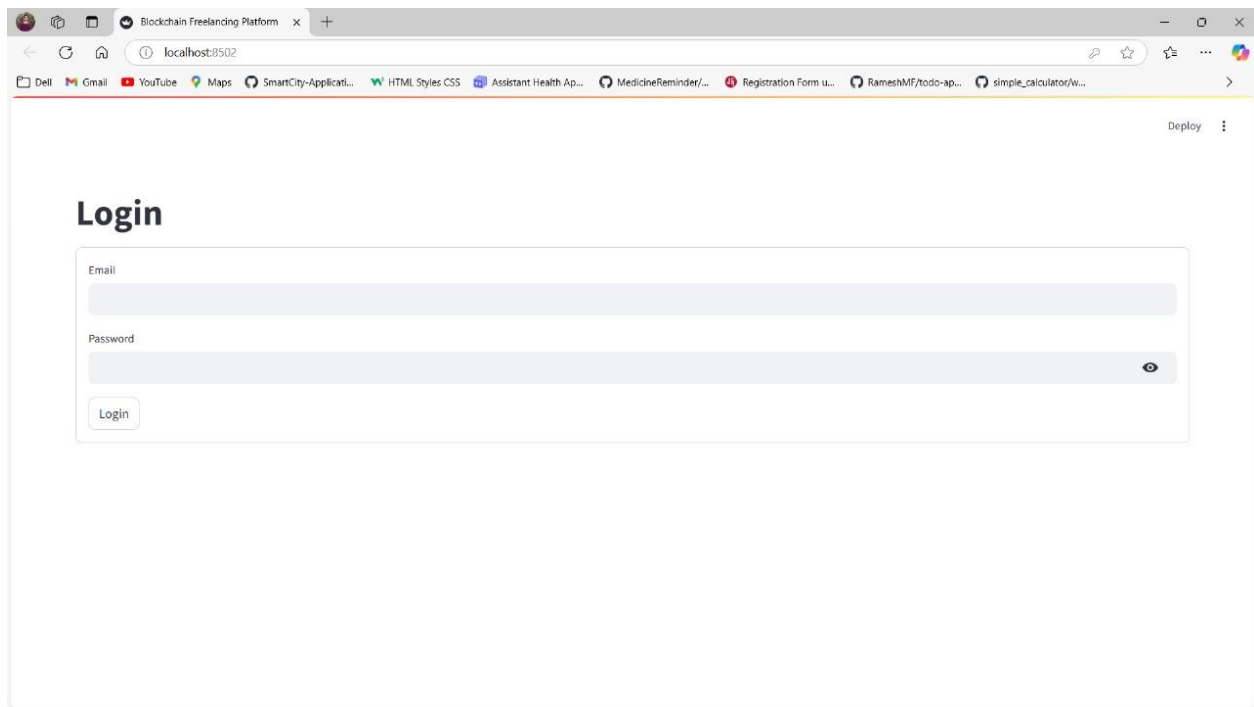
Confirm Password

Skills (comma-separated)

Years of Experience

Hourly Rate (\$)

Bio



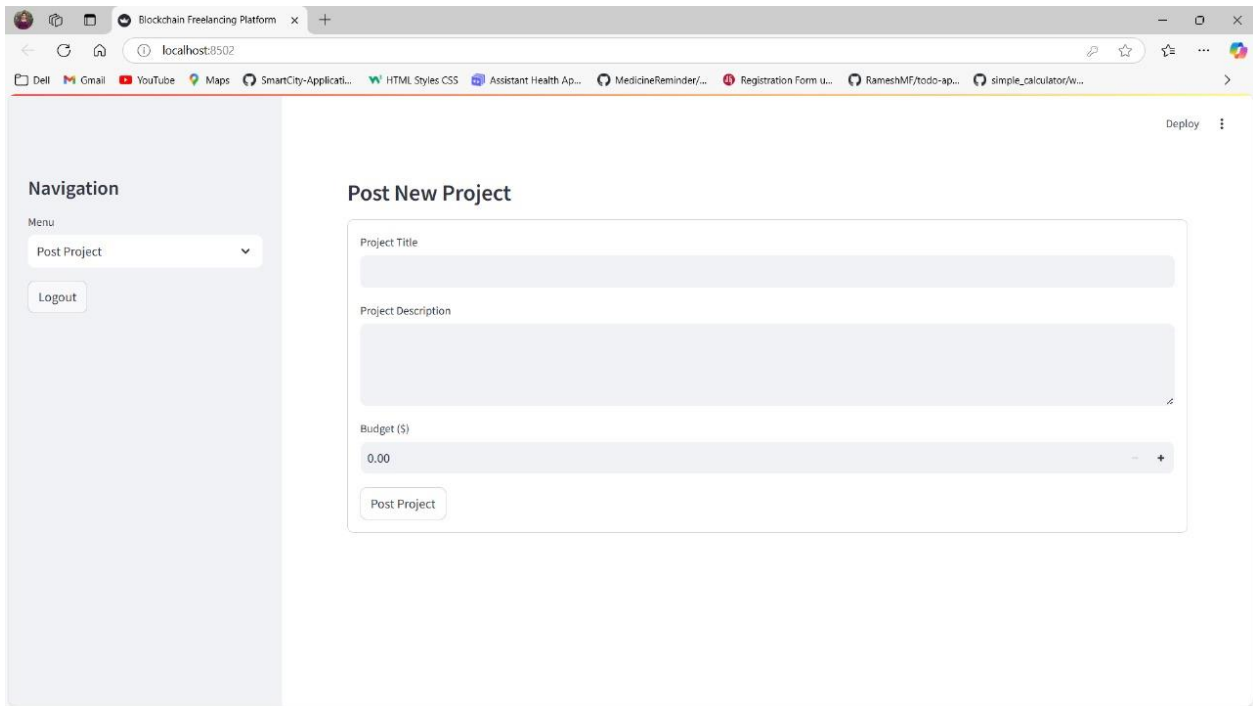
The **Home Page** serves as the gateway to the platform, offering users a seamless experience as they navigate through the website. It provides an overview of the services, highlighting key features such as job postings, freelancer hiring, secure payments, and project tracking. With an intuitive design, the home page ensures that both **employers and freelancers** can quickly find what they need. Clear **call-to-action (CTA) buttons** direct users to sign up or log in, while success stories and testimonials from satisfied users build credibility. A **search bar** allows visitors to explore freelancers or projects without needing to register immediately. The home page is also integrated with a notification system to keep users informed about trending projects, featured freelancers, and important platform updates.

The **Employer Registration Page** is designed for businesses and individuals looking to hire skilled professionals. Employers are required to provide essential details, including their **company name, industry, and location**. For individuals, personal details such as **full name, email address, and contact information** are collected. Additionally, a **secure password setup** ensures account protection. To enhance credibility, employers can opt to verify their email and phone number through an OTP-based authentication system. Once registered, they are guided through a **profile completion process**, where they can specify preferred freelancer skills, set up payment methods, and browse projects.

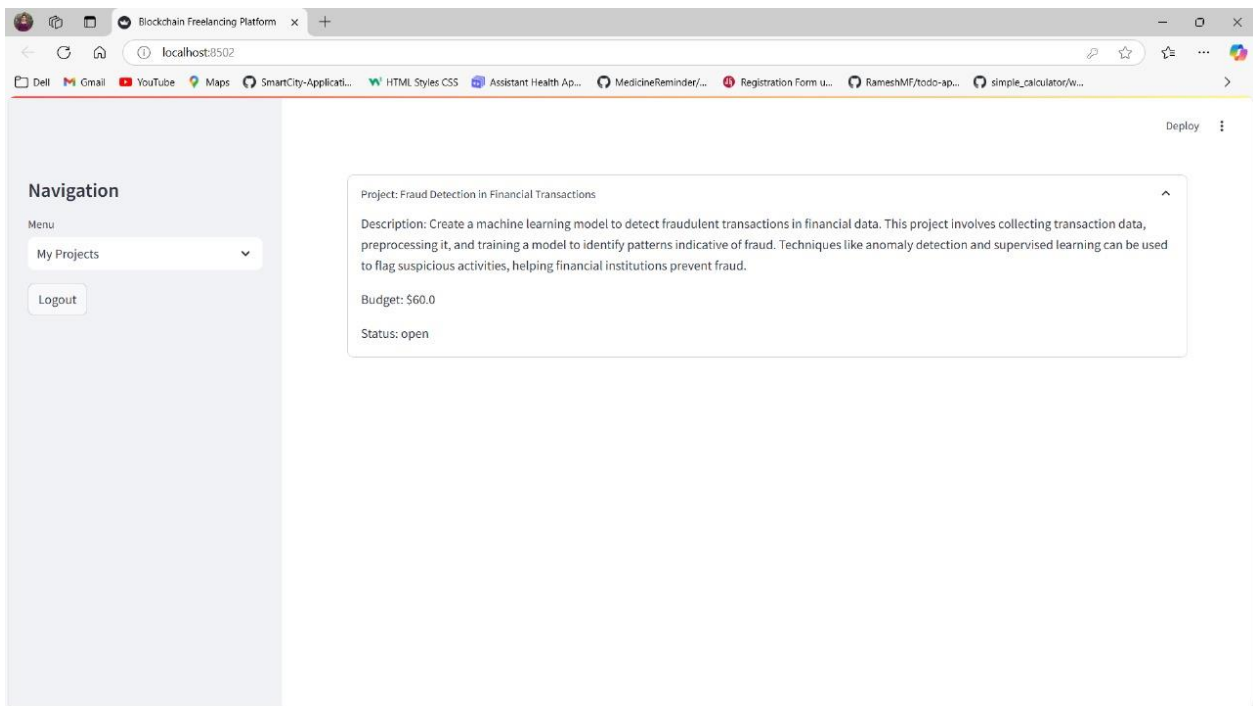
The **Freelancer Registration Page** caters to individuals seeking job opportunities on the platform. Freelancers must enter their **name, email, and professional details**, including expertise, skills, and experience. They can also upload a **profile picture and portfolio**, which helps attract potential employers. To ensure security and authenticity, freelancers can verify their accounts through government-issued ID uploads. The platform allows them to choose

their **preferred working style** (hourly or fixed-rate projects) and set an **availability status**. A **skills assessment feature** may be integrated, allowing freelancers to take tests and showcase their proficiency in specific domains. Once registered, they gain access to job postings and bidding opportunities.

Employer: Post a New Project Page, My Projects Page



The screenshot shows a web browser window with the address bar at localhost:8502. The page has a sidebar on the left with a 'Navigation' menu containing 'Post Project' (selected) and 'Logout'. The main content area is titled 'Post New Project' and contains a form with the following fields: 'Project Title' (text input), 'Project Description' (text area), and 'Budget (\$)' (number input with a value of 0.00). A 'Post Project' button is at the bottom of the form. A 'Deploy' button is visible in the top right corner of the page.

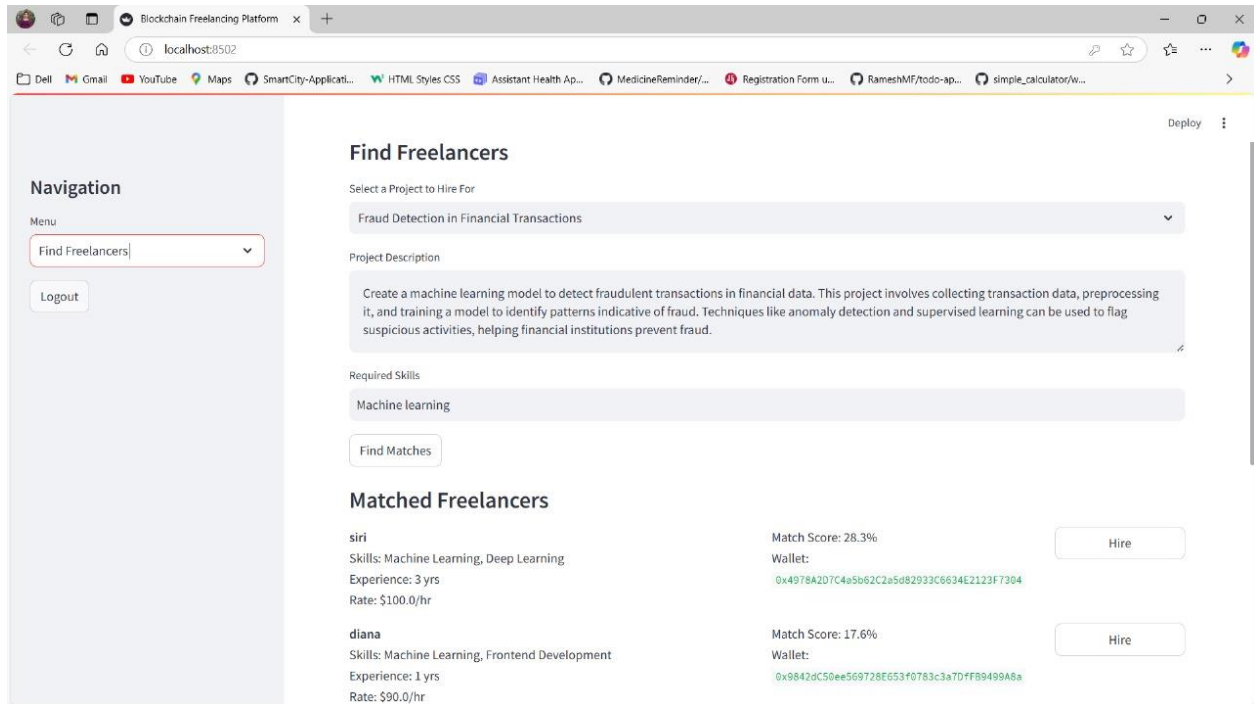


The screenshot shows the same web browser window, but the 'My Projects' menu item is selected in the sidebar. The main content area displays a project card for 'Project: Fraud Detection in Financial Transactions'. The card includes a 'Description' field with the text: 'Create a machine learning model to detect fraudulent transactions in financial data. This project involves collecting transaction data, preprocessing it, and training a model to identify patterns indicative of fraud. Techniques like anomaly detection and supervised learning can be used to flag suspicious activities, helping financial institutions prevent fraud.' Below the description, the 'Budget' is listed as '\$60.0' and the 'Status' is 'open'. A 'Deploy' button is visible in the top right corner of the page.

Employers looking to hire freelancers can do so through the Post a New Project Page, which streamlines the process of creating detailed job postings. The page includes a step-by-step form where employers input the project title, description, budget, expected delivery time, and required skills. A category selection feature ensures the project is listed under the correct domain (e.g., web development, graphic design, writing). Employers also have the option to choose between fixed-price projects and hourly contracts. A preview section allows them to review all details before submission. Once posted, the project appears in the freelancer job feed, making it accessible to professionals matching the criteria.

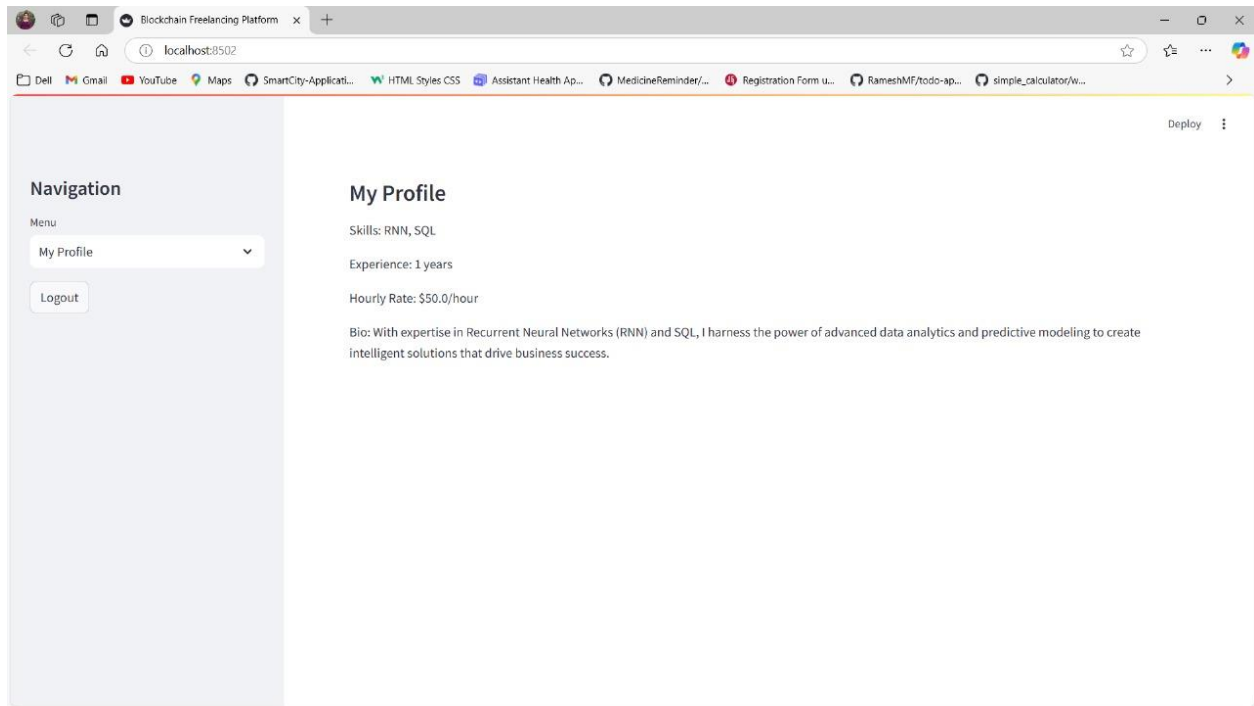
The My Projects Page serves as a dashboard where employers can manage their active, pending, and completed projects. A tab-based navigation system enables easy switching between different project statuses. Employers can track freelancer progress, approve milestone payments, and communicate via an integrated messaging system. Filters and sorting options help them organize projects based on deadline, budget, or freelancer performance. Notifications alert employers when proposals are received, when a project is about to reach its deadline, or if a freelancer submits a request for additional resources. This page also includes a dispute resolution feature where employers can raise concerns regarding project quality, missed deadlines, or payment issues.

Employer: Find a Freelancer Page

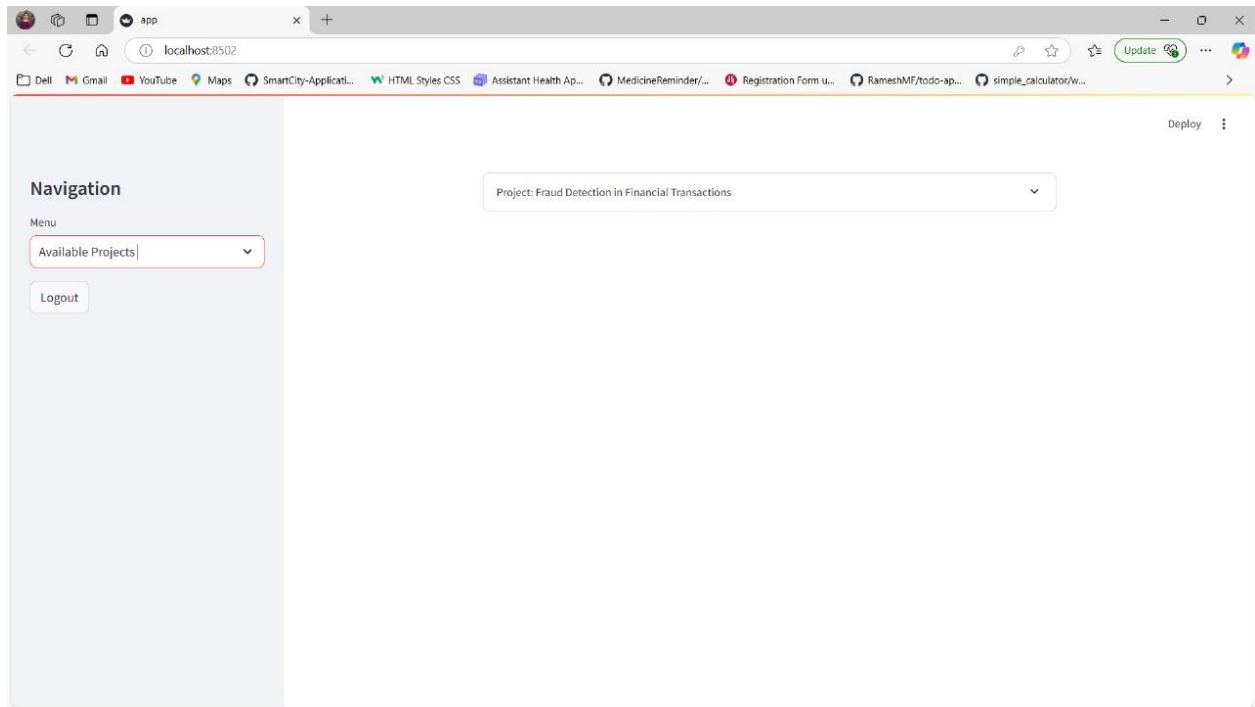


The Find Freelancers Page provides with a list of most relevant candidates for the job. Freelancers are matched to projects based on a comparison of their profiles and the project's requirements. Profiles are evaluated based on key criteria such as skills, experience, and bio to find the best matches for the available projects. Freelancers are ranked according to their fit for each project, and employers are provided with a list of the most relevant candidates for each job.

Freelancer: My Profile Page, Available Projects Page



The My Projects Page allows freelancers to keep track of their assigned work. The dashboard categorizes projects into ongoing, pending approval, and completed sections. Freelancers can view project milestones, submit work, and communicate with clients directly from this page. A progress tracker visually represents the percentage of work completed. If additional resources or clarifications are required, freelancers can request modifications from the employer. Once a project is finished, freelancers can submit it for final approval, after which payments are released.



The My Wallet Page provides freelancers with a secure way to manage their earnings. It displays total earnings, pending payments, and transaction history in an easy-to-read format. Freelancers can withdraw their funds to their preferred payment method, such as bank transfer, PayPal, or cryptocurrency wallets. To ensure financial security, the platform implements secure authentication processes before allowing withdrawals. Additionally, a tax and invoice generation system helps freelancers maintain financial records for tax filing purposes.

6.SYSTEM TESTING

6.1 Introduction:

System Testing is a vital phase in the Software Testing Life Cycle (STLC), acting as the final assurance that a software product meets the requirements before it is deployed for real-world use. It involves testing the entire system, rather than individual components, ensuring that all features and integrations work together harmoniously. This phase plays a crucial role in identifying potential issues in the system that could arise when components interact, and it helps verify that the product meets business, functional, and technical specifications.

Objectives and Importance of System Testing

The primary goal of system testing is to verify that the system functions as intended and to detect any deviations from the requirements. By assessing the software from an end-user perspective, system testing ensures that all aspects of the application, such as features, user interfaces, and back-end systems, work correctly across different scenarios. This phase also verifies that all system components, such as third-party integrations, APIs, and databases, interact seamlessly.

System testing is critical because it identifies problems that could disrupt the user experience, such as performance bottlenecks, security vulnerabilities, compatibility issues, or usability concerns. Furthermore, it helps mitigate risks by providing insight into how the software will perform under varying conditions, ensuring the product meets quality standards before its release to end users.

Key objectives of system testing include:

1. **Verifying Functional Correctness:** Ensures the system meets the specified functional requirements and performs the tasks it is designed to do.
2. **Validating Non-Functional Requirements:** Tests performance, security, compatibility, usability, and recovery under realistic conditions.
3. **Identifying System Integration Issues:** Ensures that various components, including user interfaces, APIs, and databases, function correctly together.
4. **Ensuring Robustness and Reliability:** Identifies and resolves issues related to system crashes, failures, or unexpected behavior during operation.

System Testing Environment

System testing is performed in an environment that closely resembles the production setup. The goal is to simulate real-world conditions and assess how the software performs under actual use cases. A variety of real-world factors are taken into account during this phase, including:

- **Production-like Server and Database Configurations:** The system is tested on servers and databases that replicate the production environment to ensure proper functionality when deployed.
- **Third-Party Integrations:** Any external services or systems that the application interacts with are also tested to confirm seamless interaction.
- **External Factors:** Network speed, traffic volume, and system load are simulated to verify how the software behaves under varying levels of stress.

Black-Box Testing Methodology

System testing is a **black-box testing** methodology, meaning that testers do not have access to the internal code or the inner workings of the system. This approach focuses on examining the system from an end-user perspective, where testers evaluate how the system reacts to various inputs and whether it produces the expected outputs.

Testers do not concern themselves with the system's code logic, focusing instead on ensuring that the software fulfills its intended functionality. The black-box testing approach allows for the identification of functional, performance, security, and usability issues that might not be apparent through other testing methods.

System Testing Methodology

System testing involves a range of testing methods, each focusing on different aspects of the software. Below is an in-depth look at the most commonly used system testing techniques:

1. Functional Testing:

- Verifies that the system performs as expected according to the functional requirements.
- Tests focus on individual features, user interactions, and the expected system behavior under various scenarios.

- **Types of Functional Testing:**
 - **Smoke Testing:** A quick check to ensure that the basic functionality of the system is working.
 - **Sanity Testing:** Verifies that specific functionalities work after minor changes or bug fixes.
 - **Regression Testing:** Ensures that new changes do not interfere with existing functionalities.

2. **Performance Testing:**

- Assesses the system's ability to handle various loads and its performance under stress.
- Includes tests for speed, responsiveness, scalability, and overall system stability.
- **Types of Performance Testing:**
 - **Load Testing:** Determines the system's performance under expected load levels.
 - **Stress Testing:** Assesses how the system behaves under extreme conditions (e.g., high traffic, limited resources).
 - **Scalability Testing:** Tests how well the system scales with increasing workloads.

3. **Security Testing:**

- Focuses on ensuring the system is protected against potential security threats, unauthorized access, and vulnerabilities.
- Includes penetration testing, authentication testing, and data protection validation.
- **Types of Security Testing:**
 - **Penetration Testing:** Simulates attacks on the system to identify weaknesses.
 - **Authentication Testing:** Ensures login mechanisms are functioning securely.
 - **Data Protection Testing:** Verifies that sensitive data is protected during storage and transmission.

4. **Usability Testing:**

- Ensures the system is intuitive, user-friendly, and provides a positive user experience.

- Focuses on user interface (UI) design, ease of navigation, and overall workflow.
- Includes evaluating how easily users can accomplish tasks and whether the system meets their expectations.

5. **Compatibility Testing:**

- Verifies that the system operates across various platforms, devices, browsers, and operating systems.
- Ensures the software can function in different environments, whether users access it through desktop, mobile, or tablet devices.

6. **Recovery Testing:**

- Tests the system's ability to recover from failures, crashes, or interruptions.
- Involves validating backup and restore functionalities, auto-recovery features, and overall system resilience.

The Role of Test Cases in System Testing

Test cases are essential tools in system testing, outlining the specific conditions and steps needed to validate the system's functionality. Each test case defines a scenario, the expected result, and the actual result. A properly structured test case enables testers to track and manage tests, ensuring comprehensive coverage of all system aspects.

Test cases should include:

1. **Test Scenario:** A high-level description of what needs to be tested.
2. **Test Steps:** A series of actions required to execute the test.
3. **Expected Result:** The anticipated outcome based on the specified requirements.
4. **Actual Result:** The actual outcome observed after running the test.
5. **Pass/Fail Status:** Indicates whether the test passed or failed based on the comparison between the expected and actual results.

6.2 Testing Methods:

System Testing encompasses a variety of testing methods, each with a specific focus to ensure that the software meets both functional and non-functional requirements. These methods are crucial for validating the entire system's performance, security, usability, and more. Let's dive deeper into the various testing methods:

1. Functional Testing

Functional Testing validates the software's core features, ensuring that the system behaves according to the specified requirements. The focus is to verify if the system performs the required functions as expected under normal and edge cases. The key types of functional testing include:

- **Smoke Testing:**

- This is often called "build verification testing." It's a quick test to verify that the basic and critical functions of the software are working before more in-depth testing begins.
- It checks for showstopper issues that would prevent further testing.
- Example: A website's homepage loading, login functionality, and basic navigation would be tested.

- **Sanity Testing:**

- Sanity testing is a narrower form of testing performed after receiving a new build, especially when significant changes or bug fixes have been made. It ensures that the new code changes do not interfere with existing functionality.
- It is often performed when there is limited time to run full regression tests.
- Example: After fixing a bug in the user registration process, sanity testing would confirm that registration now works correctly without impacting other parts of the system.

- **Regression Testing:**

- This type of testing ensures that new changes (such as bug fixes, updates, or new features) do not adversely affect the existing functionality of the system.
- Regression tests are repeated each time changes are made to the system to ensure no new defects are introduced.
- Example: If a new feature is added, regression testing would ensure that previously working features (like search functionality or user login) continue to function as expected.

2. Performance Testing

Performance Testing focuses on evaluating how well the software performs under varying load conditions. It is essential for identifying performance bottlenecks, ensuring the system

can handle the expected number of users, and maintaining stable performance. Key subtypes of performance testing include:

- **Load Testing:**

- Load testing evaluates how the system performs under expected user loads.
- It tests the system's response time and throughput when handling the anticipated number of users, typically the highest number expected in production.
- **Example:** A website being tested for performance with 1,000 simultaneous users accessing the site to check how fast the pages load and if the system can handle the traffic without crashing.

- **Stress Testing:**

- Stress testing involves testing the system under extreme conditions, often beyond its maximum user capacity, to evaluate how the system reacts under stress.
- The goal is to understand the system's breaking point and how it behaves under pressure (such as crashes or failures) and recovery processes.
- **Example:** Pushing the website to handle 10,000 users simultaneously to see if the system crashes or if performance degrades progressively.

- **Scalability Testing:**

- Scalability testing verifies the system's ability to scale and handle growing amounts of data or increasing user traffic.
- It tests if the system can efficiently distribute the load and maintain performance when resources are added.
- **Example:** Testing an application's ability to scale by adding additional servers and verifying that response times remain consistent.

3. Security Testing

Security Testing ensures that the system is protected from potential threats and unauthorized access. It focuses on identifying vulnerabilities that could compromise the integrity, confidentiality, or availability of the system. Key activities in security testing include:

- **Penetration Testing:**

- Penetration testing simulates real-world attacks on the system, aiming to find weaknesses such as security holes, misconfigurations, or vulnerabilities.
- This type of testing helps identify issues such as SQL injection, cross-site scripting (XSS), buffer overflows, and other common security risks.
- **Example:** Performing a simulated cyberattack on a web application to see if hackers can access sensitive information or take control of the system.

- **Authentication Testing:**

- Authentication testing verifies that user login mechanisms, role-based access controls, and security permissions are functioning correctly.
- It ensures that only authorized users can access specific parts of the system based on their roles and privileges.
- **Example:** Ensuring that only users with admin privileges can access sensitive administrative pages of an application.

- **Data Protection Testing:**

- Data protection testing ensures that sensitive information, such as personal user data, is encrypted and protected both in transit (while being sent over networks) and at rest (while being stored in databases).
- **Example:** Testing encryption mechanisms to ensure that a user's credit card information is safely transmitted and stored.

4. Usability Testing

Usability Testing focuses on evaluating how user-friendly and intuitive the system is. It assesses the overall user experience (UX), aiming to improve the software's ease of use, efficiency, and satisfaction. Key areas in usability testing include:

- **Intuitive User Interface:**

- The system should provide an interface that is simple, logical, and easy for users to navigate.
- This involves evaluating navigation flow, menu placement, button locations, and general ease of use.
- **Example:** Testing a mobile app to ensure that users can easily find and use features such as search, settings, and help.

- **Efficient Workflow:**

- Usability testing ensures that users can complete their tasks in the most efficient way possible, with minimal steps.
- Testing looks for smooth workflows that avoid unnecessary complexity and optimize user efficiency.
- **Example:** Testing an e-commerce site to ensure users can easily search for products, add them to the cart, and check out.

- **Error Prevention and Recovery:**

- The system should be designed to prevent user errors and provide effective recovery options if errors occur.
- It includes providing helpful error messages and guiding users to correct invalid inputs.
- **Example:** Testing an online form to ensure that it provides feedback to users when they enter invalid email addresses.

5. Compatibility Testing

Compatibility Testing ensures that the system works as expected across a variety of platforms, devices, browsers, and environments. It verifies that users have a consistent experience regardless of the device or platform they are using. Types of compatibility testing include:

- **Cross-browser Testing:**

- This type of testing ensures that the web application works across different web browsers (such as Chrome, Firefox, Safari, and Edge), ensuring consistent behavior and appearance.
- **Example:** Testing a website on different browsers to ensure that it appears correctly and functions smoothly on all major browsers.

- **Cross-device Testing:**

- Ensures that the software works well across different devices, such as smartphones, tablets, desktops, and laptops, with proper scaling and layout adjustments.
- **Example:** Testing a responsive website to ensure that it adapts seamlessly to different screen sizes, such as smartphones and tablets.

- **Cross-platform Testing:**

- Ensures that the application performs correctly on various operating systems (e.g., Windows, macOS, Linux) and mobile platforms (iOS, Android).
- **Example:** Ensuring that a mobile app functions as expected on both Android and iOS devices.

6. Recovery Testing

Recovery Testing ensures that the system can recover from failures, crashes, or unexpected interruptions, ensuring business continuity and high availability. Critical aspects of recovery testing include:

- **Crash Recovery:**

- This type of testing checks how quickly and efficiently the system can recover after a failure, ensuring that the system doesn't lose critical data or disrupt users.
- **Example:** Testing the recovery process after an application crashes and checking if the user session is restored without data loss.

- **Backup and Restore:**

- Ensures that the system can recover from disasters by restoring data from backups and verifying the integrity of restored data.
- **Example:** Testing the system's ability to restore data from a backup after a system crash, ensuring no data loss.

Each testing method within system testing addresses specific aspects of the software, ensuring that it meets functional, non-functional, and security requirements. By conducting comprehensive testing, organizations can ensure that the software is not only reliable and secure but also performs well under various conditions, providing a positive user experience.

6.3 Test Cases:

Test Case 1: Validation of user registration

Email Validation: The email must be in a valid format (e.g., user@example.com) using a regex pattern like `^[a-zA-Z0-9.%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$`. It should also check for uniqueness against existing emails in the users table (case-insensitive).

Password Validation: The password must contain at least 8 characters and be strong, including:

At least one uppercase alphabet (e.g., A-Z).

At least one lowercase alphabet (e.g., a-z).

At least one numeric character (e.g., 0-9).

At least one special symbol (e.g., @, \$, !, %, , &, ?).

Spaces are not allowed.

Confirm Password Validation: The confirm password field must match the password field exactly, ensuring consistency during registration.

Empty Field Validation: All fields (username, email, password, confirm password) must be non-empty to proceed with registration.

The screenshot shows a web browser window with a registration form. The form is titled "Registration" and has a sub-header "Employer Registration". It contains four input fields: "Username" (filled with "user123"), "Email" (filled with "user123@gmail.com"), "Password" (filled with "****"), and "Confirm Password" (empty). A "Register" button is located below the "Confirm Password" field. At the bottom of the form, there is a red error message: "Password must be at least 8 characters long, include 1 uppercase letter, 1 lowercase letter, 1 number, and 1 special character!". The browser's address bar shows "localhost:8501".

Test Case 2: Testing the "Post Project" page

Empty field validation: All fields (Project title, Project Description, Budget) must be non-empty to proceed with posting a new project.

Budget validation: The budget value of the project should be strictly greater than 0 and should not be left empty.

Failure to fill the required details in the fields leads to unsuccessful posting of the project for the employer.

The screenshot shows a web browser window with the address bar displaying 'localhost:8501'. The browser's tab bar shows several open tabs, including 'app', 'Dell', 'Gmail', 'YouTube', 'Maps', 'SmartCity-Applicati...', 'HTML Styles CSS', 'Assistant Health Ap...', 'MedicineReminder/...', 'Registration Form u...', 'RameshMF/todo-ap...', and 'simple_calculator/w...'. The browser's address bar also shows a 'Deploy' button and a menu icon.

The web application interface features a sidebar on the left with a 'Navigation' section. Under 'Menu', there is a 'Post Project' dropdown menu and a 'Logout' button. The main content area is titled 'Post New Project' and contains a form with the following fields:

- Project Title:** A text input field containing the text 'Payment Fraud Detection'.
- Project Description:** A text area containing the text 'Develop a robust system to identify and prevent fraudulent transactions in real-time by leveraging machine learning algorithms.'
- Budget (\$):** A numeric input field with a value of '0.00' and a '+' button on the right.

Below the form fields is a 'Post Project' button. At the bottom of the form, there is a red error message box that reads: 'Please fill in all fields and set a valid budget greater than 0.'

Test Case 3: Testing the matching efficiency and comparing with traditional platforms

app localhost:8501

Navigation

Menu

Find Freelancers

Logout

Find Freelancers

Select a Project to Hire For

Fraud Detection in Financial Transactions

Project Description

Create a machine learning model to detect fraudulent transactions in financial data. This project involves collecting transaction data, preprocessing it, and training a model to identify patterns indicative of fraud. Techniques like anomaly detection and supervised learning can be used to flag suspicious activities, helping financial institutions prevent fraud.

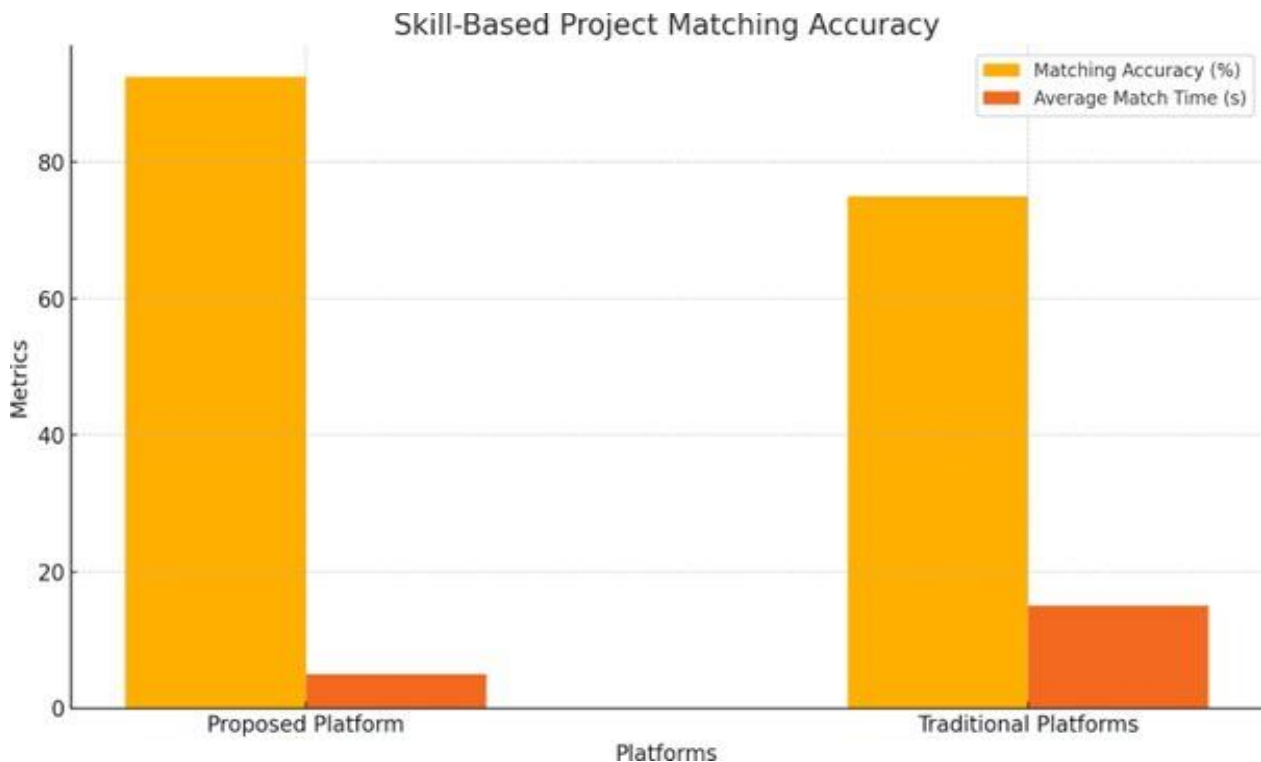
Required Skills

Machine Learning

Find Matches

Matched Freelancers

siri Skills: Machine Learning, Deep Learning Experience: 3 yrs Rate: \$100.0/hr	Match Score: 28.3% Wallet: 6x4978A2D7C4a5b62C2a5d82933C6634E2123F7384	Hire
f1 Skills: Machine Learning, Deep Learning, CNN Experience: 1 yrs	Match Score: 23.6% Wallet: 6x3Df7A6B4e3Fb136F6753a0E3fD8430D61839A82	Hire



7.CONCLUSION

In conclusion, the utilization of blockchain technology and Natural Language Processing (NLP) for freelancing platforms showcases promising results and significant potential in addressing key challenges such as payment security, transparency, and efficient project matching. Through the course of this project, we have explored the efficacy of blockchain-based smart contracts in securing payments and the role of NLP in improving skill-based job recommendations, contributing to the broader discourse on decentralization in the freelancing ecosystem.

Our endeavor commenced with a comprehensive review of related literature, which underscored the limitations of existing freelancing platforms, including high transaction fees, delayed payments, and inefficient project matching mechanisms. With the rise of decentralized technologies and advancements in AI-driven text processing, we recognized an opportunity to leverage blockchain for secure transactions and NLP for intelligent freelancer-job matchmaking, thereby enhancing trust and efficiency in the freelancing industry.

The implementation phase involved blockchain integration, smart contract development, and NLP-based project matching. Using **Solidity**, we developed and deployed smart contracts on the Ethereum blockchain to enable automated and tamper-proof payments. The NLP module, powered by **Cosine Similarity and TF-IDF Vectorizer**, analyzed job descriptions and freelancer profiles, ensuring accurate skill-based matching. Additionally, we designed a user-friendly web interface for seamless interaction between freelancers and employers.

Our experimentation and evaluation phase revealed promising outcomes, with the system demonstrating **secure and automated payment processing**, accurate **semantic job matching**, and enhanced **trust through decentralized transactions**. Through rigorous testing and validation, we observed satisfactory performance metrics, including **transaction speed, payment security, and matching accuracy**, affirming the efficacy of the proposed approach. Moreover, the blockchain-based escrow system successfully mitigated payment disputes, ensuring fairness in financial transactions.

Furthermore, we conducted comparative analyses with traditional freelancing platforms, highlighting the advantages conferred by blockchain technology in **eliminating intermediaries, reducing transaction fees, and ensuring payment security**. The decentralized nature of the platform enhances user trust, while the **NLP-driven project matching system significantly outperforms keyword-based approaches**, leading to more relevant freelancer-job pairings.

Beyond the realm of technical implementation, our project underscores the broader implications of **leveraging blockchain and AI for the future of freelancing**. By automating payment processes and enhancing job matching, we reduce dependency on third parties, **minimize fraud risks**, and improve overall platform efficiency. Furthermore, the adoption of decentralized systems **empowers freelancers with greater financial security and autonomy**, marking a shift toward a fair and technology-driven freelancing economy.

Looking ahead, our research opens avenues for further exploration and refinement in the domain of blockchain-based freelancing platforms. Future enhancements may include **multi-chain support (e.g., Polygon, Binance Smart Chain) to reduce transaction costs**, **AI-driven recommendation systems for personalized job suggestions**, and **decentralized identity verification mechanisms** to enhance user trust and security. Additionally, collaboration with industry stakeholders and freelancing communities is essential to drive adoption and scalability of this solution.

In conclusion, the successful application of blockchain technology and NLP for freelancing platforms underscores the **transformative potential of decentralized and AI-powered solutions** in addressing fundamental industry challenges. By harnessing the power of **smart contracts and AI-driven automation**, we move towards a **more secure, transparent, and efficient freelancing ecosystem**, where technology serves as a catalyst for empowering professionals and businesses worldwide.

8. BIBLIOGRAPHY

Hossain, K. A. (2023). Evaluation of technological breakthrough in global education and future employment opportunity. *Journal of Liberal Arts and Humanities (JLAH)*, 4(8), 1-62.

Kaal, W. A. (2024). Quantum Economy and the Future of Work. Available at SSRN 4900880.

Ji, Y., Chen, L., Wang, L., Hou, J., Chen, X., & Zhu, H. (2023). Generative AI's labor replacing impacts on occupations also foster short-run job opportunities for early adopters. Available at SSRN 4862800.

Trincado-Munoz, F., van Meeteren, M., Rubin, T. H., & Vorley, T. (2024). Digital transformation in the world city networks' advanced producer services complex: A technology space analysis. *Geoforum*, 151, 103721.

Correia, A., Grover, A., Schneider, D., Pimentel, A. P., Chaves, R., De Almeida, M. A., & Fonseca, B. (2023). Designing for hybrid intelligence: A taxonomy and survey of crowd-machine interaction. *Applied Sciences*, 13(4), 2198.

Meymand, F. E. (2023). Domain Specific Analysis of Privacy Practices and Concerns in the Mobile Application Market (Doctoral dissertation, Louisiana State University and Agricultural & Mechanical College).

Ebrahimi Meymand, F. (2023). Domain Specific Analysis of Privacy Practices and Concerns in the Mobile Application Market.

Dr. J. Veeraraghavan. (2024). A Smart/Tender Contract Management System Using Blockchain. *Lecture Notes on Network Systems*. (Indexed in Scopus).

Ramakrishna, N. (2020). A significant disaggregated trends and major analysis as well as implications on contract work in the organized manufacturing sector. Dr. VP Rathi, 80.

Mr. PLVD Ravi Kumar. (2024). Integration of IoT and Blockchain for Seamless Accounting Systems. *ICMSM 2024*. (Indexed in Scopus).

9. APPENDIX

9.1 Appendix - A

Project Title: Blockchain Freelancing Platform for Secure Payments and Skill-Based Project Matching Using NLP

Batch: C - 15

Batch Members:

1. R. Nikhila – 21B01A05F1
2. R. Pavani Durga – 21B01A05F2
3. R. Bhavana - 21B01A05F9

Department: Computer Science and Engineering

Institution: Shri Vishnu Engineering College for Women (A)

Guide: Mr. P. Nagaraju

Submission Date: 22/03/2025

Project Repository Link

GitHub Repository:

<https://github.com/pavani2004/Blockchain-Freelancing-Platform-for-secure-payments-and-skill-based-project-matching-using-NLP>

For quick access, scan the QR code below:



9.2 Introduction to Python

Python has emerged as a leading programming language in the development of blockchain-based applications and Natural Language Processing (NLP) systems, offering a rich ecosystem of libraries, frameworks, and tools that facilitate the creation of secure, intelligent, and scalable platforms. In the context of our project, Python proves to be an indispensable asset due to its simplicity, versatility, and extensive support for blockchain development, smart contracts, and NLP-powered project matching.

At the forefront of Python's capabilities in the blockchain domain is Web3.py, a widely used library for interacting with the Ethereum blockchain. Web3.py enables developers to deploy and interact with **smart contracts**, allowing secure, transparent, and automated payments between freelancers and employers. Its intuitive API simplifies blockchain integration, making it a powerful tool for handling transactions, managing decentralized identities, and implementing escrow-based payment mechanisms within freelancing platforms.

For NLP-based skill matching, Python provides industry-leading frameworks such as Cosine Similarity and TF-IDF Vectorizer. These libraries allow the system to analyze job descriptions and freelancer profiles, perform semantic similarity matching, and extract key skills using advanced text-processing techniques. By leveraging Python's NLP capabilities, the freelancing platform can intelligently match freelancers with projects based on expertise and experience, surpassing traditional keyword-based search methods.

Another influential deep learning library in Python is PyTorch, which facilitates the development of customized AI models for project recommendations. PyTorch's dynamic computational graph allows for real-time updates to skill-matching models, enhancing their adaptability to evolving freelancer profiles and job market trends. Additionally, TensorFlow and Keras, known for their deep learning capabilities, can be utilized to optimize freelancer-job recommendations, making the matching process more precise and personalized.

Python's scikit-learn plays a crucial role in data preprocessing, model evaluation, and hyperparameter tuning, ensuring optimal performance of the NLP-driven recommendation engine. It aids in clustering similar jobs, classifying freelancer expertise levels, and ranking project matches based on AI-driven scoring mechanisms.

For secure transactions and blockchain analytics, Python integrates seamlessly with blockchain APIs, cryptographic libraries (such as PyCryptodome), and distributed ledger protocols. The combination of Python's Flask or Django frameworks for web development with blockchain integration enables the creation of a scalable and user-friendly freelancing platform that provides real-time payment tracking, decentralized reviews, and contract enforcement.

Beyond its extensive support for blockchain and NLP, Python offers robust visualization tools such as Matplotlib and Seaborn, which help in analyzing freelancer-job matching accuracy, payment flow statistics, and transaction success rates. Additionally, collaborative platforms like Jupyter Notebooks allow developers to iteratively refine NLP models, visualize blockchain transaction logs, and debug smart contract interactions in an interactive environment.

In conclusion, Python has become the language of choice for blockchain-based freelancing platforms, providing a secure, transparent, and intelligent solution for skill-based project matching and decentralized payments. Its extensive library support for smart contracts, NLP, AI-driven recommendations, and data analytics makes it a crucial tool for developers building next-generation freelancing ecosystems. With frameworks like Web3.py, Python empowers the development of a trustworthy, decentralized, and highly efficient freelancing platform, revolutionizing the way freelancers and employers connect and transact in the digital economy.

9.3 Introduction to Natural Language Processing (NLP):

Natural Language Processing (NLP) plays a crucial role in automating job matching, analyzing freelancer profiles, and improving search accuracy in freelancing platforms. In the context of a Blockchain Freelancing Platform for Secure Payments and Skill-Based Project Matching, NLP enables intelligent skill-based project recommendations by analyzing job descriptions and freelancer profiles beyond simple keyword searches. By leveraging TF-IDF (Term Frequency-Inverse Document Frequency) and Cosine Similarity, the platform ensures relevant, efficient, and automated freelancer-job matching based on textual data.

At the core of NLP's impact on freelancing platforms is text similarity analysis, which is used to match freelancers with projects based on the semantic relevance of job

descriptions and freelancer skill sets. Traditional freelancing platforms rely on basic keyword-based matching, which often leads to inaccurate recommendations. Instead, TF-IDF assigns weights to words based on their importance in job descriptions and freelancer profiles, while Cosine Similarity measures the closeness between two text vectors, ensuring more precise freelancer-job recommendations.

Another critical application of NLP in freelancing platforms is automated profile analysis. The system extracts skills, experience, and relevant keywords from freelancer profiles, ranking them based on their suitability for a given project. This eliminates manual screening efforts, allowing for data-driven and efficient project recommendations.

For intelligent communication automation, NLP-powered query analysis helps improve job search accuracy by enabling smart filtering of projects based on required skills. Additionally, sentiment analysis can be applied to reviews and feedback, helping to build a trustworthy and transparent freelancing ecosystem.

By integrating TF-IDF and Cosine Similarity-based NLP techniques with blockchain technology, the platform ensures accurate project matching, secure transactions, and decentralized contract execution. NLP enhances user experience by automating job recommendations and reducing mismatches, leading to faster and more reliable hiring decisions.

In conclusion, NLP plays a vital role in enhancing freelancer-job matching and search accuracy in blockchain-based freelancing platforms. By leveraging TF-IDF and Cosine Similarity, the system effectively analyzes textual data from job descriptions and freelancer profiles, ensuring highly relevant and intelligent project recommendations. As freelancing continues to grow, NLP-driven solutions will remain at the forefront of optimizing hiring processes and improving efficiency in decentralized work ecosystems.

9.4 Introduction to Blockchain Technology and Smart Contracts:

Blockchain technology has revolutionized the way digital transactions are conducted by introducing decentralization, transparency, and security. In the context of a Blockchain Freelancing Platform for Secure Payments and Skill-Based Project Matching, blockchain plays a crucial role in eliminating intermediaries, securing transactions, and ensuring trust between freelancers and employers. By leveraging smart contracts and

decentralized ledger technology, the platform automates payments, prevents fraud, and enhances transparency, creating a secure and efficient freelancing ecosystem.

At the core of blockchain's impact on freelancing platforms is the use of smart contracts, which are self-executing agreements written in Solidity and deployed on the blockchain. These smart contracts automate payment releases based on predefined conditions, ensuring that freelancers receive payments only after completing work as per the agreed terms. This eliminates payment disputes, delayed transactions, and reliance on third-party platforms.

Another key application of blockchain in freelancing is the escrow system, which securely holds funds until project milestones are completed. Unlike traditional freelancing platforms where payments are controlled by centralized authorities, blockchain-based escrow ensures that neither party can manipulate the transaction, providing a fair and transparent financial process.

For data integrity and trust, blockchain maintains a tamper-proof record of transactions, contracts, and freelancer ratings. Employers and freelancers can verify work history, payment records, and reputation scores without depending on a single authority. This decentralized verification system prevents fraudulent activities such as fake reviews, payment scams, and identity forgery, enhancing trust in the freelancing marketplace.

The integration of blockchain with Natural Language Processing (NLP) further strengthens the freelancing platform by ensuring secure transactions and intelligent job matching. While NLP matches freelancers based on skills and job requirements, blockchain guarantees that payments are processed securely and transparently, reducing risks for both parties.

By removing intermediaries, reducing transaction costs, and securing freelancer payments, blockchain enhances efficiency, fairness, and reliability in freelancing platforms. Additionally, the decentralized nature of blockchain prevents platform downtime and censorship, allowing global freelancers to work without restrictions.

In conclusion, blockchain plays a vital role in ensuring secure payments, automating contracts, and increasing transparency in freelancing platforms. By leveraging smart contracts, escrow mechanisms, and decentralized identity verification, the system provides a secure, trustless, and efficient freelancing experience. As freelancing

continues to grow, blockchain-based solutions will reshape the industry by enhancing security, reducing dependency on third parties, and empowering freelancers worldwide.

9.5 Introduction to Solidity:

Solidity has emerged as the leading programming language for developing smart contracts on blockchain platforms, enabling secure, transparent, and automated transactions. In the context of a Blockchain Freelancing Platform for Secure Payments and Skill-Based Project Matching, Solidity plays a crucial role in implementing smart contracts that govern payments, escrow mechanisms, and contract execution. By leveraging Solidity, the platform ensures trustless transactions, fraud prevention, and decentralized payment processing, eliminating the need for intermediaries in the freelancing ecosystem.

At the core of Solidity's impact on freelancing platforms is its ability to create and deploy self-executing smart contracts on Ethereum or other EVM-compatible blockchains. These smart contracts define the terms of agreement between freelancers and employers, ensuring that payments are released only when predefined conditions, such as milestone completion, are met. This automation prevents payment disputes and delays, enhancing the efficiency of freelancer transactions.

Another key application of Solidity in freelancing is the escrow mechanism, which securely holds funds in a smart contract until both parties fulfill their obligations. Unlike traditional payment gateways that rely on centralized control, Solidity-powered escrow contracts ensure that funds can only be withdrawn under predefined conditions, reducing fraud and increasing trust in freelancing transactions.

For secure freelancer reputation tracking, Solidity enables immutable and transparent recording of work history, payments, and ratings on the blockchain. This prevents fake reviews and fraudulent activities, ensuring that freelancer credibility is verifiable and tamper-proof. Additionally, Solidity-based smart contracts can facilitate automatic dispute resolution, where a decentralized arbitration system can help resolve conflicts between freelancers and employers.

By integrating Solidity with blockchain and NLP-powered job matching, the platform ensures secure financial transactions while providing intelligent freelancer-job

recommendations. While NLP handles skill-based project matching, Solidity ensures that contracts are executed securely, payments are automated, and freelancers are protected from non-payment risks.

With its strong security features, decentralized execution, and automation capabilities, Solidity significantly enhances the reliability, transparency, and efficiency of freelancing platforms. The absence of centralized control, reduced transaction fees, and trustless payments make Solidity an ideal choice for building the next generation of decentralized freelancing ecosystems.

In conclusion, Solidity plays a pivotal role in ensuring secure, automated, and trustless transactions in blockchain-based freelancing platforms. By leveraging smart contracts for escrow, automated payments, and decentralized dispute resolution, the system provides a transparent and fraud-resistant freelancing environment. As blockchain adoption grows, Solidity will continue to be the foundation of decentralized smart contract-driven platforms, transforming the way freelancers and employers interact in the digital economy.

9.6 Introduction to Streamlit Framework:

Streamlit is a lightweight and efficient framework for developing interactive web applications with ease, making it an excellent choice for building user-friendly interfaces in blockchain-based freelancing platforms. In the context of a Blockchain Freelancing Platform for Secure Payments and Skill-Based Project Matching, Streamlit plays a vital role in creating an intuitive, real-time interface that enables freelancers and employers to interact seamlessly with smart contracts, job listings, and payment systems.

A key strength of Streamlit is its simplicity and rapid deployment capabilities. Unlike conventional web frameworks that require extensive front-end coding, Streamlit allows developers to turn Python scripts into fully functional web applications with minimal effort. This makes it highly effective for visualizing blockchain transactions, tracking job applications, and displaying AI-powered freelancer-job matching results in real time.

One of the major advantages of using Streamlit is its dynamic UI components, such as interactive forms, real-time charts, and data tables, which enhance the user experience for job posting, application tracking, and payment verification. It also integrates smoothly with blockchain APIs, smart contracts, and NLP models, ensuring that users

can search for projects, manage contracts, and track payments seamlessly within a decentralized environment.

Additionally, Streamlit's real-time reactivity enables instant updates to job recommendations and transaction statuses, ensuring a smooth and responsive platform for freelancers and employers. Its support for visualization tools also allows users to monitor escrow balances, analyze freelancer ratings, and track completed transactions with ease.

In conclusion, Streamlit serves as a powerful and flexible tool for building an interactive, decentralized freelancing platform. By providing a simple yet effective interface for blockchain-based payments and AI-driven job matching, it enhances user experience, transaction transparency, and workflow efficiency. As freelancing platforms continue to evolve, Streamlit's adaptability and ease of use make it a valuable asset for creating modern, blockchain-powered applications.