

FREELANCEHUB

Mini Project Report

Submitted by

VARSHA SHAJI

Reg. No.: AJC20MCA-I056

In Partial fulfillment for the Award of the Degree of

INTEGRATED MASTER OF COMPUTER APPLICATIONS

(INMCA)

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY



AMAL JYOTHI COLLEGE OF ENGINEERING AUTONOMOUS

KANJIRAPPALLY

[Affiliated to APJ Abdul Kalam Technological University, Kerala. Approved by AICTE,
Accredited by NAAC. Koovappally, Kanjirappally, Kottayam, Kerala – 686518]

2024-2025

DEPARTMENT OF COMPUTER APPLICATIONS
AMAL JYOTHI COLLEGE OF ENGINEERING AUTONOMOUS
KANJIRAPPALLY



CERTIFICATE

This is to certify that the Mini Project report, “**FREELANCEHUB**” is the bonafide work of **VARSHA SHAJI (Regno: AJC20MCA-I056)** in partial fulfillment of the requirements for the award of the Degree of Integrated Master of Computer Applications under APJ Abdul Kalam Technological University during the year 2024-25.

Ms.Anit James

Internal Guide

Mr.Binumon Joseph

Coordinator

Rev. Fr. Dr. Rubin Thottupurathu Jose

Head of the Department

DECLARATION

I hereby declare that the mini project report “**FREELANCEHUB**” is a bona fide work done at Amal Jyothi College of Engineering, towards the partial fulfilment of the requirements for the award of the **Integrated Master of Computer Applications (INMCA)** from **APJ Abdul Kalam Technological University**, during the academic year **2024-2025**.

Date: 07/11/2024

KANJIRAPPALLY

VARSHA SHAJI

Reg: AJC20MCA-I056

ACKNOWLEDGEMENT

First and foremost, I thank God almighty for his eternal love and protection throughout the project. I take this opportunity to express my gratitude to all who helped me in completing this project successfully. It has been said that gratitude is the memory of the heart. I wish to express my sincere gratitude to our Director (Administration) **Rev.Fr.Dr. Roy Abraham Pazhayaparampil** and Principal **Dr. Lillykutty Jacob** for providing good faculty for guidance.

I owe a great depth of gratitude towards our Head of the Department **Rev. Fr. Dr. Rubin Thottupurathu Jose** for helping us. I extend my whole hearted thanks to the project coordinator **Mr. Binumon Joseph, Assistant professor** for his valuable suggestions and for overwhelming concern and guidance from the beginning to the end of the project. I would also express sincere gratitude to my guide **Ms. Anit James, Assistant professor** for her inspiration and helping hand.

I thank our beloved teachers for their cooperation and suggestions that helped me throughout the project. I express my thanks to all my friends and classmates for their interest, dedication, and encouragement shown towards the project. I convey my hearty thanks to my family for the moral support, suggestions, and encouragement to make this venture a success.

VARSHA SHAJI

ABSTRACT

Freelance Hub is an Online Freelance Job Management System designed to enhance collaboration between clients and freelancers by streamlining project posting, searching, and project management. The platform offers a variety of essential functionalities, including efficient job posting and proposal management, an integrated chat system for real-time communication, and an AI-powered chatbot to assist users with common queries.

For premium users, Freelance Hub provides advanced features such as enhanced visibility for job postings to attract qualified freelancers, certification badges that boost credibility, and customized job alerts for timely notifications of new postings. The platform also includes robust administrative tools for managing user accounts and resolving disputes effectively. Additionally, secure payment processing is facilitated through an escrow system, ensuring transparency in financial transactions. A feedback and rating system promotes accountability, allowing clients and freelancers to evaluate each other's performance, thereby fostering a trustworthy ecosystem for successful collaborations.

In managing projects, Freelance Hub uses Scrum project management principles to enhance collaboration and client satisfaction. Clients initiate projects by posting detailed descriptions, which freelancers respond to with proposals. Once a freelancer is selected, clients add specific tasks for the freelancer to complete. Freelancers provide progress updates during Daily Scrums, and clients review the completed tasks. After reviewing, clients can mark tasks as completed. This process ensures that projects meet client expectations and lead to successful outcomes.

CONTENT

SL. NO	TOPIC	PAGE NO
1	INTRODUCTION	1
1.1	PROJECT OVERVIEW	2
1.2	PROJECT SPECIFICATION	2
2	SYSTEM STUDY	5
2.1	INTRODUCTION	6
2.2	LITERATURE REVIEW	6
2.3	PROPOSED SYSTEM	9
2.4	ADVANTAGES OF PROPOSED SYSTEM	9
3	REQUIREMENT ANALYSIS	10
3.1	FEASIBILITY STUDY	11
3.1.1	ECONOMICAL FEASIBILITY	11
3.1.2	TECHNICAL FEASIBILITY	11
3.1.3	BEHAVIORAL FEASIBILITY	12
3.1.4	FEASIBILITY STUDY QUESTIONNAIRE	12
3.2	SYSTEM SPECIFICATION	13
3.2.1	HARDWARE SPECIFICATION	13
3.2.2	SOFTWARE SPECIFICATION	13
3.3	SOFTWARE DESCRIPTION	14
3.3.1	DJANGO	14
3.3.2	MARIADB	14
4	SYSTEM DESIGN	15
4.1	INTRODUCTION	16
4.2	UML DIAGRAM	16
4.2.1	USE CASE DIAGRAM	16
4.2.2	SEQUENCE DIAGRAM	17
4.2.3	ACTIVITY DIAGRAM	19
4.2.4	CLASS DIAGRAM	21
4.2.5	OBJECT DIAGRAM	22
4.3	USER INTERFACE DESIGN USING FIGMA	24
4.4	DATABASE DESIGN	25
4.4.1	RDBMS	25

4.4.2	NORMALIZATION	25
4.4.3	SANITIZATION	29
4.4.4	INDEXING	29
4.5	TABLE DESIGN	30
5	SYSTEM TESTING	44
5.1	INTRODUCTION	45
5.2	TEST PLAN	45
5.2.1	UNIT TESTING	45
5.2.2	INTEGRATION TESTING	45
5.2.3	VALIDATION TESTING	46
5.2.4	USER ACCEPTANCE TESTING	46
5.2.5	AUTOMATION TESTING	46
5.2.6	SELENIUM TESTING	46
6	IMPLEMENTATION	63
6.1	INTRODUCTION	64
6.2	IMPLEMENTATION PROCEDURE	64
6.2.1	USER TRAINING	65
6.2.2	TRAINING ON APPLICATION SOFTWARE	65
6.2.3	SYSTEM MAINTENANCE	66
6.2.4	HOSTING	66
7	CONCLUSION & FUTURE SCOPE	69
7.1	CONCLUSION	70
7.2	FUTURE SCOPE	70
8	BIBLIOGRAPHY	71
9	APPENDIX	74
9.1	SAMPLE CODE	75
9.2	SCREEN SHOTS	81
9.2	GIT LOG	84

List of Abbreviations

IDE	-	Integrated Development Environment
HTML	-	Hyper Text Markup Language
CSS	-	Cascading Style Sheet
UML	-	Unified Modeling Language
MVT	-	Model-View-Template
NLP	-	Natural Language Processing
MariaDB	-	Maria Database
JS	-	JavaScript
AI	-	Artificial Intelligence
API	-	Application Programming Interface
SQL	-	Structured Query Language
UI	-	User Interface
UX	-	User Experience
CRUD	-	Create, Read, Update, Delete
JWT	-	JSON Web Token
SEO	-	Search Engine Optimization
SSL	-	Secure Sockets Layer
CDN	-	Content Delivery Network
AJAX	-	Asynchronous JavaScript and XML
DB	-	Database
ORM	-	Object-Relational Mapping
CSRF	-	Cross-Site Request Forgery
WSGI	-	Web Server Gateway Interface
ASGI	-	Asynchronous Server Gateway Interface
CBV	-	Class-Based View
FBV	-	Function-Based View

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

Freelance Hub is a web-based platform designed to streamline collaboration between clients and freelancers. It simplifies job posting, proposal management, and project tracking, enabling clients to post jobs, assign tasks, and review completed work. Freelancers can search for relevant projects, submit proposals, and complete tasks efficiently. The platform includes key features such as real-time chat for seamless communication, an AI-powered chatbot to handle common queries, payments, and a rating system that fosters trust and transparency between users.

The platform also adopts Scrum project management principles to ensure continuous progress tracking and client feedback. Clients and freelancers work together through iterative cycles, with regular updates and evaluations to ensure projects meet expectations. Future developments include a mobile app for enhanced accessibility and premium features like increased visibility for freelancers. Additionally, there is a focus on research to improve job recommendations using machine learning, further enhancing the platform's efficiency and user satisfaction.

1.2 PROJECT SPECIFICATION

Freelance Hub is an online platform that connects clients and freelancers for seamless collaboration, allowing clients to post detailed job listings and freelancers to submit tailored proposals. The platform includes an integrated chat system for communication, an escrow payment system for secure transactions, and features like task management, real-time notifications, and feedback/rating systems to ensure transparency and trust. It follows a Scrum-based project management approach, where clients assign tasks to freelancers, monitor progress, and review completed work. Built using Django with a MariaDB database, the platform's frontend is developed with HTML, CSS, and JavaScript, with plans for a future mobile app using Flutter.

Users in the Project

In Freelance Hub, there are three primary users:

- **Client:** Individuals or organizations posting job opportunities and managing projects.
- **Freelancer:** Independent professionals delivering project outcomes.
- **Administrator:** Oversees operations, manages interactions, and ensures integrity.

Modules

Client Features

- **Profile Management:** Clients can manage their profile details, including name,

email, phone number, and project history.

- **Project Posting:** Clients create comprehensive job descriptions specifying projectscope, requirements, deadlines, and budget.
- **Proposal Management:** Clients review and manage proposals submitted byfreelancers.
- **Rating and Review:** Clients provide feedback on freelancer performance upon project completion.
- **Complaint Reporting:** Clients report issues or concerns about freelancer conduct.
- **Progress Tracking:** Clients monitor project progress through updates fromfreelancers.
- **Task Management:** Clients define and manage project tasks or user stories.
- **Payment:** Clients process payments to freelancers securely through the platform.
- **Scrum Meeting Reminders:** Clients can initiate and receive reminders for Scrum meetings with freelancers.
- **Notifications:** Clients receive real-time notifications about project updates and communications.
- **Chat System:** Integrated chat facilitates direct communication with freelancers.
- **Password Reset:** Clients can reset their passwords securely.
- **Premium Upgrade:** Clients can access enhanced features through a premium subscription.
- **Logout:** Clients can securely log out of their accounts.

Freelancer Features

- **Profile Management:** Freelancers manage their profiles, including skills, experience, and portfolio.
- **Project Search and Proposal Submission:** Freelancers search for projects and submit tailored proposals.
- **To-Do List:** Freelancers track project tasks, deadlines, and priorities.
- **Daily Progress Updates:** Freelancers provide daily updates to clients about project status.
- **Rating and Review Clients:** Freelancers rate and review clients based on interactions.
- **Complaint Reporting:** Freelancers report issues or concerns about clients.
- **Notifications:** Freelancers receive real-time notifications about projects and communications.
- **Chat System:** Integrated chat facilitates direct communication with clients.
- **Password Reset:** Freelancers can reset their passwords securely.
- **Premium Upgrade:** Freelancers can access enhanced features through a

premiumsubscription.

- **Logout:** Freelancers can securely log out of their accounts.

Administrator Features

- **User Management:** Administrators manage user accounts, including activation and permissions.
- **Complaint Handling:** Administrators review and resolve client and freelancer complaints.
- **Analytics Dashboard:** Administrators access detailed analytics for platform performanceand user activity.

CHAPTER 2

SYSTEM STUDY

2.1 INTRODUCTION

The system study is a critical analysis of the proposed Freelance Hub, aimed at understanding its components, functionalities, and interactions. This process identifies potential challenges and opportunities while defining goals and objectives to enhance overall efficiency and effectiveness. The study will utilize various techniques, such as stakeholder interviews, surveys, and data analysis, to gather comprehensive insights into user needs and system performance.

The primary purpose of this study is to assess the platform's ability to meet user requirements, streamline job management, and improve communication between clients and freelancers. The introduction will outline the study's objectives and provide an overview of the Freelance Hub as an online platform for freelance job management. Additionally, it will define the scope and limitations of the study, alongside the methods and tools employed for data collection. This framework sets the stage for a thorough evaluation of the system's potential and performance.

2.2 LITERATURE REVIEW

Current freelance job platforms face significant challenges, such as high platform fees, limited real-time collaboration tools, slow or biased dispute resolution, and lack of skill verification, which undermine trust and efficiency. These platforms often struggle with fragmented processes, inadequate communication channels, and insecure payment methods, leading to misunderstandings and inefficiencies between clients and freelancers. The absence of structured project tracking and integrated systems further exacerbates misaligned expectations, while insufficient search filters and unbalanced accountability systems hinder the overall user experience for both parties. These gaps highlight the need for a more transparent, secure, and efficient platform that addresses these issues to enhance trust and collaboration [8][1].

Freelance Hub addresses these critical challenges through a comprehensive, web-based platform that integrates essential features, including secure payment processing, seamless job posting, and direct communication channels. By fostering efficient interactions through a rating system and transparent project management, Freelance Hub enhances alignment between client expectations and freelancer capabilities. Real-time collaboration tools and advanced communication features, such as live chat and proposal submission, streamline the project management process, promoting clearer understanding between both parties. Additionally, the platform's integrated rating and review system introduces an extra layer of accountability, encouraging both clients and freelancers to uphold quality standards, thereby supporting a healthier and more trustworthy freelancing ecosystem [2].

One notable advancement in Freelance Hub is the implementation of automated resume parsing and dynamic portfolio generation. By utilizing Natural Language Processing (NLP) and PDF processing tools like spaCy and PyMuPDF, the platform efficiently extracts structured data from resumes, including skills, experience, and education, converting them into visually appealing, interactive portfolio websites. This automation reduces the need for manual data entry, minimizing errors while allowing freelancers to present a more nuanced view of their qualifications. The NLP-based resume parsing system captures key resume sections and enables Freelance Hub to display skills and achievements in an engaging and accessible format for clients. This innovative system supports more reliable talent matching and enhances collaboration, benefiting both freelancers seeking new opportunities and clients in search of skilled professionals [3].

Freelance Hub's solution is designed to overcome limitations found in existing platforms by offering a modern, comprehensive freelancing platform optimized for efficient collaboration. The platform provides competitive, lower fees, making freelancing more accessible and financially viable. It also includes built-in real-time collaboration tools, such as live document editing and task management features, allowing freelancers and clients to work closely on project tasks. Freelance Hub's fair and efficient dispute resolution process relies on transparent, unbiased mechanisms to resolve issues effectively, while its robust skill verification system, complete with tests and certifications, ensures the credibility of freelancer profiles [9]. Advanced search filters in Freelance Hub enable clients to locate freelancers with specific skills and relevant experience, simplifying the process of finding the right match for projects. Additionally, the platform's centralized project repository supports secure file sharing and storage, ensuring easy access to essential documents, while a transparent refund system enhances accountability and protects client investments. By maintaining a balanced accountability system, Freelance Hub fosters a professional and respectful freelancing environment for both freelancers and clients [10].

Natural Language Processing (NLP) Techniques

Natural Language Processing (NLP) has become foundational in many resume parsing systems because of its ability to process unstructured data, transform it into structured information, and categorize it effectively. NLP techniques include linguistic patterns and rule-based extraction methods that help classify information. Studies highlight the advantages of NLP libraries like spaCy, which are particularly useful for segmenting resumes into sections like "Experience" and "Skills" due to their efficiency and speed [4]. Common techniques in NLP-based systems include text segmentation to break up resumes into sections, Named Entity Recognition (NER) to identify key entities (such as names and organizations), part-of-speech tagging for analyzing grammatical

structure, and rule-based pattern matching to locate specific keywords associated with different resume sections. However, these NLP techniques can be limited when handling diverse resume formats, which calls for the inclusion of machine learning to improve accuracy and flexibility [5].

Machine Learning (ML) Models

Machine learning models add a layer of predictive power to resume parsing, making them suitable for structured data classification tasks like categorizing resumes by job type or skills. Supervised learning models, including Support Vector Machines (SVM) and Decision Trees, demonstrate high accuracy in processing structured resume data, with some studies achieving success rates of over 90% [6]. Additionally, deep learning models, such as Long Short-Term Memory (LSTM) networks, capture long-range dependencies in text, which enhances the ability to interpret complex text relationships within resumes. LSTM networks have shown to improve resume parsing accuracy by 5-10% compared to traditional machine learning models, providing a more nuanced understanding of resume content [7]. These machine learning methods are essential in processing complex datasets and diverse formats, which are common in freelance profiles and client job listings.

Hybrid NLP-ML Systems

To address the limitations of NLP and ML individually, many resume parsing systems now use a hybrid approach, combining the strengths of both NLP and ML techniques. Hybrid systems apply NLP techniques like Named Entity Recognition and rule-based matching for feature extraction, which allows them to handle varied and unstructured resume formats efficiently. The extracted features are then processed through ML algorithms for classification and prediction, enhancing accuracy and adaptability. This combined approach is particularly effective for complex and varied datasets, as it balances the structural understanding provided by NLP with the predictive power of ML [11]. Research has shown that hybrid NLP-ML systems can improve parsing accuracy by 10- 15% over traditional methods and are better suited for processing diverse resume formats. Such hybrid models are highly adaptable and offer better contextual understanding, making them ideal for platforms like Freelance Hub that manage a wide range of client and freelancer profiles [12]. FreelanceHub is an innovative freelancing platform designed to facilitate seamless collaboration between freelancers and clients. Offering lower fees, integrated real-time collaboration tools, and a robust skill verification system, FreelanceHub enhances project efficiency. With advanced search filters, a centralized project repository, and a transparent refund system, it ensures accountability and quality, making

freelancing accessible and reliable.

2.3 PROPOSED SYSTEM

FreelanceHub is an innovative freelancing platform designed to facilitate seamless collaboration between freelancers and clients. Offering lower fees, integrated real-time collaboration tools, and a robust skill verification system, FreelanceHub enhances project efficiency. With advanced search filters, a centralized project repository, and a transparent refund system, it ensures accountability and quality, making freelancing accessible and reliable.

2.4 ADVANTAGES OF PROPOSED SYSTEM

- Lower, more competitive fees for freelancers.
- Built-in real-time collaboration tools like live editing and task management.
- Fair and efficient dispute resolution process.
- Strong skill verification through tests and certifications.
- Advanced search filters for specific skills and experience.
- Balanced accountability system for both freelancers and clients.
- Centralized project repository for secure file sharing.
- Transparent refund system for clients in case of project cancellations or disputes.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 FEASIBILITY STUDY

Feasibility is defined as the practical extent to which a project can be performed successfully. To evaluate feasibility, a feasibility study is performed, which determines whether the solution considered to accomplish the requirements is practical and workable in the software [1]. Information such as resource availability, cost estimation for software development, benefits of the software to the organization after it is developed and cost to be incurred on its maintenance are considered during the feasibility study.

Various types of feasibility that are commonly considered include :

- Technical feasibility
- Operational feasibility
- Economic feasibility

3.1.1 Economical Feasibility

Economic feasibility determines whether the required software is capable of generating financial gains for an organization. It involves the cost incurred on the software development team, estimated cost of hardware and software, cost of performing feasibility study, and so on.

The Freelance Hub project is economically feasible as development costs are reasonable, leveraging open-source technologies and existing team skills. The platform promises long-term financial gains by improving job matching, communication, and secure payments. With minimal additional hardware costs and potential revenue from premium features, the investment is justified by expected benefits and user growth.

3.1.2 Technical Feasibility

Technical feasibility assesses the current resources (such as hardware and software) and technology, which are required to accomplish user requirements in the software within the allocated time and budget. The Freelance Hub will use established technologies to ensure technical feasibility. The backend will be developed with Django, handling job postings, proposal management, and user authentication. HTML, CSS, and JavaScript will be used for the frontend to create intuitive interfaces. Initially, SQLite will be the database, with scalability to PostgreSQL as needed. These technologies are chosen for their stability, widespread use, and the team's proficiency, ensuring effective development and maintenance within the project's resources and timeline.

3.1.3 Behavioral Feasibility

Operational feasibility assesses the extent to which the required software performs a series of steps to solve business problems and user requirements. This feasibility is dependent on human resources and involves visualizing whether the software will operate after it is developed and be operative once it is installed.

Operational feasibility for the Freelance Hub focuses on enhancing efficiency and user experience in freelance job management. The platform will streamline job posting, proposal submission, and communication through integrated chat functionalities. By addressing inefficiencies like fragmented communication and cumbersome administrative tasks, it aims to improve user satisfaction and operational efficiency. Stakeholders support these improvements, recognizing the platform's potential to optimize workflow and foster better collaboration between clients and freelancers.

3.1.4 Feasibility Study Questionnaire

1. What types of freelance jobs are most popular among clients right now?

The most popular freelance jobs include sales, billing, accounting, web development, graphic design, content writing, and digital marketing.

2. How do clients typically decide which freelancer to hire for their projects?

Clients typically decide based on the freelancer's resume, relevant experience, portfolio, reviews and ratings from previous clients, and their proposal.

3. What do freelancers usually look for when searching for freelance job opportunities?

Freelancers look for jobs that match their skills and experience, offer competitive compensation, and provide clear project descriptions. They also value clients who have a history of timely payments and positive feedback from other freelancers.

4. How do freelancers prefer to communicate with clients during projects?

Freelancers prefer to use integrated real-time chat systems within the platform for quick and efficient communication. They also use email and video calls for more detailed discussions and project updates.

5. Can you describe how payments between clients and freelancers are usually

handled? Payments are typically handled through secure payment gateways like Stripe or PayPal. Clients often pay a portion upfront and the remainder upon project

completion.

6. In your experience, what features or tools do freelancers find most helpful on freelance job platforms?

Freelancers find features like detailed job postings, efficient proposal management, real-time chat systems, secure payment processing, and a rating and review system most helpful.

7. How important are client reviews and freelancer ratings in the freelance job market?

Reviews and ratings are extremely important as they build trust and credibility. High ratings can significantly increase a freelancer's chances of being hired, while clients rely on reviews to gauge the reliability and quality of freelancers.

8. What essential features should a freelance job platform have?

Essential features include robust search functionality, comprehensive profile management, secure payment gateways, a rating and review system, real-time communication tools, etc.

9. How do clients and freelancers usually resolve disputes or issues?

Disputes are typically resolved through the platform's resolution center, where both parties can present their case. Administrators or mediators then review the information and make a decision.

10. How do clients track the progress of their freelance projects?

Clients track progress through the feedback and updates provided by the feedback. Clients track progress through the feedback and updates provided by the feedback.

3.2 SYSTEM SPECIFICATION

3.2.1 Hardware Specification

Processor	- Intel Core i5 or higher
RAM	- 8GB or higher
Hard disk	- 500GB solid-state drive (SSD) or higher

3.2.2 Software Specification

Front End -	HTML, CSS, JavaScript, Bootstrap, jQuery, AJAX
Backend -	Django, MariaDB for database management
Client on PC	-Windows 10 and above.
Technologies used -	HTML, CSS, JavaScript, Django, MariaDB, AJAX, NLP

3.3 SOFTWARE DESCRIPTION

3.3.1 Django

Django is a high-level Python web framework that enables rapid development of secure and scalable web applications. It follows the Model-View-Template (MVT) architectural pattern and provides built-in features for authentication, database handling, and URL routing, making development efficient. Django promotes reusability, less code repetition, and emphasizes the principle of "Don't Repeat Yourself" (DRY). It's widely used for creating robust web applications and supports integration with various databases[11].

3.3.2 MariaDB

MariaDB is an open-source relational database management system (RDBMS) that originated as a fork of MySQL. It is known for its reliability, performance, and scalability, making it a popular choice for web applications. MariaDB supports SQL for database queries and offers additional features, improved security, and performance optimizations compared to MySQL. It is fully compatible with MySQL, allowing seamless migration between the two. MariaDB is commonly used with frameworks like Django to manage and store large volumes of structured data[12].

CHAPTER 4

SYSTEM DESIGN

4.1 INTRODUCTION

System design involves creating a detailed plan for a system's architecture, components, and data flow to meet specific requirements. It addresses technical aspects like performance and scalability, and non-technical ones such as usability and maintainability. The goal is to ensure the system operates efficiently and can handle future changes[13].

UML (Unified Modeling Language) diagrams are the most common tools used in system design. They include use case diagrams, class diagrams, sequence diagrams, and activity diagrams to visualize interactions, structure, workflows, and event flows. Other diagrams, like ER diagrams and data flow diagrams, further clarify system elements, helping developers and stakeholders communicate and understand the design effectively.

4.2 UML DIAGRAM

A Unified Modeling Language (UML) diagram is a standardized visual tool used to represent the structure, behavior, and interactions within a system or software application. It uses specific symbols and notations to depict elements such as objects, classes, and relationships. UML diagrams are essential for documenting, analyzing, and designing systems, enabling developers, architects, and business analysts to communicate designs clearly to stakeholders. They support various phases of the software development lifecycle, helping ensure that systems are designed to meet user requirements efficiently. UML can model anything from simple apps to large enterprise systems [15].

4.2.1 USE CASE DIAGRAM

A use case diagram visually represents the system's functional requirements, showing interactions between users (actors) and the system to achieve specific goals (use cases). It identifies the system's boundaries, actors, and use cases, highlighting what the system must accomplish. This diagram is commonly used in the early stages of development to clarify user needs, define system functionality, and ensure the system meets requirements [16].

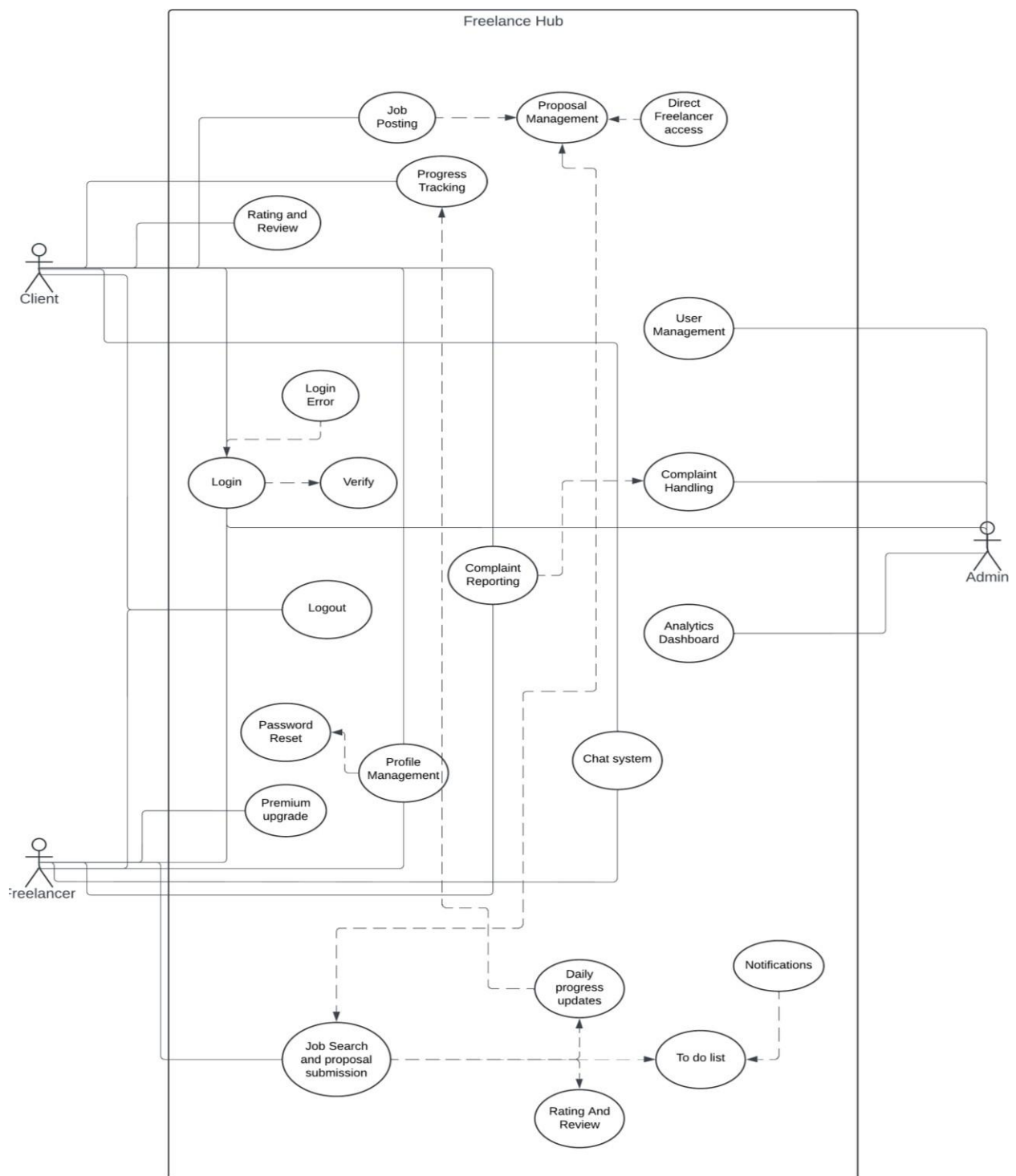


Figure 1:Use case diagram

4.2.1 SEQUENCE DIAGRAM

A sequence diagram is an interaction diagram that shows how objects or components in a system interact over time. It represents the sequence of events in a scenario, using vertical lifelines for objects and horizontal arrows for messages exchanged between them. These diagrams help visualize the flow of control, making it easier to design, analyze, and communicate complex

systems. Sequence diagrams are commonly used in software development to model system behavior and document interactions between components [17].

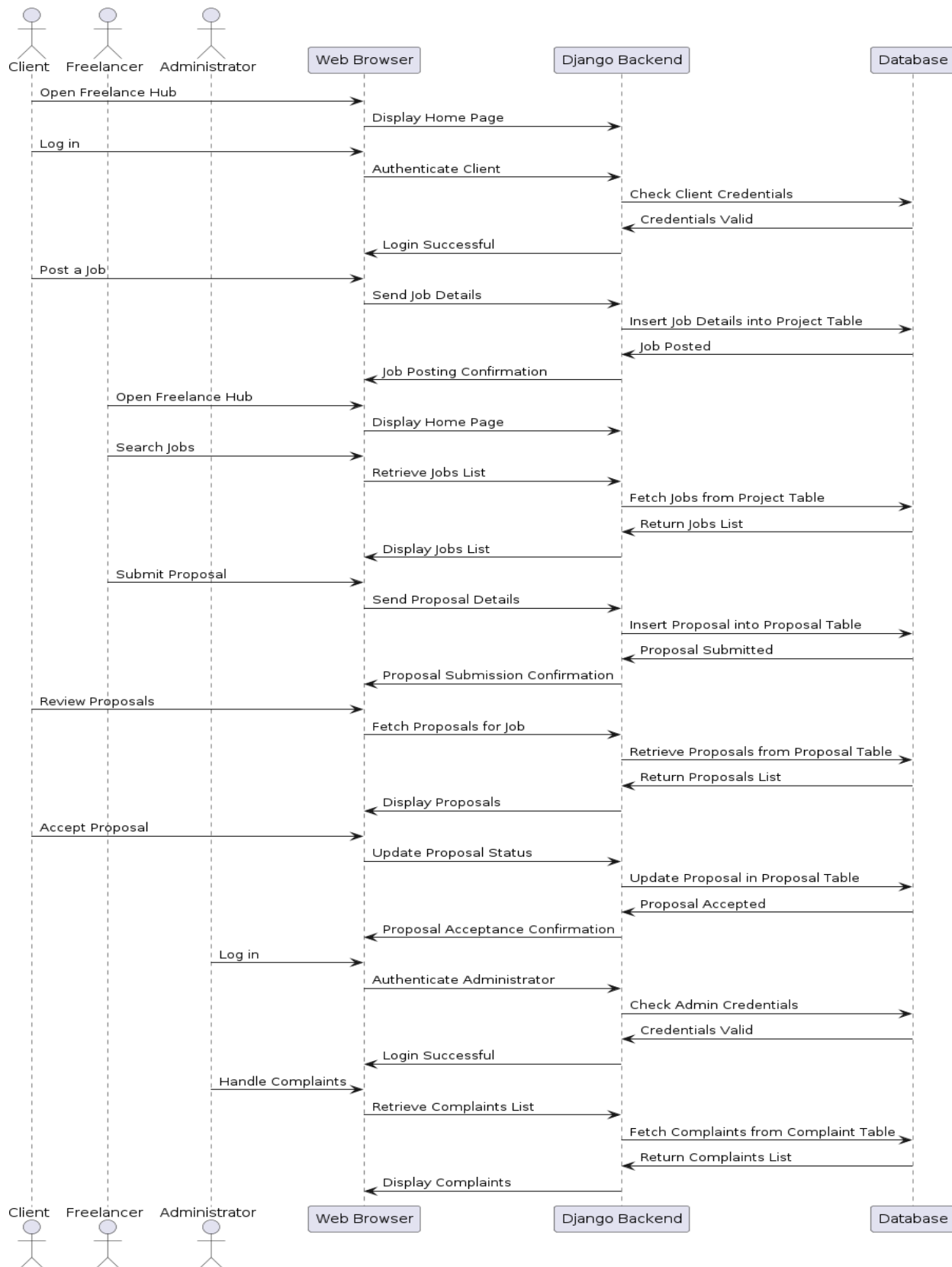


Figure 2 : Sequence diagram

4.2.2 ACTIVITY DIAGRAM

An activity diagram is a graphical representation of workflows and processes that shows the flow of control or object flow. It visually depicts the steps, actions, and decision points involved in a process or system. It is used to model, visualize, and analyze the behavior of a system, and to capture the sequence of activities or steps that occur during the execution of a use case or business process. The activity diagram is an important tool in software development, business process modeling, and system analysis, as it helps stakeholders to understand complex processes and workflows in a clear and concise manner [18].

Activity Diagram of Admin

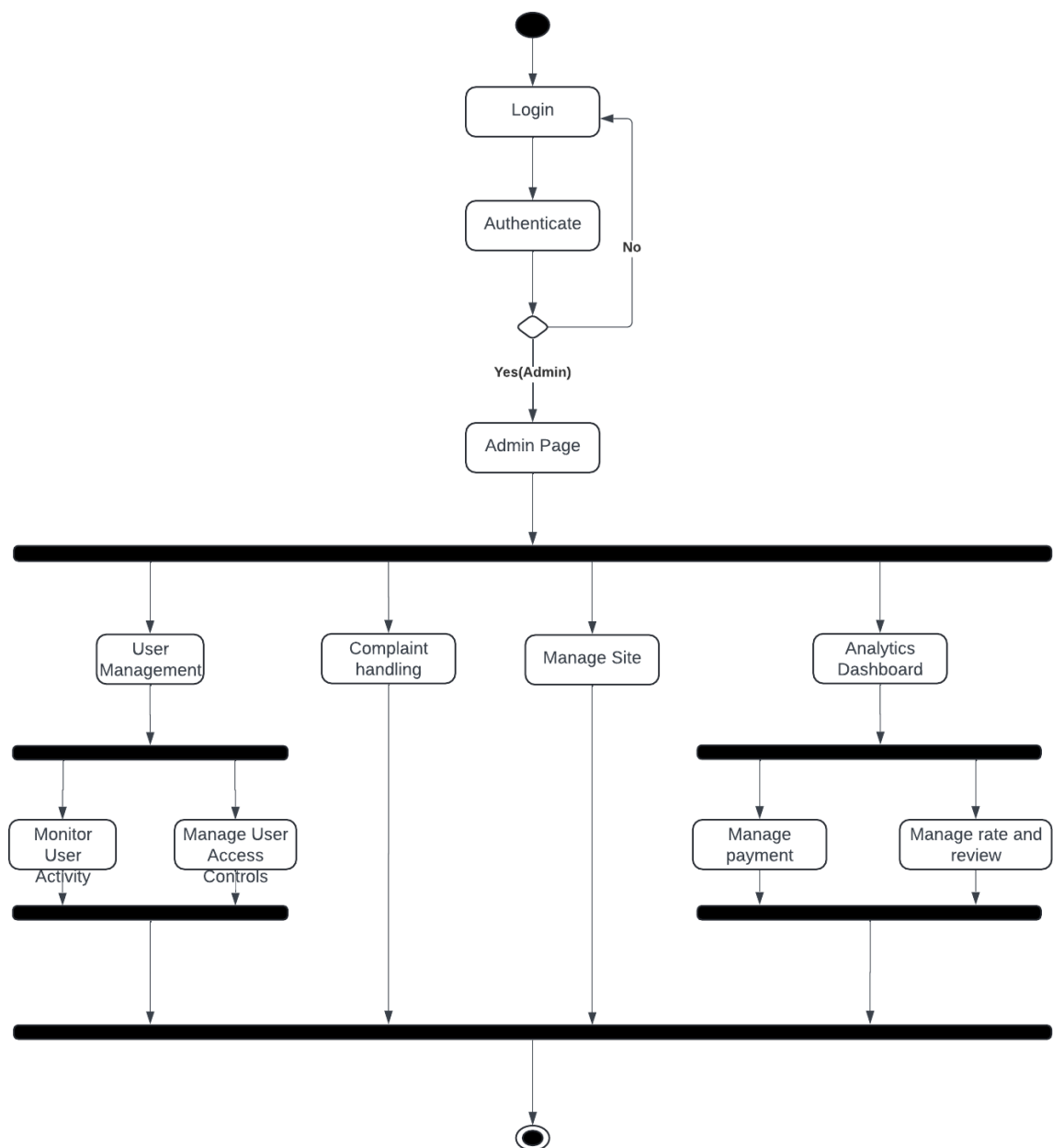


Figure 3: Activity diagram of admin

Activity Diagram of Client

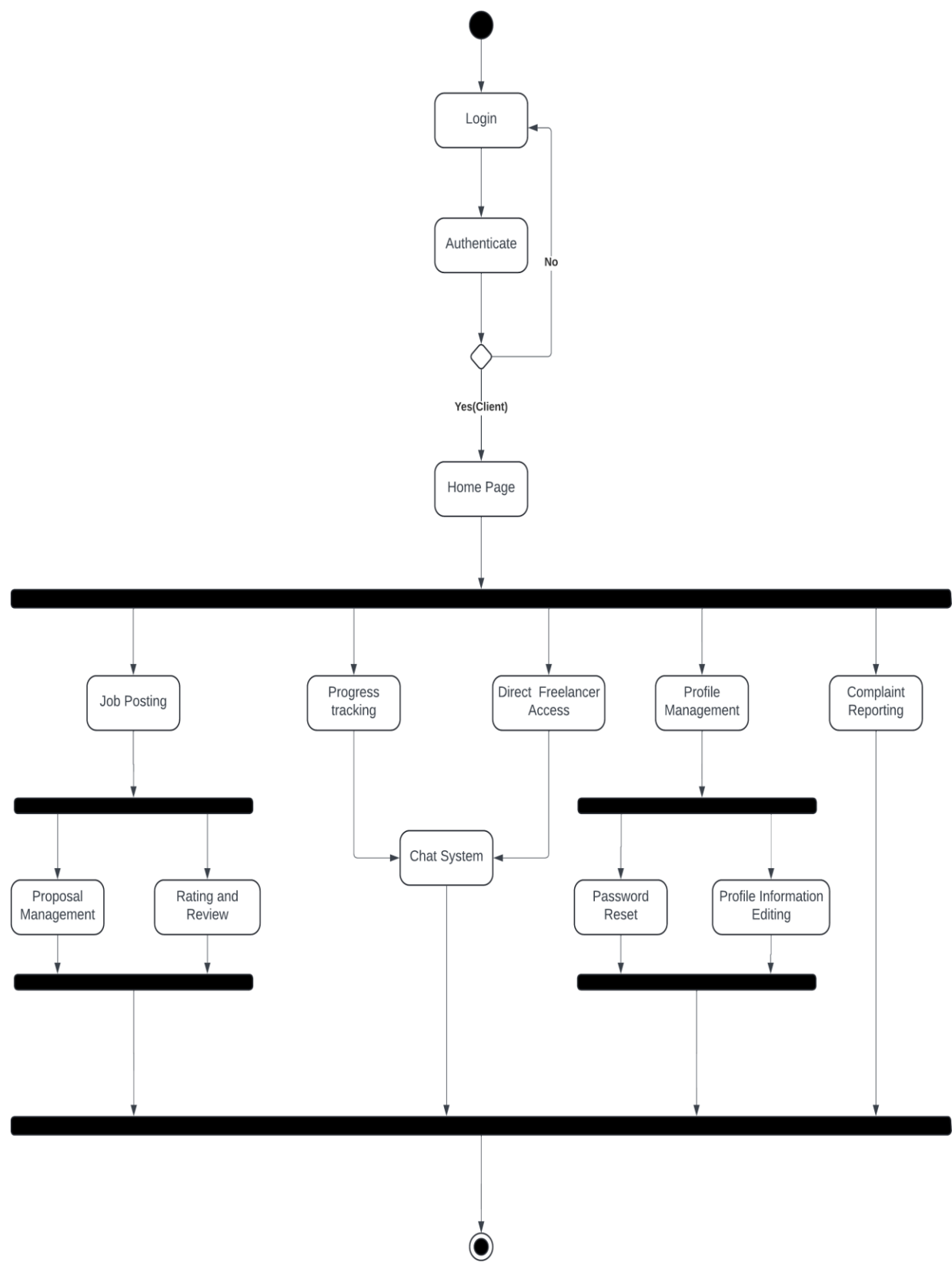


Figure 4:Activity diagram of client

Activity Diagram of Freelancer

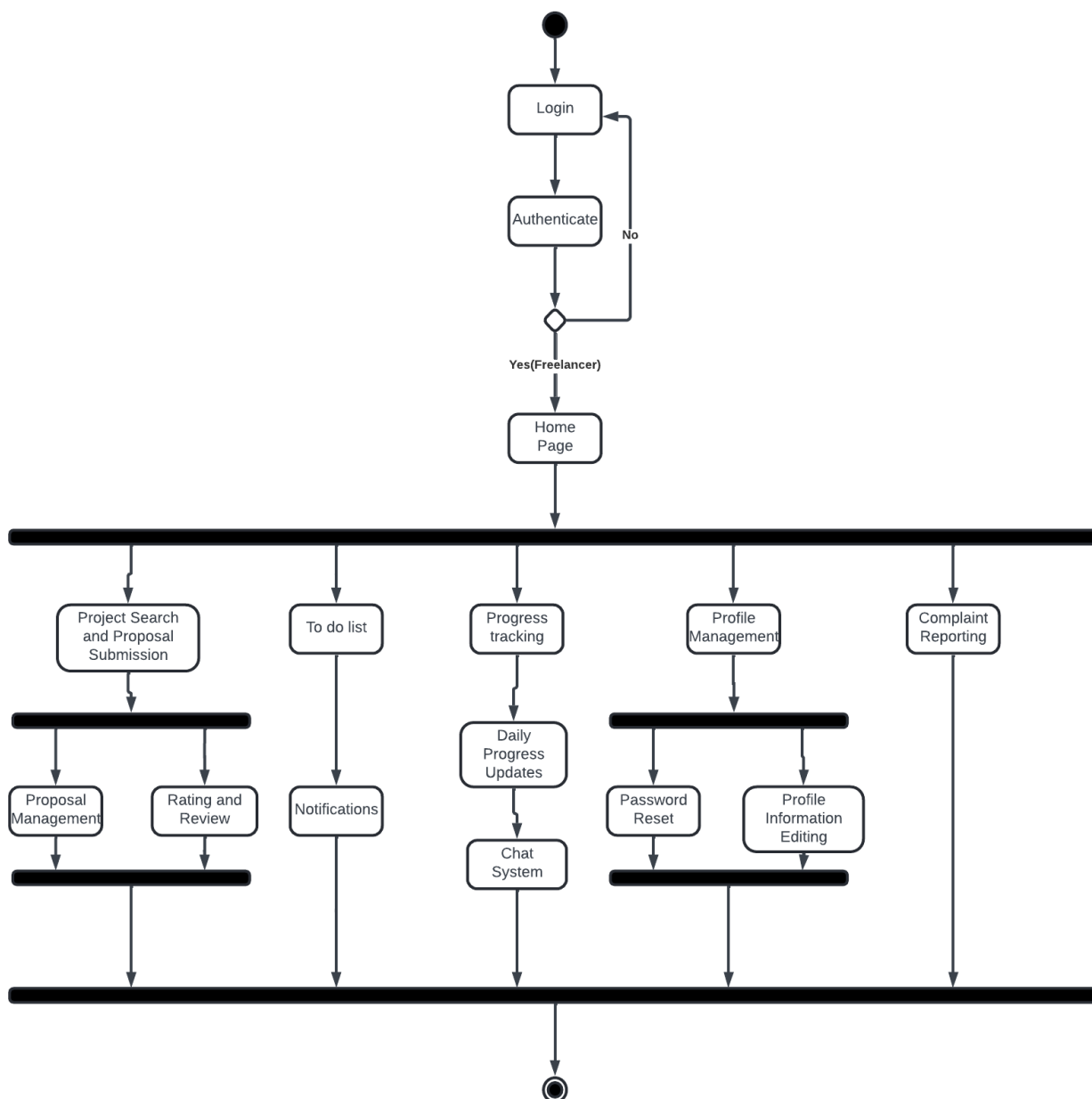


Figure 5:Activity diagram of freelancer

4.2.3 CLASS DIAGRAM

A class diagram is a type of diagram used in object-oriented programming to illustrate the structure of a system by showing the classes and their relationships to each other. It provides a visual representation of the system's classes, their attributes, and the methods or operations that can be performed on them. The class diagram allows developers to design and plan the system's architecture, define the relationships between classes, and identify the attributes and methods that will be used in the system. It is an essential tool for software developers to communicate and collaborate with stakeholders and other team members involved in the project [19].

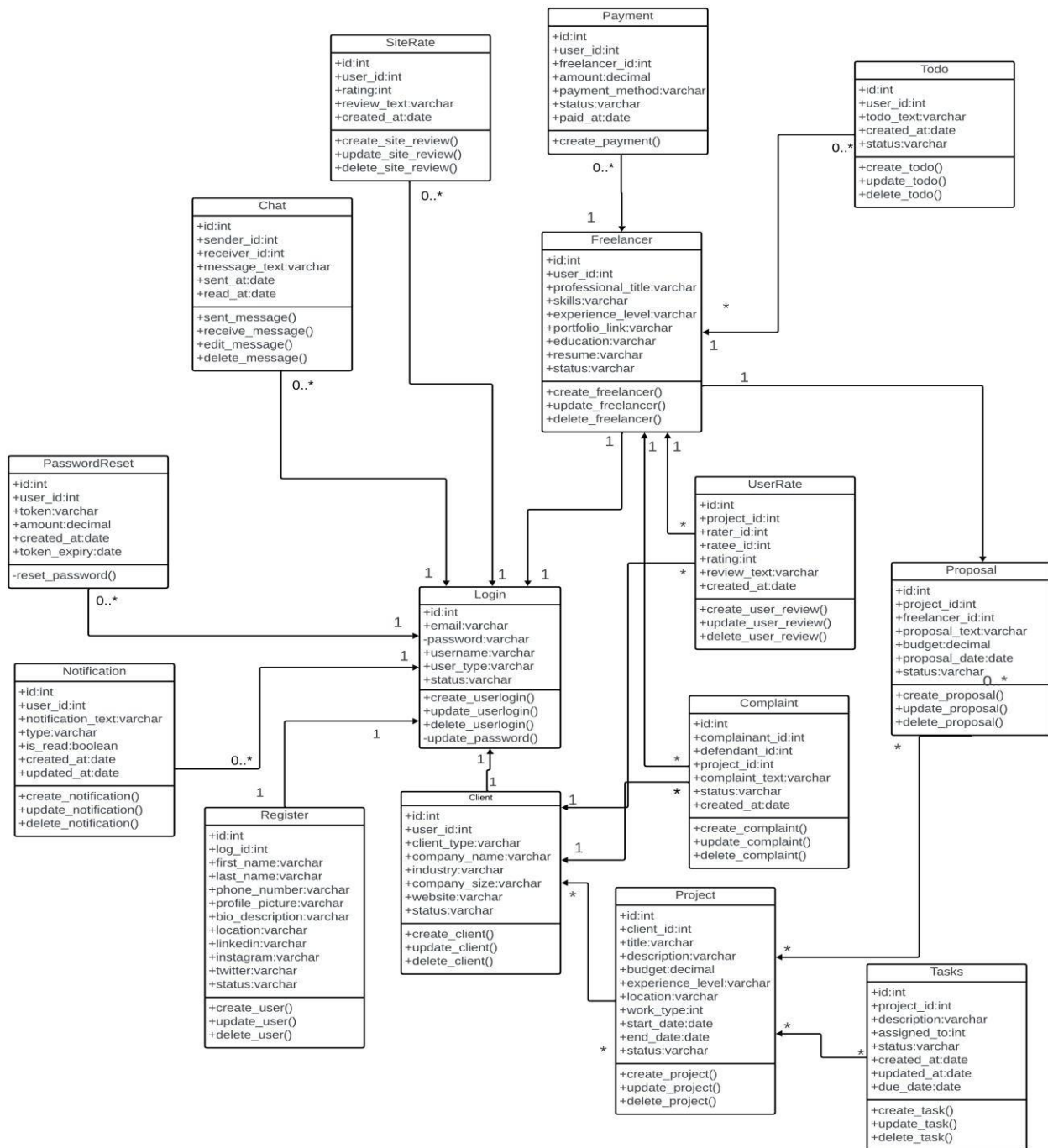


Figure 6:class diagram

4.2.4 OBJECT DIAGRAM

An object diagram is a type of static structure diagram in UML (Unified Modeling Language) that represents a snapshot of the system at a particular point in time. It shows instances of classes (objects) and their relationships, including object attributes and links between objects. Object diagrams help visualize real-world scenarios, illustrating how objects interact in a specific context, providing insights into system behavior and relationships at runtime [20].

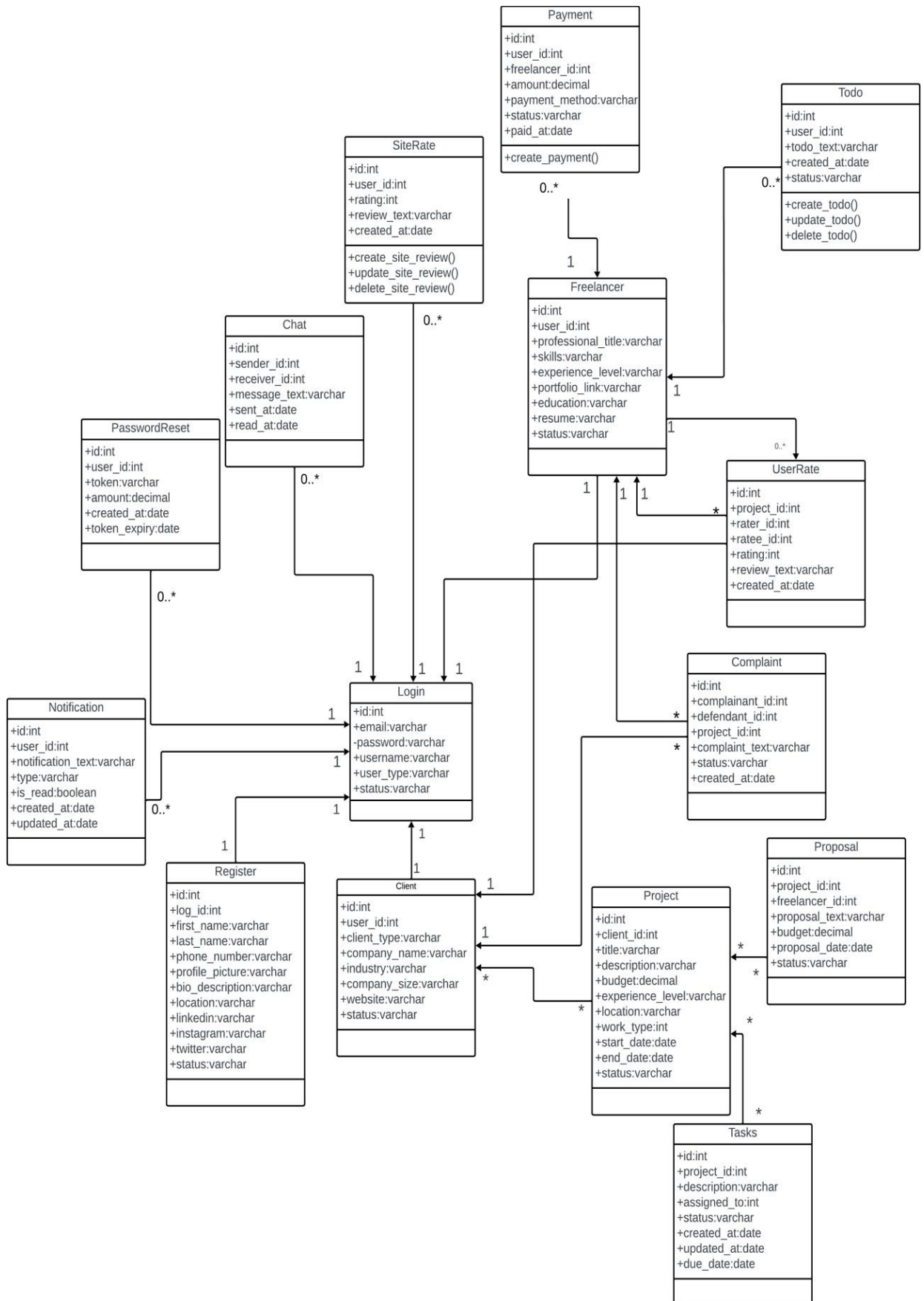
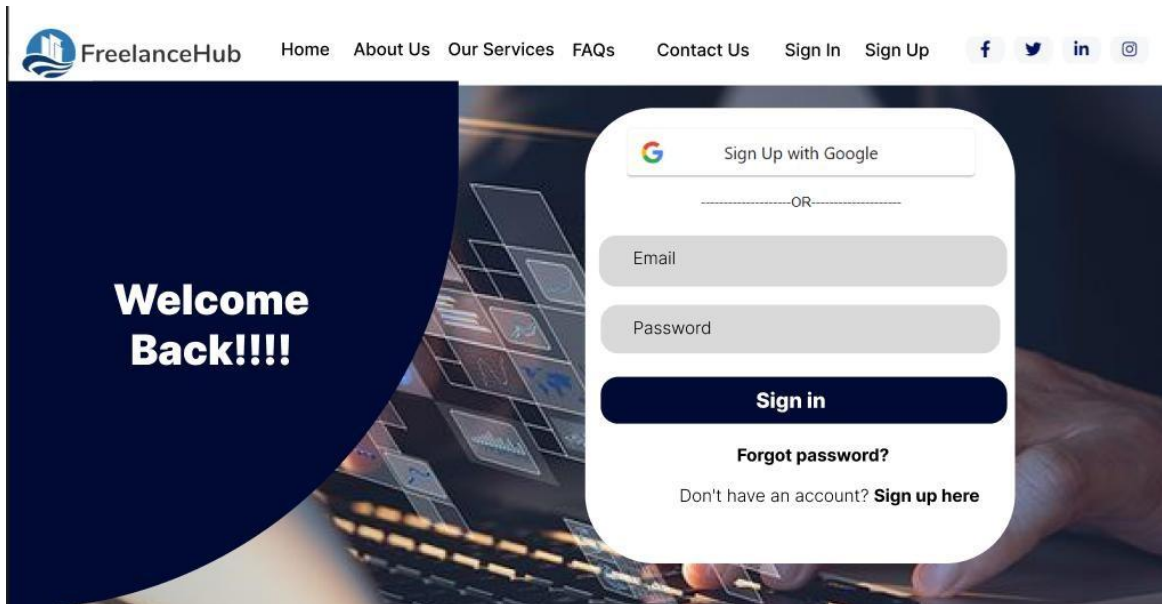


Figure 7:object diagram

4.3 USER INTERFACE DESIGN USING FIGMA

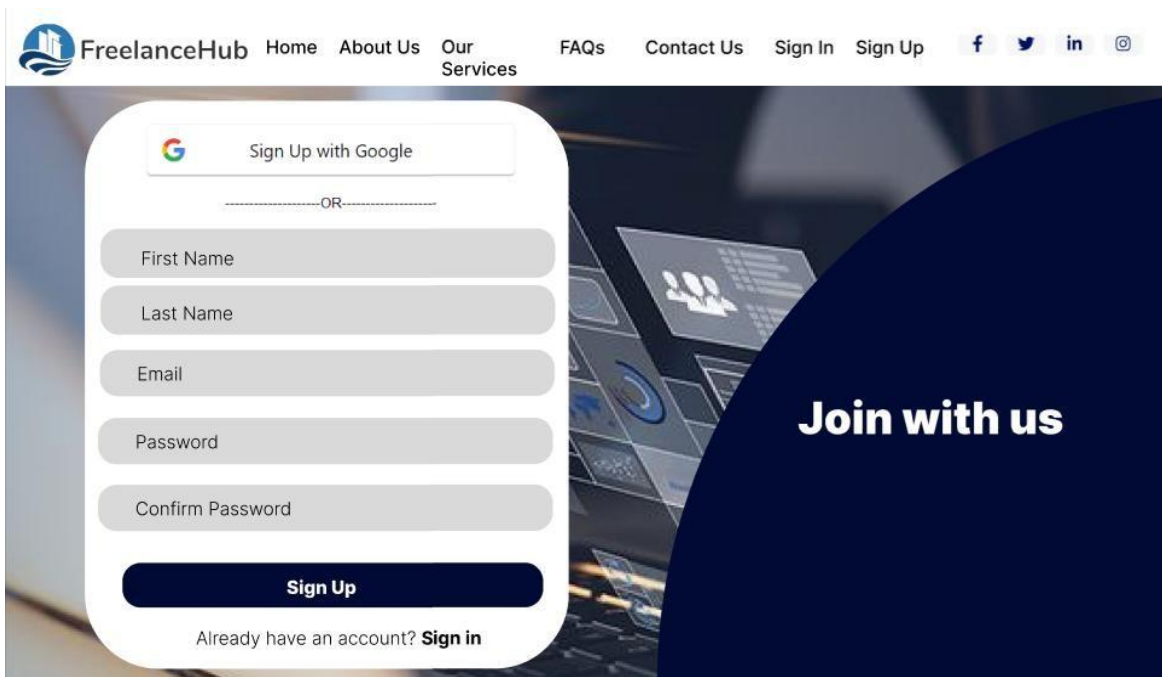
Form Name: Login



The image shows the login page of the FreelanceHub website. The header includes the FreelanceHub logo and navigation links: Home, About Us, Our Services, FAQs, Contact Us, Sign In, and Sign Up. Social media icons for Facebook, Twitter, LinkedIn, and Instagram are also present. The main content area features a large dark blue curved shape on the left with the text "Welcome Back!!!!". On the right, there is a white rounded rectangle containing the login form. The form includes a "Sign Up with Google" button, an "OR" separator, input fields for "Email" and "Password", a "Sign in" button, a "Forgot password?" link, and a "Don't have an account? Sign up here" link.

Figure 8: Interface design of login page

Form Name: Register



The image shows the registration page of the FreelanceHub website. The header is identical to the login page. The main content area features a large dark blue curved shape on the right with the text "Join with us". On the left, there is a white rounded rectangle containing the registration form. The form includes a "Sign Up with Google" button, an "OR" separator, input fields for "First Name", "Last Name", "Email", "Password", and "Confirm Password", a "Sign Up" button, and a link for "Already have an account? Sign in".

Figure 9: Interface design of registration page

4.4 DATABASE DESIGN

4.4.1 Relational Database Management System (RDBMS)

Relational Database Management System (RDBMS) is a type of database management system that stores and manages data in a tabular form, with relationships between tables. It consists of a set of tables, each with a unique name, and columns or fields that represent attributes of the entity. RDBMS uses a structured query language (SQL) to perform operations on data, such as adding, deleting, updating, and querying data. RDBMS also provides the ability to enforce data integrity and ensure consistency through the use of constraints, such as primary keys, foreign keys, and unique constraints. Overall, RDBMS is a powerful tool for managing large amounts of structured data in a highly efficient and scalable manner

4.4.2 Normalization

Normalization is a process of organizing data in a database to eliminate redundancy and improve data integrity. It involves dividing large tables into smaller, more manageable tables and defining relationships between them. There are several levels of normalization, each with its own set of rules, but the primary goal of normalization is to minimize data redundancy and ensure that each piece of information is stored in only one place. By doing so, normalization reduces the likelihood of data inconsistencies and anomalies and improves the accuracy and efficiency of database operations.

Normalization is typically divided into several levels, or "normal forms," each of which imposes a set of rules on how data should be organized. The most common normal forms are:

- First Normal Form (1NF)
- Second Normal Form (2NF)
- Third Normal Form (3NF)
- Boyce-Codd Normal Form
- Fourth Normal Form
- Fifth normal form

First Normal Form (1NF)

First Normal Form (1NF) is the basic level of database normalization, ensuring that each table has a unique primary key and that all columns contain atomic (indivisible) values. This means no multi-valued or repeating groups of data are allowed in any column. To achieve 1NF,

repeating data is moved to separate tables. The goal is to eliminate redundancy, improve data integrity, and make the database easier to manage and maintain.

For example: Relation STUDENT in table 1 is not in 1NF because of multi-valued attribute STUD_PHONE. Its decomposition into 1NF has been shown in table 2.

STUD_NO	STUD_NAME	STUD_PHONE	STUD_STATE	STUD_COUNTRY
1	RAM	9716271721, 9871717178	HARYANA	INDIA
2	RAM	9898297281	PUNJAB	INDIA
3	SURESH		PUNJAB	INDIA

Table 1

Conversion to first normal form

STUD_NO	STUD_NAME	STUD_PHONE	STUD_STATE	STUD_COUNTRY
1	RAM	9716271721	HARYANA	
1	RAM	9871717178	HARYANA	INDIA
2	RAM	9898297281	PUNJAB	INDIA
3	SURESH		PUNJAB	INDIA

Table 2

Second Normal Form (2NF)

Second Normal Form (2NF) builds on First Normal Form (1NF) by ensuring that every non-key attribute is fully dependent on the entire primary key, not just part of it. This eliminates partial dependencies in tables with composite primary keys. By achieving 2NF, data redundancy is reduced, data integrity is improved, and the database becomes easier to maintain and update. It also enhances query efficiency by organizing data more consistently.

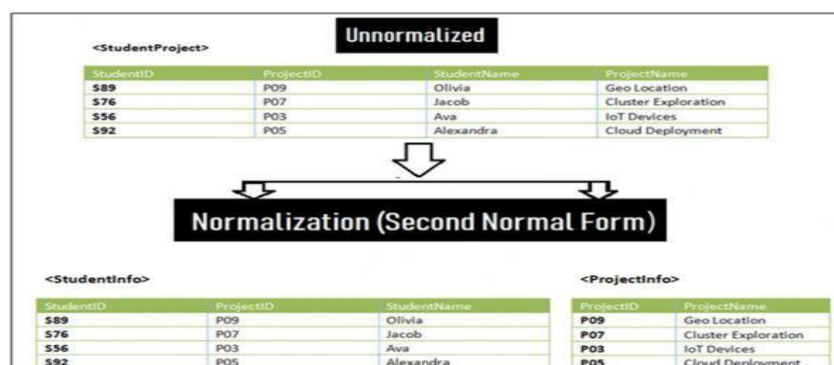
For Example:

The prime key attributes are **StudentID** and **ProjectID**.

As stated, the non-prime attributes i.e. **StudentName** and **ProjectName** should be functionally dependent on part of a candidate key, to be Partial Dependent.

The **StudentName** can be determined by **StudentID**, which makes the relation Partial Dependent. The **ProjectName** can be determined by **ProjectID**, which makes the relation Partial Dependent.

Therefore, the <StudentProject> relation violates the 2NF in Normalization and is considered a bad database design.



Third Normal Form (3NF)

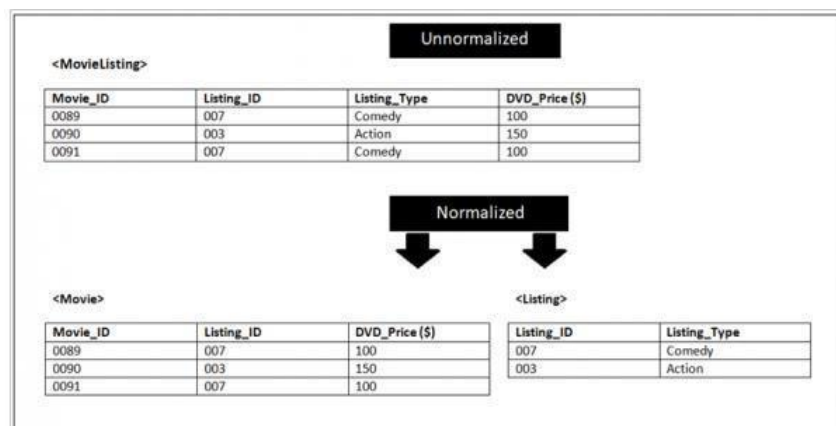
Third Normal Form (3NF) builds on the principles of First Normal Form (1NF) and Second Normal Form (2NF) by ensuring that all non-key attributes in a table are independent of each other, eliminating transitive dependencies. This means each non-key attribute should depend solely on the primary key and not on other non-key attributes. Achieving 3NF involves breaking tables into smaller, specialized tables focused on specific data aspects. This reduces data redundancy, enhances data integrity, and improves database efficiency. However, pursuing 3NF may complicate the data model, making it harder to understand and manage, and it may not be necessary for every database.

For Example:

The table is not in 3NF because it has a transitive functional dependency –

[Movie_ID -> Listing_ID](#) [Listing_ID -> Listing_Type](#)

Therefore, **Movie_ID -> Listing_Type** i.e. transitive functional dependency.



Boyce-Codd Normal Form

Boyce-Codd Normal Form (BCNF) is a higher level of normalization than Third Normal Form (3NF) that requires every determinant in a table to be a candidate key, ensuring it uniquely determines all non-key attributes. This eliminates non-trivial functional dependencies between non-key attributes, reducing data anomalies and inconsistencies.

Achieving BCNF may involve splitting tables into multiple smaller tables, each with its own primary key. While this can create a more complex schema, it helps maintain data integrity and reduces redundancy. BCNF is particularly important for large databases that need high performance and consistency, optimizing query execution and minimizing data errors. In summary, BCNF enhances data integrity and database performance by ensuring that all determinants are candidate keys.

For example consider relation $R(A, B, C)$

$A \rightarrow BC,$

$B \rightarrow A$

A and B both are super keys so above relation is in BCNF.

Fourth Normal Form

Fourth Normal Form (4NF) is a level of database normalization that builds on Third Normal Form (3NF) by eliminating multi-valued dependencies between attributes. Its main goal is to ensure that each non-key attribute in a table is functionally dependent on the primary key and that no non-trivial multi-valued dependencies exist.

A multi-valued dependency occurs when three attributes (A, B, and C) are present, with A as the primary key and B and C as non-key attributes that are independent of each other but both depend on A. To achieve 4NF, the table must be split into two separate tables, each containing the primary key and one of the non-key attributes. This process reduces redundancy and helps maintain data consistency and accuracy by eliminating anomalies. Overall, 4NF enhances the integrity of the database.

A common example of a table that could benefit from 4NF is a student course schedule table.

Let's say we have a table named "Course_Schedule" that contains the following fields:

- Student_ID (Primary Key)
- Course_ID (Primary Key)
- Semester
- Grade

In this table, a student can take multiple courses in the same semester, and a course can be taken by multiple students. Therefore, we have a many-to-many relationship between students and courses. However, each course in a semester is likely to have a different instructor, and the instructor can also change from semester to semester.

This creates a multivalued dependency between the Course_ID and Instructor fields. To normalize this table to 4NF, we need to split it into two separate tables:

1. Course_Schedule table:

- Student_ID (Primary Key)
- Course_ID (Primary Key)
- Semester
- Grade

2. Course_Instructor table:

- Course_ID (Primary Key)
- Semester (Primary Key)
- Instructor

By doing this, we have removed the multivalued dependency and ensured that each piece of data in the database is atomic and not redundant. The Course_Instructor table can now be updated independently of the Course_Schedule table, and the data in both tables will remain consistent.

Fifth normal form

Fifth Normal Form (5NF), or Project-Join Normal Form, is the highest level of normalization that addresses complex relationships between entities, especially in many-to-many relationships. It requires decomposing the database into smaller tables, each with its own primary key, focusing on single attributes or relationships to be "fact-oriented."

Achieving 5NF involves using join dependency techniques, ensuring that tables can be reconstructed through join operations. This reduces redundancy and improves data consistency while enhancing flexibility. However, achieving 5NF can be complex and may not be practical for all databases.

4.4.3 Sanitization

Sanitization in database design refers to the process of removing potentially harmful or unwanted characters from user input before it is stored in the database. This is done to prevent various forms of attacks, such as SQL injection and cross-site scripting (XSS), which can compromise the security and integrity of the database. Sanitization can be achieved through various methods, including input validation, parameterized queries, and using prepared statements. Proper sanitization of user input is crucial for maintaining the security and reliability of a database, and should be a fundamental aspect of any database design.

4.4.4 Indexing

Indexing in database design is the process of creating data structures that improve the efficiency of data retrieval operations. Indexes allow users to quickly find specific data in a large database by creating a mapping between the values of one or more columns and their corresponding locations in the database. When a user performs a search or query that involves the indexed column(s), the database can quickly locate the relevant data without having to scan the entire database. Indexing is an important consideration in database design because it can significantly

improve the performance of database operations, particularly in large or complex databases with many tables and columns. However, it is important to carefully consider which columns to index, as indexing too many columns can slow down database write operations and increase storage requirements.

4.5 TABLE DESIGN

1. **tbl_customuser**

Primary Key: id

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	username	Varchar (128)	Not Null	Username of the user
3	Email	Varchar (255)	Unique, Not Null	Email of the user
4	password	Varchar (128)	Not Null	Password of the user
5	status	Varchar (50)	Default 'active'	User status (active/inactive)
6	joined	DateTime	Auto Add	Date when the user joined
7	role	Varchar (20)	Nullable	Role of the user (admin, client, freelancer)
8	welcome_email_sent	Boolean	Default False	Whether the welcome email has been sent
9	permission	Boolean	Default False	Permission status
10	email_verified	Boolean	Default False	Email verification status
11	google	Boolean	Default False	Logged in via Google
12	complaint_count	Int	Default 0	Number of complaints against the user

2. **tbl_register**

Primary key: **id**

Foreign key: **user_id** references table **tbl_customuser(id)**

N o:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	user_id	Int	Foreign Key	Foreign key referencing tbl_customuser
3	first_name	Varchar (150)	Nullable	First name of the user

4	last_name	Varchar (150)	Nullable	Last name of the user
5	phone_number	Varchar (10)	Nullable	Phone number of the user
6	profile_picture	Varchar (255)	Nullable	Profile picture URL
7	bio_description	Text	Nullable	Bio description
8	location	Varchar (255)	Nullable	User's location
9	linkedin	Varchar (255)	Nullable	LinkedIn profile link
10	instagram	Varchar (255)	Nullable	Instagram profile link
11	twitter	Varchar (255)	Nullable	Twitter profile link

3. tbl_passwordreset

Primary key: **id**

Foreign Key: **user_id** references table **tbl_customuser(id)**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	user_id	Int	Foreign Key	Foreign key referencing tbl_customuser
3	token	Varchar (50)	Default None	Token for password reset
4	created_at	DateTime	Auto Add	Token creation timestamp
5	expiry	DateTime	Not Null	Expiry date/time of the token

4. tbl_emailverification

Primary key: **id**

Foreign key: **user_id** references table **tbl_customuser**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	user_id	Int	Foreign Key	Foreign key referencing tbl_customuser
3	token	Varchar (50)	Default None	Token for email verification
4	created_at	DateTime	Auto Add	Token creation timestamp

5	expiry	DateTime	Not Null	Expiry date/time of the token
---	--------	----------	----------	-------------------------------

5. tbl_event

Primary key: **id**

Foreign Key: **user_id** references table **tbl_customuser(id)**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	title	Varchar (200)	Not Null	Title of the event
3	start_time	DateTime	Not Null	Event start time
4	end_time	DateTime	Not Null	Event end time
5	description	Text	Nullable	Description of the event
6	color	Varchar (7)	Default '#ffffff'	Color code of the event
7	user_id	Int	Foreign Key	Foreign key referencing tbl_customuser

6. tbl_notification

Primary key: **id**

Foreign key: **user_id** references table **tbl_customuser**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	user_id	Int	Foreign Key	Foreign key referencing tbl_customuser
3	message	Varchar (255)	Not Null	Notification message
4	is_read	Boolean	Default False	Indicates if the notification has been read
5	created_at	DateTime	Auto Now Add	Timestamp when the notification was created

7. tbl_site_review

Primary key: **id**

Foreign key: **user_id** references table **tbl_customuser**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table

2	user_id	Int	Foreign Key	Foreign key referencing tbl_customuser
3	review_text	Text	Not Null	Review text provided by the user
4	rating	Positive Int	Not Null	Rating given by the user
5	created_at	DateTime	Default Now	Timestamp when the review was created

8. tbl_cancellation_request

Primary key: **id**

Foreign key: **project_id** references table **tbl_project**, **requested_by** references table **tbl_customuser**, **approver** references table **tbl_customuser**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	project_id	Int	Foreign Key	Foreign key referencing the project
3	requested_by	Int	Foreign Key	Foreign key referencing tbl_customuser (the user who requested)
4	approver	Int	Foreign Key	Foreign key referencing tbl_customuser (the user who approves)
5	status	Varchar (10)	Default 'Pending'	Status of the cancellation request
6	requested_date	DateTime	Default Now	Timestamp when the request was made
7	response_date	DateTime	Nullable	Timestamp when the request was responded to
8	reason	Text	Nullable	Reason for cancellation

9. tbl_refundpayment

Primary key: **id**

Foreign key: **pay_to** references table **tbl_customuser**, **user** references table **tbl_customuser**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	amount	Decimal(10, 2)	NOT NULL	The amount to be refunded
3	pay_to	Int	Foreign Key	Foreign key referencing tbl_customuser (Recipient of the refund)
4	is_paid	Boolean	Default False	Whether the refund payment has been paid

5	payment_date	DateTime	Nullable	Date and time when the refund payment was made
6	razorpay_order_id	Varchar(255)	Nullable	Razorpay order ID for the refund payment
7	razorpay_payment_id	Varchar(255)	Nullable	Razorpay payment ID for the refund payment
8	user	Int	Foreign Key	Foreign key referencing tbl_customuser (User who created the refund payment)
9	total_paid	Decimal(10, 2)	Default 0.00	Total amount that has been paid
10	compensation_amount	Decimal(10, 2)	Default 0.00	Compensation amount included in the refund

10. tbl_clientprofile

Primary key: **id**

Foreign key: **user_id** references table **tbl_customuser**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	user	Int	Foreign Key	Foreign key referencing tbl_customuser
3	client_type	Varchar(50)		Type of client ('Individual' or 'Company')
4	company_name	Varchar(255)	Nullable	Name of the company, if applicable
5	website	Varchar(255)	Nullable	Website URL of the client
6	license_number	Varchar(255)	Nullable	License number of the client
7	aadhaar_document	Varchar (file)	Nullable	Aadhaar document file

11. tbl_project

Primary key: **id**

Foreign key: **user_id** references table **tbl_reg**, **freelancer** references table **tbl_customuser**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	title	Varchar(255)	Null	Title of the project
3	description	Text	Null	Description of the project

4	budget	Int	Null	Budget allocated for the project
5	category	Varchar(605)	Null	Category of the project
6	allow_bid	Boolean	Null	Whether bidding is allowed on the project
7	end_date	Date	Nullable	End date for the project
8	file_upload	Varchar (file)	Nullable	File upload for the project
9	created_at	DateTime	Default Now	Timestamp when the project was created
10	User_id	Int	Foreign Key	Foreign key referencing tbl_customuser (Client)
11	status	Varchar(20)	Default:open	Status of the project (open/closed)
12	start_date	Date	Nullable	Start date of the project
13	project_end_date	Date	Nullable	End date of the project
14	project_status	Varchar(50)	Default 'Not Started'	Current status of the project
15	freelancer	Int	Foreign Key Nullable	Foreign key referencing tbl_customuser (Freelancer)
16	git_repo_link	Varchar(200)	Nullable	URL to the project's Git repository
17	gst_rate	Decimal (5,2)	Default 18.00	GST rate applied to the project
18	gst_amount	Decimal (15,2)	Default 0.00	GST amount calculated
19	total_including_gst	Decimal (15,2)	Default 0.00	Total amount including GST
20	client_review_given	Boolean	Default False	Whether the client review is given
21	freelancer_review_giv en	Boolean	Default False	Whether the freelancer review is given
22	scope	Varchar(10)	Default 'medium'	Scope of the project (low/medium/high)

12. tbl_repository

Primary key: **id**

Foreign key: **created_by** references table **tbl_customuser**, **project_id** references table **tbl_project**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	project_id	Int	Foreign Key	Foreign key referencing tbl_project
3	name	Varchar(255)	Not Null	Name of the repository

4	created_at	DateTime	Default Now	Timestamp when the repository was created
5	created_by	Int	Foreign Key	Foreign key referencing tbl_customuser

13. tbl_sharedfile

Primary key: **id**

Foreign key: **uid** references table **tbl_customuser**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	repository	Int	Foreign Key	Foreign key referencing tbl_repository
3	file	Varchar(file)		File uploaded
4	description	Text	Nullable	Description of the file
5	uploaded_by	Int	Foreign Key	Foreign key referencing tbl_customuser
6	uploaded_at	DateTime	Default Now	Timestamp when the file was uploaded

14. tbl_sharedurl

Primary key: **id**

Foreign key: **repository** references table **tbl_repository**, **shared_by** references table **tbl_customuser**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	repository	Int	Foreign Key	Foreign key referencing tbl_repository
3	url	Varchar(200)		URL shared within the repository
4	description	Text	Nullable	Description of the URL
5	shared_by	Int	Foreign Key	Foreign key referencing tbl_customuser
6	shared_at	DateTime	Default Now	Timestamp when the URL was shared

15. tbl_sharednote

Primary key: **id**

Foreign Key: **repository** references table **tbl_repository**, **added_by** references table **tbl_customuser**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	repository	Int	Foreign Key	Foreign key referencing tbl_repository
3	note	Text		Note added to the repository
4	added_by	Int	Foreign Key	Foreign key referencing tbl_customuser
5	added_at	DateTime	Default Now	Timestamp when the note was added

16. tbl_task

Primary key: **id**

Foreign key: **project** references table **tbl_project**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
3	title	Varchar(255)		Title of the task
4	description	Text	Nullable	Description of the task
5	start_date	Date	Nullable	Start date of the task
6	due_date	DateTime		Due date of the task
7	status	Varchar(20)	Default 'Pending'	Status of the task ('Pending', 'InProgress', etc.)
8	created_at	DateTime	Default Now	Timestamp when the task was created
3	title	Varchar(255)	Not null	Title of the task
9	updated_at	DateTime	Default Now	Timestamp when the task was last updated
10	progress_percentage	Float	Default 0.0	Progress percentage of the task

17. tbl_freelancecontract

Primary key: **id**

Foreign key: **client** references table **tbl_customuser**, freelancer references table

tbl_customuser,project references table **tbl_project**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table

2	client	Int	Foreign Key	Foreign key referencing tbl_customuser (Client)
3	freelancer	Int	Foreign Key	Foreign key referencing tbl_customuser (Freelancer)
4	project	Int	One-to-One	Foreign key referencing tbl_project
5	client_signature	Varchar(file)	Nullable	File for client signature
6	freelancer_signature	Varchar(file)	Nullable	File for freelancer signature
7	contract_date	Date	Default Now	Date when the contract was signed
8	pdf_version	Varchar(file)	Nullable	PDF version of the contract

18. tbl_paymentinstallment

Primary key: **id**

Foreign key: **contract** references table **tbl_freelancecontract**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	contract	Int	Foreign Key	Foreign key referencing tbl_freelancecontract
3	amount	Decimal(10, 2)		Amount for the payment installment
4	due_date	Date		Due date of the payment installment
5	status	Varchar(20)	Default 'pending'	Status of the installment (pending/paid)
6	razorpay_order_id	Varchar(255)	Nullable	Razorpay order ID for the installment
7	razorpay_payment_id	Varchar(255)	Nullable	Razorpay payment ID for the installment
8	paid_at	Date	Nullable	Date when the installment was paid

19. tbl_review

Primary key: **id**

Foreign key: **project** references table **tbl_project**, **reviewer** references table

tbl_customuser, **reviewee** references table **tbl_customuser**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table

2	project	Int	Foreign Key	Foreign key referencing tbl_project
3	reviewer	Int	Foreign Key	Foreign key referencing tbl_customuser (Reviewer)
4	reviewee	Int	Foreign Key	Foreign key referencing tbl_customuser (Reviewee)
5	review_text	Text		Text content of the review
6	overall_rating	Float		Overall rating given in the review
7	quality_of_work	Float	Nullable	Rating for quality of work
8	communication	Float	Nullable	Rating for communication
9	adherence_to_deadlines	Float	Nullable	Rating for adherence to deadlines
10	professionalism	Float	Nullable	Rating for professionalism
11	problem_solving_ability	Float	Nullable	Rating for problem-solving ability
12	review_date	DateTime	Default Now	Timestamp when the review was made

20. tbl_chatroom

Primary key: **id**

Foreign key: **project** references table **tbl_project**, **participants** references table **tbl_customuser**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	participants	Many-to-Many	Foreign Key	Foreign key referencing tbl_customuser
3	project	Int	Foreign Key	Foreign key referencing tbl_project

21. tbl_message

Primary key: **id**

Foreign key: **chat_room** references table **tbl_chatroom**, **sender** references table **tbl_customuser**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	chat_room	Int	Foreign Key	Foreign key referencing tbl_chatroom

3	sender	Int	Foreign Key	Foreign key referencing tbl_customuser
4	content	Text	Nullable	Text content of the message
5	image	Varchar (file)	Nullable	Image file in the message
6	file	Varchar (file)	Nullable	File in the message
7	timestamp	DateTime	Default Now	Timestamp of when the message was sent

22. tbl_complaint

Primary key: **id**

Foreign key: complaineer references table **tbl_customuser**, user references table **tbl_customuser**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	user	Int	Foreign Key	Foreign key referencing tbl_customuser (Complainant)
3	complaineer	Int	Foreign Key	Foreign key referencing tbl_customuser (Complaineer)
4	complaint_type	Varchar(20)	NOT NULL	Type of complaint (Client/Freelancer/Site Issue)
5	subject	Varchar(100)	NOT NULL	Subject of the complaint
6	description	Text	NOT NULL	Description of the complaint
7	status	Varchar(10)	Default 'Pending'	Status of the complaint (Pending/Resolved/Rejected)
8	created_at	DateTime	Default Now	Timestamp of when the complaint was created
9	resolved_at	DateTime	Nullable	Timestamp of when the complaint was resolved

23. tbl_template

Primary key: **id**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	name	Varchar(255)		The name of the template

3	file	Varchar(255)		Path to the template file
4	cover_image	Varchar(255)	Nullable	Path to the cover image of the template

24. tbl_freelancerprofile

Primary key: **id**

Foreign key: **user_id** references table **tbl_customuser**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	user_id	Int	Foreign Key	Foreign key referencing tbl_customuser
3	professional_title	Text(255)		Freelancer's professional title
4	skills	Text(255)		Freelancer's skills
5	experience_level	Varchar(50)	Nullable	Freelancer's experience level
6	portfolio_link	Varchar(255)	Nullable	Link to the freelancer's portfolio
7	education	Text	Nullable	Freelancer's education details
8	resume	Varchar(255)	Nullable	Path to the resume file
9	aadhaar_document	Varchar(255)	Nullable	Path to the Aadhaar document
10	status	Varchar(20)	Default 'active'	Current status of the freelancer profile (active/inactive)
11	work_type	Varchar(10)	Default 'part_time'	Work type (Full-time/Part-time)

25. tbl_todo

Primary key: **id**

Foreign key: **user_id** references table **tbl_customuser**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	user_id	Int	Foreign Key	Foreign key referencing tbl_customuser
3	title	Varchar(50)	Default None	Title of the to-do item
4	is_completed	Boolean	Default False	Status of the to-do item (Completed

				or not)
5	created_at	DateTime	Auto Add	Timestamp when the to-do item was created

26. tbl_proposal

Primary key: **id**

Foreign key: **project_id** references table **tbl_project**, **freelancer_id** references table

tbl_customuser

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	project_id	Int	Foreign Key	Foreign key referencing tbl_project
3	freelancer_id	Int	Foreign Key	Foreign key referencing tbl_customuser
4	date_issued	Date	Default Now	Date the proposal was issued
5	proposal_details	Text (Rich)		Details of the proposal
6	budget	Decimal(10, 2)		Proposed budget
7	deadline	Date		Proposal deadline
8	status	Varchar(10)	Default 'Pending'	Status of the proposal (Pending/Accepted/Rejected)
9	fancy_num	Varchar(5)	Unique, Nullable	Unique number associated with the proposal
10	proposal_file	Varchar(255)	Nullable	Path to the proposal file
11	locked	Boolean	Default False	Whether the proposal is locked or not

27. tbl_proposalfile

Primary key: **id**

Foreign key: **proposal_id** references table **tbl_proposal**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	proposal_id	Int	Foreign Key	Foreign key referencing tbl_proposal

3	file	Varchar(255)		Path to the uploaded file
4	uploaded_at	DateTime	Auto Add	Timestamp when the file was uploaded

28. tbl_document

Primary key: **id**

Foreign key: **template_id** references table **tbl_template**, **user_id** references table **tbl_customuser**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	id	Int (Auto)	Primary Key	Primary key of the table
2	user_id	Int	Foreign Key	Foreign key referencing tbl_customuser
3	resume_file	Varchar(255)	Nullable	Path to the resume file
4	portfolio_file	Varchar(255)	Nullable	Path to the portfolio file
5	template_id	Int	Foreign Key	Foreign key referencing tbl_template
6	created_at	DateTime	Auto Add	Timestamp when the document was created
7	cover_image	Varchar(255)	Nullable	Path to the cover image

CHAPTER 5

SYSTEM TESTING

5.1 INTRODUCTION

Testing is a vital process in software development that ensures product quality and functionality by identifying errors, defects, or requirement gaps. Its primary goal is to detect and resolve bugs to ensure the software meets user needs.

There are various types of testing, including unit, integration, system, acceptance, and regression testing, each focusing on different development stages. Testing is essential for ensuring software reliability, security, and adherence to quality standards. It also mitigates risks, enhances customer satisfaction, and improves overall product quality. In summary, testing is crucial for delivering reliable, secure software that meets user requirements, ultimately saving time and money while boosting customer satisfaction.

5.2 TEST PLAN

A test plan is a formal document that outlines the strategy, objectives, scope, and schedule for testing a software application. It serves as a roadmap for the testing team, detailing the testing environment, test cases, data, tools, and team responsibilities. The main goal of a test plan is to ensure an organized and effective testing process, helping to identify and resolve issues in the software. The test plan should also provide information on the different levels of testing. These typically include:

- Unit Testing
- Integration Testing
- System Testing
- Acceptance Testing

5.2.1 Unit Testing

Unit testing is a software testing technique that focuses on testing individual units or components of code in isolation from the rest of the system. Its goal is to verify that each unit performs as intended and meets specified requirements. This typically involves writing and executing automated test cases to identify errors early in the development cycle. By isolating and testing units in a controlled environment, developers can quickly identify and fix issues, ensuring the code is reliable, maintainable, and functions correctly.

5.2.2 Integration Testing

Integration testing is a software testing technique that examines the interactions between individual components of a system to ensure they work correctly together. It aims to identify defects in interfaces and interactions not detected during unit testing. This involves testing data and control

flows between modules, subsystems, and external interfaces, using approaches like top-down, bottom-up, or a combination of both. The goal is to ensure seamless functionality of the system's components before production release.

5.2.3 Validation Testing or System Testing

Validation testing and system testing are crucial in software development. Validation testing evaluates whether the software meets stakeholder requirements and user needs, ensuring it performs intended functions. In contrast, system testing assesses the entire system's functionality, performance, and reliability by testing interactions between components. Both types of testing are vital for delivering high-quality software that fulfills user expectations.

5.2.4 Output Testing or User Acceptance Testing

Output testing, or User Acceptance Testing (UAT), is a critical phase in software development where end-users test the software to ensure it meets business requirements before release. During UAT, users verify that the system functions correctly, meets both functional and non-functional requirements, and offers a user-friendly experience. The goal is to validate the software's usability, reliability, and efficiency, making it the final step before deployment.

5.2.5 Automation Testing

Automation testing uses software tools and scripts to test applications, increasing efficiency and accuracy over manual testing. It quickly identifies defects across various types of tests, such as functional and regression testing, while reducing time and costs. This approach enhances overall software quality by detecting issues early in development.

5.2.6 Selenium Testing

Selenium is an open-source framework for automating web browsers, commonly used for functional and regression testing of web applications. It supports multiple programming languages like Java, Python, and C#, and can simulate user interactions such as clicks and typing across various browsers like Chrome and Firefox. Selenium offers features for test script creation, execution, and reporting, and integrates well with tools like TestNG, JUnit, and Jenkins, making it a popular choice for automated testing in the software industry.

Example:**Test Case 1****Code**

```
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time
driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()))
login_data = [{"email": "admin@gmail.com", "password": "Admin@12", "expected_url":
    "http://127.0.0.1:8000/administrator/admin_view/"}, {"email": "ava.robinson@gmail.com",
    "password": "Ava@1234", "expected_url":
    "http://127.0.0.1:8000/freelancer/freelancer_view/"}, {"email": "samuel@gmail.com",
    "password": "Sam@1234", "expected_url":
    "http://127.0.0.1:8000/client/client_view/"}, {"email": "samuel@gmail.com", "password":
    "Sam@1235", "expected_url": "http://127.0.0.1:8000/login/"}, ]
try:
    for credentials in login_data:
        driver.get("http://127.0.0.1:8000/login/")
        email_input = WebDriverWait(driver, 10).until(
            EC.visibility_of_element_located((By.NAME, "mail")))
        password_input = WebDriverWait(driver, 10).until(
            EC.visibility_of_element_located((By.NAME, "pass")))
        email_input.clear()
        password_input.clear()
        email_input.send_keys(credentials['email'])
        password_input.send_keys(credentials['password'])
        password_input.send_keys(Keys.RETURN)
        time.sleep(2)
        if driver.current_url == credentials['expected_url'] and
        driver.current_url!="http://127.0.0.1:8000/login/":
            print(f"Login successful for {credentials['email']}.")
            time.sleep(2)
            logout_button = WebDriverWait(driver, 10).until(
                EC.element_to_be_clickable((By.LINK_TEXT, "Logout")))
            logout_button.click()
        else:
            print(f"Login failed. Incorrect URL after login.")
finally:
    driver.quit()
```

Screenshot

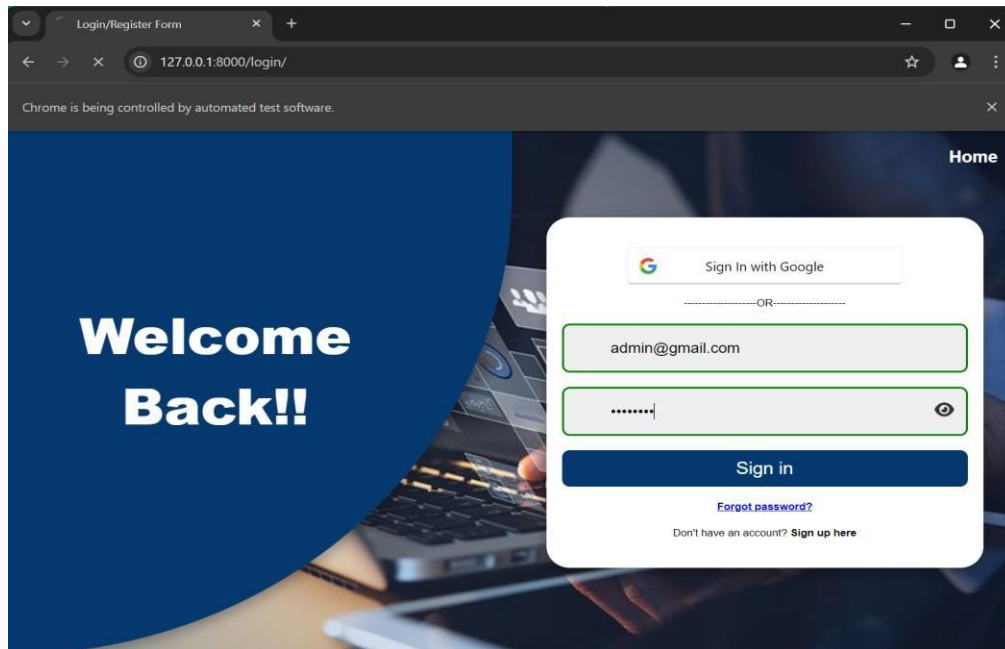


Figure 10 :interface of login page with credentials entered

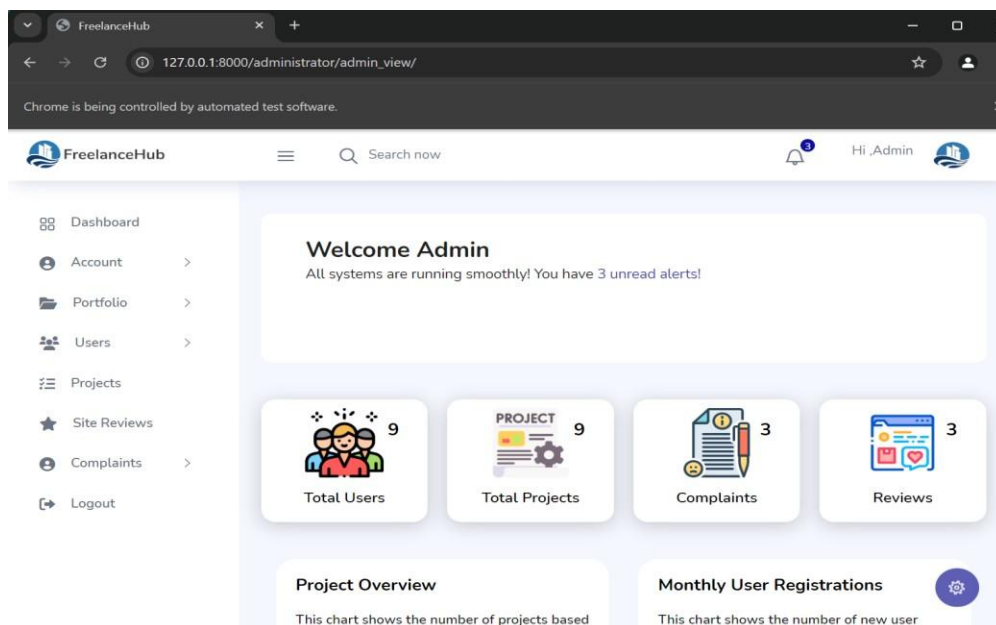


Figure 11:index page after successful login

```
PS C:\Users\LENOVO\Desktop\freelancehub> & "C:/Program Files/python/python.exe" c:/Users/LENOVO/Desktop/freelancehub/tests/test_login.py

DevTools listening on ws://127.0.0.1:58105/devtools/browser/5b40227f-9736-4eae-8047-e6d30eb8d806
Login successful for admin@gmail.com.
Created TensorFlow Lite XNNPACK delegate for CPU.
Attempting to use a delegate that only supports static-sized tensors with a graph that has dynamic-sized tensors (tensor#58 is a dynamic-sized tensor).
Login successful for ava.robinson@gmail.com.
Login successful for samuel@gmail.com.
Login failed. Incorrect URL after login.
```

Figure 11:Result of testing login page

Test Report

Test Case 1					
Project Name:FreelanceHub					
Login Test Case					
Test Case ID: Test_1			Test Designed By: Varsha Shaji		
Test Priority(Low/Medium/High):High			Test Designed Date: 25/09/2024		
Module Name: Login Page			Test Executed By : Ms.Anit James		
Test Title : Verify Login with valid email and password			Test Execution Date: 27/09/2024		
Description: Login page Testing					
Pre-Condition :User has valid email and password					
Step	Test Step	Test Data	Expected Result	Actual Result	Status(Pass/Fail)
1	Navigate to login page		Display Login page	Login page displayed	Pass
2	Valid email entered	Email: admin@gmail.com	User should be able to login	User logged in and navigate to the admin dashboard page.	Pass
3	Valid password entered	Password: Admin@12			
4	Click on Login Button				
Post-Condition: Email id and password is checked with database values for successful Login					

Test Case 2:**Code**

```

from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time

```

```
driver =
webdriver.Chrome(service=Service(ChromeDriverManager().install()))
registration_data = [
    {
        "fname": "Mary", "lname": "Doe",email": "mary.doe@gmail.com", "password":
        "Mary@123","repassword": "Mary@123","expected_url":
        "http://127.0.0.1:8000/add_user_type/65"
    },
    {
        "fname": "Noah","lname": "Wilson","email": "noah.wilson@gmail.com", "password":
        "Noah@123","repassword": "Noah@123","expected_url": "http://127.0.0.1:8000/register/"
    }
]

try:
for data in registration_data:
    driver.get("http://127.0.0.1:8000/register/")

    fname_input = WebDriverWait(driver, 10).until(
        EC.visibility_of_element_located((By.NAME, "fname")))
    lname_input = WebDriverWait(driver, 10).until(
        EC.visibility_of_element_located((By.NAME, "lname")))
    email_input = WebDriverWait(driver, 10).until(
        EC.visibility_of_element_located((By.NAME, "email")))
    )
    password_input = WebDriverWait(driver, 10).until(
        EC.visibility_of_element_located((By.NAME,"password")))
    repassword_input = WebDriverWait(driver, 10).until(
        EC.visibility_of_element_located((By.NAME, "repassword")))
    fname_input.clear()
    lname_input.clear()
    email_input.clear()
    password_input.clear()
    repassword_input.clear()
    fname_input.send_keys(data['fname'])
    lname_input.send_keys(data['lname'])
    email_input.send_keys(data['email'])
    password_input.send_keys(data['password'])
    repassword_input.send_keys(data['repassword'])
    driver.find_element(By.ID, "subbtn").click()
    time.sleep(3)
    if driver.current_url == data['expected_url'] and "add_user_type" in data['expected_url']:
        driver.find_element(By.ID, "create-account-btn").click()
        time.sleep(3)
```

```
if driver.current_url == "http://127.0.0.1:8000/login/":
    print(f"Test Passed :Registration successful for {data['email']}")
else:
    print(f"Test failed for {data['email']}")
elif driver.current_url == data['expected_url'] and data['expected_url'] ==
"http://127.0.0.1:8000/register/":
    print(f"Test passed for existing email {data['email']}")
else:
    print(f"Test failed for {data['email']}")
finally:
    driver.quit()
```

Screenshot

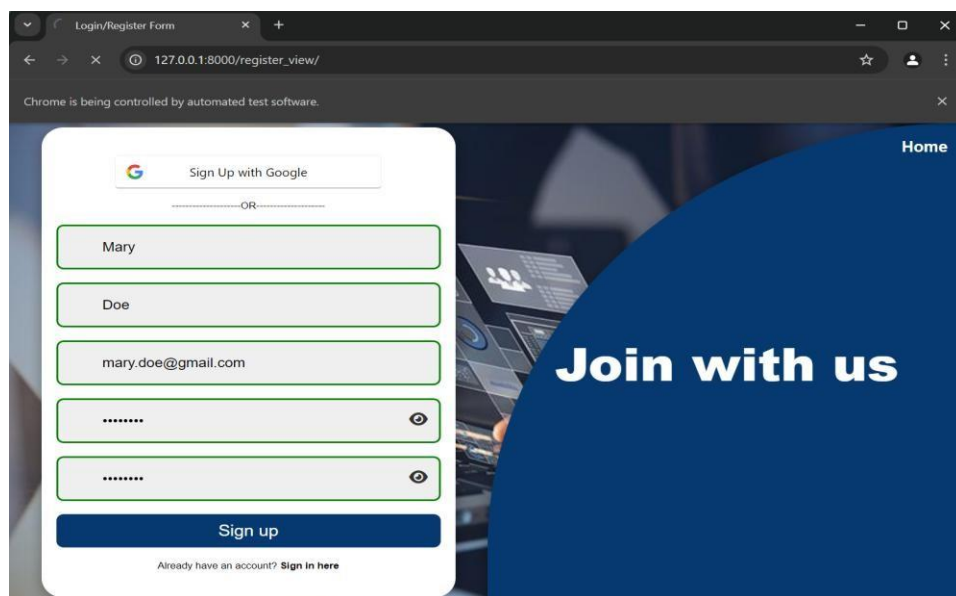


Figure 12:interface of register page with credentials entered

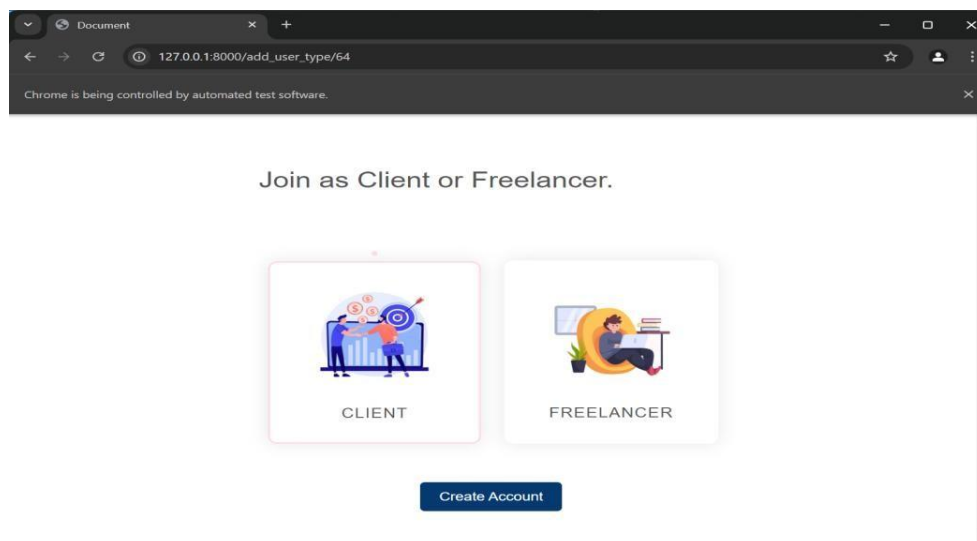


Figure 13:interface of User type page after successful registration

```

PS C:\Users\LENOVO\Desktop\freelancehub> & "C:/Program Files/python/python.exe" c:/Users/LENOVO/
/Desktop/freelancehub/tests/test_register.py

DevTools listening on ws://127.0.0.1:58097/devtools/browser/0250af10-83d5-4bb3-8ff6-26a6c817f70
3
Test Passed :Registration successful for mary.doe@gmail.com
Created TensorFlow Lite XNNPACK delegate for CPU.
Attempting to use a delegate that only supports static-sized tensors with a graph that has dyna
mic-sized tensors (tensor#58 is a dynamic-sized tensor).
Test passed for existing email noah.wilson@gmail.com

```

Figure 14:Result of testing registration page

Test report

Test Case 2					
Project Name:FreelanceHub					
Register Test Case					
Test Case ID: Test_2			Test Designed By: Varsha Shaji		
Test Priority(Low/Medium/High):High			Test Designed Date: 25/09/2024		
Module Name: Registration Page			Test Executed By : Ms.Anit James		
Test Title : Verify registration with valid credentials			Test Execution Date: 27/09/2024		
Description: Registration page Testing					
Pre-Condition :User has valid credentials					
Step	Test Step	Test Data	Expected Result	Actual Result	Status(Pass/ Fail)
1	Navigate to register page		Display registration page	Registration page displayed	Pass
2	Valid details are entered	fname:Mary lname:Doe email: mary.doe@gmail.com password:Mary@123 confirm password: Mary@123	User should be able to register	User navigate to the user type selecting page.	Pass
3	Click on register Button				
4	Navigate to add user type page	Display add user type page	Add user type page displayed	Add user type Page displayed	Pass

5	Select user type	user_type:client	User should be able to add user type	User navigate to the user type selecting page.	Pass
6	Click on create account Button				
Post-Condition: Details are validated with database values for successful registration					

Test Case 3

Code

```

from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager
driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()))
try:
    driver.get('http://127.0.0.1:8000/login/')
    email_input = WebDriverWait(driver, 10).until(EC.element_to_be_clickable((By.NAME, 'mail')))
    email_input.send_keys('admin@gmail.com')
    password_input = WebDriverWait(driver, 10).until(EC.element_to_be_clickable((By.NAME,
    'pass'))))
    password_input.send_keys('Admin@12')
    password_input.send_keys(Keys.RETURN)
    WebDriverWait(driver, 10).until(EC.url_contains('/administrator/admin_view/'))
    driver.get('http://127.0.0.1:8000/administrator/add_template/')
    WebDriverWait(driver, 10).until(EC.presence_of_element_located((By.ID, 'project_form')))
    title_input = WebDriverWait(driver, 10).until(EC.element_to_be_clickable((By.ID, 'title')))
    title_input.send_keys('Template2')
    file_input1 = WebDriverWait(driver, 10).until(EC.element_to_be_clickable((By.ID,
    'template_file'))))
    file_input1.send_keys('C:/Users/LENOVO/Downloads/index.html')
    file_input2 = WebDriverWait(driver, 10).until(EC.element_to_be_clickable((By.ID,
    'cover_image'))))
    file_input2.send_keys('C:/Users/LENOVO/Downloads/openstack.pdf')
    driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")
    WebDriverWait(driver, 4).until(EC.presence_of_element_located((By.ID, 'submit_btn')))
    submit_button = driver.find_element(By.ID, 'submit_btn')
    if submit_button.get_attribute('disabled'):

```

```

    print('Test Passed: Submit button is disabled due to an error in form filling.')
else:
    submit_button.click()
WebDriverWait(driver, 10).until(EC.url_contains('/administrator/template_list/'))
if driver.current_url == 'http://127.0.0.1:8000/administrator/template_list/':
    print('Test Passed: Submit button was enabled and redirected successfully.')
else:
    print('Test Failed: Did not redirect to the expected URL.')
except Exception as e:
    print(f"Test Failed: {e}")
finally:
    driver.quit()

```

Screenshot

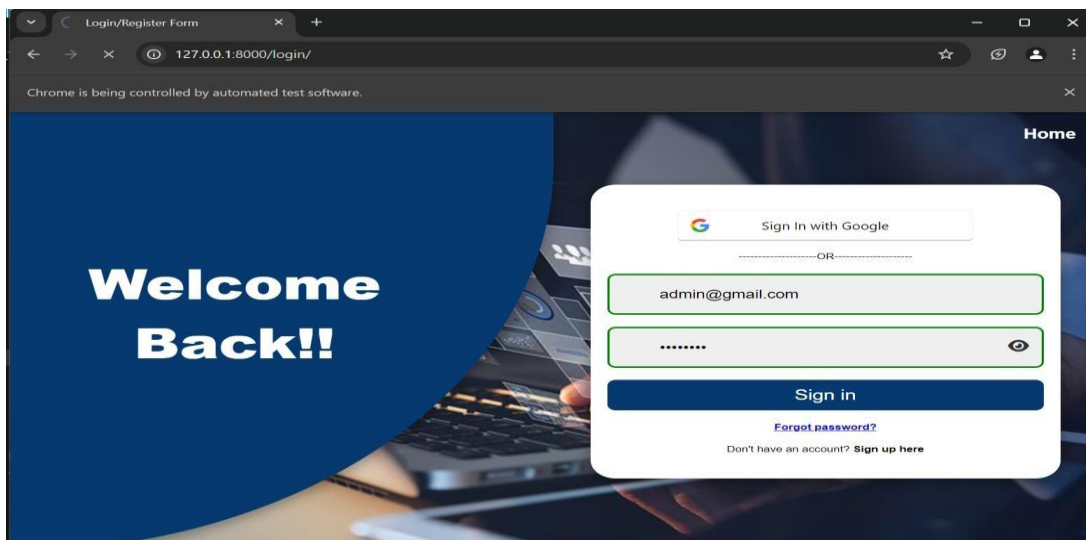


Figure 15:interface of login page with credentials entered

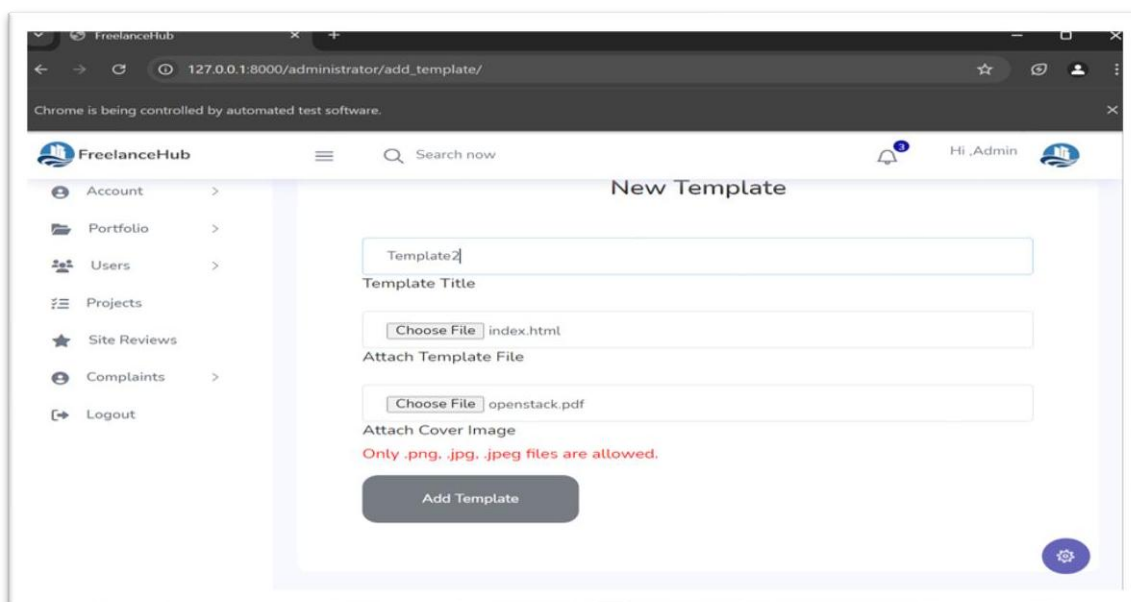


Figure 16:interface of new template adding page with credentials entered

```

PS C:\Users\LENOVO\Desktop\freelancehub> & "C:/Program Files/python/python.exe" c:/Users/LENOVO/Desktop/freelancehub/tests/test_add_template.py

DevTools listening on ws://127.0.0.1:58880/devtools/browser/856a91cf-503f-43af-a0e4-f6b05ef9f013
Logged in successfully.
Test Passed: Submit button is disabled due to an error in form filling.

```

Figure 17:Result of testing add new template page

Test Report

Test Case 3					
Project Name:FreelanceHub					
Portfolio Template Adding Test Case					
Test Case ID: Test_3			Test Designed By: Varsha Shaji		
Test Priority(Low/Medium/High):High			Test Designed Date: 25/09/2024		
Module Name: Portfolio Template Adding Page			Test Executed By: Ms.Anit James		
Test Title: Verify Template adding with invalid credentials			Test Execution Date: 27/09/2024		
Description: Portfolio Template Adding Testing					
Pre-Condition: Providing Invalid credentials					
Step	Test Step	Test Data	Expected Result	Actual Result	Status(Pass/Fail)
1	Navigate to login page		Display login page	Login page displayed	Pass
2	Valid details are entered	email: admin@gmail.com password:Admin@12	User should be able to login	User navigate to the admin dashboard	Pass
3	Click on login Button				
4	Navigate to add portfolio template page	Display add portfolio template page	Display add portfolio template	Add portfolio template displayed	Pass

5	Datas entered	title: Template2 template_file:index.htm cover_image: openstack.pdf	User should be able to add template	User navigate to the add template selecting page.	Pass
6	Click on submit Button				
Post-Condition: Admin cannot add template because of invalid credentials					

Test Case 4

Code

```

from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.support.ui import WebDriverWait, Select
from selenium.webdriver.support import expected_conditions as EC
import time
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager

driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()))
try:
    driver.get('http://127.0.0.1:8000/login/')
    email_input = WebDriverWait(driver, 10).until(EC.element_to_be_clickable((By.NAME, 'mail')))
    email_input.send_keys('tech.innovations@gmail.com')
    password_input = WebDriverWait(driver, 10).until(EC.element_to_be_clickable((By.NAME,
    'pass'))))
    password_input.send_keys('Tech@123')

    password_input.send_keys(Keys.RETURN)
    WebDriverWait(driver, 10).until(EC.url_contains('/client/client_view/'))
    print("Logged in successfully")
    driver.get('http://127.0.0.1:8000/client/add_new_project/')
    WebDriverWait(driver, 10).until(EC.presence_of_element_located((By.ID, 'project_form')))
    title_input = WebDriverWait(driver, 10).until(
        EC.element_to_be_clickable((By.ID, 'title'))
    )
    title_input.send_keys('Test Project')
    description_input = driver.find_element(By.ID, 'description')
    description_input.send_keys('This is a test project description.')
    budget_input = driver.find_element(By.ID, 'budget')
    budget_input.send_keys('5000')

```



```

category_dropdown = Select(driver.find_element(By.ID, 'category'))
category_dropdown.select_by_visible_text('Web Development')
end_date_input = WebDriverWait(driver, 10).until(EC.element_to_be_clickable((By.ID,
    'end_date'))))
end_date_input.clear()
end_date_input.send_keys('20-02-2024')
end_date_input.send_keys(Keys.TAB)
file_input = WebDriverWait(driver, 10).until(EC.element_to_be_clickable((By.ID, 'file'))))
file_input.send_keys('C:/Users/LENOVO/Downloads/openstack.pdf')
scope_dropdown = Select(driver.find_element(By.ID, 'scope'))
scope_dropdown.select_by_visible_text('Medium')
driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")
time.sleep(4)
submit_button = driver.find_element(By.ID, 'addProject')
WebDriverWait(driver, 10).until(EC.element_to_be_clickable((By.ID, 'addProject'))).click()
if driver.current_url == 'http://127.0.0.1:8000/client/add_new_project/':
    print('Test passed!New project not aaded beacause of error in form filling!')
elif driver.current_url == 'http://127.0.0.1:8000/client/project_list/':
    print('Test was successful! New project added')
except Exception as e:
    print(f"Test Failed")
finally:
    time.sleep(3)
    driver.quit()

```

Screenshot

The screenshot shows the 'New Project' form on the FreelanceHub website. The form is titled 'New Project' and contains several input fields and a submit button. The fields are:

- Title:** Test Project
- Description:** This is a test project description.
- Budget:** 5000
- Category of Work:** Web Development
- Proposal Submission End Date:** 20-02-2024 (with a red error message: 'End date must be after today')
- Scope of Project:** Medium
- Attach Additional Files:** Choose File | openstack.pdf

A black 'Add Project' button is located at the bottom of the form. The left sidebar shows navigation options: Dashboard, Account, Project, Repositories, Freelancers, Calendar, Payments, Chat, Complaints, and Logout.

Figure 18:interface of a new project page with credentials

```
PS C:\Users\LENOVO\Desktop\freelancehub> & "C:/Program Files/python/python.exe" c:/Users/LENOVO/Desktop/freelancehub/tests/test_new_project.py

DevTools listening on ws://127.0.0.1:64497/devtools/browser/03b3611d-c125-4270-8ef3-c1bd9d1aaf00
Logged in successfully
Test passed!New project not aaded beacause of error in form filling!
```

Figure 19: Result of testing new project adding page

Test Report

Test Case 4					
Project Name:FreelanceHub					
New Project Adding Test Case					
Test Case ID: Test_4			Test Designed By: Varsha Shaji		
Test Priority(Low/Medium/High):High			Test Designed Date: 25/09/2024		
Module Name: Adding new project Functionality			Test Executed By : Ms.Anit James		
Test Title : client adding project			Test Execution Date: 27/09/2024		
Description:Adding new project Functionality Testing					
Pre-Condition :Client provides Invalid datas					
Step	Test Step	Test Data	Expected Result	Actual Result	Status(Pass/Fail)
1	Navigate to login page		Display login page	Login page displayed	Pass
2	Valid details are entered	email: tech.innovations@gmail.com password: Tech@123	User should be able to login	User navigate to the client dashboard	Pass
3	Click on login Button				
4	project	Display add new project page	Display add new project page	Add new proje displayed	Pass
5	Datas entered	title:Test Project description: 'This is a test proje description' Budget: 5000 Category: Web Development File: openstack.pdf Scope:Medium	User should'nt be able to add project	User doesn't navigate to the project list page.	Pass

		End date:20-02-2024			
6	Click on add project Button				
Post-Condition: Client couldn't able to add new project because of invalid credentials					

Test Case 5

Code

```

from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.support.ui import WebDriverWait, Select
from selenium.webdriver.support import expected_conditions as EC
import time
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager

driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()))
try:
    driver.get('http://127.0.0.1:8000/login/')
    email_input = WebDriverWait(driver, 10).until(
        EC.element_to_be_clickable((By.NAME, 'mail'))
    )
    email_input.send_keys('noah.wilson@gmail.com')
    password_input = WebDriverWait(driver, 10).until(
        EC.element_to_be_clickable((By.NAME, 'pass'))
    )
    password_input.send_keys('Noah@123')
    password_input.send_keys(Keys.RETURN)
    WebDriverWait(driver, 10).until(EC.url_contains('/freelancer/freelancer_view/'))
    print("Logged in successfully. Current URL:", driver.current_url)
    driver.get('http://127.0.0.1:8000/freelancer/template_list/')

    template_card = WebDriverWait(driver, 10).until(
        EC.element_to_be_clickable((By.CLASS_NAME, 'card'))
    )
    use_template_button = template_card.find_element(By.XPATH, "//*[@button[contains(@onclick, 'openModal')]]")
    use_template_button.click()
    upload_modal = WebDriverWait(driver, 10).until(
        EC.visibility_of_element_located((By.ID, 'uploadModal')) )

```

```
file_input = upload_modal.find_element(By.ID, 'file')
file_input.send_keys(r"C:\Users\LENOVO\Downloads\MC_Assignment 2.docx")
submit_button = upload_modal.find_element(By.ID, 'subbtn')

if not submit_button.is_enabled():
    print("Test Case 1 Passed: DOCX correctly rejected.")
else:
    print("Test Case 1 Failed: DOCX rejection failed.")

time.sleep(1)

close_button = upload_modal.find_element(By.ID, 'close')
close_button.click()

WebDriverWait(driver, 10).until(EC.invisibility_of_element_located((By.ID, 'uploadModal'))))

driver.get('http://127.0.0.1:8000/freelancer/template_list/')
template_card = WebDriverWait(driver, 10).until(
    EC.element_to_be_clickable((By.CLASS_NAME, 'card'))
)
use_template_button = template_card.find_element(By.XPATH, "//*[@button[contains(@onclick, 'openModal')]]")
use_template_button.click()
upload_modal = WebDriverWait(driver, 10).until(
    EC.visibility_of_element_located((By.ID, 'uploadModal')) )
file_input = upload_modal.find_element(By.ID, 'file')
file_input.send_keys(r"C:\Users\LENOVO\Downloads\Varsha Shaji_INT MCA_Amal Jyothi College.pdf")
submit_button = upload_modal.find_element(By.ID, 'subbtn')
submit_button.click()

try:
    WebDriverWait(driver,
        10).until(EC.url_matches(r'http://127\.\0\.\0\.\1:8000/freelancer/process_resume/\d+/'))
    print("Test Case 2 Passed: PDF uploaded and portfolio generated successfully.")
except:
    print("Test Case 2 Failed: PDF upload failed.")

time.sleep(2)

except Exception as e:
    print(f"Test Failed: {e}")
finally:
    driver.quit()
```

Screenshot

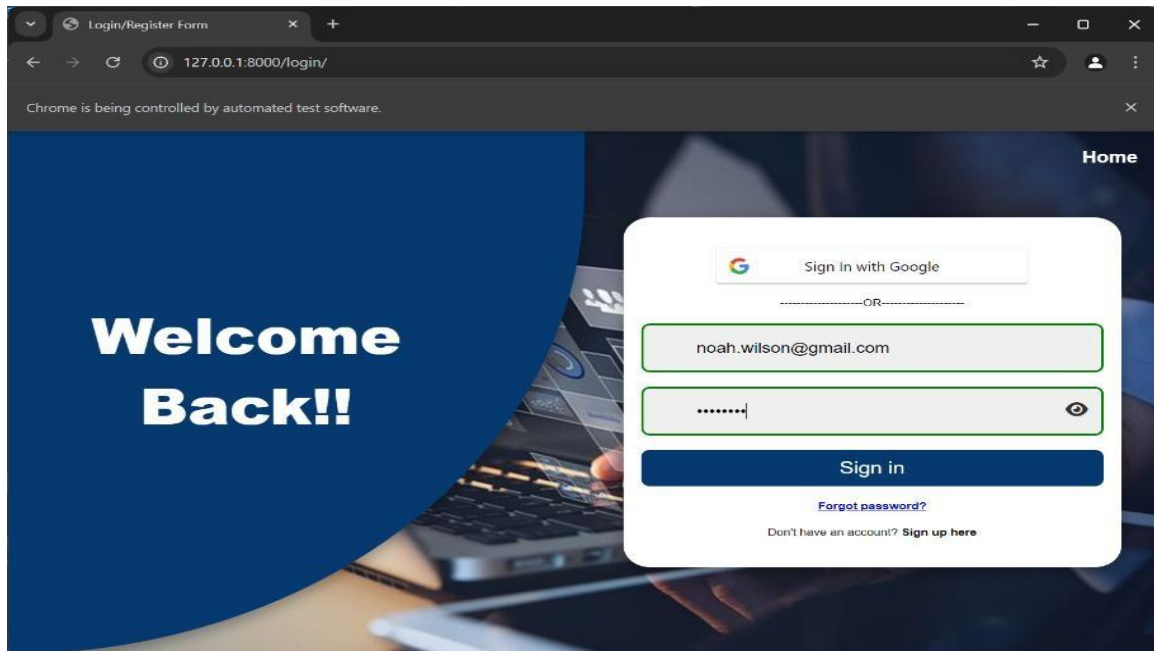


Figure 20:interface of a login page with credentials

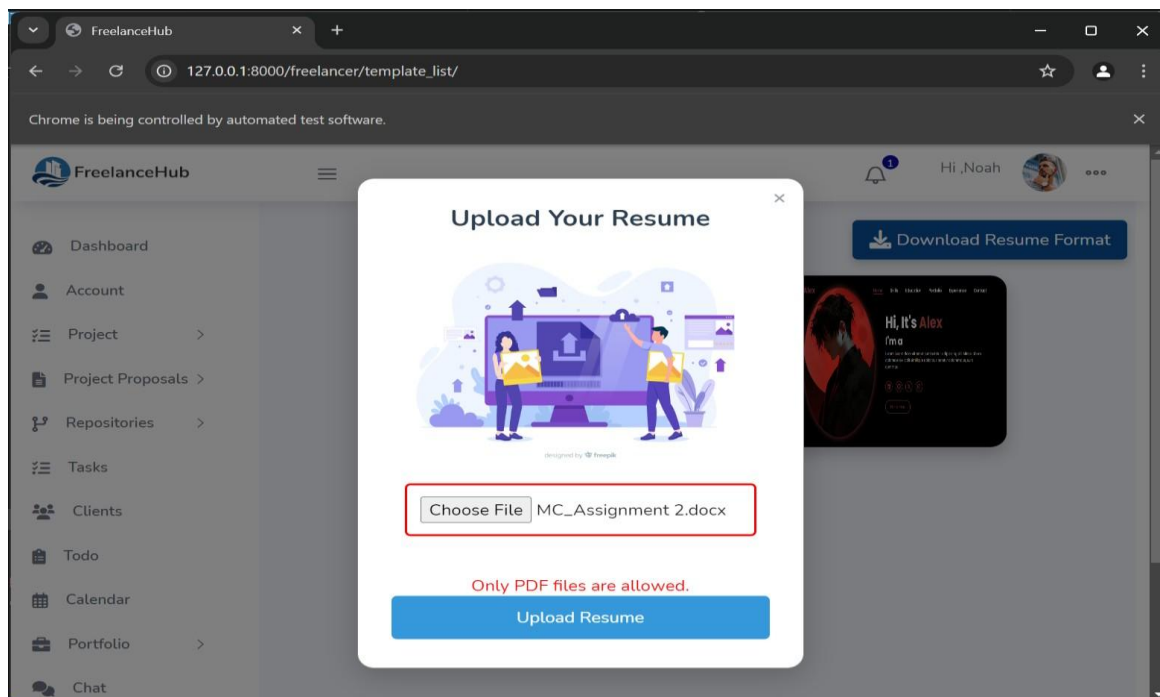


Figure 21:interface of a create portfolio page with credentials

```
PS C:\Users\LENOVO\Desktop\freelancehub> & "C:/Program Files/python/python.exe" c:/Users/LENOVO/Desktop/freelancehub/tests/test_portfolio_file.py

DevTools listening on ws://127.0.0.1:58538/devtools/browser/32b5426f-6d61-40a8-97f0-5f0f4b24e292
Logged in successfully. Current URL: http://127.0.0.1:8000/freelancer/freelancer_view/
Test Case 1 Passed: DOCX correctly rejected.
Created TensorFlow Lite XNNPACK delegate for CPU.
Test Case 2 Passed: PDF uploaded and portfolio generated successfully.
```

Figure 22:Result of testing creating new portfolio

Test Report

Test Case 5					
Project Name:FreelanceHub					
Portfolio Creation Test Case					
Test Case ID: Test_5			Test Designed By: Varsha Shaji		
Test Priority(Low/Medium/High):High			Test Designed Date: 25/09/2024		
Module Name: Adding new project Functionality			Test Executed By : Ms.Anit James		
Test Title : freelancer creating portfolio			Test Execution Date: 27/09/2024		
Description:Creating portfolio Functionality Testing					
Pre-Condition :Freelancer provides Invalid data's					
Step	Test Step	Test Data	Expected Result	Actual Result	Status(Pass/Fail)
1	Navigate to login page		Display login page	Login page displayed	Pass
2	Valid details are entered	email: noah.wilson@gmail.com password: Noah@123	User should be able to login	User navigate to the freelancer dashboard	Pass
3	Click on login Button				
4	Navigate to create portfolio page and selecting a template and modal is opened	Display create portfolio page and selecting a template and modal is opened	Display create portfolio page and selecting a template and modal is opened	create portfolio displayed and selecting a template and modal is opened	Pass
5	Data's Entered	File:MC_Assignment 2.docx	User shouldn't navigate to the process_resume page	User doesn't navigate to the process_resume page	Pass
6	Click on add project Button				
Post-Condition: Freelancer couldn't able to create portfolio site because of invalid credentials					

CHAPTER 6

IMPLEMENTATION

6.1 INTRODUCTION

Implementation in software development is the process of transforming design specifications into a working product. This phase involves writing code, integrating different modules, testing the functionality, and deploying the software. Success relies on effective communication, collaboration, and thorough planning. All stakeholders must be involved throughout to ensure the product meets both functional and business requirements.

During implementation, developers focus on writing maintainable and scalable code while ensuring the project's requirements are met. Rigorous testing helps identify and fix bugs, and the deployment process installs the software on the target platform, ensuring it works as expected. A successful implementation ensures the final product is stable, meets the goals, and can be easily maintained and scaled.

6.2 IMPLEMENTATION PROCEDURES

The following are the typical procedures for implementing software:

Planning and Preparation: Before starting the implementation process, it is essential to plan and prepare thoroughly. This includes developing a detailed implementation plan, identifying project requirements, assigning responsibilities, and ensuring that all necessary resources are available.

- **Development and Testing:** The development team writes and tests code, ensuring it meets functional requirements and integrates all components seamlessly, identifying and fixing bugs along the way.
- **User Acceptance Testing:** End-users test the software to confirm it aligns with their requirements, is user-friendly, and functions as expected in real-world scenarios.
- **Deployment:** After passing tests, the final product is deployed into the production environment, making it available for actual use by all intended users.
- **Post-Deployment Testing:** Testing is conducted after deployment to ensure the software operates correctly in the live environment and continues to meet functional requirements.
- **Maintenance and Support:** Ongoing maintenance ensures the software remains efficient, secure, and adaptable to evolving business needs while addressing any arising issues.
- **Evaluation:** Feedback from users and stakeholders is collected to evaluate the software's performance and identify areas for potential improvements or enhancements.
- **Communication and Collaboration:** Effective communication among developers, stakeholders, and users is essential for ensuring smooth progress and successful implementation of the software.

6.2.1 User Training

User training is an essential part of software implementation, as it enables end-users to use the new software effectively and efficiently. The following are the typical steps involved in user training:

- **Training Needs Assessment:** This initial step involves identifying the training requirements of end-users by determining which software features they will use and the tasks they need to perform.
- **Training Plan Development:** A training plan is created based on the assessment, outlining objectives, training methods, and required materials to guide the training process.
- **Training Delivery:** The training is conducted for end-users using various methods, such as classroom sessions, online courses, or one-on-one training to suit different learning preferences.
- **Training Evaluation:** After training, its effectiveness is assessed by gathering feedback from end-users to determine if training objectives were met and identify areas for improvement.
- **Follow-Up Training:** Additional training sessions are provided based on feedback to fill any knowledge or skill gaps identified during the evaluation process.

Effective user training ensures that end-users can use the new software with confidence, leading to increased productivity and improved business outcomes. It is important to ensure that end-users receive adequate training and support during the implementation phase to ensure a smooth transition to the new software.

6.2.2 Training on the Application Software

Training on application software is crucial for ensuring that end-users can use the software effectively and efficiently. The following are the typical steps involved in training on application software:

- **Orientation:** An overview of the software is provided, highlighting its features, functionalities, and benefits, to generate interest and enthusiasm among end-users.
- **Hands-on Training:** End-users engage in practical training through live demonstrations or by accessing a test environment, allowing them to practice using the software effectively.
- **Role-Based Training:** Training is tailored to different roles and responsibilities, ensuring that each end-user receives relevant instruction aligned with their job functions.
- **Customized Training Materials:** Relevant and comprehensive training materials are developed, making them easy to understand and ensuring they meet the specific needs of end-

users.

- **Evaluation:** The effectiveness of the training is assessed to identify areas for improvement, confirming that end-users have gained the necessary knowledge and skills for software usage.

Effective training on application software ensures that end-users can use the software with confidence, leading to increased productivity and improved business outcomes. It is important to provide adequate training and support to end-users during the implementation phase to ensure a smooth transition to the new software.

6.2.3 System Maintenance

System maintenance of project refers to the ongoing process of ensuring that the system functions optimally and meets the business needs. The following are some of the essential activities involved in system maintenance:

- **Regular Updates:** The system should be updated regularly with bug fixes, security patches, and new features to ensure that it remains efficient and secure.
- **Backup and Recovery:** Regular backups of data should be taken to prevent data loss in case of system failure. A disaster recovery plan should also be in place to ensure business continuity in case of unexpected system failures.
- **Performance Monitoring:** Regular monitoring of the system's performance should be carried out to identify any bottlenecks, potential issues, or areas for improvement.
- **User Support:** A help desk or support team should be available to respond to user queries and issues related to the system.
- **Security Measures:** The system should be protected with appropriate security measures, such as firewalls, antivirus software, and intrusion detection systems.
- **Training and Documentation:** End-users should receive ongoing training and support to ensure that they can use the system effectively. User manuals and documentation should also be regularly updated to reflect any changes or new features in the system.

Regular system maintenance ensures that the project operates smoothly and remains efficient and secure. It also helps to prevent any potential issues or downtime that could impact the business operations.

6.2.4 Hosting

The Django-based FreelanceHub project is hosted on Koyeb, a cloud platform optimized for modern web applications, ensuring scalable and reliable access for freelancers and clients alike.

This cloud hosting setup supports automated deployments, efficient resource scaling, and easy configuration of environment variables to manage the project's unique requirements. FreelanceHub integrates with a MariaDB database hosted on Files.io, providing robust data storage for user profiles, project details, and messaging features. With SSL encryption, regular backups, and continuous monitoring, this hosting solution delivers a secure, high-performance environment, keeping FreelanceHub responsive, accessible, and secure as its user base expands.

Procedure for hosting a website on Koyeb.app:

- Create an account at [Koyeb](https://www.koyeb.com) and log in.
- Click "Create Service" and select GitHub to connect your repository.
- Ensure your Django project runs locally, and create a `requirements.txt` with `pip freeze > requirements.txt`.
- Add a `Procfile` with `web: gunicorn myproject.wsgi --bind 0.0.0.0:\$PORT` (replace `myproject` with your Django project name).
- Set `ALLOWED_HOSTS` in `settings.py` to `["*"]` or the Koyeb domain, and set `DEBUG = False` and configure your database credentials to connect to the hosted MariaDB database on `files.io`.
- Configure static files in `settings.py` using WhiteNoise or an external storage service.
- Click Deploy on Koyeb and monitor the logs for any issues.
- Once deployed, use the provided Koyeb URL to access your live Django application.
- Push updates to your GitHub repository to automatically redeploy on Koyeb.

Hosted Website: Railwayapp, Koyeb.app, render.com

Hosted Link:

- <https://freelancehub-production.up.railway.app/>
- <https://freelancehub.koyeb.app/>
- <https://freelancehub-5tya.onrender.com/>

Hosted Link QR Code:



Screenshot

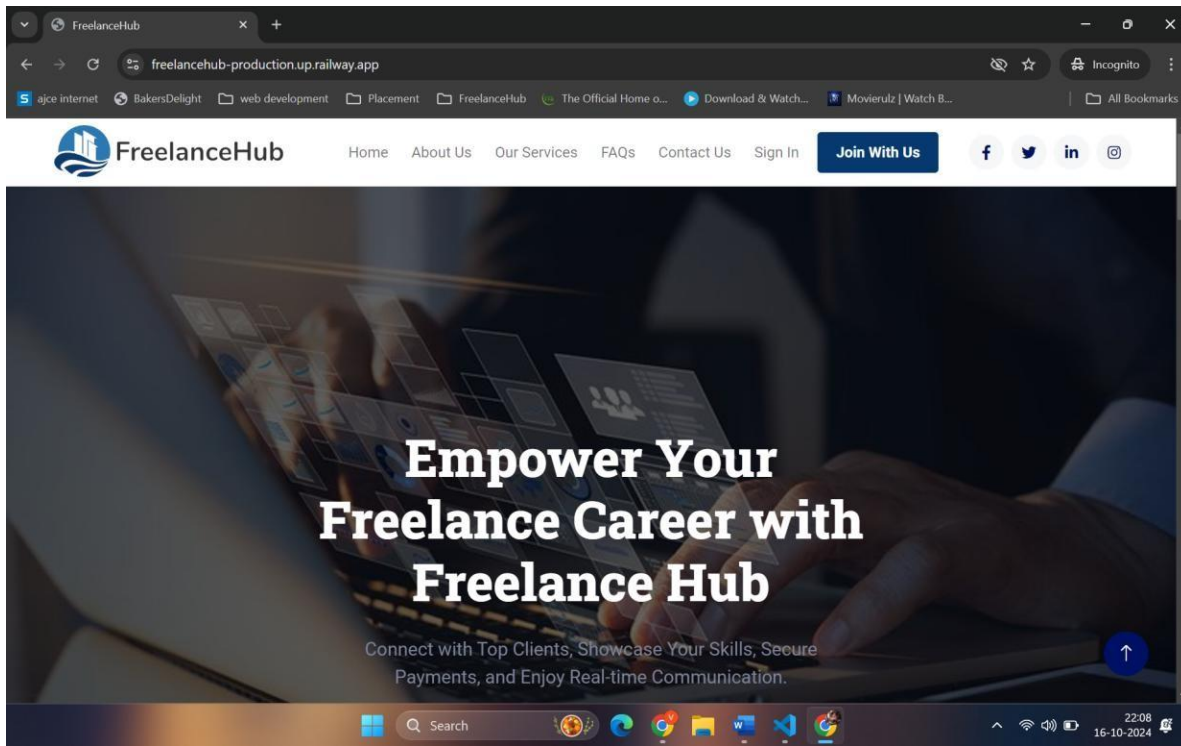


Figure 20:interface of index page after hosting

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

In conclusion, FreelanceHub is a user-friendly Online Freelance Job Management System designed to enhance collaboration between clients and freelancers. The platform streamlines essential functions such as project posting, proposal management, and real-time communication through an integrated chat system. A comprehensive feedback and rating system fosters accountability, allowing clients and freelancers to evaluate each other's performance and build trust within the ecosystem. By adopting Scrum project management principles, FreelanceHub ensures projects are completed on time and align with client expectations. After freelancers submit their proposals, an agreement is generated that includes all project details, including payment installments, which must be signed by both parties. Clients can also lock submitted proposals, preventing freelancers from making further edits, thus enhancing transparency and establishing clear expectations.

An exciting feature of FreelanceHub is that freelancers can upload their resumes, which are automatically converted into personalized portfolio websites. This allows freelancers to showcase their skills and experience effectively, attracting potential clients. The platform is built using Django for the back-end, with MariaDB as the database management system, ensuring secure storage and management of user data. For payments, Razorpay is integrated to facilitate secure financial transactions. Real-time updates and communications are powered by AJAX and WebSockets, providing a seamless user experience. By leveraging these technologies and methodologies, FreelanceHub offers a reliable and effective solution for managing freelance projects, ultimately ensuring successful outcomes for both clients and freelancers.

7.2 FUTURE SCOPE

In the future, FreelanceHub plans to introduce premium features like enhanced job visibility, certification badges, and customized alerts. An AI-powered chatbot will assist users, while team collaboration tools will enable freelancers to work together seamlessly. Additionally, the platform will offer User Training and Resources, including courses and webinars, to help users enhance their skills and effectively navigate the platform. These enhancements will strengthen FreelanceHub, fostering a thriving freelance community and ensuring successful project outcomes.

CHAPTER 8

BIBLIOGRAPHY

REFERENCES:

- [1] Ahmad, R. et al. (2020). "Named Entity Recognition using CNN and LSTM for Resume Parsing." *International Journal of Advanced Computer Science and Applications*, 11(2), 52-60.
- [2] Siddiqui, M. & Jain, R. (2019). "Hybrid Approach for Resume Information Extraction using NLP and Rule-based Parsing." *Journal of Intelligent Information Systems*, 54(3), 112-122.
- [3] Kumar, S. et al. (2018). "Using SVM and Decision Trees for Automated Resume Classification." *Applied Artificial Intelligence*, 32(7), 543-562.
- [4] Gupta, N. et al. (2019). "Building Portfolio Websites from Parsed Resumes Using Django Framework." *Proceedings of the ACM International Conference on Web Technology*, 34(1), 25-32.
- [5] Liu, Y. & Wang, Z. (2017). "AI-Powered Resume Screening: A Comparative Analysis of SVM and Random Forests." *IEEE Transactions on Systems, Man, and Cybernetics*, 47(9), 1965-1973.
- [6] Shen, W. et al. (2020). "Unstructured Resume Parsing Challenges: Solutions Using NLP and Heuristics." *Journal of Natural Language Processing*, 46(1), 81-92.
- [7] Zhou, J. & Li, X. (2018). "Scalable Cloud-Based Framework for Resume Processing: An Experimental Study." *Journal of Cloud Computing*, 8(1), 54-67.
- [8] Ahmed, A. & Kapoor, P. (2020). "Integrating Resume Parsing with Interactive Portfolio Websites." *Computational Intelligence*, 36(2), 184-194.
- [9] Patel, R. et al. (2021). "A Study on Large-scale Resume Parsing using SVM and LSTM." *International Journal of Machine Learning and Computing*, 10(4), 435-440.
- [10] Ali, M. & Shah, R. (2017). "Efficient Resume Screening using AI and NLP Techniques." *Expert Systems with Applications*, 79(1), 334-345.
- [11] Django for Beginners, William S. Vincent, 2018.
- [12] MariaDB Essentials: Building a High-Performance Database, Emilien Gagnon, 2017.
- [13] System Design Interview: An Insider's Guide, Lewis C. Lin, 2017.
- [14] The Unified Modeling Language User Guide, Grady Booch, James Rumbaugh, Ivar Jacobson, 1999.

-
- [15] Use Case Driven Object Modeling with UML: A Practical Approach, Doug Rosenberg, 1999.
- [16] Learning UML 2.0, Russ Miles, Kim Hamilton, 2006.
- [17] Business Process Modeling with UML, Michael K. L. St. John, 2008.
- [18] Object-Oriented Modeling and Design with UML, Michael Blaha, James Rumbaugh, 2005.
- [19] UML 2 and the Unified Process: Practical Object-Oriented Analysis and Design, Jim Arlow, Ila Neustadt, 2004.
- [20] UML Distilled: A Brief Guide to the Standard Object Modeling Language, Martin Fowler, 2003.

WEBSITES:

- <https://www.upwork.com/>
- <https://www.freelancer.in/>
- <https://www.toptal.com/freelance-jobs>
- <https://www.guru.com/>
- <https://www.tutorialspoint.com/django/index.htm>
- <https://www.w3schools.com/django/>
- <https://www.servicescape.com/>
- <https://www.w3schools.com/django/>

CHAPTER 9

APPENDIX

9.1 SAMPLE CODE

Login and registration

Login.html

```
{% load static% }
{% load socialaccount %}
{% providers_media_js %}
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Login/Register Form</title>
</head>
<body>
  <div id="container" class="container">
    <div class="row">
      <a href="{% url 'index' %}" class="home-link">
        Home
      </a>
      <div class="col align-items-center flex-col sign-up">
        <div class="form-wrapper align-items-center">
          <div class="form sign-up">
            <br>
            <a href="{% provider_login_url 'google' %}" class="btn btn-google">Sign Up with
            Google</a> <br>
            <p>.....OR.....</p>
            <form action="{% url 'register' %}" id="register" method="post">

              {% csrf_token %}
              <div class="input-group">
                <i class="bx bxs-user"></i>
                <input type="text" placeholder="First Name" id="fname" name="fname">
                <br><span id="error_fna" class="error"></span>
              </div>
              <div class="input-group">
                <i class="bx bxs-user"></i>
                <input type="text" placeholder="Last Name" id="lname" name="lname">
                <br><span id="error_lna" class="error"></span>
              </div>
              <div class="input-group">
                <i class="bx bx-mail-send"></i>
                <input type="email" placeholder="Email" id="email" name="email">
                <br><span id="error_em" class="error"></span>
              </div>
              <div class="input-group">
                <i class="bx bxs-lock-alt"></i>

                <input type="password" placeholder="Password" id="pass" name="password"
                  class="password">
                <span class="password-toggle-icon"><i class="fas fa-eye"></i></span>
              </div>
            </form>
          </div>
        </div>
      </div>
    </div>
  </div>
</body>
</html>
```

```

        <span id="error_pass" class="error"></span>
    </div>
    <button id="subbtn">
        Sign up
    </button>
    <br><span id="reg_err" class="error"></span>
</form>
<p>
    <span>
        Already have an account?
    </span>
    <b onclick="toggle()" class="pointer">
        Sign in here
    </b>
</p>
</div>
</div>
<div class="col align-items-center flex-col sign-in">
    <div class="form-wrapper align-items-center">

        <div class="form sign-in">
            <br>
            <a href="{% provider_login_url 'google' %}" class="btn btn-google">Sign In with
                Google</a><br>
            <p>.....OR.....</p>
            <form action="{% url 'login' %}" id="login" method="post">
                {% csrf_token %}
                <div class="input-group">
                    <i class="bx bxs-user"></i>
                    <input type="email" placeholder="Email" id="mail" name="mail">
                    <br><span id="error_ma" class="error"></span>
                </div>
                <div class="input-group">
                    <i class="bx bxs-lock-alt"></i>
                    <input type="password" placeholder="Password" id="pas" class="password"
                        name="pass">
                    <span class="password-toggle-icon"><i class="fas fa-eye"></i></span>
                    <br><span id="error_pswd" class="error"></span>
                </div>
                <button name="submit" class="sub" id="sub">
                    Sign in
                </button>
                <br><span id="log_err" class="error"></span></td>
            </form>
            <p>
                <b>
                    <a href="{% url 'send_forget_password_mail' %}" class="btn btn-primary">
                        Forgot password?</a></b>
                </p>
                <p>
                    <span>
                        Don't have an account?
                    </span>
                    <b onclick="toggle()" class="pointer">
                        Sign up here

```

```

        </b>
    </p>
</div>
</div>
<div class="form-wrapper">
</div>
</div>

</div>

<div class="row content-row">
    <div class="col align-items-center flex-col">
        <div class="text sign-in">

            <h2 style="margin-left:-90px;margin-top:-20px;">
                Welcome Back!!
            </h2>
        </div>
        <div class="img sign-in">

        </div>
    </div>
    <div class="col align-items-center flex-col">
        <div class="img sign-up">

        </div>
        <div class="text sign-up">
            <h2>
                Join with us
            </h2>
        </div>
    </div>
</div>
</div>
</body>
</html>

```

Views.py

```

def login(request):
    if request.method == 'POST':
        email = request.POST.get('mail')
        password = request.POST.get('pass')

        user = authenticate(request, email=email, password=password)
        if user is not None:
            auth_login(request, user, backend='django.contrib.auth.backends.ModelBackend')
            if not user.welcome_email_sent:
                user.welcome_email_sent = True
                user.save()

            send_welcome_email(request, user)
            return redirect_based_on_user_type(request, user)
        else:
            return render(request, 'login.html', {'error_msg': 'Invalid credentials', 'page': 'sign-in'})

```

```
elif request.user.is_authenticated:
    user = request.user
    user.email_verified = True
    user.google = True
    user.save()

    if not user.welcome_email_sent:
        user.welcome_email_sent = True
        user.save()

    #send_welcome_email(request, user)

    return redirect_based_on_user_type(request, user)

url_parts = urlparse(request.get_full_path())
query = parse_qs(url_parts.query)
if 'next' in query:
    query.pop('next')
    url_parts = url_parts._replace(query=urlencode(query, doseq=True))
    clean_url = urlunparse(url_parts)
    return HttpResponseRedirect(clean_url)

return render(request, 'login.html', {'page': 'sign-in'})

def redirect_based_on_user_type(request, user):
    user.backend = 'django.contrib.auth.backends.ModelBackend'
    print(f"Redirecting user: {user}")

    existing_entry = Register.objects.filter(user_id=user.id).first()
    if not existing_entry:
        user2 = Register(user_id=user.id)
        user2.save()
        print(f"Created Register entry for user: {user}")

    print(f"User role: {user.role}")
    if not user.role:
        print(f"User role not set, redirecting to add_user_type for user: {user}")
        return add_user_type(request, user.id)
    if user.status != 'active':

        print(f"User status is not active, logging out user: {user}")
        return redirect('logout')
    if user.role == 'admin':
        auth_login(request, user)
        request.session['uid'] = user.id
        print(f"Redirecting admin user: {user}")
        return redirect('administrator:admin_view')
    elif user.role == 'client':
        auth_login(request, user)
        request.session['uid'] = user.id
        print(f"Redirecting client user: {user}")
        return redirect('client:client_view')
    elif user.role == 'freelancer':
        auth_login(request, user)
```

```

    request.session['uid'] = user.id
    print(f"Redirecting freelancer user: {user}")
    return redirect('freelancer:freelancer_view')
else:
    print(f"User role is undefined, redirecting to login for user: {user}")
    return redirect('login')

def register(request):
    if request.method == 'POST':
        fname = request.POST.get('fname')
        lname = request.POST.get('lname')
        email = request.POST.get('email')
        password = request.POST.get('password')

        if CustomUser.objects.filter(username=fname).exists() or
CustomUser.objects.filter(email=email).exists():
            return render(request, 'login.html', {'page':'sign-up','error_msg': 'User with this email already
exists'})
        else:
            user_type="
            user = CustomUser.objects.create_user(username=fname,email=email,
password=password,role=user_type)
            user.role = user_type
            user.save()

            uid=user.id
            reg=Register.objects.create(user_id=uid,first_name=fname,last_name=lname)
            reg.save()
            return redirect('add_user_type', uid=user.id)

    else:
        return redirect(register_view)

```

Portfolio Template adding from admin side

```

{% extends 'Admin/base.html' %}
{% block 'admin_content' %}
{% load static %}

<div class="row" style="background-color:white;padding:10px;border-radius:10px;">
  <div class="tab-content" id="myTabContent"
    style="background-color:white;padding:30px;border-radius:10px;border:none;">
    <h3 align="center">New Template</h3>
    <br>
    <div class="tab-pane active" id="update" role="tabpanel" aria-labelledby="update-tab">

      <div class="card">
        <div class="card-body">
          <form method="post" action="{% url 'administrator:add_template' %}"
enctype="multipart/form-data" id="project_form">
            {% csrf_token %}
            <div class="row gy-3">
              <div class="col-12 col-md-12">
                <div class="form-floating mb-3">

```

```

        <input type="text" class="form-control" id="title" name="template_name"
placeholder="Title" required>
        <label for="title" class="form-label">Template Title</label>
        <br><span id="error_title" class="error" style="color:red;"></span>
    </div>
</div>
<div class="col-12">
    <div class="form-floating mb-3">
        <input type="file" class="form-control" accept=".html" id="template_file"
name="template_file" required>
        <label for="template_file" class="form-label">Attach Template File</label>
        <br><span id="error_template_file" class="error" style="color:red;"></span>
    </div>
</div>

<div class="col-12">
    <div class="form-floating mb-3">
        <input type="file" class="form-control" id="cover_image" name="cover_image" required>
        <label for="cover_image" class="form-label">Attach Cover Image</label>
        <br><span id="error_cover_image" class="error" style="color:red;"></span>
    </div>
</div>

<div class="col-12">
    <div class="d-grid">
        <button type="submit" class="btn btn-dark btn-lg" id="submit_btn">Add
Template</button>
    </div>
</div>
</div>
<span id="project_err" class="error" style="color:red;"></span>
</form>
</div>
</div></div>
</div>
{% endblock %}

```

Views.py

```

from django.shortcuts import render, redirect
from .models import Template

```

```

def add_template(request):
    if request.method == 'POST':
        name = request.POST['template_name']
        file = request.FILES['template_file']
        file2 = request.FILES['cover_image']
        Template.objects.create(name=name, file=file, cover_image=file2)
        return redirect('administrator:template_list')

    return render(request, 'Admin/AddTemplate.html')

```


9.2 SCREEN SHOTS

Landing page

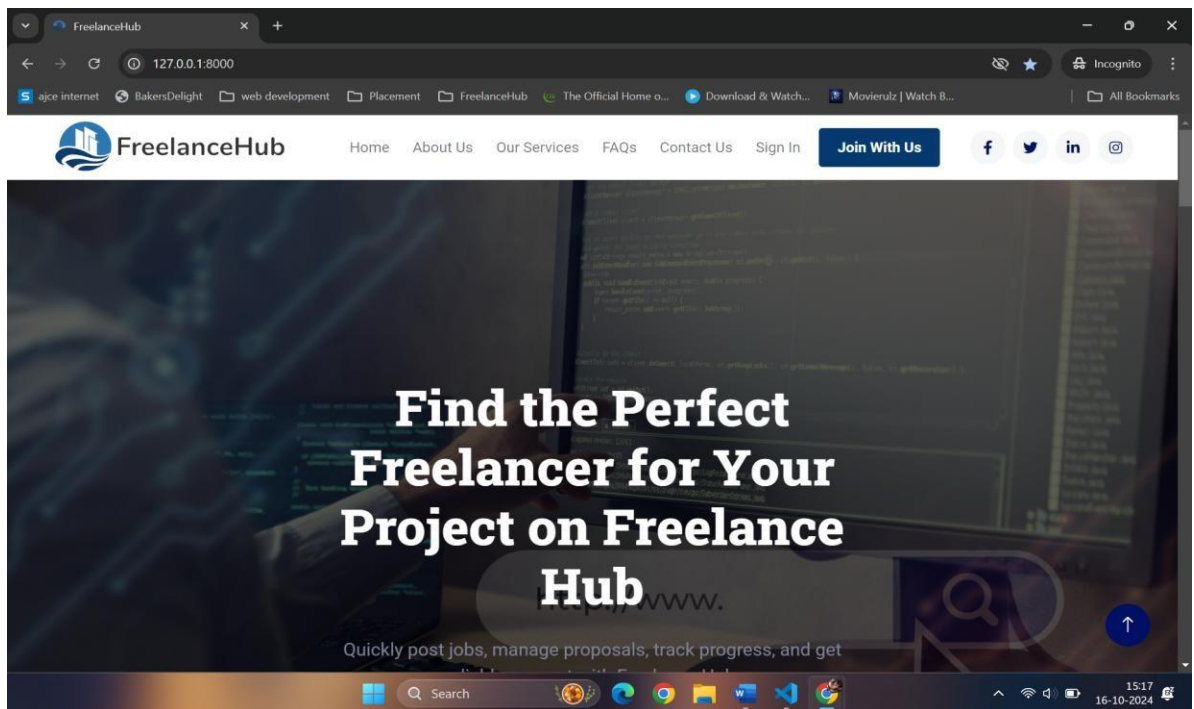


Figure 21:index page

Freelancer Dashboard

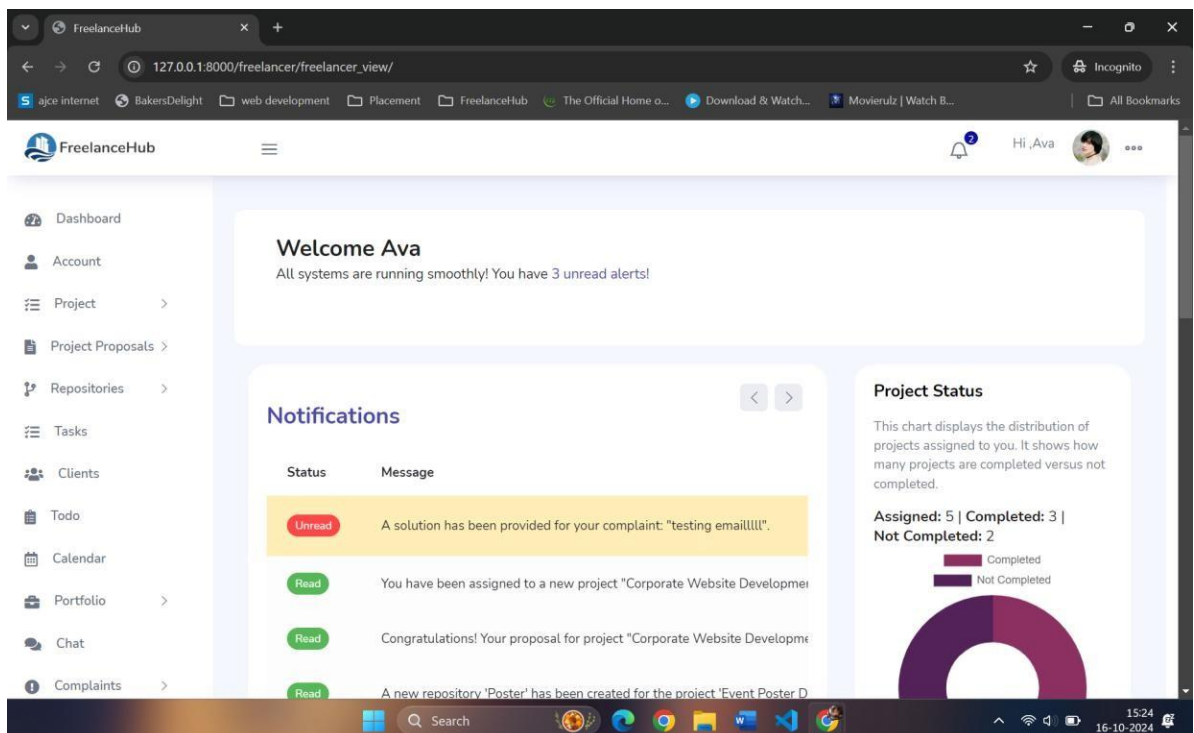


Figure 22:dashboard of freelancer

Project Listing in freelancer Dashboard

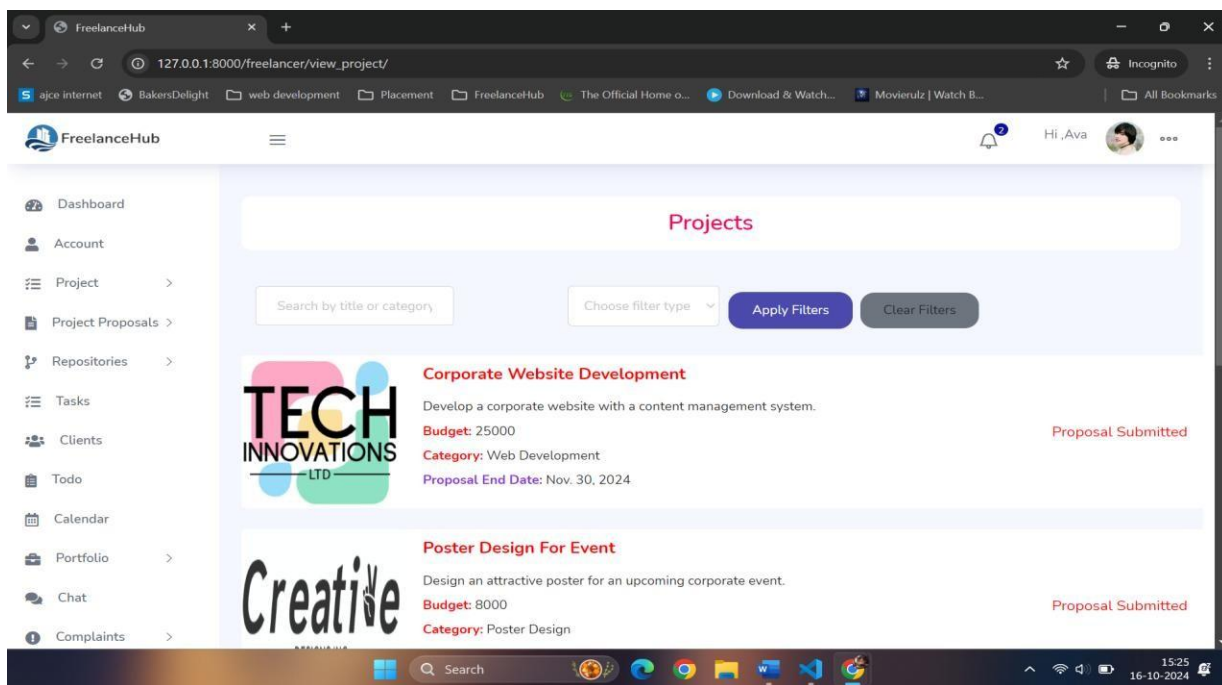


Figure 23: Project Listing in freelancer Dashboard

Client Dashboard

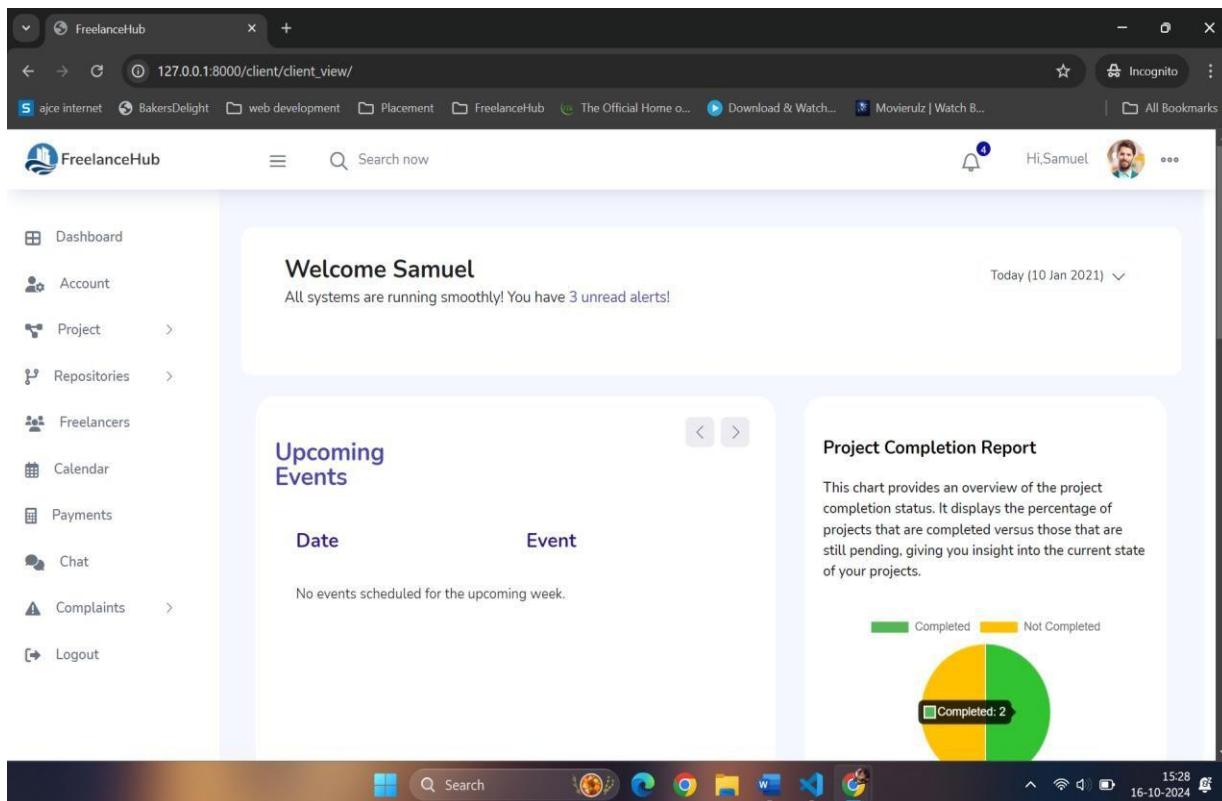


Figure 24:Dashboard of client

Project list in client side

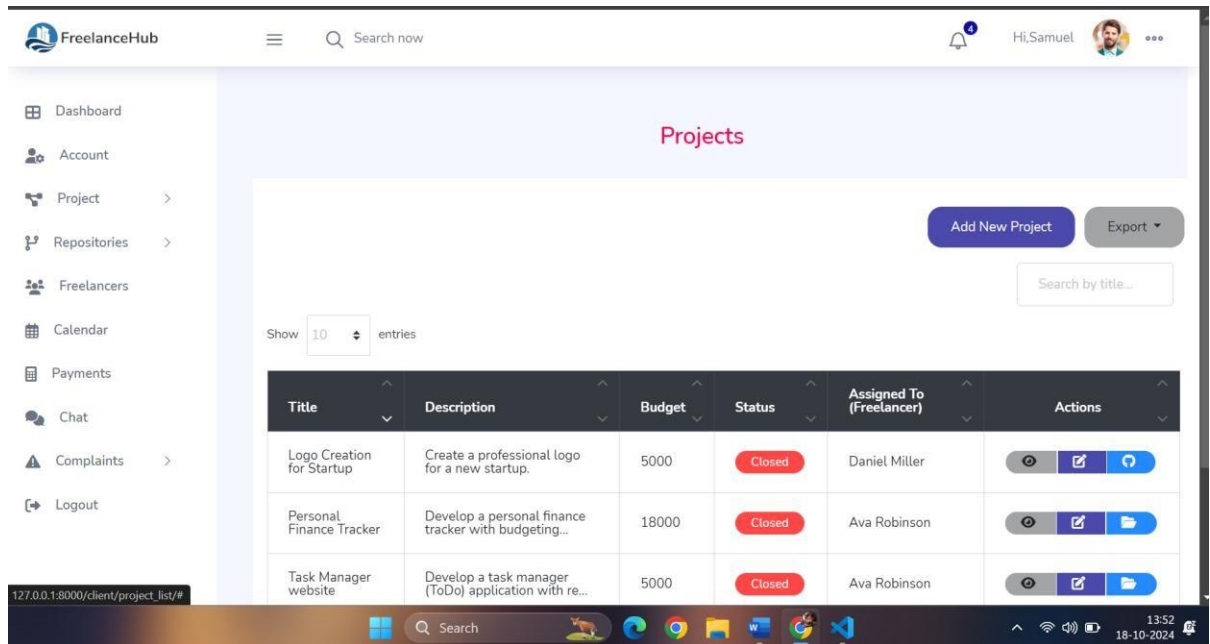


Figure 25:Project list page in client side

Admin Dashboard

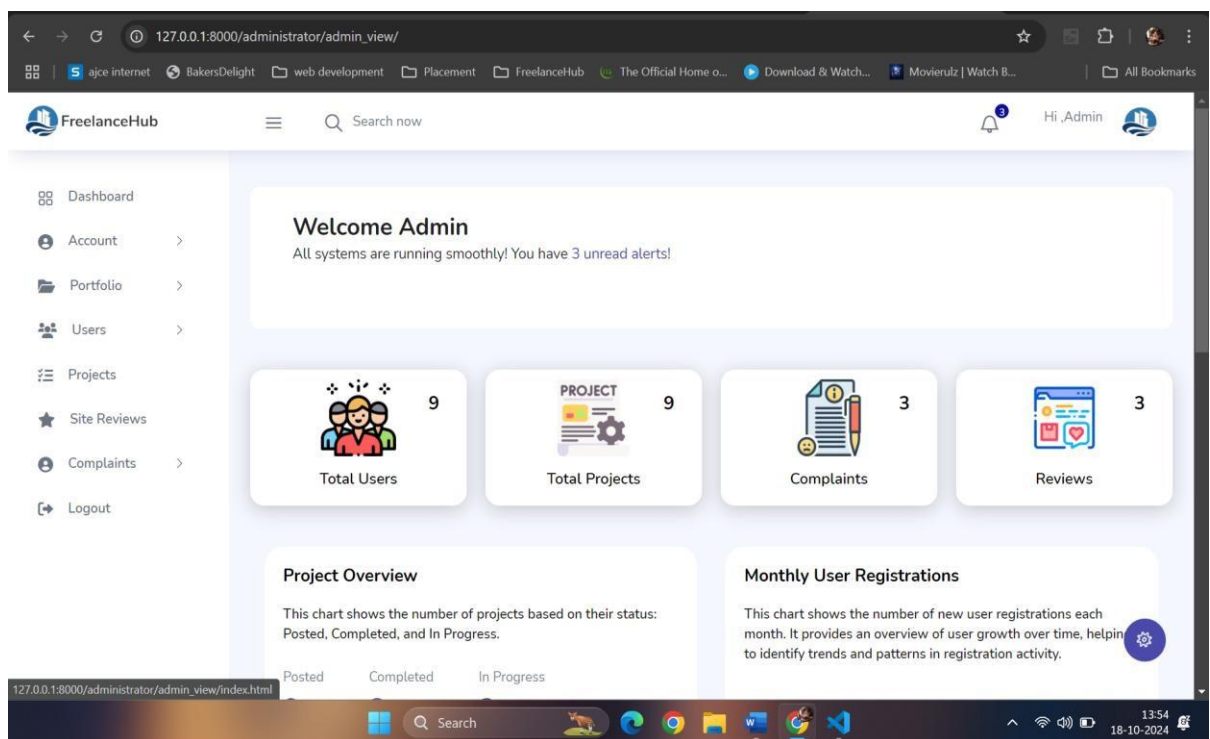


Figure 26:Admin Dashboard

9.3 GIT LOG

varshashji3 / FreelanceHub

<> Code

Issues

Pull requests

Actions

Projects

Wiki

Security 1

Insights

Commits

main

All users

All time

Commits on Aug 5, 2024

Add files via upload

Verified86ad201

varshashji3 authored on Aug 5

Add files via upload

Verified0d041d5

varshashji3 authored on Aug 5

Commits on Jul 26, 2024

Add files via upload

Verified0afb861

varshashji3 authored on Jul 26

Add files via upload

Verifiedc344139

varshashji3 authored on Jul 26

Commits on Jun 20, 2024

2nd commit

Verifiedd5d373f

varshashji3 authored on Jun 20

Commits on Jun 15, 2024



Abstract






Verifiedf8d1ffd









varshashji3 authored on Jun 15

Amal Jyothi College of Engineering Autonomous, Kanjirappally

Department of Computer Applications

 varshashaji3 / FreelanceHub



 Code  Issues  Pull requests  Actions  Projects  Wiki  Security 1  Insights





Commits





 main ▾





 All users ▾





 All time ▾





🔗 Commits on Nov 5, 2024

Project Done
 0832e97  <> 
 varshashaji3 authored 2 days ago




project done
 b284a95  <> 
 varshashaji3 authored 2 days ago

project done
 819d340  <> 
 varshashaji3 authored 2 days ago



project done
 a7e3e88  <> 
 varshashaji3 authored 2 days ago























project done
 52b6127  <> 
 varshashaji3 authored 2 days ago

🔗 Commits on Oct 8, 2024

invoice generation done
824201a  <> 
 varshashaji3 committed on Oct 8 · ❌ 0 / 1

🔗 Commits on Oct 4, 2024

new selenium tests,cover_image for portfolio done
50b7222  <> 

	 varshashaji3 committed on Oct 4 ·  0 / 1
🔗	Commits on Oct 3, 2024
	<div>refunding done 3111380   ...  varshashaji3 committed on Oct 3</div>
🔗	Commits on Sep 29, 2024
	<div>modifications done,testing done d4d1d65   ...  varshashaji3 committed on Sep 29</div>
🔗	Commits on Sep 23, 2024
	<div>NLP implementation done 7c77d5c   ...  varshashaji3 committed on Sep 23</div>
🔗	Commits on Sep 17, 2024
	<div>chat ,complaints done 7437dc3   ...  varshashaji3 committed on Sep 17</div>
🔗	Commits on Sep 6, 2024
	<div>chat ,complaints done 3d1e7d6   ...  varshashaji3 committed on Sep 6</div>
🔗	Commits on Sep 2, 2024
	<div>userreview and sitereview done 5f55564   ...  varshashaji3 committed on Sep 2</div>
🔗	Commits on Aug 26, 2024
	<div>upto payment 5bc9aa2   ...</div>

 varshashaji3 committed on Aug 26

Commits on Aug 13, 2024

new commit

f6fdb7b  

...

 varshashaji3 committed on Aug 13

Commits on Aug 11, 2024

calendar and notifications done

9e1c774  

...

 varshashaji3 committed on Aug 11

upto proposal approval

2f20c42  

...

 varshashaji3 committed on Aug 11

Delete manage.py

Verified

ce79628  

...

 varshashaji3 authored on Aug 11

Delete templates directory

Verified


ebc9f5c  

...

 varshashaji3 authored on Aug 11

Delete static directory

Verified

6ecbb96  

...

 varshashaji3 authored on Aug 11

Delete media directory

Verified

9be4176  

...

 varshashaji3 authored on Aug 11

Delete freelancer directory

Verified

9bc400f  

...


 varshashaji3 authored on Aug 11

Delete core directory

Verified

e7fc562  

...

 varshashaji3 authored on Aug 11

<div>Delete client directory</div> <div>Verified272bde5<div><div></div><></div></div> <div><div></div> varshashaji3 authored on Aug 11</div>	...
<div>Delete administrator directory</div> <div>Verified728bfa7<div><div></div><></div></div> <div><div></div> varshashaji3 authored on Aug 11</div>	...
<div>Delete FreelanceHub directory</div> <div>Verifiedde60cb0<div><div></div><></div></div> <div><div></div> varshashaji3 authored on Aug 11</div>	...

Commits on Aug 5, 2024

<div>Delete static/hi</div> <div>Verifiedc7bd87a<div><div></div><></div></div> <div><div></div> varshashaji3 authored on Aug 5</div>	...
<div>Add files via upload</div> <div>Verified8a9aa95<div><div></div><></div></div> <div><div></div> varshashaji3 authored on Aug 5</div>	...
<div>Create hi</div> <div>Verified349363a<div><div></div><></div></div> <div><div></div> varshashaji3 authored on Aug 5</div>	...
<div>Delete static</div> <div>Verified35aa55e<div><div></div><></div></div> <div><div></div> varshashaji3 authored on Aug 5</div>	...
<div>Create static</div> <div>Verified8da5c75<div><div></div><></div></div> <div><div></div> varshashaji3 authored on Aug 5</div>	...
<div>Add files via upload</div> <div>Verified00cd6cf<div><div></div><></div></div> <div><div></div> varshashaji3 authored on Aug 5</div>	...
<div>Add files via upload</div> <div>Verified2ff5141<div><div></div><></div></div> <div></div>	...
<div></div> <div><div></div> varshashaji3 authored on Aug 5</div>	
<div>Add files via upload</div> <div>Verifiedda57e94<div><div></div><></div></div> <div><div></div> varshashaji3 authored on Aug 5</div>	...
<div>Add files via upload</div> <div>Verified3ab728c<div><div></div><></div></div> <div><div></div> varshashaji3 authored on Aug 5</div>	...