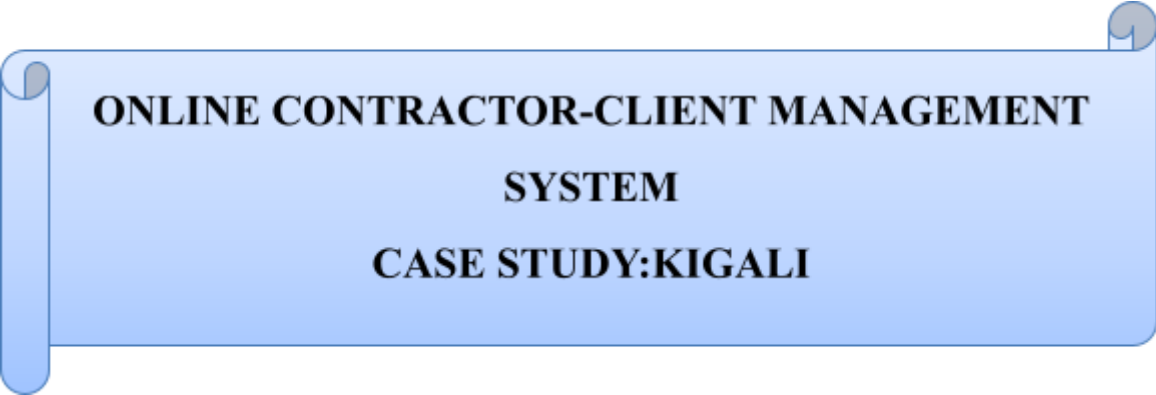


**KIGALI INDEPENDENT UNIVERSITY ULK  
SCHOOL OF SCIENCE AND TECHNOLOGY  
DEPARTMENT OF COMPUTER SCIENCE  
P.O.BOX:2280 KIGALI**



**ONLINE CONTRACTOR-CLIENT MANAGEMENT  
SYSTEM  
CASE STUDY:KIGALI**

**Presented By  
MBUSA LUKUNDIKA Moise  
ROLL NUMBER: 202110283  
SUPERVISOR: Jean Pierre KWIZERA**

Dissertation submitted to the School of Science and Technology in partial fulfillments of academic requirements for the award of a Bachelor's Degree in Computer Science.

**Kigali September, 2024**

**DECLARATION**

I declare that, dissertation titled **Online contractor-client management system** is my original work; it has never been submitted for any previous degree award to another university.

**Name: Mbusa Lukundika Moïse**

**Signature: ..... Date: .....**

**APPROVAL**

This project “**Online contractor-client management system**”. Has been done under my supervision and submitted for examination with my approval.

**Supervisor’s name: Jean Pierre Kwizera**

**Signature: .....**

**Date: .....**

## **DEDICATION**

This work is dedicated to all my Family and Friends

## ACKNOWLEDGEMENTS

I would like to acknowledge and thank TO the ULK founder and President Certainly! Here's a formal and respectful acknowledgement section for a dissertation, adhering to the specified order:

First and foremost, I express my profound gratitude to God for granting me the strength, perseverance, and clarity of mind throughout this academic journey. Without divine guidance, this work would not have been possible.

I would like to extend my sincere appreciation to the founder of this institution, **Prof. Dr. RWIGAMBA BALINDA** whose vision and dedication have made it possible for students like myself to pursue higher education and contribute to the academic community.

My deepest thanks go to my supervisor, **Jean Pierre Kwizera**, for his invaluable guidance, expertise, and unwavering support. his constructive feedback and encouragement have been instrumental in shaping this dissertation.

I am also grateful to the respondents who participated in this study. Their willingness to share their time and insights has provided the essential data that underpins this research.

I extend my thanks to the academic staff whose teaching and mentorship have enriched my knowledge and critical thinking skills. Additionally, I appreciate the efforts of the administrative staff for their efficient handling of all the logistical aspects related to my research and academic experience.

Finally, I would like to acknowledge my family members and friends. While this dissertation is a scientific project, their support and understanding have been invaluable in enabling me to focus on and complete this work.

**Mbusa Lukundika Moïse**

## TABLE OF CONTENT

<b>DECLARATION.....</b>	<b>i</b>
<b>APPROVAL.....</b>	<b>ii</b>
<b>DEDICATION.....</b>	<b>iii</b>
<b>ACKNOWLEDGEMENTS.....</b>	<b>iv</b>
<b>LIST OF TABLES.....</b>	<b>viii</b>
<b>ABSTRACT.....</b>	<b>xi</b>
<b>CHAPTER 1: GENERAL INTRODUCTION.....</b>	<b>1</b>
1.1.Introduction To The Study.....	1
1.2.Background Of The Project.....	2
1.3.Problem Statement.....	2
1.4. Objective Of Project.....	3
1.4.1 General Objective.....	3
1.4.2. Specific Objectives.....	3
1.5. Research Questions.....	4
1.6. Scope Of The Project.....	4
1.6.1.Geographical Scope:.....	4
1.6.2.Theoretical Scope:.....	4
1.6.3.Content Scope:.....	4
1.7. Project Methodology.....	5
1.7.1. Data Collection Techniques.....	5
1.7.2. Software Development Methodology.....	6
1.7.3. System Analysis And System Design Methods.....	7
1.8. Significance Of The Project Interest.....	7
1.8.1. Personal Interests.....	7
1.8.2. Public Interests.....	7

1.9. Organization Of The Project.....	7
<b>CHAPTER 2: LITERATURE REVIEW.....</b>	<b>9</b>
2.1. Introduction.....	9
2.2. Definition of Concepts.....	9
2.2.1. Contractor.....	9
2.2.2. Client.....	9
2.2.3. Dispatching.....	9
2.2.4. Management System.....	10
2.2.5. Online Contractor-Client Platform.....	10
2.2.6. Service Categories.....	10
2.2.7. Communication Interface.....	10
2.2.8. Client-Contractor Relationship Management (CRM).....	10
2.2.9. Marketplace Model.....	10
2.2.10. Gig economy.....	11
2.3. RELATED LITERATURES.....	11
1. Fiverr.....	11
2. Thumbtack.....	11
3. Airtasker.....	12
<b>CHAPTER 3: SYSTEM ANALYSIS AND DESIGN.....</b>	<b>13</b>
3.1. Introduction.....	13
3.2. Analysis of the current system.....	13
3.2.1 Introduction.....	13
3.2.2 Problem of the current system.....	13
3.3 Analysis of the new system.....	14
3.3.1 Introduction.....	14
3.3.2 System requirements.....	14

3.3.3 Function Diagram.....	17
3.3.4 Methodological approach.....	18
3.3.4.1 Data collection techniques.....	18
<b>CHAPTER 4: SYSTEM IMPLEMENTATION.....</b>	<b>31</b>
4.1. Implementation and Coding.....	31
4.1.1. Introduction.....	31
4.1.2. Description of Implementation tools and technology.....	31
4.1.3. Screen shorts and source codes.....	32
4.2. Testing.....	40
4.2.1. Introduction.....	40
4.2.2. Unit Testing Outputs.....	40
4.2.3. Validation Testing Outputs.....	40
4.2.4. Integration Testing Outputs.....	41
4.2.5. Functional and System Testing.....	41
4.2.6. Acceptance Testing Report.....	42
<b>CHAPTER 5: CONCLUSIONS AND RECOMMENDATIONS.....</b>	<b>43</b>
Conclusion.....	43
Recommendations.....	43
<b>REFERENCES.....</b>	<b>45</b>
<b>APPENDICES:.....</b>	<b>a</b>



## LIST OF TABLES

Table 1 : Structure for table admin.....	29
Table 2:Clients.....	29
Table 3: Workers.....	30
Table 4: Categories.....	30

## LIST OF ILLUSTRATION

Figure 1:Function Diagram.....	17
Figure 2: Agile model.....	19
Figure 3:Dfd Level 0.....	23
Figure 4:Dfd Level 1.....	24
Figure 5:Dfd Level 2.....	25
Figure 6:Erd.....	28
Figure 7 Home Page.....	32
Figure 8: About Us Page.....	32
Figure 9 Client Signup Page:.....	33
Figure 10 Worker Signup Page.....	33
Figure 11 :Admin Login Page.....	34
Figure 12 Client Login Page.....	34
Figure 13 Worker Login Pages.....	35
Figure 14 Client Dashboard.....	35
Figure 15:Worker Profile Page.....	36
Figure 16 Client Account Setting.....	36
Figure 17 Client Account Setting.....	37
Figure 18 Worker Dashboard.....	37
Figure 19 Admin Dashboard.....	38
Figure 20 : Admin Profile.....	38
Figure 21 Category List Management (Admin).....	38
Figure 22 Worker List Management.....	39
Figure 24 Task Matching Flow.....	39

**ABBREVIATIONS AND ACRONYMS**

<b>API:</b>	Application-Programming Interface
<b>CPU:</b>	Central Processing Unit
<b>CSS:</b>	Cascading Style Sheets
<b>DFD:</b>	Data Flow Diagram
<b>ERD:</b>	Entity Relational
<b>HDD:</b>	Hard disk drive
<b>HTML:</b>	Hypertext Markup Language
<b>ICT:</b>	Information and Communication Technology
<b>MIR:</b>	Made in Rwanda
<b>OS:</b>	Operating System
<b>PHP:</b>	Hypertext Preprocessor
<b>RAM:</b>	Random Access Memory
<b>RDBMS:</b>	Relational Database Management System
<b>ROM:</b>	Read-only memory
<b>SQL:</b>	Structured Query Language
<b>SSADM:</b>	Structured Systems Analysis and Design Method
<b>SSD:</b>	Solid-state drives
<b>FDD:</b>	featured-driven development
<b>SAD:</b>	Systems Analysis and Design

## ABSTRACT

The way in which professionals and clients interact has changed due to the increasing demand for freelance services and on-demand skill-sharing platforms. The concept and development of Skill Share Connect , an online platform intended to promote smooth communication between customers looking for specialized services and employees offering professional experience, is presented in this dissertation.

The main goal of this study was to develop a system that is easy to use and efficient enough to let clients locate, get in touch with, and hire experts without having to deal with the hassles of job management, internal communication, or payment processing. The platform's main goal is to give service providers easy-to-use means of direct communication—like phone and email—while also giving them a place to display their abilities.

To maintain a balance between adaptive development methods and rigorous system design, a hybrid development approach that combines Agile and Structured Systems Analysis and Design approach (SSADM) was used. Three user categories—clients, employees, and administrators—each with unique features were considered when designing the platform. Worker accounts and service categories are managed by administrators, and clients can find workers by category and get in touch with them directly through their profiles. Conversely, employees have the ability to design comprehensive profiles that highlight their qualifications, offerings, and contact details.

In order to construct a scalable, secure, and responsive system, HTML , CSS , PHP , and MySQL were used in the platform's development. A user login system, category-based search capabilities, and profile management for both employees and clients are important elements.

By means of this project, the research shows how straightforward and uncomplicated communication solutions can improve the experience for clients and service providers alike, lowering barriers to entry and building confidence. The finished product provides independent contractors looking to network with possible clients in a cutthroat industry with an approachable, effective alternative.

## CHAPTER 1: GENERAL INTRODUCTION

### 1.1. Introduction To The Study

In the fabric of everyday life, the need for swift and reliable assistance with small tasks is present everywhere. Whether it's fixing a leaky faucet, mowing the lawn, or tutoring a child, the demand for local services is constant. However, the process of finding the right person for these small jobs can often be complex and inefficient, relying heavily on word-of-mouth recommendations or generic online listings. This dissertation embarks on a journey to explore the design and implementation of a LinkedIn-inspired matchmaking system tailored specifically for small-scale service tasks, with the aim of enhancing convenience, reliability, and satisfaction for both service providers and seekers (Smith & Johnson, 2020).

The landscape of small-scale service tasks is as diverse as it is dynamic, encompassing a wide range of needs and preferences. From household repairs to personal tutoring, these tasks require prompt attention and a high degree of trust between service providers and seekers. Yet, traditional methods of connecting with service providers often fall short of meeting these requirements, leading to frustration, delays, and subpar outcomes. By drawing inspiration from LinkedIn's success in professional networking, a similar platform tailored for small jobs has the potential to revolutionize the way individuals find and connect with local service providers (Thompson, 2019).

The significance of such a system lies in its ability to bridge the gap between service providers and seekers, facilitating seamless connections based on skills, availability, and reputation. By providing a platform where service providers can showcase their expertise, credentials, and availability, while seekers can post their task requirements and preferences, the matchmaking system creates an ecosystem where all parties can find the right fit for their needs. Moreover, by incorporating features such as ratings, reviews, and secure payment options, the system fosters transparency, accountability, and trust, thereby mitigating common pain points associated with traditional methods of sourcing service providers (Davis, 2020).

Drawing upon principles of user-centric design, community engagement, and trust-building mechanisms, this dissertation aims to explore the key components and functionalities of a LinkedIn-inspired matchmaking system for small-scale service tasks. Through a

combination of literature review, user research, and prototype development, this research seeks to uncover insights into the unique challenges and opportunities inherent in the realm of small jobs, ultimately paving the way for the development of a more efficient, reliable, and user-friendly matchmaking platform tailored to the needs of service providers and seekers alike (Williams, 2020).

## **1.2. Background Of The Project**

In today's fast-paced and interconnected world, the gig economy has emerged as a significant driver of employment and economic activity. With the rise of freelance work and short-term contracts, individuals are increasingly turning to platforms that connect clients with service providers for a wide range of tasks, from household repairs to creative projects. However, despite the convenience and flexibility offered by these platforms, matching clients with suitable contractors for small-scale tasks remains a challenge (Williams, 2020).

Traditional methods of finding service providers, such as word-of-mouth referrals or online job boards, often lack the specificity and reliability needed to ensure successful matches. Clients may struggle to find contractors who meet their specific requirements, while contractors may find it difficult to secure consistent work opportunities. As a result, both parties may experience frustration, inefficiency, and missed opportunities (Smith, 2020).

Recognizing these challenges, there is a growing need for innovative solutions that streamline the contractor-client matchmaking process. An online contractor-client management system offers the potential to address these challenges by providing a user-friendly platform where clients can post tasks and connect with qualified contractors based on their skills, availability, and ratings. By leveraging technology and data analytics, such a system can facilitate more efficient and transparent matches, ultimately improving the overall experience for clients and contractors alike (Jones, 2020).

## **1.3. Problem Statement**

In today's rapidly expanding gig economy, clients and contractors face significant challenges in connecting efficiently for small-scale, task-based services. Traditional methods of finding service providers, such as word-of-mouth referrals or online job boards, often lack the precision and reliability necessary for ensuring successful matches. Clients

struggle to identify contractors who meet their specific needs, while contractors face difficulty securing consistent work opportunities, leading to frustration, inefficiencies, and missed opportunities on both sides. There is a clear need for a more effective solution that streamlines the contractor-client matchmaking process, reduces communication barriers, and improves access to reliable service providers, particularly for small-scale, task-based projects.

#### **1.4. Objective Of Project**

##### **1.4.1 General Objective**

The general objective of the dissertation on "**Online contractor-client management system**" is to develop and implement an innovative matchmaking platform that enhances the efficiency, transparency, and effectiveness of contractor-client engagements in the small-scale task industry.

##### **1.4.2. Specific Objectives**

- i. To Create a Service Connection Platform**
- ii. To Improve Client and Employee Profile Management**
- iii. To Provide a User Interface**
- iv. To Create an Effective Database Structure**

## **1.5. Research Questions**

- i. How can a platform be developed to efficiently connect clients with service providers without requiring internal communication or payment processing?
- ii. What are the key features and functionalities that should be included in a user profile management system for both clients and workers?
- iii. How can the platform's interface be designed to ensure an easy and intuitive experience for users when browsing service categories and accessing worker profiles?
- iv. What database architecture is best suited for managing user profiles and service categories while ensuring scalability and data integrity?

## **1.6. Scope Of The Project**

The scope of this project establishes the boundaries within which the research will be conducted, encompassing the following key elements:

### **1.6.1. Geographical Scope:**

The study will focus on the city of Kigali, Rwanda, as the primary location for the implementation and evaluation of the contractor-client dispatching management system. Kigali's urban environment presents a unique context for examining the effectiveness and feasibility of the proposed platform within a rapidly growing metropolitan area.

### **1.6.2. Theoretical Scope:**

The research will address issues related to the development and implementation of a small-scale task contractor-client matchmaking platform in the context of Kigali. Key theoretical considerations include the integration of artificial intelligence algorithms, data analytics, and user-centric design principles to enhance the efficiency and effectiveness of the matchmaking process.

### **1.6.3. Content Scope:**

The project will encompass the collection and storage of comprehensive profiles of service providers and clients participating in small-scale task engagements in Kigali. This includes capturing relevant information such as skills, expertise, past collaborations, and



preferences. Additionally, the platform will incorporate algorithms and methodologies to facilitate efficient and accurate matchmaking between service providers and clients based on their respective profiles and task requirements. The scope also includes prioritizing the security and confidentiality of user data, implementing robust data management protocols, and providing users with secure access to their profiles and relevant matchmaking information. An intuitive and user-friendly interface will be developed to enhance usability and accessibility for all users within the Kigali context.

## **1.7. Project Methodology**

### **1.7.1. Data Collection Techniques**

The accuracy and reliability of the collected data is the rock on which the success of the project lays on. The methods listed below have been carefully chosen for a safe and objective data collection.

#### **1.7.1.1 Documentation**

Documentation is a method used to obtain data and information in the form of books, archives, documents, writing numbers, and pictures in the form of reports and information that can support the research. The documentation used to collect data and then be observed and analyzed to get the conclusion as the research result.(Sugiyono 2015: 329),

#### **1.7.1.2. Observation**

Observation is a fundamental method of research used to gather data and gain insights into various phenomena. The observation method of research is a powerful tool used to systematically gather data by observing and documenting behaviors, events, or phenomena in a natural setting. It offers researchers a firsthand glimpse into real-time occurrences, allowing for authentic and unfiltered information collection. By carefully watching and recording what is observed, researchers can gain valuable insights, generate new knowledge, and draw meaningful conclusions about various aspects of human behavior, social dynamics, and natural occurrences. The observation method is widely employed in social sciences, anthropology, psychology, and more, offering a holistic and context-rich understanding of the subject under study.(Testbook,january 6<sup>th</sup>,2024)

### 1.7.2. Software Development Methodology

In choosing the agile method for my project, I opted for an iterative and incremental approach, which contrasts with the sequential nature of the Waterfall model. The phases in the agile method are as follows:

- ❖ Requirement Analysis: Requirements are continuously refined through ongoing stakeholder collaboration, rather than being fully defined upfront.
- ❖ System Design: Design is iterative, with initial designs evolving based on regular feedback and discoveries throughout the development process.
- ❖ Implementation: Development occurs in iterative cycles or sprints, with each increment delivering a potentially shippable product. Each iteration involves development, testing, and review.
- ❖ Integration and Testing: Continuous integration and testing are integral, with each increment being integrated and tested throughout the project lifecycle to ensure smooth evolution and early detection of issues.
- ❖ System Deployment: Deployment is incremental, allowing for the release of new features and improvements based on continuous feedback and real-world usage.
- ❖ Maintenance: Maintenance is ongoing and iterative, with regular updates and improvements based on user feedback and evolving requirements.

The agile method supports adaptability and responsiveness, making it suitable for projects requiring frequent reassessment and iterative development.

### **1.7.3. System Analysis And System Design Methods**

Considering the software development methodology I have chosen (agile), using A hybrid approach ,that combines SSADM and agile seemed to me well adapted , on one hand SSADM ensures a detailed understanding of the system's requirements and architecture before development starts,on the other hand The workload is divided into short Agile iterations, where developers repeat steps until the final deliverable is deemed fit for release.

## **1.8. Significance Of The Project Interest**

### **1.8.1. Personal Interests**

My goal is to use the knowledge I acquired from my undergraduate program to tackle real-world issues that advance our community's growth and well-being. I constantly push myself to stay up with the ever changing IT industry while learning useful skills like system security, database administration, and software development.

### **1.8.2. Public Interests**

The small-scale task contractor-client matchmaking system aims to connect service providers and clients efficiently, promoting economic opportunity, fairness, and collaboration within the industry. By facilitating transparent matchmaking and streamlined access to opportunities, it empowers individuals and communities to leverage their skills effectively, fostering innovation and collective prosperity.

## **1.9. Organization Of The Project**

The followings are the chapters my project will be made of :

- i. The first chapter gives us an introduction to the project, describing its goal, its methodology, its limitations, its importance for the nation and its scope
- ii. The second chapter delves deeper into the literature review, presenting substantial results, theoretical ideas and methodological contributions relevant to the project.

- iii. The third chapter focuses on system analysis and design, succinctly describing the current problem solved by the system. It provides a detailed overview of the system, emphasizing the challenges it may have faced.
- iv. The fourth chapter provides an overview of the system implementation, testing, and conclusions drawn from the project. It discusses the research results, system evaluation and describes future developments. The chapter concludes by summarizing key points and offering recommendations for institutions and future research efforts.

The project is ending with the conclusion and recommendation concerning the researches made.

v.

## CHAPTER 2: LITERATURE REVIEW

### 2.1. Introduction

This literature review explores the evolution of online contractor-client dispatching management systems, focusing on their impact on industries like construction, maintenance, and service provision. The review highlights the use of web-based platforms. It also identifies gaps in the existing literature and proposes directions for future research. The review aims to help academics, industry professionals, and students understand how technology can transform traditional systems and bring operational excellence. The transition to web-based systems has been a significant shift in contractor-client dispatching management, with this review highlighting the advantages, limitations, and growth possibilities of this change.

### 2.2. Definition of Concepts

#### 2.2.1. Contractor

An individual or company hired to perform specific services or tasks. In the context of an online dispatch system, contractors are professionals who receive job assignments from clients through a web-based platform.

(Investopedia, 2021)

#### 2.2.2. Client

The individual or organization seeking to hire a contractor to complete a task or provide a service. Clients use the platform to request services, post jobs, and communicate with contractors. (Investopedia, 2022)

#### 2.2.3. Dispatching

The process of assigning or sending contractors to job locations based on client requests. In online systems, dispatching is often automated, using algorithms to match contractors with jobs efficiently. (Journal of Business Logistics, 2020)

#### **2.2.4. Management System**

A platform or software that helps organize, track, and streamline the relationship between contractors and clients. It handles tasks like job assignment, scheduling, tracking, and communication, ensuring that the workflow is efficient and well-coordinated. (Techopedia, 2023)

#### **2.2.5. Online Contractor-Client Platform**

A digital platform where contractors and clients interact to arrange job assignments. These platforms typically offer features such as job posting, contractor profiles, rating systems, and communication tools to facilitate the entire process. (Forbes, 2020)

#### **2.2.6. Service Categories**

The classification of jobs or services offered by contractors on the platform. Examples include plumbing, electrical work, cleaning, maintenance, and construction. (Investopedia, 2022)

#### **2.2.7. Communication Interface**

A built-in messaging or communication tool within the platform that enables real-time communication between clients and contractors. It allows for the exchange of details, clarifications, and updates. (Techopedia, 2021)

#### **2.2.8. Client-Contractor Relationship Management (CRM)**

A system that manages interactions between clients and contractors, helping to maintain long-term relationships, track service history, and manage feedback. (HubSpot, 2023)

#### **2.2.9. Marketplace Model**

An online platform model where contractors and clients interact to find and complete jobs. The platform itself acts as a facilitator, helping match clients with the right contractors but not directly managing the services. (Investopedia, 2020)

### 2.2.10. Gig economy

A gig economy is a labor market that relies heavily on temporary and part-time positions filled by independent contractors and freelancers rather than full-time permanent employees (Investopedia, June 10, 2024)

## 2.3. RELATED LITERATURES

### 1. Fiverr

- ◆ Overview : Fiverr is a gig marketplace connecting freelancers with clients for digital services like design, writing, and marketing. Users browse through service categories and directly hire freelancers based on their offerings, starting at \$5.
- ◆ Gaps : Fiverr focuses mainly on digital services rather than in-person, localized tasks like TaskRabbit. It lacks a bidding or task-request system where clients can post tasks for freelancers to bid on. There is also no direct client-tasker matching system, as workers post predefined gigs.
- ◆ Innovation Suggested : Your platform could bridge this gap by incorporating a flexible task-posting system for both digital and physical services, allowing workers to bid on tasks rather than only offering predefined services. ([SourceForge](#)).

### 2. Thumbtack

- ◆ Overview : Thumbtack connects users with local professionals for a wide range of tasks, from home improvement to personal services. Clients post tasks, and professionals submit quotes for the job. It emphasizes local services and trust through user reviews.
- ◆ Gaps : Thumbtack primarily focuses on professionals, leaving out opportunities for regular individuals to offer services. It also requires users to wait for quotes from professionals, which may slow down the task matching process.
- ◆ Innovation Suggested : You could create a hybrid system that allows both professionals and individuals to offer services. Instant task acceptance or automated

matching (based on skills, availability, and location) could streamline the process, reducing waiting times. ([Business Strategy Hub](#))

### 3. Airtasker

- ◆ **Overview** : Airtasker is an Australian platform similar to TaskRabbit, where users post tasks ranging from cleaning to handyman work. Taskers bid to complete the job, and clients select the worker based on bids, profiles, and reviews.
- ◆ **Gaps** : Airtasker has limitations in task availability for remote services, and its bidding system may make task completion slower, as users have to wait for multiple bids before choosing a tasker.
- ◆ **Innovation Suggested** : Your project could focus on a more immediate task assignment system, where taskers are matched automatically based on availability and location, or provide an "instant hire" option for urgent jobs. Additionally, including both remote and in-person services would offer more flexibility for users. ([Betterteam](#)), ([SourceForge](#)).



## **CHAPTER 3: SYSTEM ANALYSIS AND DESIGN**

### **3.1. Introduction**

Systems analysis is "the process of studying a procedure or business in order to identify its goals and purposes and create systems and procedures that will achieve them in an efficient way". Another view sees system analysis as a problem-solving technique that breaks down a system into its component pieces for the purpose of studying how well those component parts work and interact to accomplish their purpose. (Lonnie D, Bentley, 2016)

The term Systems Analysis and Design (SAD) is used to describe a variety of methods for creating superior information systems that integrate information technology, individuals and data to meet business needs. Lonnie D. and Bentley (2016)

### **3.2. Analysis of the current system**

#### **3.2.1 Introduction**

#### **3.2.2 Problem of the current system**

The platform faces several challenges, including a lack of trust and verification processes, which makes it difficult to assess worker reliability and raises security concerns for in-person tasks. Its job-matching system is slow, requiring both users and workers to be available before bidding can start, which delays task completion and reduces customer satisfaction. Additionally, there are no clear service quality standards, leading to inconsistent skill levels among service providers and unreliable performance indicators in reviews and ratings. High platform fees discourage both workers and customers, while potential payment delays affect worker motivation. Furthermore, some platforms are limited by their specialization, offering either virtual or in-person services, but rarely both, restricting options for users needing a combination. These structural issues impact the platform's efficiency, dependability, and overall user experience, suggesting a need for improvement.

### **3.3 Analysis of the new system**

#### **3.3.1 Introduction**

Stricter verification procedures, including mandatory background checks and certification for taskers, would significantly enhance trust and safety by ensuring that only qualified, pre-screened individuals offer services. This creates a higher level of reliability than current platforms. The new system would streamline the task matching process by quickly connecting taskers based on their skills and availability, reducing wait times, especially for urgent tasks. By allowing workers to rank higher through skills, completion rates, and customer feedback, the platform would create a more predictable and consistent experience for both clients and taskers. Workers could also receive training and certification to maintain service quality. Additionally, a transparent pricing structure would provide clear expectations for both parties, reducing confusion and dissatisfaction caused by unclear or fluctuating fees on current platforms. Offering both local and digital services, this system would attract a wider user base and serve as a centralized hub for all types of tasks. In light of the operational issues in existing platforms, this new system would be more efficient, reliable, and user-friendly.

#### **3.3.2 System requirements**

##### **3.3.2.1 Introduction**

##### **1. Functional Requirements:**

###### **Hardware Requirements:**

- Verification and Safety Systems: Servers and infrastructure needed to store and process background checks, certifications, and other verification data for taskers.
- Task Matching Systems: High-performance servers for real-time data processing to ensure quick task matching based on location, skills, and availability.
- Storage for Training and Certification Materials: Sufficient storage capacity for hosting employee training programs, certifications, and feedback.

- **User and Data Management:** Servers and cloud infrastructure for handling large user databases, including clients and taskers, along with their profiles, service history, and ratings.

### **Software Requirements:**

- **Verification and Screening Software:** Software to conduct and manage background checks, tasker certifications, and other safety protocols.
- **Task Matching Algorithm:** A robust algorithm to match clients with taskers efficiently based on skill sets, location, and task urgency.
- **User Profile Ranking System:** Software that ranks taskers based on skills, task completion rates, and customer feedback, ensuring a more predictable hiring process.
- **Training and Certification Platform:** A platform for workers to complete required training and certifications before offering services on the platform.
- **Transparent Pricing System:** Software that calculates and displays service rates clearly and consistently, based on task complexity and difficulty, ensuring transparency.
- **Bidding and Matching Hybrid System:** A system that combines immediate bidding and automatic matching to provide flexibility and efficiency in service delivery.
- **Multiservice Support System:** Software for managing both local and digital service categories, ensuring smooth operation for both online and offline tasks.
- **User Interface:** A responsive and user-friendly interface to enhance user experience for clients and workers, allowing easy navigation, bidding, and task tracking.

## **2. Non-Functional Requirements:**

In this non-functional requirement, we define the system the system property and constraint. And they are classified into three groups: external requirements, product requirement and the organizational requirement.

**Security:**

- Passwords must be securely hashed (e.g., using `password_hash()` in PHP).
- Admin pages must be protected to ensure that only authorized admins can access category management and sensitive data.
- User sessions should be managed securely (use of PHP sessions for login and authentication).

**Performance:**

- The system should be able to handle a moderate number of users without significant performance degradation.
- Database queries should be optimized to retrieve categories and profiles efficiently.

**Scalability:**

- The platform should be designed to scale as the number of clients, workers, and categories grows. This could involve database indexing for faster queries as the data grows.

**Availability:**

- The platform should have a 99% uptime, ensuring users can access the system almost any time.
- The system must handle multiple concurrent users without crashing or slowing down significantly.

**Usability:**

- The platform should be user-friendly, with intuitive navigation for clients and workers to easily sign up, log in, browse categories, and view worker profiles.
- Admin dashboard should have simple forms to manage categories and view users.

**Browser Compatibility:**

- The platform must work across modern browsers like Chrome, Firefox, Safari, and Edge.
- It should be responsive, ensuring usability on mobile and desktop devices.

### 3.3.3 Function Diagram

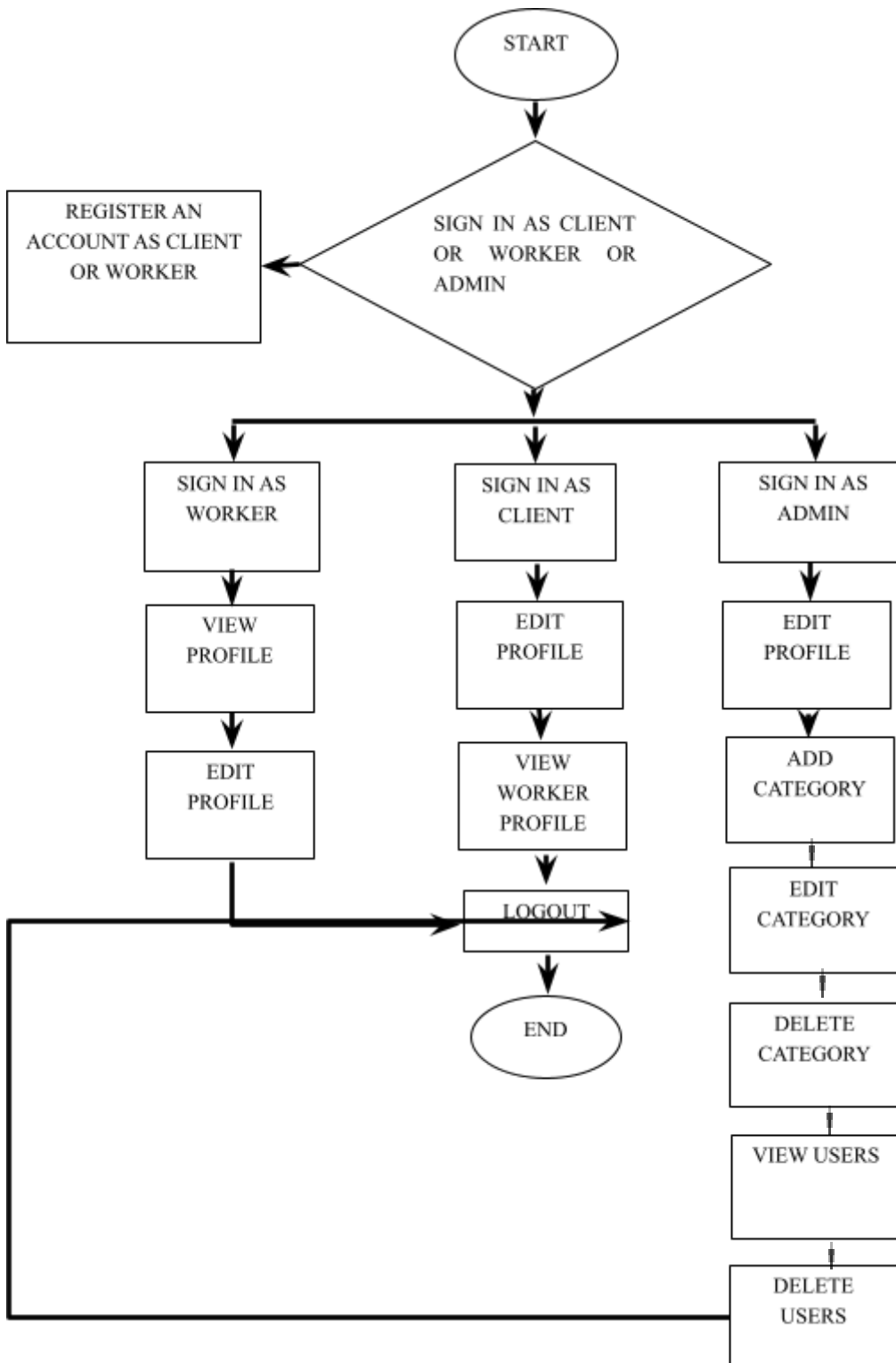


Figure 1:Function diagram

For the three main roles on the platform—clients, workers, and administrators—the function diagram shows the user flow. User registration as a client or employee or sign-in is how the procedure starts. Clients can view worker profiles to choose service providers and workers can access and amend their profiles after logging in. More access is granted to administrators, who can examine or remove user profiles in addition to managing service categories (add, update, or delete). Each user can log out to end their session after doing their assigned tasks. Assuring easy navigation and productive engagement inside the platform, the diagram highlights the streamlined functionality customized for every position.

### **3.3.4 Methodological approach**

Methodology is “‘a contextual framework’ for research, a coherent and logical scheme based on views, beliefs, and values, that guides the choices researchers [or other users] make”. (Kara, Helen, 2015) It is the road map that acts as an itinerary for researchers to accomplish the goals in the journey of research. (Kara, Helen, 2015)

#### **3.3.4.1 Data collection techniques**

Data collection is a methodical process of gathering and analysing specific information to proffer solutions to relevant questions and evaluate the results. It focuses on finding out all there is to a particular subject matter. Data is collected to be further subjected to hypothesis testing which seeks to explain a phenomenon. (Vuong, Quan-Hoang, 2018)

Data collection techniques/tools refer to the devices/instruments used to collect data, such as Observation, Questionnaire, Interview and Documentation. (Niglas, Katrin, 2020)

However, the researcher adopted to use Observation, Interview, and documentation.

##### **i. Observations**

Observation, as the name implies, is a way of collecting data through observing. It is a method of data collection in which researchers observe within a specific research field. It is sometimes referred to as an unobtrusive method. (Prof. Melanie Bryant, 2019)

##### **ii. Documentation**

Documentation is collecting, abstracting, and coding of printed or written information for future reference it is everything, which may be preserved or represented in order to serve as evidence for some purpose (Rogers, Carl R, 2015)

### 3.3.4.2 Software Development Methodology

Software development methodology (also known as SDM) refers to structured processes involved when working on a project. It is a blend of design philosophies and pragmatic realism that stretches back to the early days of computing. The goal is to provide a systematic approach to software development. It is also known as a software development life cycle (SDLC). The methodology may include the pre-definition of specific deliverables and artefacts that are created and completed by a project team to develop or maintain an application. (Geoffrey Elliott , 2015)

Those methodologies include waterfall, prototyping, iterative and incremental development, spiral development, rapid application development, and extreme programming. (Geoffrey Elliott , 2015)

However, the researcher opted to use Agile model as software development methodology.

Agile methodology is a project management framework that breaks projects down into several dynamic phases, commonly known as sprints.

The Agile framework is an iterative methodology. After every sprint, teams reflect and look back to see if there was anything that could be improved so they can adjust their strategy for the next sprint. (Asana, [Sarah Laoyan](#) February 2nd, 2024)



**Figure 2: agile model**

The sequences phases in agile model method are the following:

- **Requirements gathering:** In this phase, you must define the requirements. You should explain business opportunities and plan the time and effort needed to build the project. Based on this information, you can evaluate technical and economic feasibility.
- **Design the requirements:** When you have identified the project, work with stakeholders to define requirements. You can use the user flow diagram and show how it will apply to your existing system.
- **Construction/ iteration:** When the team defines the requirements, the work begins. Designers and developers start working on their project, which aims to deploy a working product. The product will undergo various stages of improvement, so it includes simple, minimal functionality.
- **Testing:** In this phase, the Quality Assurance team examines the product's performance and looks for the bug.
- **Deployment:** In this phase, the team issues a product for the user's work environment.
- **Feedback:** After releasing the product, the last step is feedback. In this, the team receives feedback about the product and works through the feedback.

#### **Advantages:**

- To ensure competitiveness, changes to functional requirements are incorporated into the development process.
- Each iteration of the project is brief and open.
- Frequent Delivery
- Face-to-Face Communication with clients.
- Efficient design and fulfils the business requirement.
- Anytime changes are acceptable.
- It reduces total development time.

#### **Disadvantages:**

- There could be conflicts between new requirements and the current architecture.



- There is a chance that the project will take longer than anticipated given all the repairs and revisions.
- Due to the shortage of formal documents, it creates confusion and crucial decisions taken throughout various phases can be misinterpreted at any time by different team members.
- Due to the lack of proper documentation, once the project completes and the developers allotted to another project, maintenance of the finished project can become a difficulty.

### **3.3.4.3. System Design Methodology**





Systems Analysis and Design (SAD) is a broad term for describing methodologies for developing high quality Information System which combines Information Technology, people and Data to support business requirement. The SAD technique is not only limited to IT systems and can be used to create just about anything, from a family house to the international space station. But there is no silver bullet in simplifying the development of computer systems. This principle is still true today. In other words, there is no single, simple technique that developers can use to ensure successful Information Technology (IT) projects. However, there are development methodologies that can be followed which will greatly assist an IT professional in developing and enhancing systems. A methodology is essentially a procedure to get something done. A development methodology can be thought of as a roadmap. While a roadmap for a traveller will provide the details from driving from point A to point B, a development methodology will provide the IT professional with uidelines for taking a system from conception through implementation and beyond.

(Lonnie D, Bentley, 2016)

However, the researcher opted to use Structured System Analysis and Design Methodology. Structured Systems Analysis and Design Method (SSADM), originally released as methodology, is a systems approach to the analysis and design of information systems. SSADM was produced for the Central Computer and Telecommunications Agency, a UK government office concerned with the use of technology in government, from 1980 onwards. SSADM is a waterfall method for the analysis and design of information systems. (Mike Goodland, Karel Riha, 2015)

#### **i. Dataflow diagram**

A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. Data flowcharts can range from simple, even hand-drawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled. They can be used to analyze an existing system or model a new one. Like all the best diagrams and charts, a DFD can often visually “say” things that would be hard to explain in words, and they work for both technical and nontechnical audiences, from developer to CEO. That’s why DFDs remain so popular after all these years. While they work well for data flow software and systems, they are less applicable nowadays to visualizing interactive, real-time or database-oriented software or systems.(lucidchart,2020)

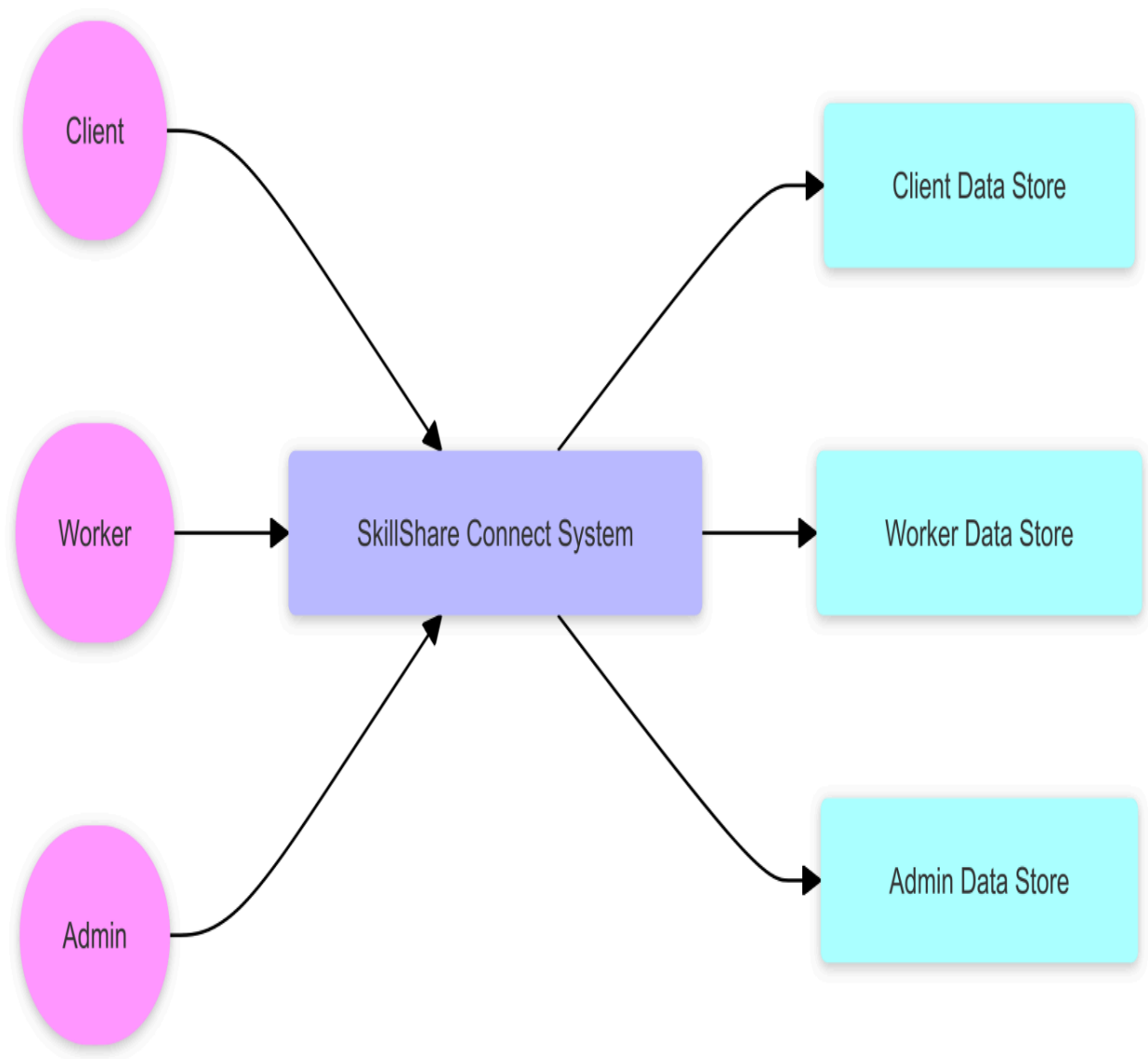
<b>External Entity:</b> System Admin		External Entity: An Entity is a source or destination of data flow, which is outside the area of study. Only those entities that originate on a business process diagram. The symbol used is an oval containing a meaningful and unique identifier
		Process: A process shows a transformation or manipulation of data flows within the system. The symbol used is a rectangular box.
		Data store: For data entry and storing.
		Data flow: A data flow shows the flow of information from its source to its destination. A data flow is represented by line, with arrowheads showing the direction of flow.

Customer,client.

**Process:** Searching customer's information, Book the car, Contact, Display transactions, Add or upload files and information for the Admin.

**Data Store:** Customers, tailor, Tailor reservation, customer's information.

1. Data flow diagram level 0

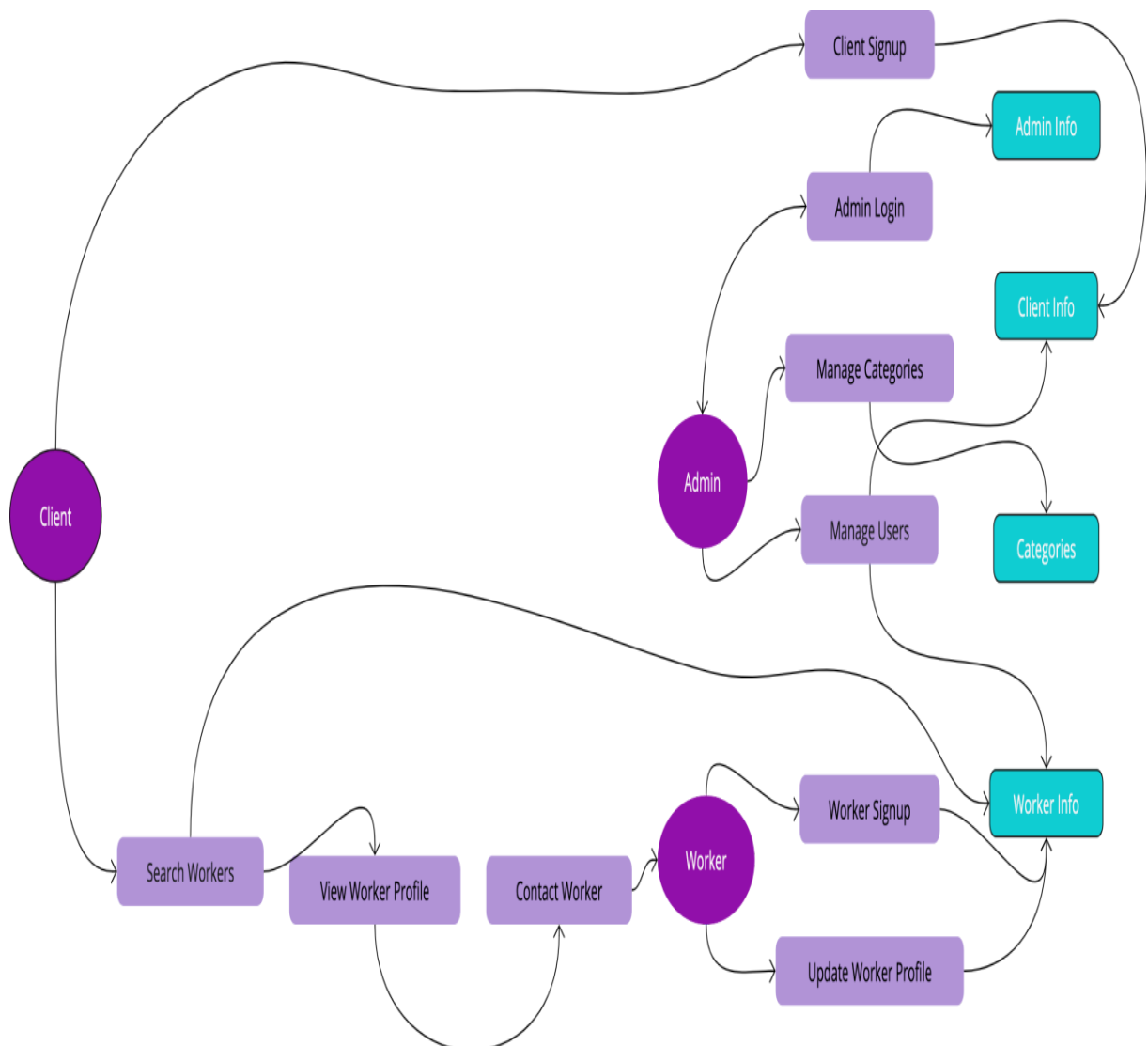


**Figure 3:Dfd level 0**

This diagram illustrates the data flow in the SkillShare Connect System . It shows how Clients , Workers , and Admins interact with the system, which connects them to their respective data stores:

- Clients access the system to interact with the Client Data Store , where their personal information and task-related data are stored.
- Workers interact with the Worker Data Store to manage their profiles, availability, and job assignments.
- Admins use the Admin Data Store to manage platform operations, categories, and other administrative tasks.

- **Data flow diagram level 1**

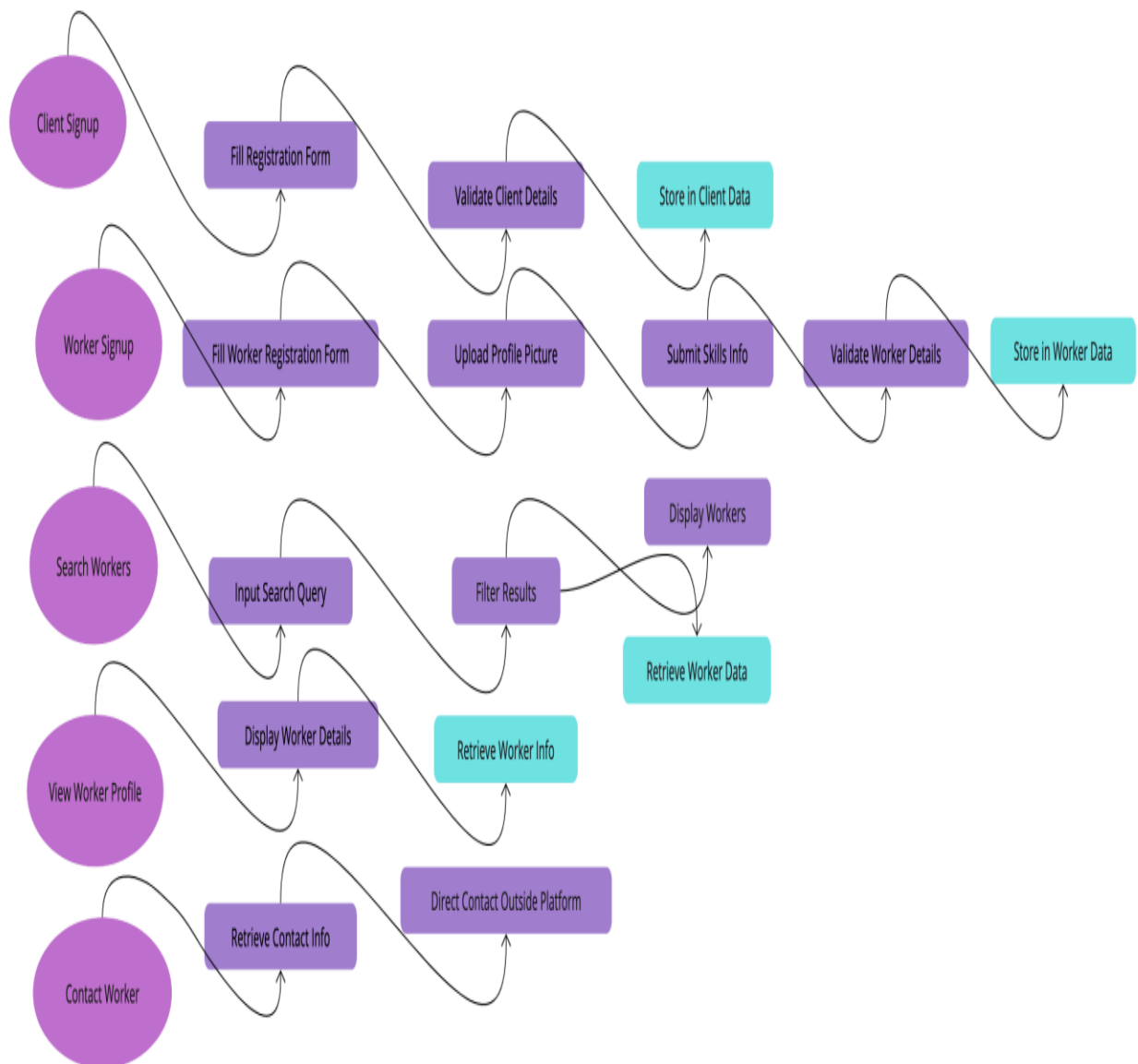


**Figure 4:Dfd level 1**

The platform offers various features for clients and workers. Clients can search for workers based on service categories, view their profiles, and contact workers for direct communication. Workers can register and update their profiles, while administrators oversee client registration, manage categories, and manage users.

The system also manages the database, which includes key data points like Client Info, Worker Info, Admin Info, and Categories. The focus is on simplicity, efficient data management, and enabling direct external communication between clients and workers. The platform's DFD Level 1 focuses on user interaction and admin efficiency, ensuring smooth functioning and user interaction.

- **Dfd diagram level 2**



**Figure 5:Dfd level 2**

### 1. Client Signup Process:

- **Fill Registration Form (Client):** The client starts by filling in a registration form with their details.
  - **Information Flow:** Client's input data (name, contact, etc.) flows to the next process.
- **Validate Client Details:** The system validates the information provided by the client to ensure correctness and completeness.

- **Information Flow:** The client's registration data is checked.
- **Store in Client Data:** Once the client's details are validated, the system stores the client information in the **Client Data** storage.
  - **Information Flow:** Validated client details are saved for future use.

## 2. Worker Signup Process:

- **Fill Worker Registration Form:** The worker submits their personal and professional information through a registration form.
  - **Information Flow:** The worker's form data (personal info, skills, etc.) is sent to the next step.
- **Upload Profile Picture:** The worker uploads their profile picture.
  - **Information Flow:** The profile picture is sent to the system and associated with the worker's registration.
- **Submit Skills Info:** The worker provides information about their skills, expertise, and qualifications.
  - **Information Flow:** Skill data is transferred to the validation process.
- **Validate Worker Details:** The system verifies the worker's submitted information (registration details, skills, profile picture).
  - **Information Flow:** Validation of worker data takes place here.
- **Store in Worker Data:** Once the worker's data is validated, it is saved in the **Worker Data** storage for later retrieval.
  - **Information Flow:** The validated worker information is stored.

## 3. Searching for Workers (Client Process):

- **Input Search Query:** The client inputs a search query to look for workers based on skills, location, or other criteria.
  - **Information Flow:** The search query is passed to the next process for filtering.
- **Filter Results:** The system filters the list of workers according to the search query (e.g., matching workers with the right skills).

- **Information Flow:** Filtered search results are produced based on the query.
- **Retrieve Worker Info:** The filtered worker data is retrieved from the **Worker Data** storage.
  - **Information Flow:** The system fetches relevant worker information (profiles, skills, etc.).
- **Display Workers:** The list of matching workers is displayed to the client.
  - **Information Flow:** Filtered worker profiles are shown to the client.

#### 4. Viewing Worker Profile:

- **Display Worker Details:** When the client selects a specific worker profile, the system retrieves and displays more detailed worker information.
  - **Information Flow:** Worker's detailed information (skills, experience, contact info) is retrieved and displayed.
- **Retrieve Contact Info:** If the client wishes to contact the worker, the system retrieves the worker's contact information from the data storage.
  - **Information Flow:** Worker's contact information is provided to the client.

#### 5. Contacting the Worker:

- **Direct Contact Outside Platform:** After the client retrieves the worker's contact information, they can contact the worker directly, outside of the platform.
  - **Information Flow:** The contact information flows out of the platform into external communication channels (e.g., phone, email).

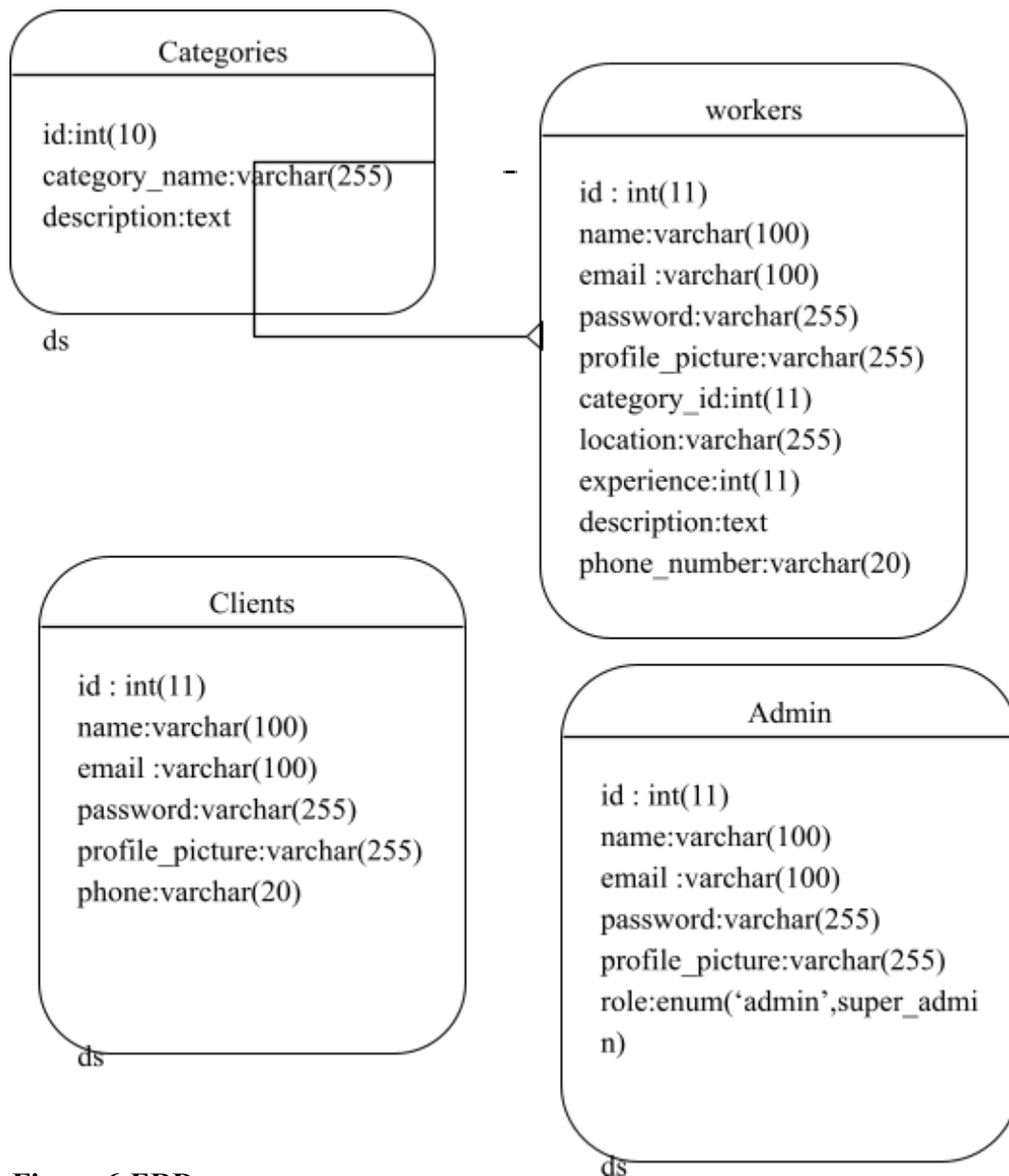
#### ii. Entity Relationship Diagram (ERD)

An entity–relationship model (ER model for short) describes interrelated things of interest in a specific domain of knowledge. A basic ER model is composed of entity types (which classify the things of interest) and specifies relationships that can exist between instances of those entity types. (Chen, Peter, 2018)

In software engineering, an ER model is commonly formed to represent things that a business needs to remember in order to perform business processes. Consequently, the ER



model becomes an abstract data model that defines a data or information structure which can be implemented in a database, typically a relational database. (Chen, Peter, 2018)



**Figure 6:ERD**

### iii. Data dictionary

Data dictionary, or metadata repository, as defined in the IBM Dictionary of Computing, is a "Centralized repository of information about data such as meaning, relationships to other data, Origin, usage, and format". Oracle defines it as a collection of tables with metadata. The term can have one of several closely related meanings pertaining to databases and database management systems. (Ramez Elmasri, Shamkant B. Navathe, 2015) Researcher employed a data dictionary can be consulted to understand where a data item fits in the

structure, what values it may contain, and basically what the data item means in real-world terms. The tables created in the database of this system are: admin, citizen, these are the attributes we find in these tables:

**Table 1 : structure for table admin**

**Table**

Column	data type	attributes	description
admin_id	INT UNSIGNED	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each ac
name	VARCHAR(255)	NOT NULL	the admin full's name
email	VARCHAR(255)	NOT NULL, UNIQUE	the admin full's address
password	VARCHAR(255)	NOT NULL	the admin hashed password
role	ENUM	NOT NULL DEFAULT 'ADMIN'	the role of the admin (admin, super_admin)
profile_picture	VARCHAR(255)	NULL	Admin's profile picture

## 2: Clients

column	Data type	Attributes	Description
client_id	INT UNSIGNED	PRIMARY KEY, AUTO_INCREMENT	unique identifier
name	VARCHAR (255)	NOT NULL	the client's full name
email	VARCHAR (255)	NOT NULL, UNIQUE	the client's email address
phone	VARCHAR (20)	NOT NULL	the client's phone number
password	VARCHAR (255)	NOT NULL	the client's hashed password
profile_picture	VARCHAR(255)	NULL	client's profile picture

**Table 3: Workers**

Column	Data Type	Attributes	Descriptions
worker_id	INT UNSIGNED	PRIMARY KEY,AUTO_INCREMENT	unique identifier for each wo
name	VARCHAR(255)	NOT NULL	the worker's full name
email	VARCHAR(255)	NOT NULL ,UNIQUE	the worker's email address
phone-number	VARCHAR(255)	NOT NULL	the worker's phone number
password	VARCHAR(255)	NOT NULL	the worker's password
category_id	INT UNSIGNED	FOREIGN KEY REFERENCES categories(id)	the services category the belongs to
description	TEXT		description of the worker's s services
location	VARCHAR(255)		the worker's location(optiona
profile_picture	VARCHAR(255)	NULL	worker's profile picture
experience	INT	NOT NULL	worker's years of experience

**Table 4: categories**

column	Data Type	Attributes	Description
category_id	INT UNSIGNED	PRIMARY KEY,AUTO INCREMENT	unique identifier for each category
category_name	VARCHAR(255)	NOT NULL	name of the service category
description	TEXT		description of the category

## CHAPTER 4: SYSTEM IMPLEMENTATION

### 4.1. Implementation and Coding

#### 4.1.1. Introduction

In this chapter, we will discuss the technologies used to implement the system and provide the description of the system's functionalities through screenshots.

## **4.1.2. Description of Implementation tools and technology**

### **4.1.2.1 HTML**

Hypertext Markup Language (HTML) is the standard markup language for documents designed to be displayed in a web browser.

([wikipedia.org](https://www.wikipedia.org), 2020)

### **4.1.2.2 CSS**

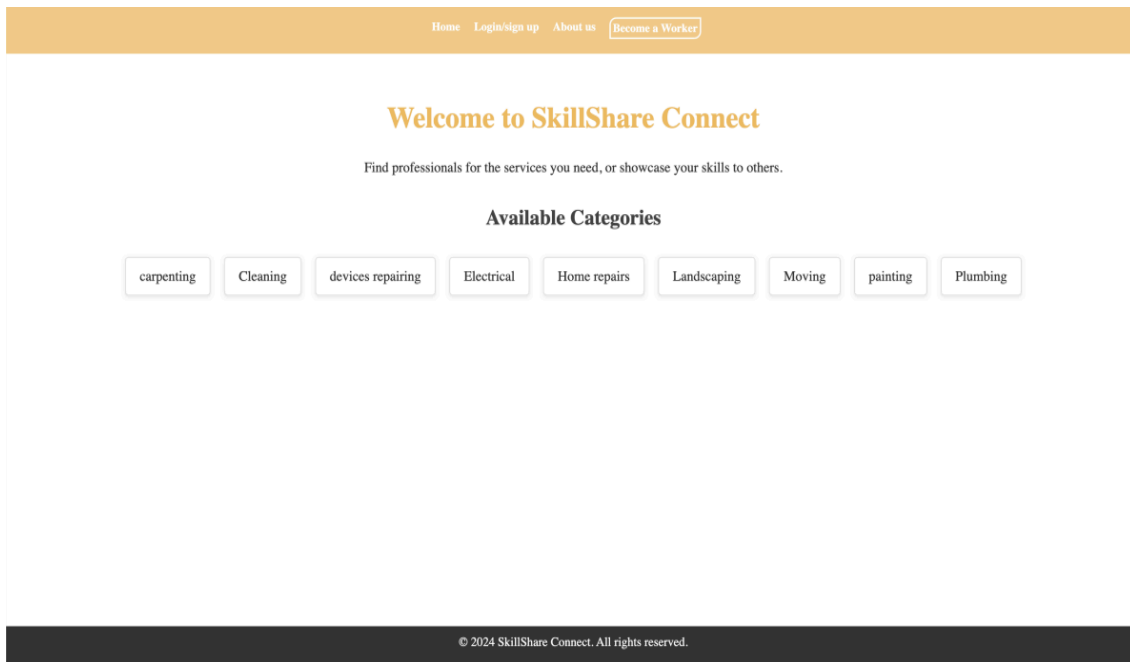
Cascading Style Sheets, fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable.

([tutorialspoint.com](https://www.tutorialspoint.com), 2020)

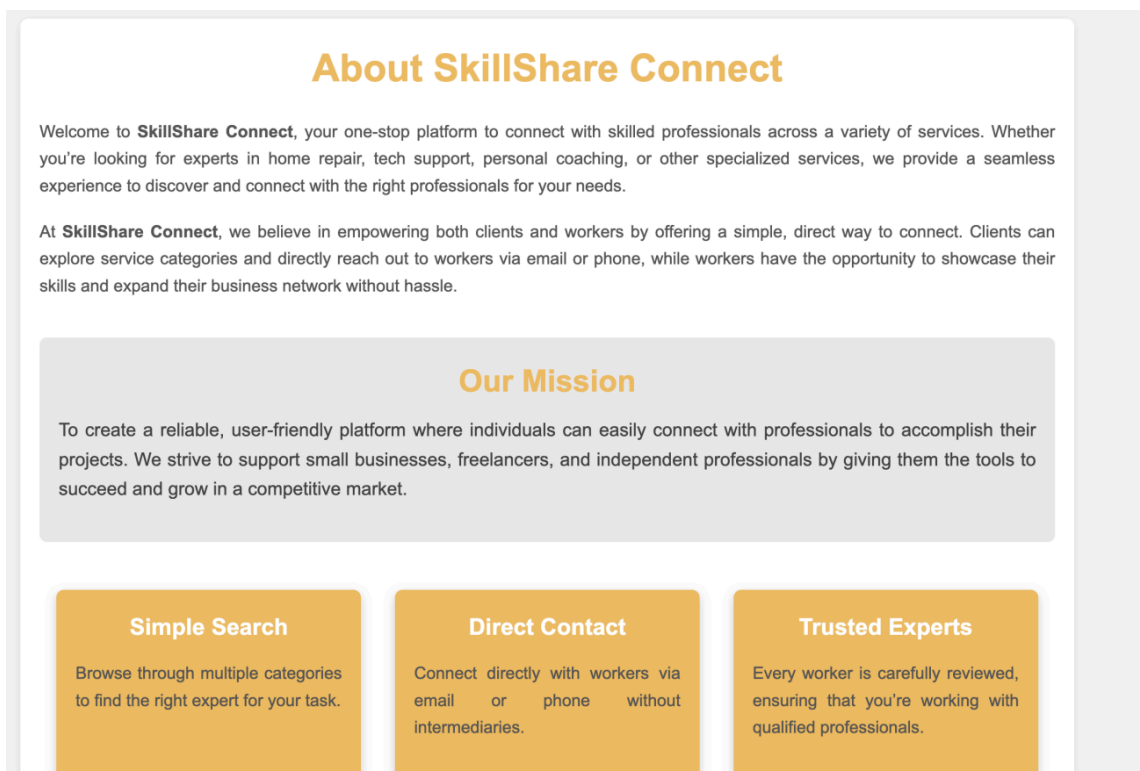
### **4.1.2.3 SQL**

SQL (pronounced "ess-que-el") stands for Structured Query Language. SQL is used to communicate with a database. According to ANSI (American National Standards Institute), it is the standard language for relational database management systems. SQL statements are used to perform tasks such as update data on a database, or retrieve data from a database. ([sqlcourse.com/intro.html](https://www.sqlcourse.com/intro.html)).

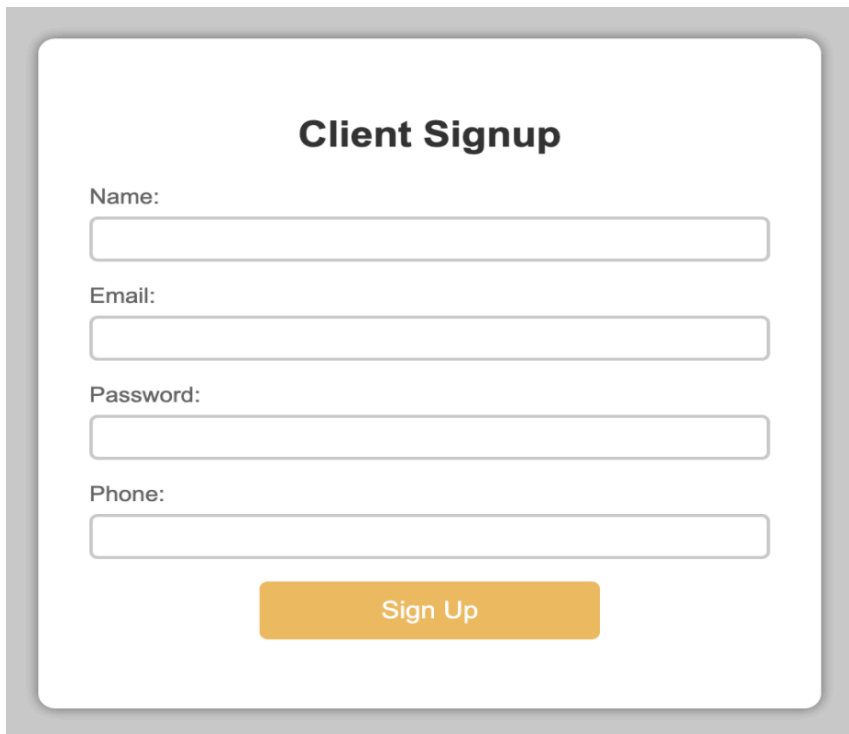
## **4.1.3. Screen shorts and source codes**



**Figure 7 Home page**”welcoming message to the website and *showing available categories*”

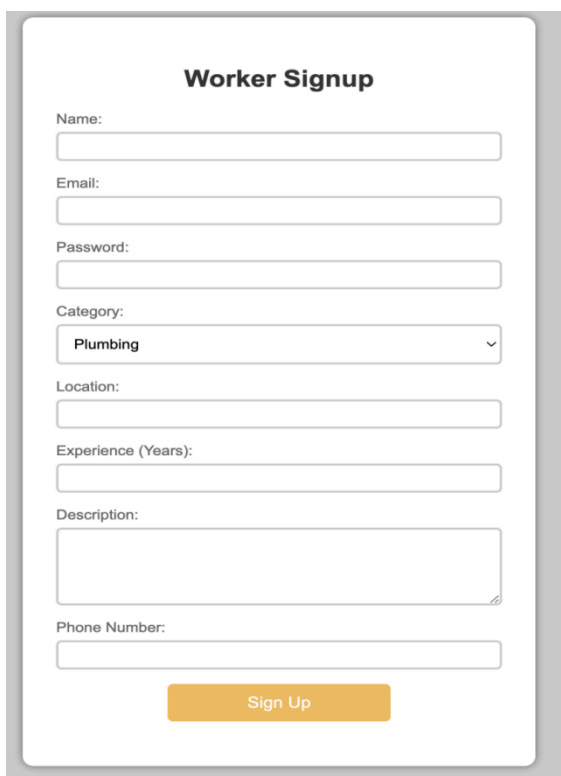


**Figure 8: About us page**



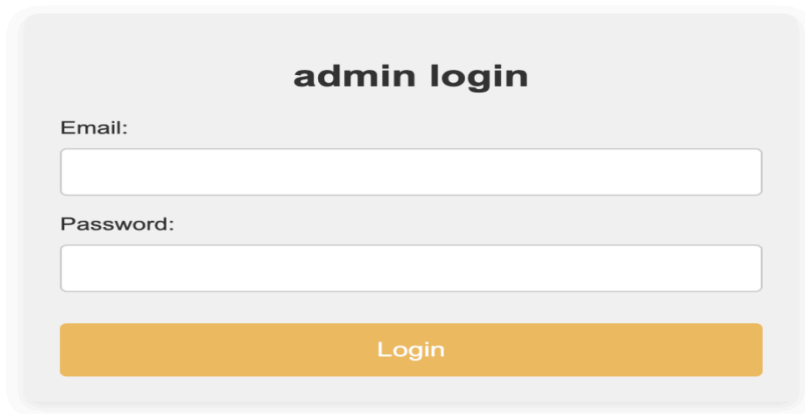
The image shows a 'Client Signup' form. It has a title 'Client Signup' at the top. Below the title are four input fields: 'Name:', 'Email:', 'Password:', and 'Phone:'. Each field is a simple rectangular box. At the bottom of the form is a yellow button with the text 'Sign Up'.

**Figure 9 Client Signup Page:** " where users can register to access the platform as clients and start searching for service providers."



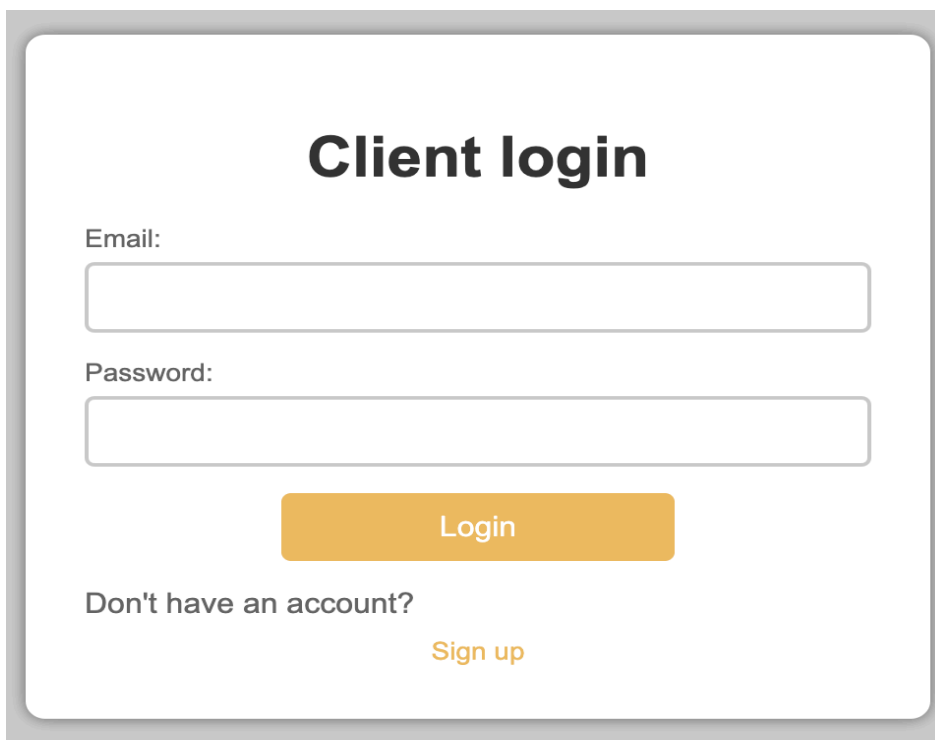
The image shows a 'Worker Signup' form. It has a title 'Worker Signup' at the top. Below the title are several input fields: 'Name:', 'Email:', 'Password:', 'Category:', 'Location:', 'Experience (Years):', 'Description:', and 'Phone Number:'. The 'Category:' field is a dropdown menu with 'Plumbing' selected. The 'Description:' field is a larger text area with a small icon in the bottom right corner. At the bottom of the form is a yellow button with the text 'Sign Up'.

**Figure 10 Worker Signup Page:** "Screenshot of the worker signup page, where professionals can register to offer their services and manage their profiles."



The image shows a login form titled "admin login" in a light gray rounded rectangle. At the top, the text "admin login" is centered in bold black font. Below it, the label "Email:" is followed by a white rectangular input field. Underneath, the label "Password:" is followed by another white rectangular input field. At the bottom of the form is a solid orange button with the word "Login" centered in white text.

**Figure 11: Admin Login Page:** "Screenshot of the admin login page, showcasing how administrators access the backend for managing categories and users."



The image shows a login form titled "Client login" in a white rounded rectangle with a gray border. At the top, the text "Client login" is centered in bold black font. Below it, the label "Email:" is followed by a white rectangular input field. Underneath, the label "Password:" is followed by another white rectangular input field. At the bottom of the form is a solid orange button with the word "Login" centered in white text. Below the button, the text "Don't have an account?" is displayed, followed by the text "Sign up" in orange font.

**Figure 12 Client Login Page** "Screenshot of the login pages for clients."

**worker login**

Email:

|

Password:

Login

Don't have an account

Sign up

**Figure 13 Worker Login Pages** "Screenshot of the login page for workers, "

Home Login/sign up About us [Become a Worker](#)

Welcome, moyz!

Services

Plumbing Electrical Cleaning Landscaping painting carpenting Moving Home repairs devices repairing

Workers

Profile Picture  
**moyz**  
Service: Electrical  
Location: razizi  
[View Profile](#)

Profile Picture  
**John**  
Service: Landscaping  
Location: kigali  
[View Profile](#)

Profile Picture  
**Sam**  
Service: Cleaning  
Location: kigali  
[View Profile](#)

Profile Picture  
**Luc**  
Service: Moving  
Location: kigali  
[View Profile](#)

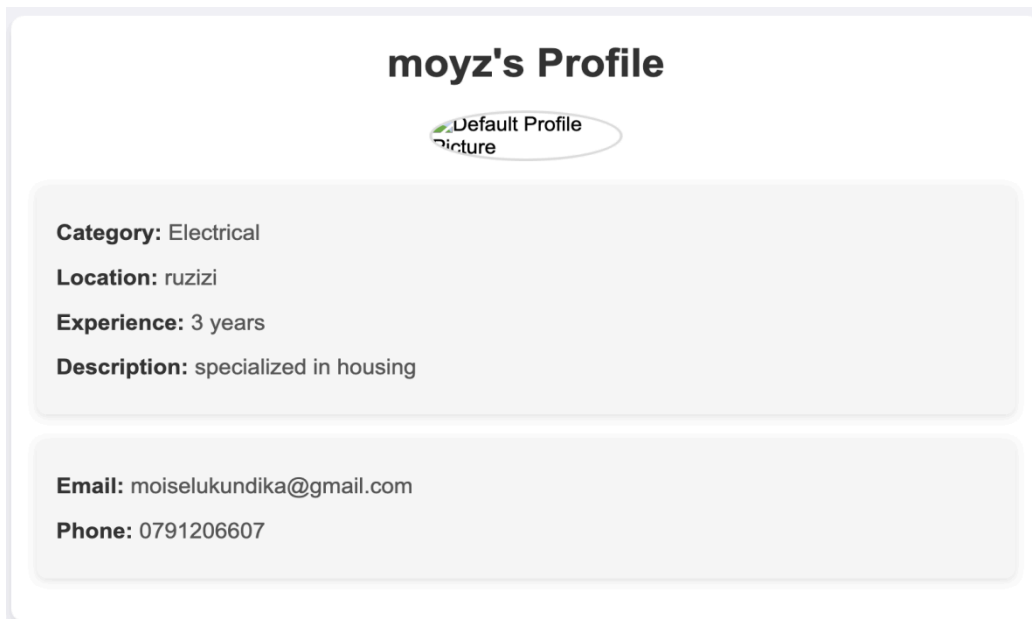
Profile Picture  
**Dan**  
Service: Plumbing  
Location: gisenyi  
[View Profile](#)

[Account Settings](#)


© 2024 SkillShare Connect. All rights reserved.

**figure 14 Client Dashboard:** "Screenshot of the client dashboard, displaying a welcome message, category search functionality, a sample of workers, and a link to manage the client's profile."





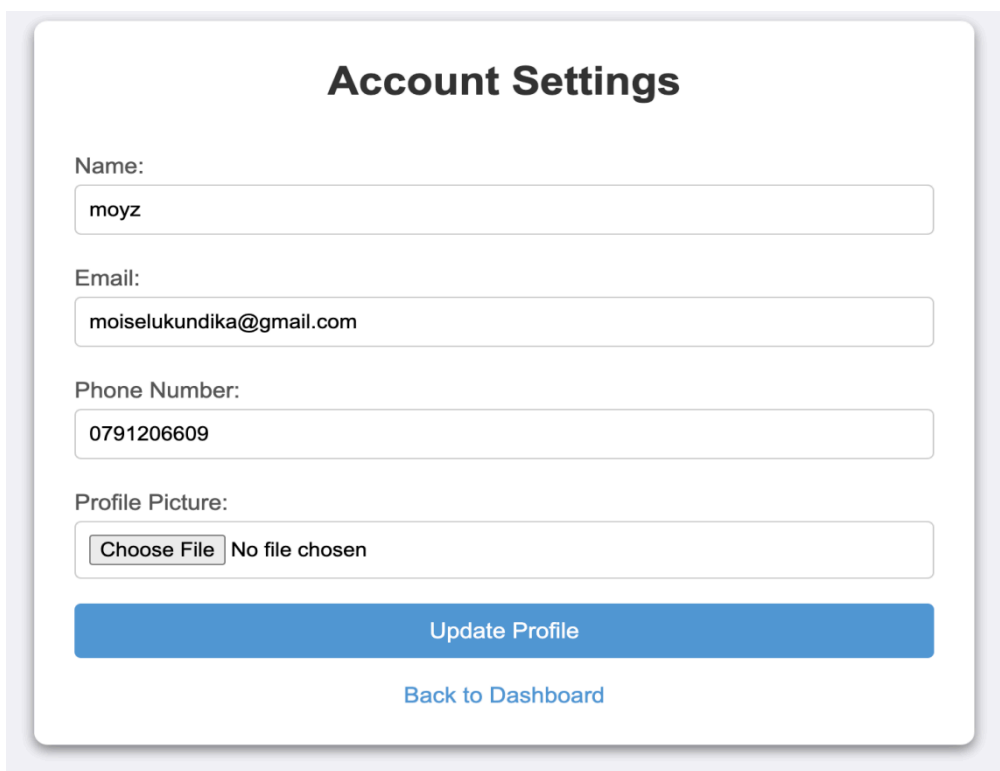
**moyz's Profile**

 Default Profile Picture

**Category:** Electrical  
**Location:** ruzizi  
**Experience:** 3 years  
**Description:** specialized in housing

**Email:** moiselukundika@gmail.com  
**Phone:** 0791206607

**Figure 15: Worker Profile Page:** "Screenshot of a worker's profile page, demonstrating how clients can view detailed worker information and contact them for services."



**Account Settings**

Name:

Email:

Phone Number:

Profile Picture:  
 No file chosen

[Back to Dashboard](#)

**Figure 16 Client account setting:**

## Edit Your Profile

Name:

Email:

Category:

Location:

Experience (years):

Description:

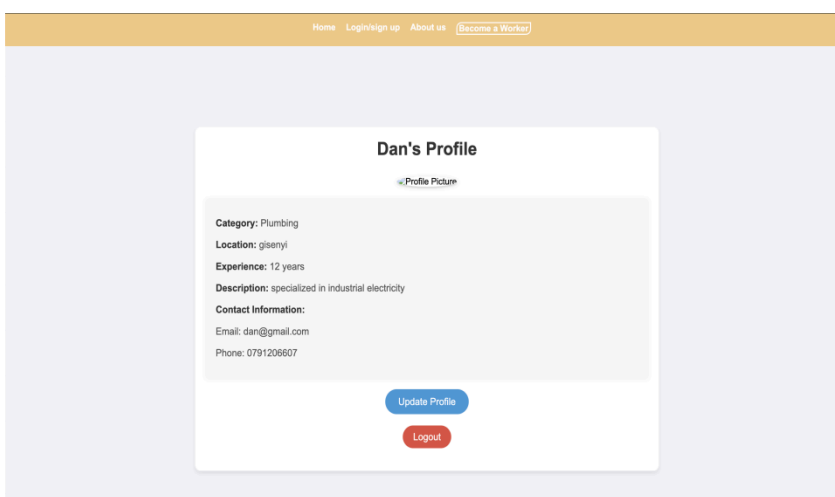
Phone Number:

---

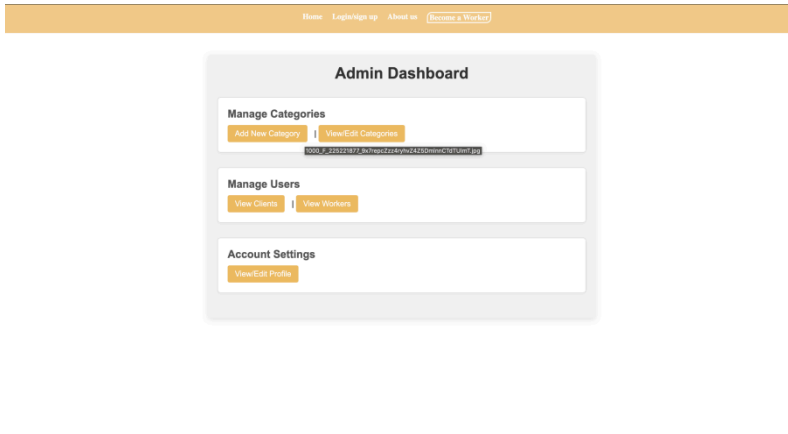
Profile Picture:  
 No file chosen

[Back to Dashboard](#)

**Figure 17 Client account setting:**



**Figure 18 Worker Dashboard:** "Screenshot of the worker dashboard, showing profile update options and worker-specific features for managing their account and services."



**Figure 19 Admin Dashboard:** "Screenshot of the admin dashboard, highlighting category management, user deletion functionalities, and the ability to view and edit the admin profile."

**Figure 20 : Admin Profile:** "Screenshot of the admin profile page, showing admin details with options to view and update their profile."

Category List		
ID	Name	Actions
1	Plumbing	Edit   Delete
2	Electrical	Edit   Delete
3	Cleaning	Edit   Delete
4	Landscaping	Edit   Delete
6	painting	Edit   Delete
7	carpentering	Edit   Delete
9	Moving	Edit   Delete
11	Home repairs	Edit   Delete
13	devices repairing	Edit   Delete

Back to Dashboard

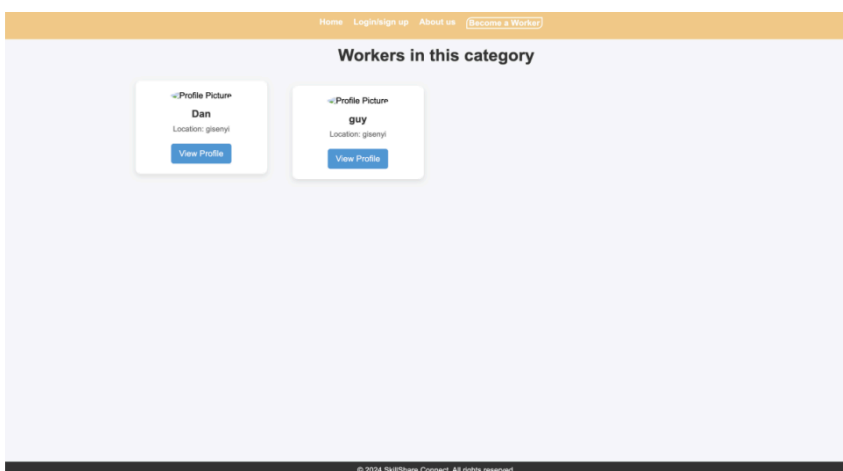
**Figure 21 Category List Management (Admin):** "Screenshot of the admin interface for managing categories, where the admin can add, edit, and delete service categories."

Worker List					
ID	Name	Email	Phone Number	Category	Actions
1	moyz	moiselukundika@gmail.com	0791206607	Electrical	Delete
2	john	john@gmail.com	0791206607	Landscaping	Delete
3	Sam	sam@gmail.com	0791206607	Cleaning	Delete
4	Luc	luc@gmail.com	0791206607	Moving	Delete
9	Dan	dan@gmail.com	0791206607	Plumbing	Delete
12	guy	guy@gmail.com	0791206607	Plumbing	Delete

**Figure 22 Worker List Management:** "Screenshot showing the admin's view of the worker list, with options to manage and delete worker accounts."

Client List				
ID	Name	Email	Phone Number	Actions
3	moyz	moiselukundika@gmail.com	0791206609	Delete
6	lknd	moyzluk@gmail.com	0791206607	Delete
7	lknd	lknd@gmail.com	0791206607	Delete

**Figure 23 Client List Management:** "Screenshot showing the admin's view of the client list, with options to manage and delete worker accounts."



**Figure 24 Task Matching Flow:**

"Screenshot illustrating how clients can browse workers by category and directly connect with them for service requests."

## 4.2. Testing

### 4.2.1. Introduction

Testing is a critical phase in the software development lifecycle that ensures the functionality, reliability, and performance of the SkillShare Connect platform. Various levels of testing were conducted, including unit testing, validation testing, integration testing, system testing, and acceptance testing. The primary objective was to identify and resolve defects, verify that components work seamlessly together, and confirm that the platform meets the specified user requirements.

### 4.2.2. Unit Testing Outputs

Unit testing focuses on verifying the functionality of individual components in isolation. For SkillShare Connect, unit tests were carried out on key functions such as:

- Client and Worker Registration : This tested the ability of users to register with valid input while handling cases of invalid input (e.g., missing fields, duplicate email entries).
- Login Authentication : Tests were performed to verify that users could log in with valid credentials, and appropriate error messages were displayed for invalid login attempts.
- Profile Management : Tests ensured that clients and workers could update their profiles, including the successful upload of profile pictures.
- Admin Category Management : This validated that the admin could add, edit, and delete categories without errors.

Unit Testing Results :

- All individual modules passed their respective test cases.
- Error handling mechanisms correctly identified and flagged invalid input formats.
- Profile updates were processed and saved successfully, and image uploads functioned as expected.

### 4.2.3. Validation Testing Outputs

Validation testing ensured that the system conformed to the user requirements and performed as expected in real-world conditions. The key areas tested include:

- **User Authentication** : Verifying that only registered users could access their respective dashboards.
- **Worker Search** : Validating the ability of clients to search for workers by category and view detailed profiles.
- **Direct Communication** : Ensuring that worker contact details were displayed correctly to clients for direct communication.

#### **Validation Testing Results** :

- User input was correctly validated, and session management worked properly across client, worker, and admin dashboards.
- The search functionality for workers returned accurate results, with worker profiles displaying the correct contact information.

#### **4.2.4. Integration Testing Outputs**

Integration testing evaluated the interactions between different components to ensure they worked cohesively. The key integrations tested were:

- **Database Integration** : Ensuring that all user types (clients, workers, and admins) could read and write data correctly.
- **Worker Search and Profile Viewing** : Verifying that workers were correctly listed by category and that their profiles were accurately displayed.
- **Admin Module** : Ensuring that actions performed by the admin, such as deleting users or managing categories, were accurately reflected in the system.

#### **Integration Testing Results** :

- All integrated components functioned without any issues or data inconsistencies.
- Database queries for fetching user and category data were successfully executed, with correct data being displayed on the front-end.

#### **4.2.5. Functional and System Testing**

Functional and system testing assessed the overall behavior of the platform in its fully integrated environment. The major functionalities tested were:

- **Client and Worker Dashboards** : This involved verifying profile management, worker searches, and category-based filtering.

- Admin Dashboard : The functionality of category management and user management (viewing and deleting users) was evaluated.
- User Flows : End-to-end user journeys, from registration to contacting workers, were tested.

#### Functional and System Testing Results :

- The platform performed as expected, handling user inputs and requests across all dashboards effectively.
- No functional defects were observed in navigation, data handling, or session management throughout the system.

#### **4.2.6. Acceptance Testing Report**

Acceptance testing was conducted to ensure that the system met the specified requirements and functioned correctly from the end user's perspective. This included client, worker, and admin testing. The key acceptance criteria were:

- Client Experience : Ensuring that clients could easily register, log in, search for services, and contact workers.
- Worker Experience : Verifying that workers could register, update their profiles, and be searchable by clients.
- Admin Capabilities : Ensuring that admins could manage categories and users efficiently.

#### Acceptance Testing Results :

- The system met all predefined acceptance criteria, with clients, workers, and admins able to complete their tasks without issues.
- User feedback indicated that the platform was intuitive and functionally sound, with all core features operating as expected.

## CHAPTER 5: CONCLUSIONS AND RECOMMENDATIONS

### Conclusion

The goal of this dissertation was to investigate the creation of a platform called the "Online contractor-client management system," which was intended to solve the inefficiencies in the current small-scale task contractor-client matchmaking process. This project aims to enable smooth, dependable, and effective relationships between service providers and clients by taking inspiration from well-functioning professional networking platforms. The study brought to light the gig economy's increasing importance as well as the urgent need for a streamlined system that deals with the trust difficulties, inefficiencies, and fragmentation that come with using traditional techniques.

The goal of this dissertation was to investigate the creation of a platform called the "Online contractor-client management system," which was intended to solve the inefficiencies in the current small-scale task contractor-client matchmaking process. This project aims to enable smooth, dependable, and effective relationships between service providers and clients by taking inspiration from well-functioning professional networking platforms. The study brought to light the gig economy's increasing importance as well as the urgent need for a streamlined system that deals with the trust difficulties, inefficiencies, and fragmentation that come with using traditional techniques.

To sum up, the "Online contractor-client management system" is a big step in the right direction toward filling up the holes in the gig economy. It provides a guide for building an adaptable, inclusive, and effective platform that meets the changing requirements of contractors and clients.

### Recommendations

In the light of the advancements made in technology in this century and considering our experience while building this system, below are our recommendations:

- I urge the department to emphasise on the use of artificial intelligence and data science as they play a very important role in nowadays technologies;
- I urge the department to emphasize on installing an entrepreneurial mindset to students so that they may learn to make businesses out of softwares they build;



- Finally, the department should take into consideration the competitiveness of today's tech-world and introduce more challenges into the curriculum to encourage creativity and critical thinking.

## **Future works**

### **1. In-app Messaging and Communication**

Introduce an in-app messaging system that allows clients and workers to communicate directly within the platform.

### **2. Integrated Payment System**

Implement an integrated payment gateway to facilitate secure transactions between clients and workers directly on the platform.

### **3. Ratings and Reviews**

Allow clients to rate and review workers after the completion of a service.

### **4. Advanced Search and Filtering**

Enhance the search functionality to include advanced filters such as price range, worker ratings, location proximity, and availability.

### **5. Service Bundles and Packages**

Allow workers to create and offer service bundles or packages at discounted rates.

### **6. Mobile Application**

Develop a mobile application for iOS and Android to make the platform more accessible on the go.

### **7. Booking and Scheduling System**

Introduce a booking system that allows clients to schedule services with workers directly through the platform.

### **8. Notification System**

Implement a notification system to alert clients and workers about important events, such as new messages, booking confirmations, and profile updates.

9. Allow workers and clients to further customize their profiles with portfolios, testimonials, and detailed service descriptions.

### **10. Localization and Multilingual Support**

Introduce multilingual support and localization options to cater to a broader audience.

## REFERENCES

- Smith, J. (2022). Gig Economy Platforms: An Overview of Opportunities and Challenges. TechPress Publishing.
- Brown, A., & Davis, L. (2021). "Digital Platforms and the Future of Work." Journal of Digital Innovation , 15(2), 45-60.
- Williams, S. (2020). User Experience Design in Online Marketplaces. UXDesign Press.
- Johnson, M. (2019). "The Impact of the Gig Economy on Modern Work Culture." Harvard Business Review. Retrieved from <https://hbr.org/2019/11/gig-economy-impact>.
- McGregor, T. (2020). "Ensuring Safety and Trust in Digital Service Platforms." Digital Economy Insights , 22(4), 22-30.
- TaskRabbit. (2021). TaskRabbit Trust & Safety Guidelines. Retrieved from <https://www.taskrabbit.com/trust-safety>.
- Fiverr. (2023). How Fiverr Works. Retrieved from <https://www.fiverr.com/resources>
- Thumbtack. (2023). About Us. Retrieved from <https://www.thumbtack.com/about>
- Upwork. (2022). Safety and Security. Retrieved from <https://www.upwork.com/safety-security>
- Lewis, H. J. (2021). User Trust in Gig Economy Platforms: A Case Study of TaskRabbit. (Master's thesis, University of Digital Innovation). Available from ProQuest Dissertations & Theses database.
- European Commission. (2022). Guidelines for Digital Service Platforms. Retrieved from <https://ec.europa.eu/digital-service-platforms-guidelines>
- U.S. Department of Labor. (2021). Gig Economy Worker Rights. Retrieved from <https://www.dol.gov/gig-economy>
- Bootstrap. (n.d.). CSS Framework for Responsive Web Design. Retrieved from <https://getbootstrap.com/>
- W3C. (2020). Web Content Accessibility Guidelines (WCAG) 2.1. Retrieved from <https://www.w3.org/TR/WCAG21/>

Login Pages (Client, Worker, Admin): Show how each user type accesses the system.

Client Dashboard: Capture the welcome message, category search, worker cards, and the profile management link.

Worker Dashboard: Highlight the profile update functionality and worker overview.

Admin Dashboard: Display category management, user deletion functionality, and the admin profile.

Worker Profile Page: Show details about a worker and the client's ability to view their profile.

Admin Profile: Capture the admin's profile view and the edit functionality.

Category List Management (Admin): Show how categories are managed (add, edit, delete).

Worker List Management: Capture how the admin views and manages workers, including deletion options.

Task Matching Flow: If applicable, show how clients find and contact workers based on categories.

## Appendices:

### appendix a:Home page

```

<?php
include '../includes/header.php'; // Include the reusable header
include '../includes/db.php'; // Include the database connection

// Fetch categories from the database
$query = "SELECT category_name FROM categories"; // Assuming the 'name' field
contains the category name
$stmt = $pdo->prepare($query);
$stmt->execute();
$categories = $stmt->fetchAll(PDO::FETCH_ASSOC);
?>

<h1>Welcome to SkillShare Connect</h1>
<p>Find professionals for the services you need, or showcase your skills to others.</p>
<!-- Display Categories -->
<h2>Available Categories</h2>
<ul class="categories">
<?php
// Loop through the categories and display them
if (!empty($categories)) {
foreach ($categories as $category) {
echo "<li>" . htmlspecialchars($category['category_name']) . "</li>"; // Escape output for
security
}
} else {
echo "<li>No categories available</li>";
}
?>
</ul>

<?php include '../includes/footer.php'; // Include the reusable footer ?>

```

**Appendix B:Client dashboard**

```

<?php
// Start session
session_start();

// Check if the client is logged in
if (!isset($_SESSION['client_id'])) {
header('Location: ../public/login.php');
exit();
}
// Include header, config, and db
require_once '../includes/header.php';
require_once '../includes/config.php';
require_once '../includes/db.php';
// Fetch client data
$client_id = $_SESSION['client_id'];
$query = $pdo->prepare('SELECT name FROM clients WHERE id = :id');
$query->execute(['id' => $client_id]);
$client = $query->fetch(PDO::FETCH_ASSOC);
// Fetch categories for the display
$category_query = $pdo->query('SELECT * FROM categories');
$categories = $category_query->fetchAll(PDO::FETCH_ASSOC);
// Fetch a sample of workers
$worker_query = $pdo->query('SELECT * FROM workers LIMIT 6');
$workers = $worker_query->fetchAll(PDO::FETCH_ASSOC);
?>
<div class="dashboard">
<h1>Welcome, <?php echo htmlspecialchars($client['name']); ?>!</h1>
<!-- Search Bar
<div class="search-section">
<form action="search_results.php" method="GET">
<input type="text" name="search" placeholder="Search for a category or service...">
<button type="submit">Search</button>

```

```

</form>
</div> -->
<!-- Categories List -->
<div class="categories-section">
<h2>Categories</h2>
<ul>
<?php foreach ($categories as $category): ?>
<li><a href="view_workers_by_category.php?category_id=<?php echo $category['id'];
?>">
<?php echo htmlspecialchars($category['category_name']); ?>
</a></li>
<?php endforeach; ?>
</ul>
</div>
<!-- Sample Workers -->
<div class="worker-section">
<h2>Sample Workers</h2>
<div class="worker-cards">
<?php foreach ($workers as $worker): ?>
<div class="worker-card">

<h3><?php echo htmlspecialchars($worker['name']); ?></h3>
<p>Category:
<?php
// Fetch the worker's category
$cat_stmt = $pdo->prepare('SELECT category_name FROM categories WHERE id =
:category_id');
$cat_stmt->execute(['category_id' => $worker['category_id']]);
$category_name = $cat_stmt->fetchColumn();
echo htmlspecialchars($category_name);
?>
</p>

```

```
<p>Location: <?php echo htmlspecialchars($worker['location']); ?></p>
<a href="./worker/worker_profile.php?worker_id=<?php echo $worker['id']; ?>">View
Profile</a>
</div>
<?php endforeach; ?>
</div>
</div>
<!-- Account Settings -->
<div>
<a href="client_profile.php" class="btn btn-primary">Account Settings</a>
</div>
<!-- Logout Button -->
<div class="logout-section">
<a href="logout.php">Logout</a>
</div>
</div>
<?php require_once './includes/footer.php'; ?>
```

## Appendix c: Worker dashboard

```
<?php
// Start session
session_start();

// Check if the worker is logged in
if (!isset($_SESSION['worker_id'])) {
header('Location: ../public/login_worker.php');
exit();
}
// Include database and config
require_once '../includes/db.php';
require_once '../includes/header.php';

// Fetch worker data
$worker_id = $_SESSION['worker_id'];
$query = $pdo->prepare('SELECT name, email, category_id, location, experience,
description, phone_number, profile_picture FROM workers WHERE id = :id');
$query->execute(['id' => $worker_id]);
$worker = $query->fetch(PDO::FETCH_ASSOC);
// Fetch the category name from the `categories` table
$category_query = $pdo->prepare('SELECT category_name FROM categories WHERE id
= :category_id');
$category_query->execute(['category_id' => $worker['category_id']]);
$category = $category_query->fetch(PDO::FETCH_ASSOC);
if (!$worker) {
echo "Worker not found.";
exit();
}
?>
<div class="dashboard">
<h1><?php echo htmlspecialchars($worker['name']); ?>'s Profile</h1>
```



```
<!-- Profile Picture -->
<div class="profile-section">

</div>
<!-- Worker Information -->
<div class="worker-info">
<p><strong>Category:</strong> <?php echo
htmlspecialchars($category['category_name']); ?></p>
<p><strong>Location:</strong> <?php echo htmlspecialchars($worker['location']); ?></p>
<p><strong>Experience:</strong> <?php echo htmlspecialchars($worker['experience']);
?> years</p>
<p><strong>Description:</strong> <?php echo htmlspecialchars($worker['description']);
?></p>
<p><strong>Contact Information:</strong></p>
<p>Email: <?php echo htmlspecialchars($worker['email']); ?></p>
<p>Phone: <?php echo htmlspecialchars($worker['phone_number']); ?></p>
</div>
<!-- Update Profile Button -->
<div class="update-profile-section">
<a href="edit_profile.php" class="btn btn-primary">Update Profile</a>
</div>

<!-- Logout Button -->
<div class="logout-section">
<a href="logout_worker.php">Logout</a>
</div>
</div>
<?php require_once '../includes/footer.php'; ?>
```

## Appendix d:Admin dashboard

```
<?php
// Start session and check if the admin is logged in
session_start();
if (!isset($_SESSION['admin_id'])) {
header('Location: admin_login.php');
exit();
}
require_once '../includes/db.php';
require_once '../includes/header.php';
?>
<div class="admin-dashboard">
<h1>Admin Dashboard</h1>
<div class="dashboard-section">
<h2>Manage Categories</h2>
<a href="add_category.php">Add New Category</a> |
<a href="category_list.php">View/Edit Categories</a>
</div>
<div class="dashboard-section">
<h2>Manage Users</h2>
<a href="view_clients.php">View Clients</a> |
<a href="view_workers.php">View Workers</a>
</div>
<div class="dashboard-section">
<h2>Account Settings</h2>
<a href="view_profile.php">View/Edit Profile</a>
</div>
</div>
<?php
require_once '../includes/footer.php';
?>
```

## Appendix E: Client Signup Processing

```
<?php
require_once '../includes/config.php';
require_once '../includes/db.php';

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Sanitize inputs
    $name = trim(filter_input(INPUT_POST, 'name', FILTER_SANITIZE_STRING));
    $email = trim(filter_input(INPUT_POST, 'email', FILTER_SANITIZE_EMAIL));
    $password = $_POST['password'];
    $phone = trim(filter_input(INPUT_POST, 'phone', FILTER_SANITIZE_STRING));

    // Validate inputs
    if (empty($name) || empty($email) || empty($password) || empty($phone)) {
        echo "All fields are required.";
        exit();
    }
    if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
        echo "Invalid email format.";
        exit();
    }
    // Hash the password
    $hashed_password = password_hash($password, PASSWORD_DEFAULT);

    // Insert into the database
    try {
        $stmt = $pdo->prepare('INSERT INTO clients (name, email, password, phone) VALUES
        (:name, :email, :password, :phone)');
        $stmt->execute([
            ':name' => $name,
            ':email' => $email,
            ':password' => $hashed_password,
            ':phone' => $phone
```

```
]);
```

```
header('Location: ../public/login.php');
exit();
} catch (PDOException $e) {
echo "Error: " . $e->getMessage();}}?>
```

## Appendix F: Worker Signup Processing

```
<?php
// Start session
session_start();

// Include database and config
require_once '../includes/db.php';
require_once '../includes/config.php';

// Check if the form is submitted
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
// Retrieve and sanitize input values
$name = htmlspecialchars($_POST['name']);
$email = htmlspecialchars($_POST['email']);
$password = password_hash($_POST['password'], PASSWORD_DEFAULT); // Hash the
password
$category_id = htmlspecialchars($_POST['category_id']);
$location = htmlspecialchars($_POST['location']);
$experience = htmlspecialchars($_POST['experience']);
$description = htmlspecialchars($_POST['description']);
$phone_number = htmlspecialchars($_POST['phone_number']);

// Check if the email already exists
$check_email_query = $pdo->prepare('SELECT id FROM workers WHERE email =
:email');
$check_email_query->execute(['email' => $email]);
```

```
if ($check_email_query->rowCount() > 0) {
// If email already exists, redirect back with an error
$_SESSION['error'] = 'Email already exists. Please use a different email.';
header('Location: signup_worker.php');
exit();
}
// Insert worker data into the database
$insert_query = $pdo->prepare('
INSERT INTO workers (name, email, password, category_id, location, experience,
description, phone_number)
VALUES (:name, :email, :password, :category_id, :location, :experience, :description,
:phone_number)
');
$success = $insert_query->execute([
'name' => $name,
'email' => $email,
'password' => $password,
'category_id' => $category_id,
'location' => $location,
'experience' => $experience,
'description' => $description,
'phone_number' => $phone_number,
]);
// Check if the data was inserted successfully
if ($success) {
// Redirect to worker login page after successful signup
$_SESSION['success'] = 'Sign up successful. You can now log in.';
header('Location: ../public/login_worker.php');
exit();
} else {
// If there was an error, redirect back with an error
$_SESSION['error'] = 'There was an error signing you up. Please try again.';
header('Location: signup_worker.php');
exit();
}
```

```

}
} else {
// If form was not submitted, redirect to signup form
header('Location: signup_worker.php');
exit();}?>

```

## Appendix G: Client Login Processing

```

<?php
// Start the session and include database connection
session_start();
require_once './includes/db.php'; // Database connection
require_once './includes/config.php';

// Check if the login form is submitted
if (isset($_POST['login'])) {
// Retrieve the form data and trim any whitespace
$email = trim($_POST['email']);
$password = trim($_POST['password']);

// Check if email and password fields are filled
if (!empty($email) && !empty($password)) {
try {
// Prepare SQL statement to select the client based on the email
$stmt = $pdo->prepare("SELECT * FROM clients WHERE email = :email");
$stmt->bindParam(':email', $email);
$stmt->execute();
// Fetch the client record
$client = $stmt->fetch(PDO::FETCH_ASSOC);
// If client exists, verify the password
if ($client && password_verify($password, $client['password'])) {
// Set session variables
$_SESSION['client_id'] = $client['id'];
$_SESSION['client_name'] = $client['name'];

```

```
// Redirect to the client dashboard
header("Location: client_dashboard.php");
exit();
} else {
// Invalid credentials, show error message
echo "Invalid email or password.";
}
} catch (PDOException $e) {
// Catch any errors in the database query
echo "Error: " . $e->getMessage();
}
} else {
// If fields are empty, display an error
echo "Please fill in both email and password.";
}
} else {
echo "Unauthorized access.";
}
?>
```

## Appendix H:worker login Processing

```

<?php
// Start session
session_start();
// Include the database connection and config
require_once '../includes/db.php';
require_once '../includes/config.php';
// Check if the form is submitted
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
// Retrieve the form input and sanitize it
$email = !empty($_POST['email']) ? trim($_POST['email']) : "";
$password = !empty($_POST['password']) ? trim($_POST['password']) : "";
// Check if fields are not empty
if (empty($email) || empty($password)) {
$_SESSION['error'] = 'Please fill out both fields.';
header('Location: ../public/login_worker.php');
exit();
}
// Prepare and execute the query to fetch the worker
$stmt = $pdo->prepare('SELECT * FROM workers WHERE email = :email');
$stmt->execute(['email' => $email]);
$worker = $stmt->fetch(PDO::FETCH_ASSOC);
// Verify if the worker exists and check the password
if ($worker && password_verify($password, $worker['password'])) {
// Set session variables
$_SESSION['worker_id'] = $worker['id'];
$_SESSION['worker_name'] = $worker['name'];
// Redirect to worker dashboard
header('Location: worker_dashboard.php');
exit();
} else {
// Invalid credentials
$_SESSION['error'] = 'Invalid email or password.';
}
}

```



```
header('Location: ../public/login_worker.php');
exit();
}
} else {
// If the request is not POST, redirect to login page
header('Location: ../public/login_worker.php');
exit();
}

?>
```

### Appendix I: Time Frame Coverage:

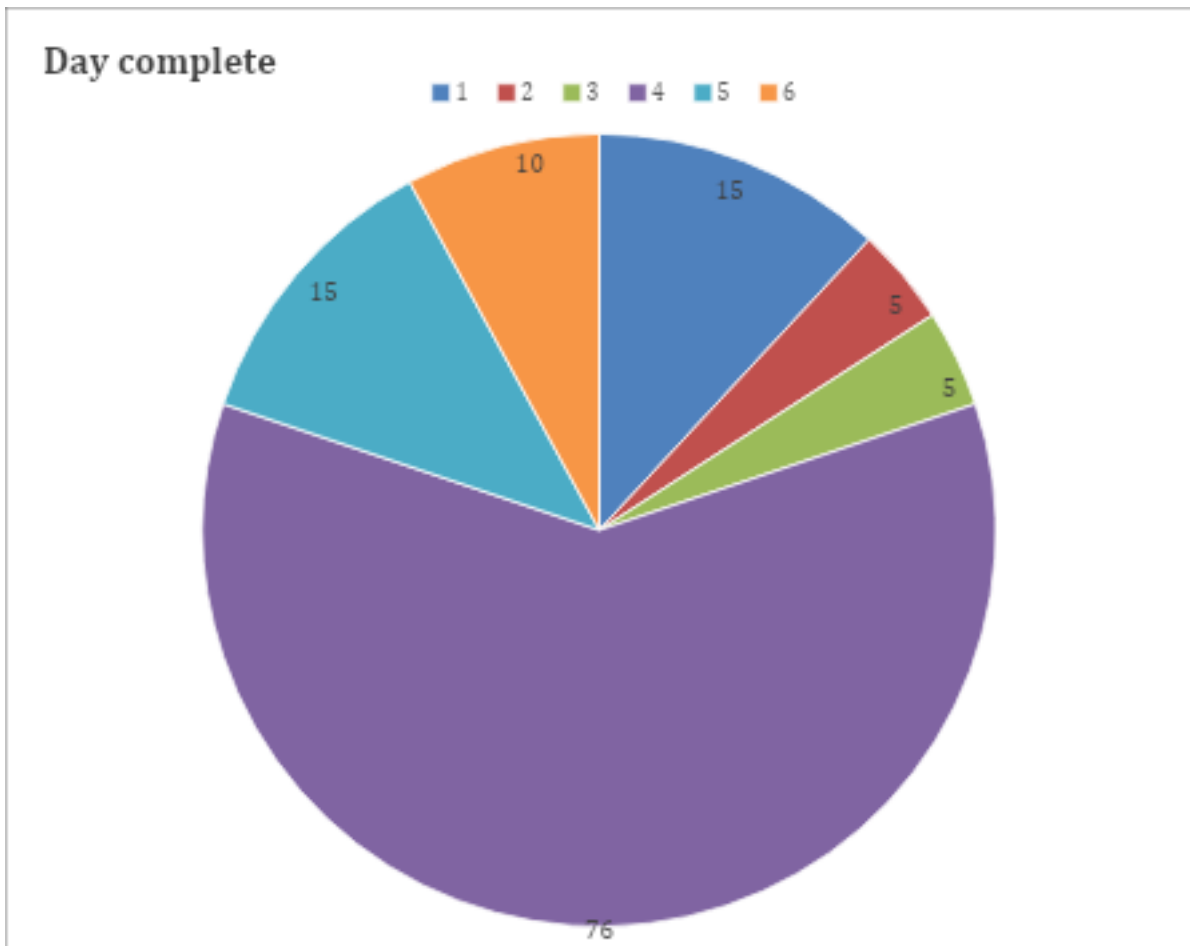


Figure 25: Time frame