

# **GIG ECONOMY SYSTEM**

**BY**

**Mr. Prasiddha Gurung**

**7-2-282-75-2021 | 13392/21**

**021BIM029**

St. Xavier's College, Maitighar

*A Project Report Submitted to*

**Faculty of Management, Tribhuvan University**

in partial fulfillment of the requirements for the degree of

**Bachelor of Information Management (BIM)**

Kathmandu

June/2025

## STUDENT DECLARATION

This is to certify that I have completed the Project entitled “*Gig Economy System*” under the guidance of “**Mr. Ganesh Dhami**” in partial fulfillment of the requirements for the degree of Bachelor of Information Management at Faculty of Management, Tribhuvan University. This is my original work and I have not submitted it earlier elsewhere.

Date:

Signature:

Name: Mr. Prasiddha Gurung

## CERTIFICATE FROM THE SUPERVISOR

This is to certify that the project entitled “*Gig Economy System*” is an academic work done by “**Mr. Prasiddha Gurung (13392/21)**” submitted in the partial fulfillment of the requirements for the degree of Bachelor of Information Management at Faculty of Management, Tribhuvan University under my guidance and supervision. To the best of my knowledge, the information presented by him in the project report has not been submitted earlier.

---

Signature of the Supervisor

Name: Mr. Ganesh Dhimi

Designation Date

## APPROVAL SHEET

This is to certify that the project titled ***Gig Economy System*** submitted by **Mr. Prasiddha Gurung (13392/21)** has been examined and approved. In our opinion, it meets the required scope and quality standards for a project submitted in partial fulfillment of the requirements for the degree of Bachelor of Information Management (BIM).

Approval Panel:

S.No.	Name	Designation	Signature
1	Mr. Ganesh Dharni	Project Supervisor	
2	Mr. Ganesh Yogi	Head of Department	
3	Mr. Sanjay Kumar Yadav	Internal Examiner	
4		External Examiner	
5		External Examiner	

Date of Defense:

Department: Department of Computer Science

Faculty:

## ACKNOWLEDGEMENTS

I am profoundly grateful to my supervisor, **Mr. Ganesh Dhami**, for his invaluable guidance, insightful feedback, and constant encouragement throughout the entire duration of this project. His expertise and thoughtful suggestions have played a crucial role in shaping and refining this work to its present form.

I would like to express my sincere gratitude to **Mr. Ganesh Yogi**, Head of the Department, for his continuous support and encouragement. His guidance has significantly contributed to both my academic growth and the successful completion of this project.

My heartfelt thanks go to **Tribhuvan University** for providing this invaluable platform to explore practical aspects of real-world problems, helping us bridge the gap between academic knowledge and practical application.

I extend my special appreciation to the **lab technician**, whose technical assistance and availability have been extremely helpful during the implementation and testing phases of this project.

I am also deeply thankful to all the **faculty members** of the department for their valuable insights, encouragement, and support throughout my academic journey.

A special note of thanks to my **friends**, whose collaboration, feedback, and motivation have been a great source of strength.

Finally, I would like to express my deepest gratitude to my **family** for their unwavering love, patience, and emotional support. Their faith in me has been a constant source of inspiration and motivation.

## ABSTRACT

The rise of the gig economy has transformed how people work, offering flexible, short-term job opportunities through digital platforms. Yet, many existing systems struggle with overly complex user interfaces, unreliable payment methods, and ineffective dispute resolution. This project introduces a web-based platform specifically designed to connect freelancers and clients in a more secure, accessible, and efficient way. Developed using technologies like **Django**, **SQLite**, **HTML**, **CSS**, and **Javascript**, the system also incorporates wallet system for handling online payments safely. Key features include smooth job posting and bidding, streamlined contract handling, user-friendly dashboards, and a dedicated conflict resolution process. The system design is supported by comprehensive object-oriented models, such as use case and sequence diagrams, to outline both functionality and structure. The platform was built following the **Agile development approach**, allowing for iterative updates based on feedback and continuous testing. While the current version targets web users, the architecture supports future expansion to mobile devices and intelligent job-matching tools. Overall, this project showcases how modern development tools and thoughtful design can help create a more trustworthy and inclusive environment for gig-based work.

*Keywords:* gig economy system, freelancers, clients, web-based platform, dispute resolution, digital platform

# Table of Contents

<i>STUDENT DECLARATION</i> .....	<i>ii</i>
<i>CERTIFICATE FROM THE SUPERVISOR</i> .....	<i>iii</i>
<i>APPROVAL SHEET</i> .....	<i>iv</i>
<i>ACKNOWLEDGEMENT</i> .....	<i>v</i>
<i>ABSTRACT</i> .....	<i>vi</i>
<i>LIST OF FIGURES</i> .....	<i>ix</i>
<i>LIST Of TABLES</i> .....	<i>x</i>
<i>LIST OF ABBRIVIATIONS</i> .....	<i>xi</i>
<b>CHAPTER I INTRODUCTION</b> .....	<b>1</b>
1.1 Background of the Study .....	1
1.2 Problem Statement .....	1
1.3 Objectives of the Project.....	1
1.4 Literature Review.....	2
1.5 Development Methodology .....	2
1.6 Scope and Limitations of the Project .....	3
1.7 Report Organization.....	3
<b>CHAPTER II SYSTEM DEVELOPMENT PROCESS</b> .....	<b>5</b>
2.1 Analysis.....	5
2.1.1 Requirement Analysis .....	5
2.1.1.2 Non-Functional Requirements .....	9
2.1.2 Feasibility Study .....	10
2.1.3 Object-Oriented Modeling .....	11
2.2 Design .....	16
2.2.1 System Architecture .....	16
2.2.2 Database Design.....	20
2.3 Implementation .....	21
2.3.1 Tools and Development Environment.....	21
2.3.2 System Modules and Their Implementation .....	21

2.3.3 Testing.....	22
<b>CHAPTER III CONCLUSION AND RECOMMENDATION .....</b>	<b>25</b>
3.1 Summary .....	25
3.2 Conclusion .....	25
3.3 Recommendations.....	25
<b>REFERENCES.....</b>	<b>26</b>



## LIST OF FIGURES

Fig 2.1 Use Case diagram .....	11
Fig 2.2 Class Diagram.....	12
Fig 2.3 Object Diagram.....	13
Fig 2.4 Sequence Diagram.....	14
Fig 2.5 Activity Diagram .....	15
Fig 2.6 Deployment Diagram .....	16
Fig 2.7 Component Diagram.....	17
Fig 2.8 Homepage .....	18
Fig 2.9 Login Page.....	18
Fig 2.10 Signup Page .....	19
Fig 2.11 Admin Dashboard .....	19
Fig 2.12 User Dashboard .....	20

## LIST OF TABLES

Fig 1.1 Report Structure Overview .....	3
Fig 2.1 Specification for User Login .....	7
Fig 2.2 Specification for User Registration .....	7
Fig 2.3 Specification for switching user roles .....	6
Fig 2.4 View or Post gigs .....	6
Fig 2.5 Manage users .....	7
Fig 2.6 Access Wallet.....	7
Fig 2.7 Change personal settings .....	8
Fig 2.8 Different Test Cases for Unit Testing .....	24

## **LIST OF ABBRIVIATIONS**

**CSRF** - Cross-Site Request Forgery

**CSS** - Cascading Style Sheets

**GDPR** - General Data Protection Regulation

**HTML** - HyperText Markup Language

**ILO** - International Labour Organizatio

**JWT** - JSON Web Token

**ORM** - Object-Relational Mapping

**PCI DSS** - Payment Card Industry Data Security Standard

**TLS/SSL** - Transport Layer Security/Secure Sockets Layer

**UI** - User Interface

**UT** - Unit Testing

**XSS** - Cross-Site Scripting

# **CHAPTER I INTRODUCTION**

## **1.1 Background of the Study**

The gig economy has changed the worldwide job market in a big way. Digital networks have made it possible for people to work in flexible, short-term, and on-demand ways. It lets people work on their own and provide their services directly to clients through online platforms. Even while the gig economy is becoming more and more common, the frameworks that are already in place have trouble with making the user interface easy to use, processing payments safely, and settling disputes quickly.

This project solves these problems by building a unique Gig Economy System. The platform will connect freelancers with consumers, let them talk to each other safely, make sure payments are always made on time, and provide quick ways to settle disagreements. All of this will make users happier and build trust in the gig economy.

## **1.2 Problem Statement**

Traditional job structures can make it harder for skilled people to find flexible jobs. Also, companies have trouble quickly finding qualified freelancers for project-based work. Many gig sites still use manual booking systems, which can lead to double bookings and problems with scheduling.

- i. Interfaces that are hard to understand and not friendly.
- ii. Payment methods that are not safe or reliable.
- iii. A conflict resolution technique that doesn't work.

## **1.3 Objectives of the Project**

The goals of this project are as follows.

- i. To build and put into action a safe, user-friendly, and scalable web-based platform that connects freelancers with clients for short term work.
- ii. To set up a system that makes it easy for clients, freelancers, and admins to register and log in.
- iii. To create a web platform that is safe, easy to use, and can grow with the needs of freelancers and clients. This platform should allow for job posting, bidding, payment processing, and resolving disputes.

## 1.4 Literature Review

Digital platforms that allow for flexible, on-demand work have changed the global job market in a big way through the gig economy. De Stefano (2016) calls this tendency the rise of a "just-in-time workforce," which stresses how unstable gig employment is, since it typically doesn't come with social safety or job stability. Sundararajan (2016) talks about the rise of the gig economy as part of a bigger change toward crowd-based capitalism, where digital platforms take the place of traditional jobs and people can make money from their skills on their own.

A number of research have looked into how digital platforms change the way people work and their health. Katz and Krueger (2019) state that there has been a huge surge in nonstandard jobs in the U.S. This is mostly because of gig work that is done on platforms. Friedman (2014) warns about the rise of "shadow corporations," which are companies where workers are independent contractors but don't have the benefits and protections that come with normal jobs. Rosenblat (2018) also talks about how algorithmic management is becoming more significant on gig platforms like Uber. This changes how workers respond and how well they execute their tasks in ways that aren't always evident.

There are good and bad things about the gig economy. It helps people find work and start enterprises, especially in developing countries like India and Nepal (Graham et al., 2017), but it also makes inequality worse by making it difficult for some people to gain the skills and technology they need. Wood et al. (2019) talk on the trade-offs between independence and algorithmic control, saying that many gig workers feel both free and exploited. Schor (2017) says that sharing economies can be flexible, but they also run the risk of making social and economic differences worse. Kenney and Zysman (2016) say that platform economies are not just changes in technology; they are also changes in institutions that challenge current labor norms and need new laws and policies.

These observations show that gig platforms need to find a balance between innovation and fairness so that all users have access, efficiency, and protection. The Gig Economy System and other systems like it try to solve these problems by providing easy-to-use interfaces, secure payment processing, and built-in ways to settle disputes. This makes sure that both clients and freelancers work in a fair and open digital workspace.

## 1.5 Development Methodology

This project uses the Agile development process to make sure that integration, feedback, and iterative improvement happen all the time. The steps in the process are:

- i. Collecting requirements through interviews and surveys.
- ii. An examination of feasibility from technical, legal, operational, and economic points of view.

- iii. Develop cycles based on sprints, with regular testing and improvement.
- iv. Prototyping and testing with target users in small steps.

## 1.6 Scope and Limitations of the Project

### Scope

- i. Managing profiles for freelancers and clients.
- ii. Make sure that job postings, bids, and selections are safe.
- iii. A payment gateway that works with other systems.
- iv. A framework for resolving conflicts.
- v. Responsive web design with a Django backend and Bootstrap.

### Limitations

- i. The first focus is only on the web interface (there is no separate mobile app).
- ii. The platform can only offer services in categories that were set when it was first launched.
- iii. Restrictions by region are based on legal compliance and the availability of payment gateways.

## 1.7 Report Organization

To make it easy to read and understand, the report is broken up into simple sections. The first pages are formal ones, like the Title Page, the Student's Declaration, and the Supervisor's Recommendation. It has an Abstract, a Table of Contents, and listings of figures, tables, and abbreviations.

**Table 1.1 Report Structure Overview**

Chapter No.	Chapter Title	Description
<b>Chapter I</b>	<b>Introduction</b>	Offers a summary of the project, including its aims, problem statement, scope, and relevance. Examines current research, technologies, and pertinent works that underpin the project.
<b>Chapter II</b>	<b>System Development Process</b>	
	<b>System Analysis</b>	Explains user requirements, functional & non-functional requirements, and feasibility analysis.
	<b>System Design</b>	Details system architecture, and diagrams.
	<b>Implementation</b>	Covers coding standards, naming conventions, frameworks, and database design used in development.

	<b>Testing &amp; Evaluation</b>	Describes testing strategies, test cases, and system evaluation results.
<b>Chapter III</b>	<b>Conclusion &amp; Recommendations</b>	Summarizes findings, achievements, challenges, and future enhancements.
<b>References</b>		Lists of books, research papers, websites, and other sources cited in the report
<b>Appendices</b>		Includes Screenshots, user manuals, and additional information.

## CHAPTER II SYSTEM DEVELOPMENT PROCESS

### 2.1 Analysis

This section goes into great detail about the system analysis for the Gig Economy System. It includes analyzing the needs (both functional and non-functional), figuring out if the project is doable, modeling the system in a structured or object-oriented way, and writing down the system's specifications.

System analysis also lets technical teams and stakeholders talk to one another, which makes it easier to plan, budget, and allocate resources.

#### 2.1.1 Requirement Analysis

Requirement analysis is the process of figuring out what users want and what the system can do. It makes sure that the system works as it should and serves the demands of all users. Usually, the process is split into functional requirements, which tell the system what it should do, and non-functional requirements, which tell the system how well it should do it.

##### Actors

- **Guest Users** can browse freelance jobs, explore freelancer or client profiles, and access login or registration features.
- **Freelancers** can register, set up service profiles, bid on projects, communicate with clients, and ultimately deliver work.
- **Clients** are responsible for registering, posting job opportunities, selecting freelancers, and overseeing both contracts and payment processes.
- **Admin** roles include managing users, moderating content and job posts, resolving disputes, and accessing system analytics.

##### 2.1.1.1 Functional Requirement

This system is designed to address the core operational needs of a party palace through digitized modules. Below is a list of functional requirements derived from interviews and surveys conducted with party palace owners and staff.

##### a) Use Case Scenario

Use case scenarios define the interactions of various users with the system to achieve particular objectives.



**Table 2.1 Specification for User Login**

<b>Use Case</b>	<b>Login to the System</b>
<b>Actors</b>	User, Admin, Staff
<b>Preconditions</b>	User must be registered
<b>Steps</b>	1. Input email, password. 3. The system verifies the input. 4. The user is sent to the user panel.
<b>Postconditions</b>	User logged in without any problem or receives the appropriate error

**Table 2.2 Specification for Users sign-up**

<b>Use Case</b>	<b>User Registration</b>
<b>Actor</b>	New User
<b>Preconditions</b>	- User should not already be registered - Must provide a unique email and phone number
<b>Steps</b>	1. Access the registration page 2. Provide required details. 3. Select the “Register” button 4. The system verifies the inputs. 5. If valid, save the user in the database. 6. Display an error notice if invalid or duplicated.
<b>Postconditions</b>	- If successful, a new user account is created - If failed, an appropriate error message is displayed

**Table 2.3 Specification for switching user roles**

<b>Use Case</b>	<b>Switch Roles</b>
<b>Actors</b>	Existing users
<b>Preconditions</b>	User is logged in
<b>Steps</b>	1. Head to dashboard 2. select the desired role (freelancer/client) 3. switches between client and freelancer dashboard
<b>Postconditions</b>	Changes the displayed dashboard depending upon the role

**Table 2.4 View or post gigs**

<b>Use Case</b>	<b>View and post gigs or job</b>
-----------------	----------------------------------

<b>Actors</b>	Existing users
<b>Preconditions</b>	User must be logged in and must set themselves to freelancer or employer role according to their preference.
<b>Steps</b>	<ol style="list-style-type: none"> <li>1. Navigate to dashboard</li> <li>2. Change the “switch role” slider to change between roles</li> <li>3. Employers can post the job from the dashboard while Freelancer can accept the jobs.</li> </ol>
<b>Postconditions</b>	<ul style="list-style-type: none"> <li>-Existing jobs listing are displayed for freelancers.</li> <li>-Employers can post a job to update the existing job listings.</li> </ul>

**Table 2.5 Manage Users**

<b>Use Case</b>	<b>Ban, remove or timeout users</b>
<b>Actors</b>	Admin, Staff
<b>Preconditions</b>	Must be logged in with admin account
<b>Steps</b>	<ol style="list-style-type: none"> <li>1. Access the admin dashboard</li> <li>2. Select "Manage users"</li> <li>3. Search and find the required user</li> <li>4. Use the provided buttons to ban, timeout or delete the user</li> </ol>
<b>Postconditions</b>	User added to respective database

**Table 2.6 Access Wallets**

<b>Use Case</b>	Access one's earning or spendings
<b>Actors</b>	users
<b>Preconditions</b>	Must be logged in
<b>Steps</b>	<ol style="list-style-type: none"> <li>1. login to respective dashboard.</li> <li>2. access the wallet section.</li> <li>3. view the spendings or earnings</li> </ol>

<b>Postconditions</b>	View the transactions
-----------------------	-----------------------

**Table 2.7 Change personal settings**

<b>Use Case Name</b>	Change profile settings, passwords and notification settings.
<b>Actors</b>	Users
<b>Preconditions</b>	Must be logged into an existing account
<b>Steps</b>	<ol style="list-style-type: none"> <li>1. Open the user dashboard</li> <li>2. Click the “settings” tab</li> <li>3. Select the settings provided and update it as your preference.</li> <li>4. save the settings</li> </ol>
<b>Postconditions</b>	Settings databases are updated

### **User Management**

- i. The system shall allow users to register and log in with role-specific access levels (clients, freelancers, administrators).
- ii. Secure authentication and authorization mechanisms shall be implemented to protect user data and restrict access appropriately.
- iii. Users shall be able to view and update their personal profiles.
- iv. Role-based dashboards shall be provided, offering tailored interfaces and functionalities for clients, freelancers, and admins.

### **Job Management**

- i. Clients shall be able to post new jobs, including details such as job title, description, budget, and deadline.
- ii. Freelancers shall be able to browse, search, and filter available job listings.
- iii. Clients shall have the capability to manage posted jobs, including editing, deleting, and updating job status.

### **Proposal and Bidding System**

- i. Freelancers shall be able to submit proposals for posted jobs, including a bid amount and proposal text.
- ii. Clients shall be able to review submitted proposals and award jobs to selected freelancers.

- iii. The system shall track proposal statuses, including submitted, shortlisted, accepted, and rejected states.

### **Contract and Work Management**

- i. Upon acceptance of a proposal, the system shall generate a contract between the client and freelancer.
- ii. Both parties shall be able to view contract terms, deadlines, and deliverables.
- iii. Contracts shall support status updates such as active, completed, or cancelled.

### **Dispute Resolution**

- i. Either party shall be able to raise disputes on contracts.
- ii. Administrators shall be empowered to review disputes, examine evidence, and resolve cases.

### **Reviewing and Rating System**

- i. After contract completion, both clients and freelancers shall be able to rate each other.
- ii. Reviews shall include a numerical rating (e.g., 1 to 5 stars) and optional comments.
- iii. User profiles should display overall ratings and review histories.

### **Administrative Features**

- i. Administrators shall manage users, including banning, verifying, and role promotion/demotion.
- ii. Admins shall moderate job postings, proposals, and dispute content.
- iii. An admin dashboard should provide analytics on platform activity and financial metrics.

## **2.1.1.2 Non-Functional Requirements**

This section shows the non-functional requirements of system, which defines the quality attributes and constraints to ensure the platform operates efficiently, securely, and reliably.

### **Performance**

- i. The system shall handle up to 10,000 concurrent users without significant degradation in response time
- ii. Real-time notifications shall be delivered with latency under 2 seconds.

### **Scalability**

- i. The system architecture shall support horizontal scaling to accommodate future growth in user base and transaction volume.

## Security

- i. User data, including authentication credentials and payment information are encrypted in transit (using TLS/SSL) and at rest.
- ii. The platform shall be protected against basic vulnerabilities like SQL injection, Cross-Site Scripting (XSS), and Cross-Site Request Forgery (CSRF).

## Usability

- i. The user interface should be intuitive and accessible.
- ii. Error messages and notifications should be clear and informative, guiding users to resolve issues.

### 2.1.2 Feasibility Study

A feasibility study assesses the practicality and viability of a proposed system before full-scale development. It helps determine whether the system is worth pursuing and whether it can be successfully implemented with the available resources. This process involves assessing several dimensions: technical, economic, operational, legal, and schedule feasibility.

- **Technical Feasibility:** This assesses whether the required technology and technical expertise are available to develop and maintain the system. The system will utilize Django for backend development, CSS combined with Javascript for a dynamic frontend, and SQLite as the database. These tools are robust, reliable, and widely supported.

- **Economic Feasibility:** This evaluates the cost-effectiveness of the project. With a reliance on open-source technologies, the project remains cost-effective. It also provides value by bridging freelancers and clients in a protected environment.

- **Operational Feasibility:** This analyzes whether the system will be accepted and effectively used by stakeholders. Thanks to its user-friendly design, little to no training is needed for users. Admins can operate efficiently using built-in dashboards.

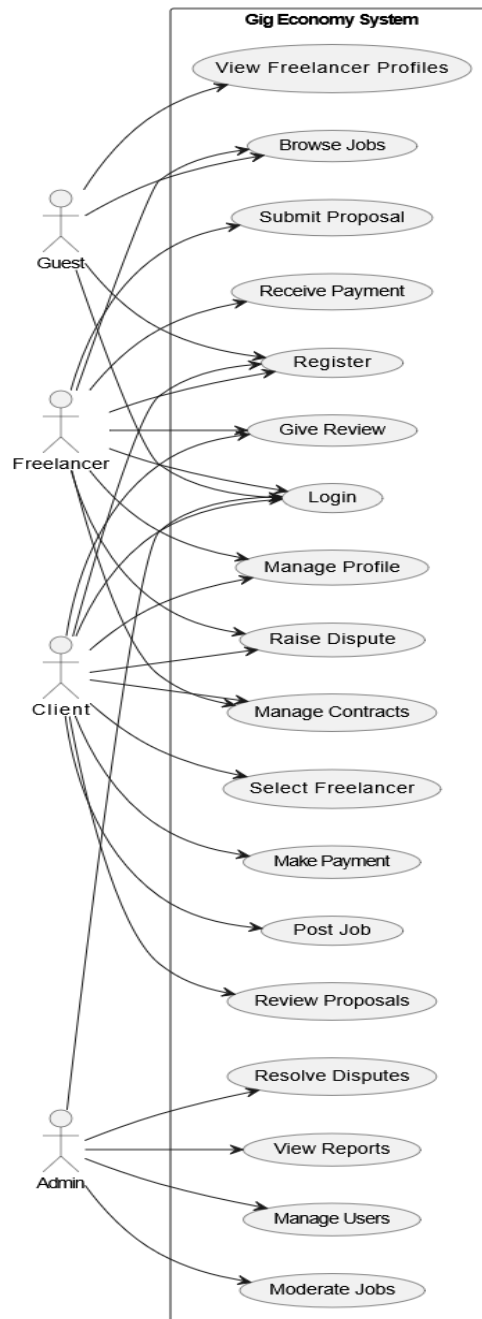
- **Legal Feasibility:** This examines the legal constraints related to the project. The system will comply with data protection regulations such as GDPR to protect user privacy. Payment processing will follow PCI DSS standards to ensure secure financial transactions. User agreements, intellectual property rights, and terms of service will be clearly defined to protect the platform and its users.

- **Schedule Feasibility:** This determines whether the project can be completed within the available time frame. The project timeline has been carefully planned with milestones for design, development, testing, and deployment. Given the scope and resources, the project is expected to be delivered on schedule.

### 2.1.3 Object-Oriented Modeling

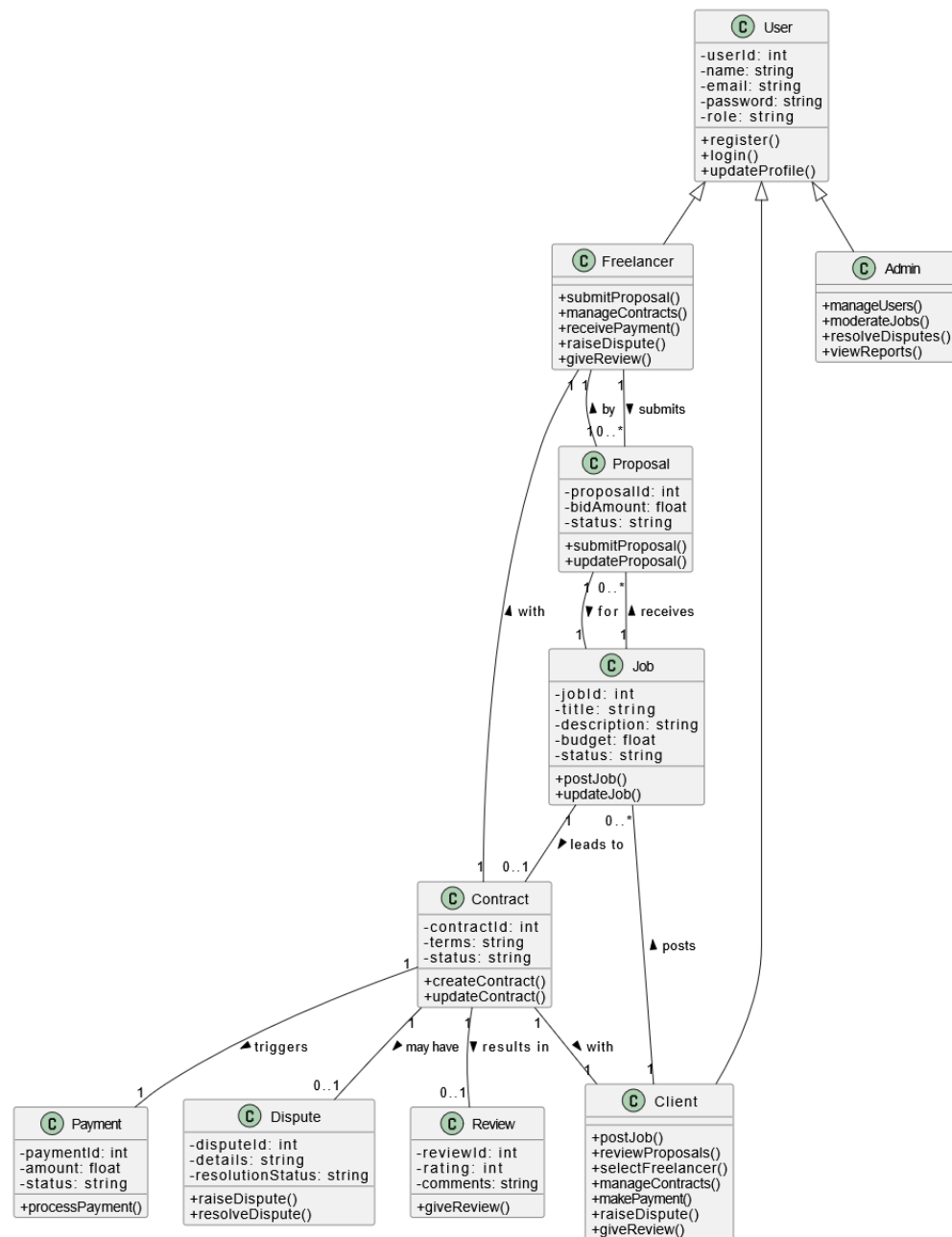
Object-oriented modeling was used extensively to define system structure. Some of the models are presented below:

- i. **Use Case Diagram** identifies users and their interactions with system functionalities.



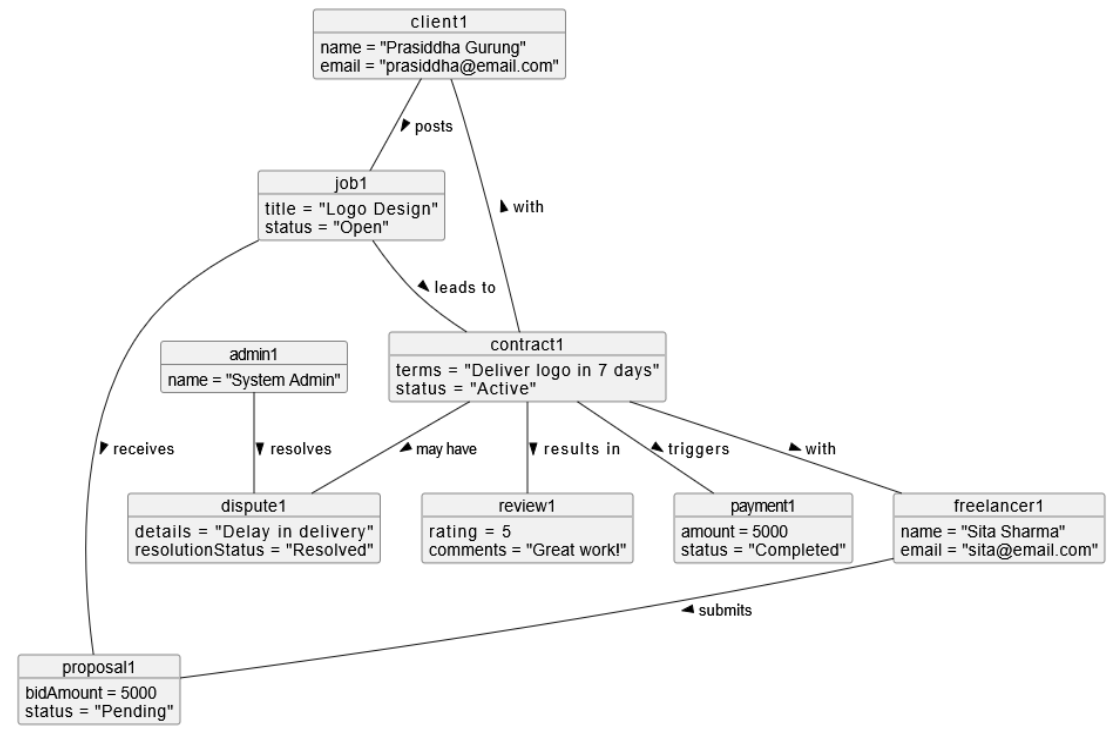
**Fig 2.1 Use Case diagram**

- ii. **Class Diagram** models the main classes, their attributes and methods, inheritance, and key relationships.



**Fig 2.2 Class Diagram**

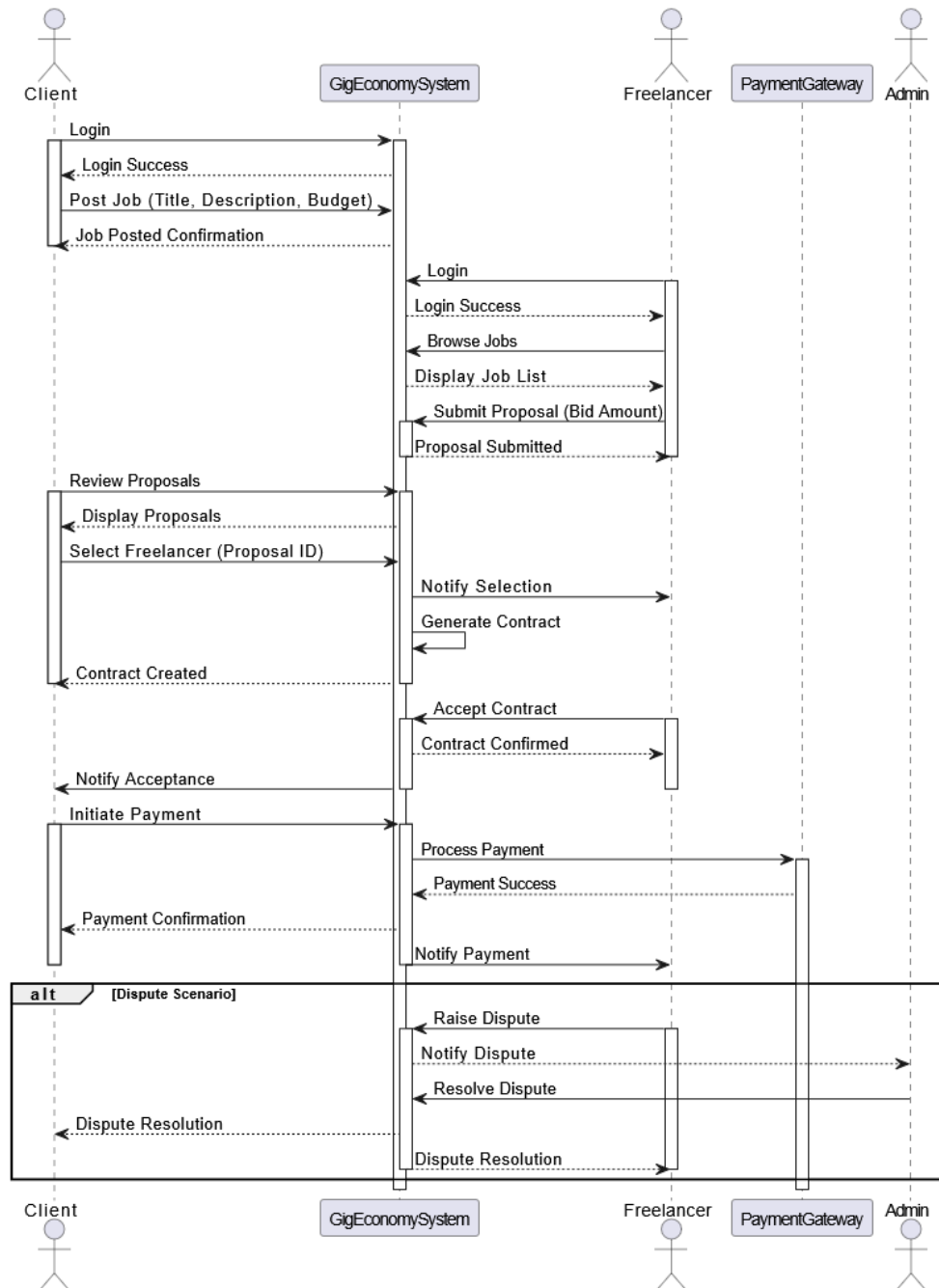
- iii. **Object Diagram** represents the instances of classes at a particular moment



**Fig 2.3 Object Diagram**

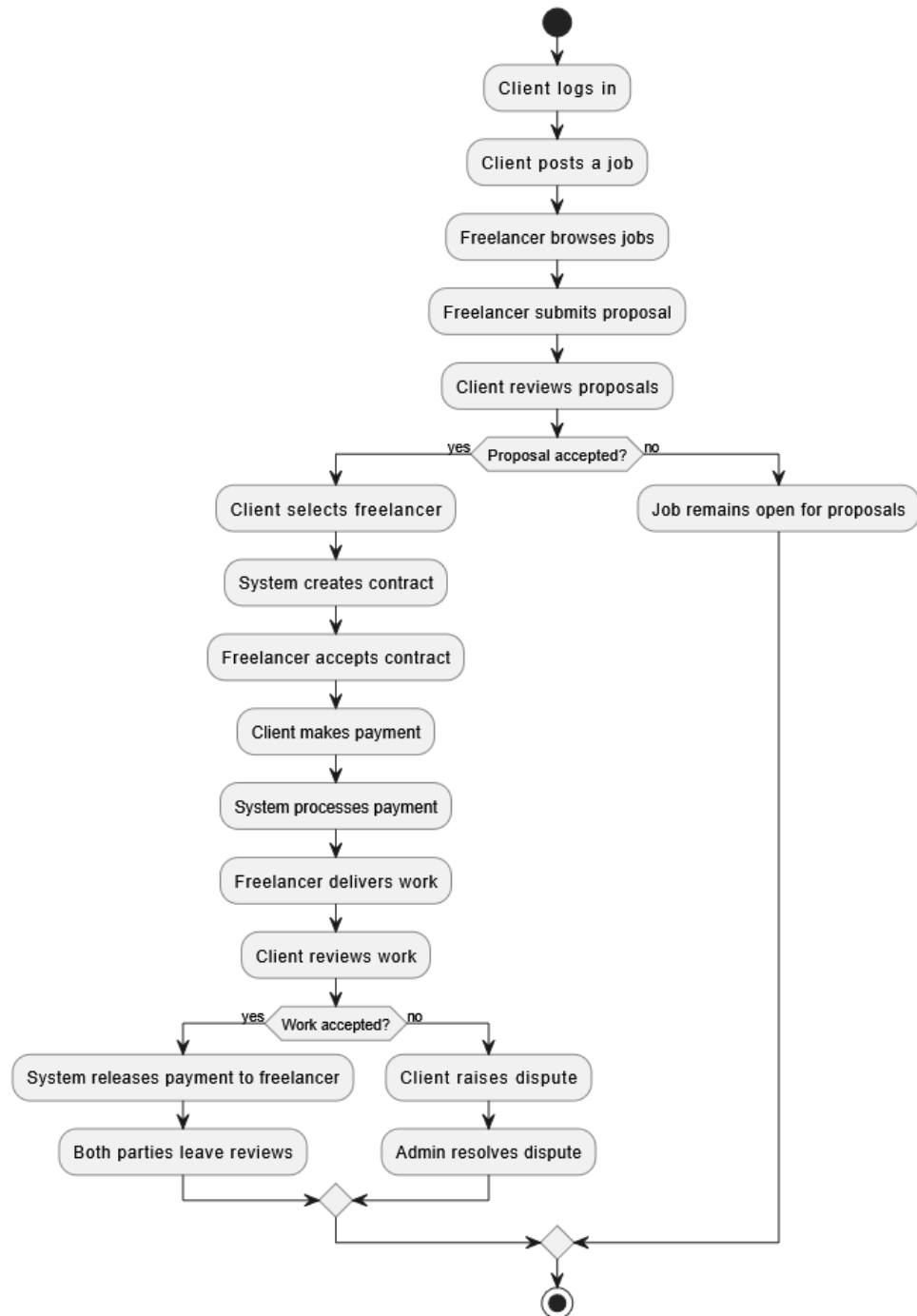


- iv. **Sequence Diagram** illustrates a typical workflow (e.g., job posting, bidding, payment):



**Fig 2.4 Sequence Diagram**

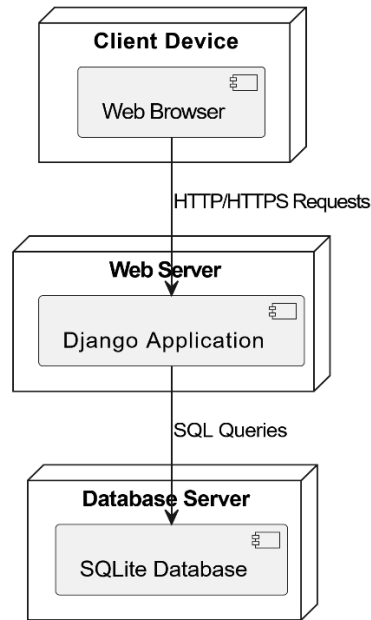
- v. **Activity Diagram** demonstrates workflow of a job from posting to completion



**Fig 2.5 Activity Diagram**

- vi. **Deployment Diagram** illustrates physical deployment of application components across devices and servers.

**Gig Economy System - Deployment Architecture (Using Wallet System)**



**Fig 2.6 Deployment Diagram**

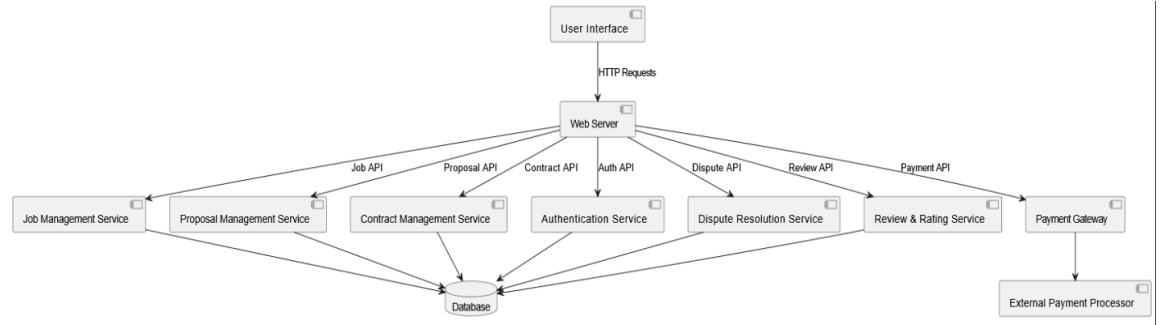
## 2.2 Design

The design phase translates the requirements specified in the analysis phase into a blueprint for building the Gig Economy System. This phase ensures that all system components, interfaces, data structures, and workflows are well-defined, coherent, and aligned with user needs and business goals. The design aims to provide a scalable, secure, and user-friendly platform that efficiently connects freelancers and clients

### 2.2.1 System Architecture

The Gig Economy System adopts a multi-tier architecture to separate concerns and enhance scalability and maintainability:

**Component Diagram** shows logical breakdown of front-end, back-end, and database layers.



**Fig 2.7 Component Diagram**

**Presentation Layer:** Implements the user interface using HTML, CSS, and JavaScript. It ensures responsive design for desktops, tablets, and smartphones.

**Application Layer:** Built with Django (Python), this layer handles business logic, user authentication, job management, bidding, contract creation, payments, dispute resolution, and review management.

**Data Layer:** Uses SQLite for secure, reliable, and efficient data storage and retrieval.

## Images of results of deployments

### i. Homepage

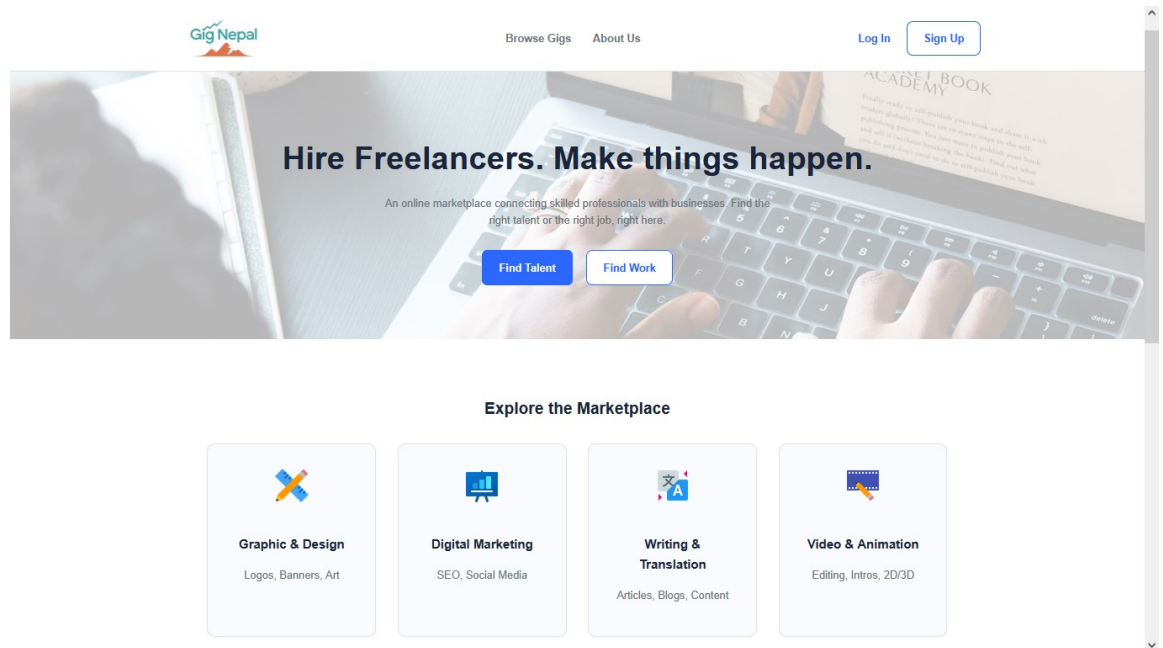


Fig 2.8 Homepage

### ii. Login page

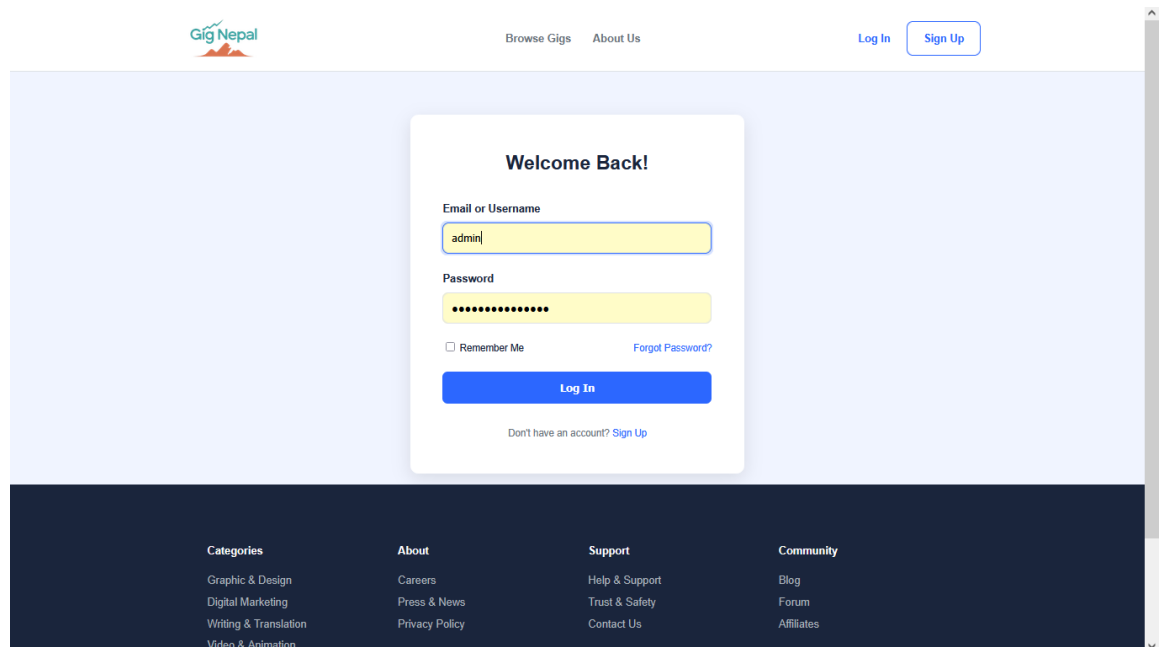
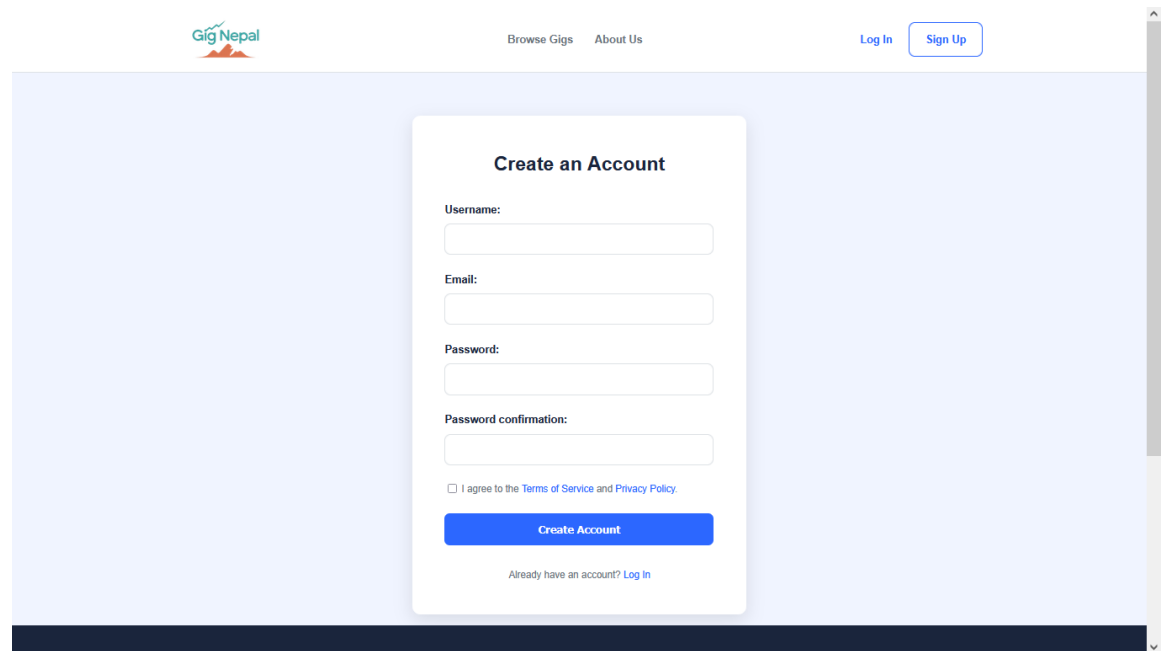


Fig 2.9 Login Page

### iii. Sign-up page



**Create an Account**

Username:

Email:

Password:

Password confirmation:

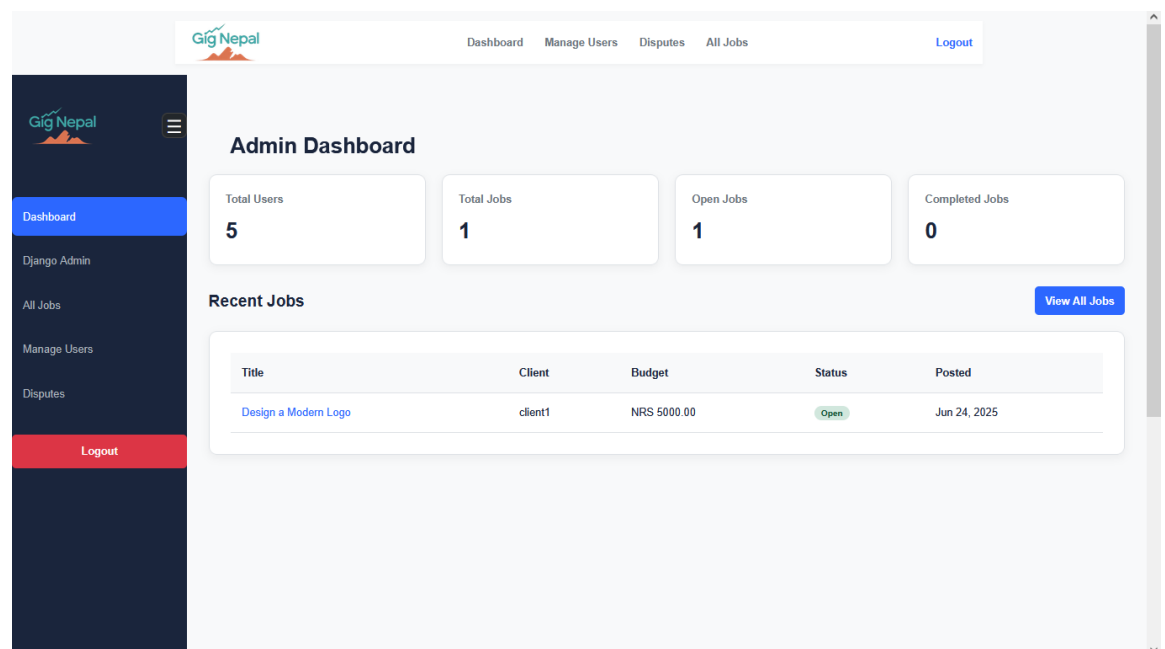
☐ I agree to the [Terms of Service](#) and [Privacy Policy](#).

[Create Account](#)

Already have an account? [Log In](#)

Fig 2.10 Signup Page

### iv. Admin Dashboard



**Admin Dashboard**

Total Users: 5

Total Jobs: 1

Open Jobs: 1

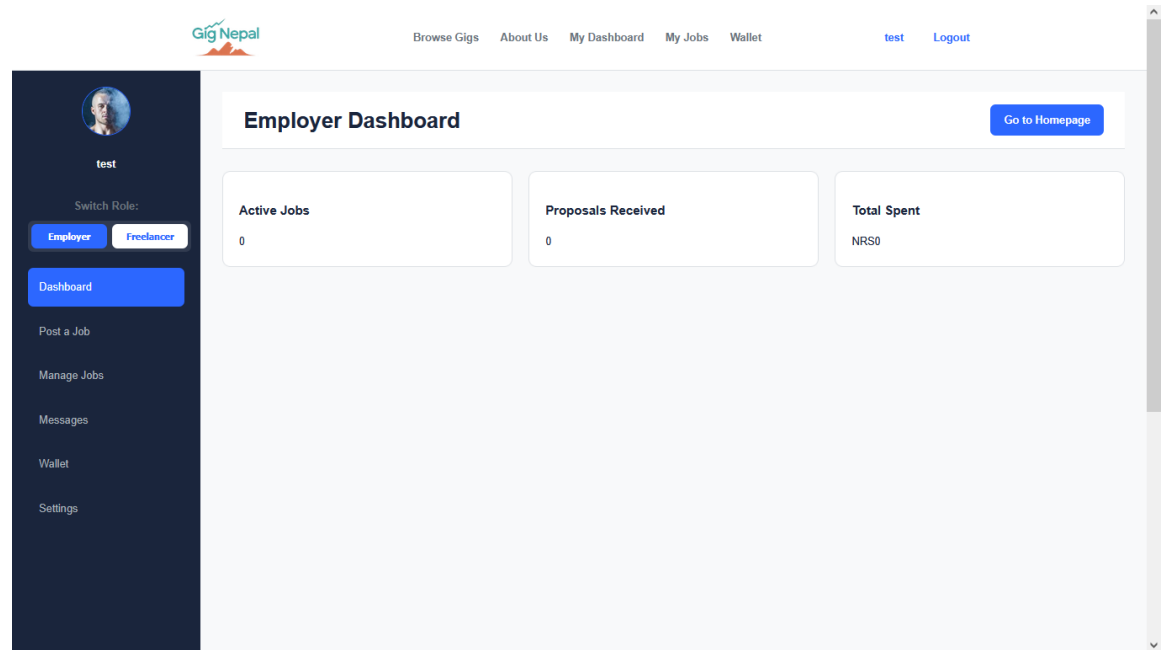
Completed Jobs: 0

**Recent Jobs**

Title	Client	Budget	Status	Posted
<a href="#">Design a Modern Logo</a>	client1	NRS 5000.00	Open	Jun 24, 2025

Fig 2.11 Admin Dashboard

## v. User Dashboard



**Fig 2.12 User Dashboard**

### 2.2.2 Database Design

The database is structured using a relational model to ensure data integrity and efficient querying. Key entities and relationships include:

#### **Relationships:**

- One client can create multiple job postings.
- Each job can receive multiple freelancer proposals.
- A freelancer may submit multiple proposals.
- Every contract connects one job with one freelancer.
- Payments are linked to contracts, and in some cases, disputes may arise.
- Reviews are generated once a contract concludes.

## 2.3 Implementation

Setting up the development environment, developing code for the front end and back end, establishing the database, integrating modules, implementing security measures, and testing the system before deployment were all part of the implementation for the Gig economy project.

### 2.3.1 Tools and Development Environment

To build and operate the system well, the following tools and settings were used:

**Visual Studio Code (VS Code):** VS Code was used as the IDE for development of this system. It works with a lot of programming languages and has features like syntax highlighting, extensions, and built-in version control making the development more productive.

**Django:** The backend of the program is built with a strong Python web framework. Its Object-Relational Mapping (ORM) technology takes care of server-side logic, URL routing, and database interfaces.

**CSS:** CSS, or Cascading Style Sheets, is used for the frontend of the project to govern the whole look of a website, including its layout, colors, fonts, spacing, and more making the website visually appealing and easy to navigate.

**JavaScript:** JavaScript is used in this project for more interactivity. Sliders, pop-ups, form validation, animations, and real-time changes to web pages, which makes them more interesting and useful for consumers is added using javascript.

**SQLite:** SQLite, a file-based database system is used in this project. Its small and easy to set up feature helped for storing and managing data along with the scalability of the database.

**GitHub:** GitHub is used in the development of this project. GitHub provided with tools for version control and to back up the progress of this project.

### 2.3.2 System Modules and Their Implementation

**User Module:** This module takes care of everything about user accounts, like signing up, logging in and out, and managing profiles. It also gives clients, freelancers, and admins their own dashboards that provide information and activities that are relevant to their positions.



**Job Module:** Clients can use the job module to make and post new gigs by filling out information such the work title, description, budget, and due dates. Freelancers can easily identify and apply for employment by browsing, searching, and filtering the jobs that are accessible.

**Proposal Module:** This module takes care of job proposals from start to finish. Freelancers send in bids with their terms, and clients can look over these bids, make a short list, accept one, or turn it down. It keeps both sides up to date on the status of each proposal.

**Contract Module:** This module makes and keeps track of contracts between clients and freelancers after a proposal is accepted. It keeps track of the status of contracts (active, completed, or canceled), deadlines, and deliverables. This makes sure that all sides are on the same page about what work needs to be done and when.

**Dispute Module:** This module gives users ways to file complaints about payments or contracts. Admins can look at facts, settle disagreements, and make fair decisions to end conflicts. The results are saved in the system.

**Review Module:** This module lets clients and freelancers rate and review each other after they finish a work. It gets feedback that helps people trust and respect you, and it shows overall ratings and comments on user profiles.

### 2.3.3 Testing

Security is vital for a system handling sensitive user and financial data. The following practices were applied:

#### 2.3.3.1 Unit Testing

Each module (e.g., login, booking, report) underwent isolated testing.

**Table 2.8 Different Test Cases for Unit Testing**

Test Case ID	Test Scenario	Test Steps	Expected Output	Actual Output	Status
UT-001	Login with correct credentials	1. Go to login page 2. Enter correct email and password 3. Login	Redirect to respective user dashboard (Admin/Staff/user)	Redirected successfully	Pass

UT-002	Login with invalid password	1.go to login page 2. Enter valid email and wrong password 3. Login	Shows error "Invalid credentials"	Displayed correct error message	Pass
UT-003	JWT token expiration	1. Log in to the system 2. Wait until token expires 3. Refresh page	Redirect to login or unauthorized message	Redirected to login	Pass
UT-004	Password hashing with bcrypt	1.Register user with password 2. Check database	Password saved in hashed format	Verified hashed password	Pass
UT-005	UI Component rendering (Dashboard)	1. Login as Admin 2. Check dashboard layout	Admin dashboard with options like booking, inventory, users visible	Correct UI rendered	Pass
UT-006	Logout clears token	1. Click logout button	Token removed, redirected to login	Token cleared and redirected	Pass
UT-007	Form validation (empty input)	1. Submit form with empty required fields	Validation message shown	Correct validation message	Pass
UT-08	Component-level error handling	1. Force API to fail 2. Observe UI response	Display generic error message without crashing app	Error handled gracefully	Pass

### 2.3.5.2 Integration Testing

- Interactions between modules were tested.
- Example: When a job is completed, the status should be updated, and the wallet must be credited

### 2.3.5.3 System Testing

System Testing revealed that the integrated system satisfies defined requirements and operates correctly under various conditions. The subsequent categories of system testing were conducted:

#### Performance Testing

The system's stability and responsiveness were evaluated through performance testing. Postman tool was utilized to assess:

- API response durations
- Page loading durations
- Rendering efficiency

The system always sent API replies within 500ms under normal loads, which made sure it worked at its best.

#### Security Testing

Security testing that looks for flaws in authentication and data transfer processes:

- a) JWT token expiration and role-based access control were verified.
- b) Input fields were tested for **SQL/NoSQL injection** and **XSS vulnerabilities**.
- c) HTTPS was recommended and tested for secure data transmission.

# **CHAPTER III CONCLUSION AND RECOMMENDATION**

## **3.1 Summary**

The Gig Economy System is a website that helps freelancers and clients find each other for project-based work. It solves some of the biggest problems with current platforms by providing a simple interface, safe payments, and a built-in system for resolving disputes. The system is built with Django, SQLite CSS, and Javascript, which makes sure it can grow, work well, and be easy to use. It was made using the Agile technique, and it sets the stage for future features like mobile apps, real-time chat, and AI-powered job matching. These will all help make the digital labor ecosystem more efficient and fair.

## **3.2 Conclusion**

This experiment shows that a web-based gig economy platform may help clients and freelancers find one other by making it easier to post jobs, bid on them, and manage contracts. Key accomplishments include safe payment processing, easy-to-use interfaces, and a strong system for resolving disputes, all of which make for a trustworthy and easy-to-use experience. The platform's modular design makes it easy to add new features in the future, and the use of modern frontend and backend technologies improves performance, security, and responsiveness. Overall, the project proves that this kind of system is possible and gives a strong base for ongoing growth and improvement.

## **3.3 Recommendations**

It is suggested that a mobile app be built for smartphones to make the Gig Economy System better and help it grow. This would make it easier for people to go to and use the system. Clients and freelancers will be able to talk to each other quickly and simply with real-time chat, which will make it easier for them to work together. AI can help people discover employment faster by making better suggestions based on their skills and past work. If the platform can manage many currencies and global payment options, it will be easier for people all over the world to use. Adding comprehensive dashboards to make reporting and analytics better would also help both users and admins learn more about how the platform is being utilized and how well it is operating. Finally, two-factor authentication and strong data encryption are examples of security methods that should always be becoming better. These will keep user data safe and help consumers trust the platform.

## REFERENCES

- De Stefano, V. (2016). The rise of the "just-in-time workforce": On-demand work, crowdwork, and labor protection in the gig economy. *Comparative Labor Law & Policy Journal*.
- Friedman, G. (2014). Workers without employers: Shadow corporations and the rise of the gig economy. *Review of Keynesian Economics*.
- Graham, M., Hjorth, I., & Lehdonvirta, V. (2017). Digital labour and development: Impacts of global digital labour platforms and the gig economy on worker livelihoods. *Transfer: European Review of Labour and Research*.
- Johnston, H., & Land-Kazlauskas, C. (2018). Organizing on-demand: Representation, voice, and collective bargaining in the gig economy. *ILO Report*.
- Kalda, A. (2025). Gig economy as a safety net in times of crisis: Evidence from Eastern Europe. *Journal of Financial Economics*.
- Katz, L. F., & Krueger, A. B. (2019). The rise of alternative work arrangements. *ILR Review*.
- Kenney, M., & Zysman, J. (2016). The rise of the platform economy. *Issues in Science and Technology*.
- Lee, M., Kim, S., & Park, J. (2020). ICT-driven transformation in event management: Leveraging the gig economy. *Technovation*.
- Manyika, J., Lund, S., Bughin, J., Robinson, K., Mischke, J., & Mahajan, D. (2016). Independent work: Choice, necessity, and the gig economy. *McKinsey Global Institute*.
- Newlands, G. (2021). Data obfuscation as resistance to platform surveillance. *New Technology, Work and Employment*.
- Omar, M., & Jamil, R. (2025). Worker well-being and satisfaction in the gig economy: A systematic review. *International Journal of Research and Innovation in Social Science (IJRISS)*.
- Petriglieri, G., Ashford, S. J., & Wrzesniewski, A. (2019). Agony and ecstasy in the gig economy: Cultivating holding environments for precarious and personalized work identities. *Academy of Management Journal*.

- Pilatti, L. A., Dominguez, S., & Carmona, C. (2024). Collective agency among gig workers: Strategies and solidarity. *Administrative Sciences*.
- Ravenelle, A. J. (2019). "We're not uber": Control, autonomy, and entrepreneurship in the gig economy. *Sociological Perspectives*.
- Rosenblat, A. (2018). *Uberland: How algorithms rewrite work rules*. University of California Press.
- Schor, J. B. (2017). Does the sharing economy increase inequality?. *Cambridge Journal of Regions, Economy and Society*.
- Sundararajan, A. (2016). *The sharing economy: The end of employment and the rise of crowd-based capitalism*. MIT Press.
- Tan, B., Lim, E. T. K., & Kim, S. (2023). Integrating e-commerce with customer relationship management in gig platforms. *Electronic Commerce Research*.
- Wood, A. J., Graham, M., Lehdonvirta, V., & Hjorth, I. (2019). Good gig, bad gig: Autonomy and algorithmic control in the global gig economy. *Work, Employment and Society*.
- Zhao, W. (2021). Using business intelligence to optimize gig economy operations. *Information & Management*.