**Shipment/delivery company**

Senior Project

by

**Antonio Al Makdissi,**

**Fawzy Ghazawy**

Submitted to the School of Engineering of the

Lebanese International University

Tripoli, Lebanon

in partial fulfillment of the requirements for the degree of

**BACHELOR OF SCIENCE IN COMPUTER ENGINEERING**

**Spring 2022-2023**

**Approved by:**

**Supervisor**  

Dr. Zaher Merhi



**Committee Member**

Dr. TBA



**DEDICATION**

First and foremost, we would like to dedicate this for ourselves, our past selves, to bring to light the progress we have made along the years starting from scratch till what we are today. Though this does not represent our endpoint in life as we are going to strive in order to achieve higher goals via further education and hard work through the many years to come.

**ACKNOWLEDGMENT**

We would like to thank our supervisor and the dean and the rest of the people who spared no effort in guiding and instructing us throughout the different steps of the project. And finally the team members who put a lot of work and dedication to help achieve this project.

**ABSTRACT**

The abstract goes here with double space 12pt font. The abstract consists of one paragraph with first line indentation. The page should be numbered. The abstract goes here with double space 12pt font. The abstract consists of one paragraph with first line indentation. The page should be numbered. The abstract goes here with double space 12pt font. The abstract consists of one paragraph with first line indentation. The page should be numbered. The abstract goes here with double space 12pt font. **The abstract consists of one paragraph with first line indentation**. The page should be numbered. The abstract goes here with double space 12pt font. The abstract consists of one paragraph with first line indentation. The page should be numbered.

**TABLE OF CONTENTS**

[**ACKNOWLEDGMENT iii**](#_heading=h.tyjcwt)

[**ABSTRACT iv**](#_heading=h.3dy6vkm)

[**TABLE OF CONTENTS v**](#_heading=h.1t3h5sf)

[**LIST OF FIGURES viii**](#_heading=h.4d34og8)

[**LIST OF TABLES ix**](#_heading=h.2s8eyo1)

[**LIST OF SYMBOLS x**](#_heading=h.17dp8vu)

[**CHAPTER 1 INTRODUCTION 1**](#_heading=h.3rdcrjn)

[1.1](#_heading=h.26in1rg) Background 1

[1.2](#_heading=h.lnxbz9) Problem Statement 1

[1.3](#_heading=h.35nkun2) General overview of the project 1

[1.4](#_heading=h.1ksv4uv) Thesis Outline 1

[**CHAPTER 2 Survey of Existing Methods and Similar Systems 2**](#_heading=h.44sinio)

[2.1](#_heading=h.2jxsxqh) Introduction 2

[2.2](#_heading=h.z337ya) Method 1 / System 1 2

[2.3](#_heading=h.3j2qqm3) Method 2 / System 2 2

[2.4](#_heading=h.1y810tw) Method 3 / System 3 2

[2.5](#_heading=h.4i7ojhp) Methods/Systems Comparison 2

[2.6](#_heading=h.2bn6wsx) Conclusion and Motivation 3

[**CHAPTER 3 System Design 5**](#_heading=h.qsh70q)

[3.1](#_heading=h.3as4poj) Introduction 5

[3.2](#_heading=h.1pxezwc) Requirements and Specification Analysis 5

[3.2.1](#_heading=h.49x2ik5) Functional Requirements 5

[3.2.2](#_heading=h.2p2csry) Use Case Diagrams 5

[3.3](#_heading=h.147n2zr) System Architecture 6

[3.4](#_heading=h.23ckvvd) Class Diagrams 6

[3.5](#_heading=h.ihv636) Sequence Diagrams 6

[3.6](#_heading=h.32hioqz) Activity Diagrams 7

[3.7](#_heading=h.1hmsyys) Entity-Relationship (ER) Diagrams 7

[3.8](#_heading=h.41mghml) Non-Technical Aspects 7

[3.8.1](#_heading=h.2grqrue) Financial Viability 7

[3.8.2](#_heading=h.3fwokq0) Stakeholders 8

[3.8.3](#_heading=h.1v1yuxt) Scope 8

[3.8.4](#_heading=h.4f1mdlm) Risks 8

[3.8.5](#_heading=h.2u6wntf) Schedule and Milestones 9

[3.8.6](#_heading=h.3tbugp1) Ethical and Social Considerations 9

[3.8.7](#_heading=h.28h4qwu) Environmental and Sustainability Considerations 9

[3.8.8](#_heading=h.nmf14n) Relevant Standards 10

[3.9](#_heading=h.37m2jsg) Conclusion 10

[**CHAPTER 4 Implementation/Simulation and Testing 11**](#_heading=h.1mrcu09)

[4.1](#_heading=h.46r0co2) Introduction 11

[4.2](#_heading=h.2lwamvv) Implementation Tools 11

[4.3](#_heading=h.111kx3o) Implementation Summary 11

[4.4](#_heading=h.3l18frh) Test Cases and Acceptance Criteria 11

[4.5](#_heading=h.206ipza) Conclusion 11

[**CHAPTER 5 Conclusion and Future Work 12**](#_heading=h.4k668n3)

[5.1](#_heading=h.2zbgiuw) Conclusion 12

[5.2](#_heading=h.1egqt2p) Future Work 12

[**APPENDIX A: Implementation Details 13**](#_heading=h.3ygebqi)

[**APPENDIX B: USER Manual 14**](#_heading=h.2dlolyb)

[**APPENDIX C: deployment and configuration Manual 15**](#_heading=h.sqyw64)

[**REFERENCES 16**](#_heading=h.3cqmetx)

**LIST OF FIGURES**

[Figure ‎1‑1:Shipyard full of cargo to be delivered [5] 2](#_Toc131358971)

[Figure ‎1‑2: web application preview [5] 4](#_Toc131358972)

[Figure ‎1‑3: paid delivery [5] 5](#_Toc131358973)

[Figure ‎2‑1: Aramex's Website [9] 8](#_Toc131358974)

[Figure ‎2‑2: Wakilni Website [8] 9](#_Toc131358975)

[Figure ‎2‑3: GoShare [10] 11](#_Toc131358976)

[Figure ‎3‑1: client use case diagram 18](#_Toc131358977)

[Figure ‎3‑2: CEO use case diagram 19](#_Toc131358978)

[Figure ‎3‑3: Branch Manager use case diagram 20](#_Toc131358979)

[Figure ‎3‑4: worker use case diagram 20](#_Toc131358980)

[Figure ‎3‑5: System Architecture 22](#_Toc131358981)

[Figure ‎3‑6: class diagram 23](#_Toc131358982)

[Figure ‎3‑7: all users sequence diagram 25](#_Toc131358983)

[Figure ‎3‑8: client sequence diagram 26](#_Toc131358984)

[Figure ‎3‑9: CEO sequence diagram 27](#_Toc131358985)

[Figure ‎3‑10: Branch Manager sequence diagram 27](#_Toc131358986)

[Figure ‎3‑11: worker sequence diagram 28](#_Toc131358987)

[Figure ‎3‑12: ER diagram 30](#_Toc131358988)

**LIST OF TABLES**

Table 2-1: Comparison Table Based on Graphical Interfaces 3

Table 2-2: Comparison Table Based on Content and Functionality 3

Table 2-3: Comparison Table Based on Features 3

**LIST OF SYMBOLS**

MUST LISTED IN ASCENDING ALPHABETICAL ORDER. Examples:

CMS: Content Management System

COVID: Corona Virus Disease

Ex: example

GUI: Graphical User Interface

ISO: International Standardization Organization

LCD: Liquid Crystal Display

LIU: Lebanese International University

UI: User Interface

UX: User Experience

WiFi: Wireless Fidelity

# INTRODUCTION

## Background

Our modern world is full of challenges. That’s why we, as humans, try our best to invent new technologies in order to make our lives less tedious and easier to go by.

Logistics companies faced various challenges before the advent of the internet. One of the major issues was the lack of real-time visibility into the status and location of shipments. This made it difficult to track shipments, causing delays and higher costs. Pre-internet, logistics companies encountered numerous obstacles, including a lack of real-time visibility into shipment status and location, impeding the ability to track shipments and resulting in delays and escalated costs [1]. Moreover, reliance on manual paper-based processes led to a host of issues, including time inefficiencies and error-prone operations that resulted in increased costs. Additionally, limited access to information and communication channels posed further difficulties in coordinating with suppliers, customers, and carriers. Collectively, these issues resulted in reduced agility and responsiveness, hampering logistics companies' abilities to cater effectively to customer demands. However, with the advent of the internet, logistics companies have been able to surmount these challenges and enhance their operations significantly. The internet's vast capabilities, including real-time tracking and communication, have facilitated increased access to information and streamlined processes, resulting in improved efficiency, reduced costs, and heightened customer satisfaction. [2]

We saw the problems that shipment companies face, such as loss of cargo either through piracy or corruption, absence of trust between customer and company due to lack of communication, and insufficient data/intel on the companies’ exchanges. [3]

Hence our attempt at developing our app: a cargo/delivery application that can address several challenges faced by logistics providers and customers, such as lack of visibility, inefficient processes, limited communication, security concerns, competition, and increasing demand. By providing real-time tracking, automating processes, improving communication and transparency, enhancing security, and offering a competitive advantage, a cargo/delivery application can improve customer satisfaction, reduce costs, optimize operations, and provide valuable data insights. [4]



Figure ‑:Shipyard full of cargo to be delivered [5]

## Problem Statement

Since its foundation, the logistics and transportation industry encounter various difficulties such as inadequate visibility, ineffective procedures, restricted communication, security threats, intense competition, and growing demand for prompt and dependable shipping services. These challenges can cause delivery delays, higher costs, and lower customer satisfaction, which may have negative effects on the profitability and expansion of logistics providers. To solve these challenges, a cargo/delivery application is necessary to deliver instant tracking, automate processes, enhance communication and transparency, boost security, and provide a competitive edge in the market [6].

* Optimize operations: The application should help logistics providers optimize their operations by reducing manual processes, increasing efficiency, and improving resource allocation.
* Reduce costs: By automating processes and improving efficiency, the application should help logistics providers reduce costs associated with labor, fuel, and other operational expenses.
* Increase revenue: By improving customer satisfaction and providing a competitive advantage, the application should help logistics providers attract new customers and increase revenue.



Figure ‑: Organized warehouse [5]

## General overview of the project

Our project aims to improve the efficiency, transparency, security, and customer satisfaction of logistics and transportation operations. This can be achieved through real-time tracking, process automation, improved communication, enhanced security measures, and a unique value proposition that sets the application apart from competitors.   
The cargo/delivery application aims to provide logistics providers and customers with a comprehensive, user-friendly platform for managing shipments and deliveries, using modern technologies and tools. The application will feature real-time tracking, automated processes, communication tools, enhanced security measures, and data insights to optimize operations and improve customer satisfaction. The real-time tracking feature will leverage GPS and other location-tracking technologies to provide customers with real-time updates on the status and location of their shipments. The process automation feature will automate manual processes involved in managing shipments, such as generating labels, managing inventory, scheduling deliveries, and handling customs documentation. The communication tools feature will provide notifications, alerts, messaging capabilities, and a customer service portal for addressing any issues that may arise. The enhanced security measures feature will include secure login credentials, real-time tracking, and integrated security systems that monitor shipments throughout the delivery process [7]. The data insights feature will provide valuable data on shipment performance, customer behavior, and other key metrics, which can be used to improve operations and make better business decisions.



Figure ‑: web application preview [5]

## Project Outline

The project aims to develop a cargo/delivery application that will provide real-time tracking, automate processes, improve communication and transparency, enhance security, and offer a competitive advantage to logistics providers. The application will help logistics providers optimize their operations, reduce costs, increase revenue, and improve customer satisfaction. Please follow us throughout the steps of realizing the project, from research to development. As first we are going to conduct a thorough analysis of the requirements and needs of logistics providers and customers to determine the key features and functionalities of the application. We will do this by surveying of existing methods and similar systems in ‎CHAPTER 2.



Figure ‑: confirmed delivery [5]

# Survey of Existing Methods and Similar Systems

## Introduction

Also called "Requirements gathering and analysis", this phase is a critical step in the development of the cargo/delivery application project. In this phase, we will research closely already existing applications (like Wakilni [8], Aramex [9] and GoShare [10]) to understand the project’s requirements and needs, identify pain points, and define the scope and goals of said project. This phase will involve conducting research, holding meetings, and gathering feedback from logistics providers and customers to develop a comprehensive set of requirements that will guide the development of the application. The project team will use this information to create use cases, user stories, and other documentation that will help ensure that the final product meets the needs of all stakeholders and delivers the desired outcomes. The requirements gathering and analysis phase will be followed by the design and development phases, where the project team will begin building the application based on the requirements identified in this phase.

## System 1: Aramex

Aramex [9] is a global logistics and transportation company based in Dubai, United Arab Emirates. It provides a range of services including express courier delivery, freight forwarding, logistics and supply chain management. Aramex operates in more than 65 countries worldwide and is known for its innovative solutions and use of technology in the logistics industry. Their website offers:

* e-commerce solutions that assist start-ups and established businesses in supply chain management, technical support, and more [9]
* Logistics Services
* Bundling, Packaging and co-packing
* Aramex Application Programming Interface (APIs)
* Import address book
* Track shipments
* Calculate rates
* Receive shipment updates
* ClickToShip: a separate desktop application without constant link to the internet
* Aramex app: a mobile application that enables Aramex customers worldwide to perform a range of tasks, including tracking deliveries, monitoring shipment progress, managing their accounts, calculating the costs and transit times of upcoming shipments, scheduling pickups and deliveries at preferred locations, dates, and times, making secure payments, and receiving a clear breakdown of delivery dates and custom charges… [9]



Figure ‑: Aramex's Website [9]

## System 2: Wakilni

Wakilni [8] is a delivery company that offers a range of innovative solutions to help individuals and businesses in the Middle East manage their daily tasks more efficiently. Founded in 2015, the company connects customers with trusted service providers who can take care of tasks like grocery delivery, laundry, home cleaning, and more. Wakilni's personalized approach and use of machine learning algorithms ensure that each customer receives customized service tailored to their unique preferences and behavior.   
Here's what their website offers:

* Donate to children feature.
* Promo code to take a discount.
* Shipment cost around 4$-8$ across Lebanon.
* Extra payment option for 24-72 hours’ delivery.
* Join our team feature if they need some workers. Ex: driver.
* Live chat feature: they will ask for your name and the email
* A marketplace page that shows all the businesses in Lebanon that are registered in Wakelin and they provide the customer with categories to find what he is searching for; and provide a link for each one of them.
* Extra mile fund donations: As of October 2020, Wakilni will donate 750 liras from each commissioned delivery to its Extra Mile Fund*.* This money will go towards helping members of the Wakilni community reboot their businesses by accessing needed services, whether in terms of online exposure, marketing, legal counseling, etc. and Wakilni will be sponsoring this access to services. [8]



Figure ‑: Wakilni Website [8]

## System 3: GoShare

GoShare [10] is a rapidly growing logistics and transportation company based in San Diego, California. Founded in 2014, the company has quickly become a leading provider of on-demand delivery and moving services in the United States.  
We personally chose GoShare as it has received numerous accolades for its innovative approach to logistics and transportation. The company was named one of the "Most Promising Startups" at the 2017 Forbes Under 30 Summit, and it has been featured in publications like the Wall Street Journal, TechCrunch, and CNBC.   
Here are some key points about GoShare:

* GoShare offers a variety of services, including local delivery, hauling, and moving. Customers can use the company's app or website to book a driver and a truck for a range of jobs, from small-item delivery to full-scale residential moves.
* One of GoShare's key selling points is its on-demand availability. The company's large network of independent contractors allows it to offer fast service, with many jobs completed within an hour of booking.
* GoShare drivers are rigorously vetted and insured, so customers can trust that their items are in safe hands. Drivers are also equipped with a range of tools and materials to help them complete jobs efficiently and effectively.
* GoShare is committed to sustainability and reducing its carbon footprint. The company's fleet of trucks includes electric and hybrid vehicles, and it encourages its drivers to adopt eco-friendly driving habits.
* In addition to its core services, GoShare also offers business solutions for companies looking to streamline their logistics operations. The company's platform allows businesses to manage deliveries, track shipments, and access real-time data on their delivery performance.

Overall, GoShare is a dynamic and rapidly expanding company that is shaking up the traditional logistics industry. With its on-demand availability, focus on sustainability, and commitment to innovation, GoShare is poised for continued success in the years to come. [10]



Figure ‑: GoShare [10]

## Methods/Systems Comparison

The purpose of this paragraph is to draw attention to the strengths and weaknesses of each of the methods mentioned earlier in relation to specific standards.

**Table 2-1: Comparison Table Based on Graphical Interfaces**

|  |  |  |  |
| --- | --- | --- | --- |
| **Criterion 1** | **Aramex** | **Wakilni** | **GoShare** |
| **Graphical Interface** |
| Good user interface | **🗸** | **🗸** | **🗸** |
| Easy and effective navigation | **🗴** | **🗸** | **🗴** |
| Simple and professional Design | **🗸** | **🗸** | **🗸** |
| Responsive | **🗴** | **🗸** | **🗸** |

A well-designed graphical user interface (GUI) is essential for the success of an application. A good user interface (UI), easy navigation, professional design, and responsiveness are crucial components. In the case of the three systems we reviewed - Aramex, GoShare, and Wakilni - we found that each had their own strengths and weaknesses in terms of GUI design. When it comes to UI, all three systems were generally easy to use, with clear icons and labels for each feature; a good and attractive GUI as it is eye catching and well laid out through the webpage. However, we found that Aramex and GoShare's navigation was slightly more confusing compared to Wakilni, which had a more intuitive layout that made it easier to find what we needed like the order part that was really accessible and available from the get-go. In terms of design, all three systems had a simple and professional look, with no obvious errors or unprofessional features. However, we noticed that Wakilni had a more modern and aesthetically pleasing design compared to Aramex and GoShare. When it came to responsiveness, we found that Aramex was slightly slower and less dynamic compared to GoShare and Wakilni. This could potentially impact user experience and satisfaction. A well-designed GUI with a good UI, easy navigation, professional design, and responsiveness improves the usability and appeal of an application.

**Table 2-2: Comparison Table Based on Content and Functionality**

|  |  |  |  |
| --- | --- | --- | --- |
| **Criterion 2** | **Aramex** | **Wakilni** | **GoShare** |
| **Content and Functionality** |
| Quality content structure | **🗸** | **🗸** | **🗸** |
| Usability | **🗸** | **🗸** | **🗴** |
| Dynamic content | **🗸** | **🗸** | **🗸** |
| Content management system | **🗸** | **🗸** | **🗸** |

Quality content structure, usability, dynamic content, and content management systems are all critical components of a successful application. In all three systems, quality content structure ensures that information is presented in a clear, logical manner, with content divided into parts, categories, and subcategories, making it easy for users to find what they need. Each system clearly lays out content throughout the page to optimize user experience. Usability refers to the ease with which users can navigate an application, complete tasks, and achieve their goals. While Aramex and Wakilni have strong usability, with an intuitive homepage that allows for easy ordering, GoShare's usability is lacking: their website requires users to navigate complex menus to place an order, making the user experience less seamless than its competitors. Dynamic content, such as personalized recommendations or real-time updates, is available in all three systems. The objects are kept up-to-date and relevant, though none of the systems have implemented truly personalized content. Finally, a content management system (CMS) [11] allows for efficient and effective management of an application's content, ensuring that it remains up-to-date and relevant. While it is not immediately clear what CMS each of the three systems is using, they appear to be using popular platforms such as WordPress, Drupal, or Joomla, or custom-built CMS systems. Together, these features help create an application that is intuitive, engaging, and valuable to its users.

**Table 2-3: Comparison Table Based on Features**

|  |  |  |  |
| --- | --- | --- | --- |
| **Criterion 3** | **Aramex** | **Wakilni** | **GoShare** |
| **Features** |
| Security measures | **🗸** | **🗸** | **🗸** |
| Third party integration | **🗸** | **🗸** | **🗸** |
| Accessible content and location | **🗴** | **🗴** | **🗴** |
| Registration form | **🗸** | **🗴** | **🗸** |

All three systems we reviewed - Aramex, GoShare, and Wakilni - prioritize user data protection by implementing HTTPS and captchas throughout their webpages to prevent unauthorized access and display the padlock icon next to the URL for added security. In addition, Aramex, GoShare, and Wakilni offer visible third-party integrations with popular platforms such as Google, Facebook, and Instagram, making it easy for users to access other services that can enhance their experience like a high-contrast mode for example. However, none of them have made explicit efforts to ensure that their content is accessible to all users, regardless of their physical abilities. Aramex and GoShare offer registration forms and Login pages, while Wakilni only offers a login page, which could be a potential drawback for new users who want to create an account. By taking these factors into consideration, developers can create a secure, functional, and personalized application that provides an optimal user experience.

## Conclusion and Motivation

Through the study of other systems, we have identified the essential components that a successful application should possess. These components include a well-designed graphical user interface (GUI) [12], quality content structure, usability, dynamic content, content management systems, security measures, third-party integration, accessible content and location, and registration forms. A well-designed GUI with a good UI, easy navigation, professional design, and responsiveness enhances the user experience and increases usage and satisfaction. Quality content structure ensures that information is presented logically, while usability and dynamic content enhance user engagement. A content management system (CMS) [11] allows for efficient management of the application's content. Security measures, third-party integration, accessible content and location, and registration forms are also important considerations. By taking these factors into account, we can create an application that is secure, functional, accessible, and personalized, leading to increased user satisfaction and engagement.

# System Design

## Introduction

The cargo/delivery application project involves developing a comprehensive platform for managing shipments and deliveries, using modern technologies and tools. To ensure that the final product meets the needs of all stakeholders and delivers the desired outcomes, several key phases will be undertaken, including requirements and specification analysis, functional requirements, use case diagrams, system architecture, class diagrams, sequence diagrams, activity diagrams, and financial viability [1]. These phases that normally involve working closely with stakeholders, gathering feedback, and developing a comprehensive set of requirements that will guide the development of the application, have been simulated via our own research. This will be followed by the design and development phases, where the project team will begin building the application based on the requirements identified in the earlier phases. Throughout the project, the team will conduct rigorous testing and quality assurance to ensure that the application is free of bugs, errors, and other issues that could affect the user experience or performance. The financial viability analysis will ensure that the project is feasible, economically viable, and sustainable in the long run…

## Requirements and Specification Analysis

By allowing clients to request specific item deliveries, offering expedited delivery options, and providing real-time tracking, clients are more likely to trust your services and become repeat customers. Additionally, the four distinct user roles and companion mobile app for workers can help streamline the delivery process, improve communication, and increase worker efficiency. The manager's ability to monitor and analyze delivery metrics can help identify areas for improvement and optimize the delivery process. Finally, features such as Captcha verification and chat communication can increase security and customer satisfaction, respectively.

### Functional Requirements

Here the app’s features:

* The platform allows clients (companies, stores, or individuals) to request delivery for specific items.
* Senders are required to provide shipment details, including package dimensions, weight, and whether the contents are fragile, and submit a picture of the item when placing an order.
* If the package is fragile, senders must take necessary precautions to prevent damage during shipment or incur an additional fee for specialized wrapping services.
* A chat feature is available for clients to communicate with the company.
* To prevent spam accounts and safeguard against DDOS attacks, a Captcha verification process is implemented.
* The App also uses md5 encryption for further protection of the passwords
* Payment options include the sender or receiver paying for the delivery.
* An expedited delivery option is available, offering same-day or next-day delivery for an additional fee.
* The platform accommodates four distinct user roles; including a CEO, branch manager, warehouse workers, and clients. Each role is granted access to specific templates and functionalities within the web platform/mobile application.
* Upon completion of each task, workers check off the corresponding item on the companion mobile app to signal readiness for the following step.
* Workers have to take a picture on the mobile app to be able to confirm the delivery that is sent via email.
* Urgent tasks are communicated to workers via notifications within the app.
* The CEO is able to monitor and manage the hiring and firing of workers, track profits, and analyze delivery metrics, including successful and failed deliveries, punctuality, and worker efficiency.
* Clients are granted access to real-time delivery tracking, including the current location of the shipment and estimated delivery time.

### Use Case Diagrams

A technique that simplifies information about a system and its users is called a use case diagram. It presents a visual representation of how different components of the system interact with one another. Use case diagrams describe the sequence of events in which the system performs its actions, but they do not provide details on how these events are executed.

A use case is an approach to identify, outline, and organize requirements for a system, such as a website for product sales and services. Use case diagrams are commonly created using UML, a standard language for representing real-world structures and systems. Compared to other types of diagrams, use case diagrams offer several advantages.

In the context of the website, the customer, CEO, branch manager and IT are the five actors depicted in the use case diagrams. The worker also has access to the companion mobile application. The following use cases illustrate the tasks and activities that each user can perform in the project.

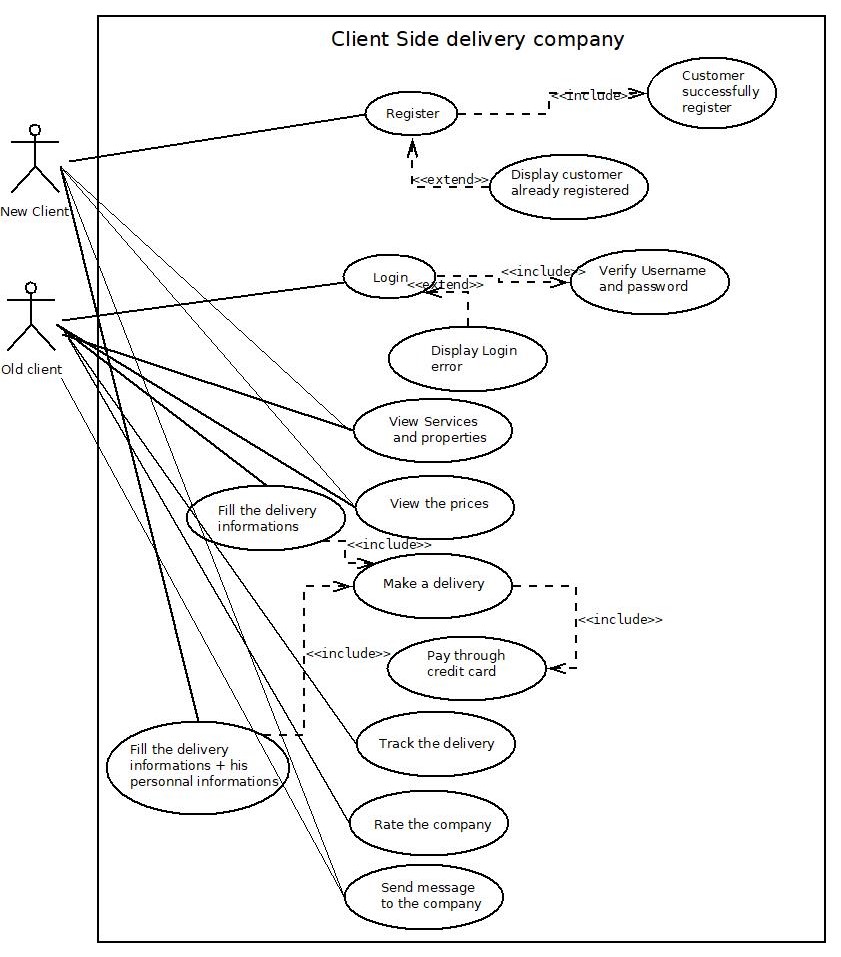


Figure ‑: client use case diagram

The users are able to register and sign in to their accounts, which allows them to save their information for future use. Upon logging in, users can view the services offered by the company, along with their corresponding prices. The user interface includes three main tabs: Track Delivery, Order Delivery, and History. Order Delivery allows users to enter the necessary information to place an order, while Track Delivery enables them to view the current location of a delivery by entering the order ID. The History tab provides users with access to their previous orders, including dates and costs. Additionally, users can provide feedback by rating the company or sending a message to the support team.

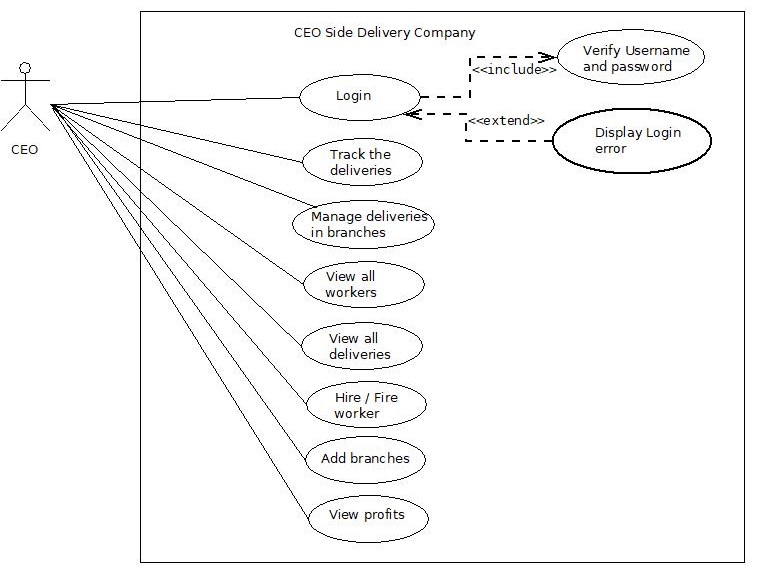


Figure ‑: CEO use case diagram

As the executive leader of the company, the CEO holds authority over a broad range of features within the application. Upon logging in to the system, the CEO is granted access to various tools that enable him to track and manage deliveries, oversee branch operations, monitor worker activity, and view financial reports related to the company's profits. In essence, the CEO is afforded the ability to exercise a high degree of control over the various aspects of the business via the application's user interface.

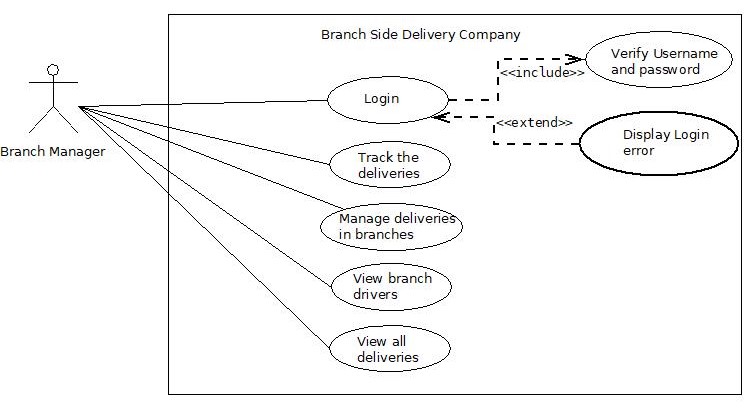


Figure ‑: Branch Manager use case diagram

The branch manager is responsible for overseeing the delivery operations within his designated branch. Through the application's user interface, the branch manager is granted access to features that enable him to manage and view deliveries within his branch, as well as monitor the activity of workers assigned to those deliveries. By tracking the progress of deliveries and reviewing worker performance, the branch manager is equipped to detect and address any issues that may arise and optimize the overall efficiency of delivery operations within his branch.

Top of Form

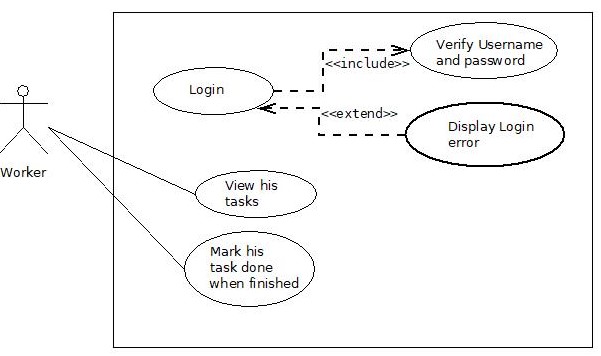


Figure ‑: worker use case diagram

The worker's role within the application is crucial to the successful management and execution of delivery operations. While their tasks may seem minimal in comparison to other users, such as the branch manager or CEO, their responsibilities are nonetheless essential to the overall success of the business. In particular, the worker's primary function is to mark the current location of deliveries within the system, enabling accurate tracking and timely updates for clients. Additionally, workers may be required to perform other tasks related to delivery management, such as verifying order details and ensuring that packages are properly labeled and accounted for.

## System Architecture

System architecture is a critical aspect of software system or application design, and it involves defining the structure of the system, including its components, interactions, and relationships with other systems. The architecture serves as a foundation for system development and maintenance, and it enables different teams to work together seamlessly. A well-designed system architecture can lead to a more efficient and effective system that provides a better user experience and allows for easier maintenance and future development.

Our project will adopt a multi-tier architecture, which consists of three main tiers: presentation, application, and database. The presentation tier will handle the user interface and interaction with the application, while the application tier will contain the business and processing logic. The database tier will be responsible for storing and managing the data used by the application. The system will utilize PHP, MySQL, HTML, CSS, JavaScript, and Bootstrap technologies, and RESTful API architecture to facilitate communication between the different tiers of the system. We will draw a figure to show the complete system architecture, which will provide a visual representation of the system and its components, and help to identify potential issues and bottlenecks, as well as to optimize the system for performance and scalability. Overall, our goal is to create a reliable, scalable, secure, and maintainable system that meets the needs of our users.

Below is a high-level diagram of the complete system architecture:

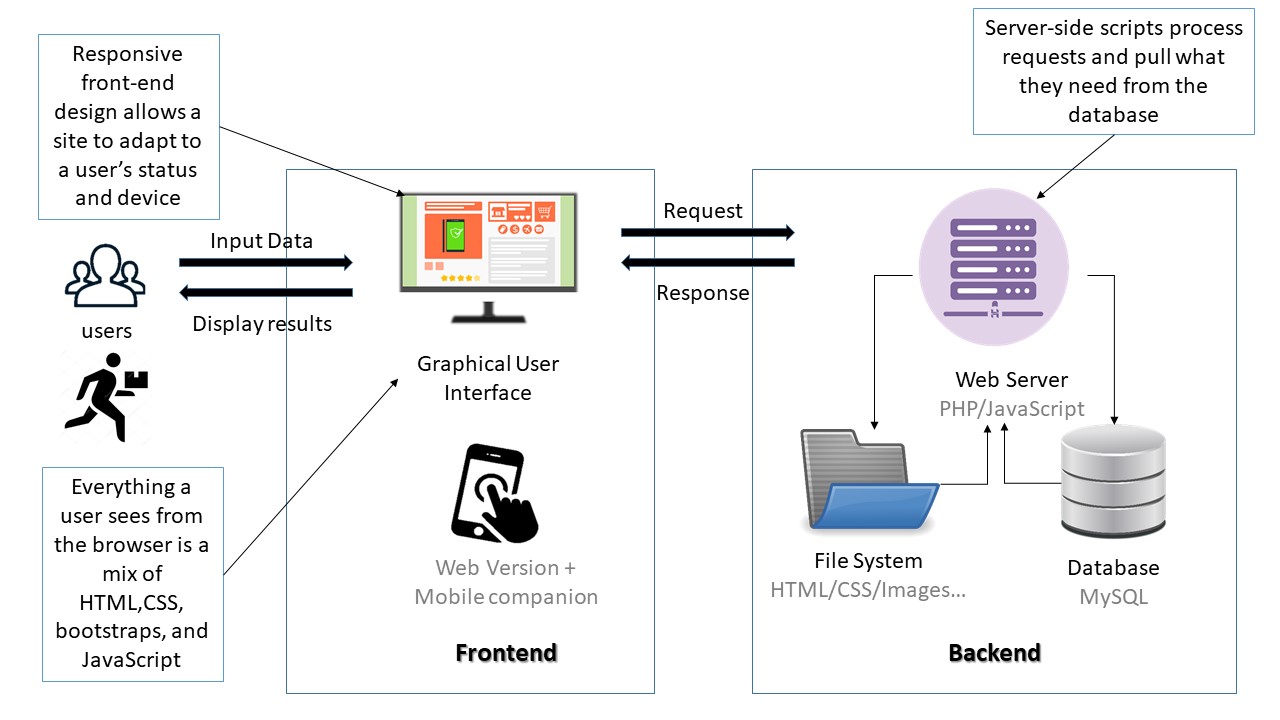


Figure ‑: System Architecture

## Class Diagrams

The class diagram is a static structural diagram in software engineering that depicts the classes, interfaces, and their relationships in a system. It provides a graphical representation of the system's object-oriented design, and helps to visualize the static view of the system. The class diagram is a part of the Unified Modeling Language (UML), which is widely used for modeling software systems. The class diagram shows the relationships among classes, such as inheritance, composition, aggregation, and association. It also shows the properties and methods of each class, which define the behavior and attributes of the objects of that class.

In our application, the class diagram might include classes such as users, orders, deliveries, payment, clients, and workers. Each class would have properties and methods that define its behavior and attributes, such as User class having properties like name, email, and password and methods like login(), signup(), and logout().

The class diagram would also show the relationships among these classes, such as the users class being composed of workers and clients’ classes, and the users class having an association with the privilege class.

This diagram is an essential component of the system design process, as it helps to visualize and communicate the object-oriented design of the system to stakeholders and development teams:

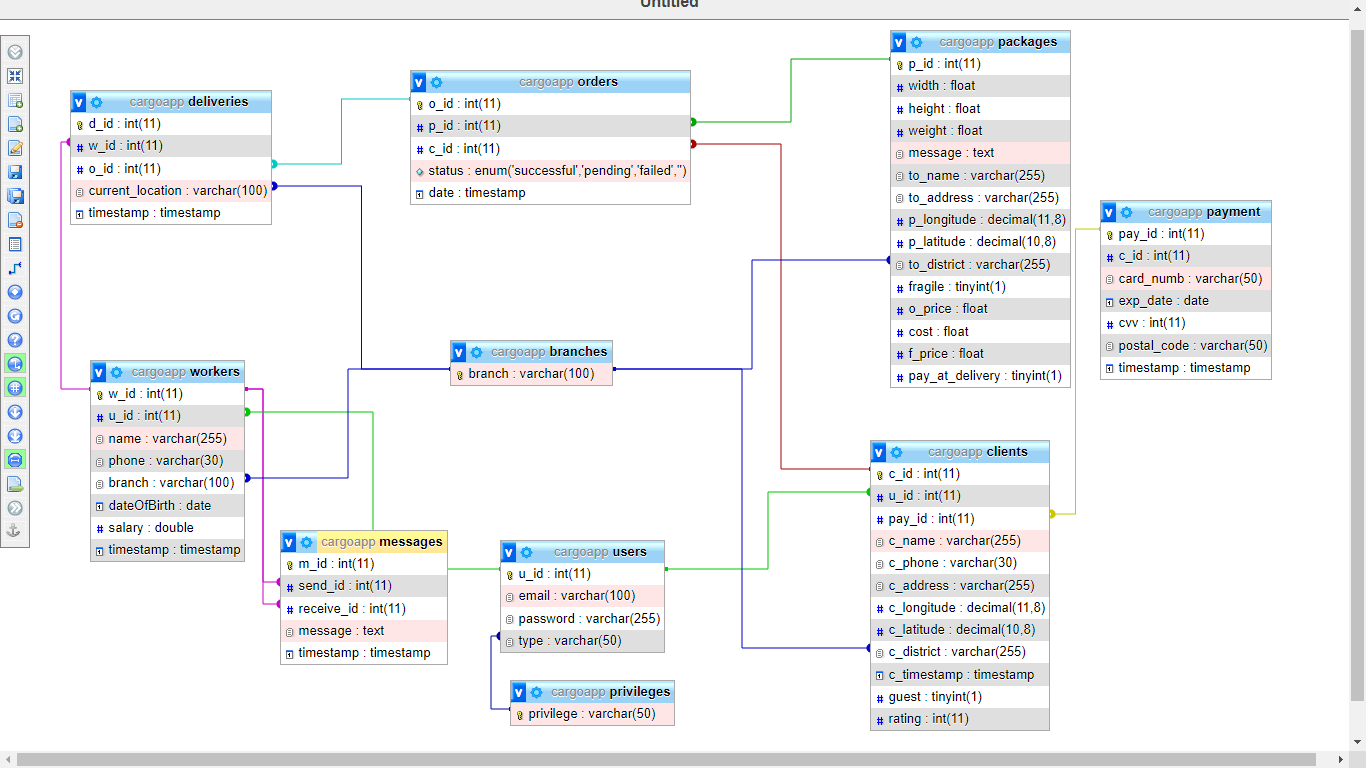


Figure ‑: class diagram

## Sequence Diagrams

A sequence diagram is a type of interaction diagram used to illustrate the interactions and relationships between different objects or components within an application or system. The diagram is time-focused and visually represents the order of interactions by using a vertical axis to depict time and horizontal arrows to indicate the flow of messages between objects or components. Sequence diagrams are useful for modeling complex processes and ensuring that different components within a system are communicating and interacting correctly. They can also help identify potential errors or areas of inefficiency in a system's design. By showing the interactions between objects or components, sequence diagrams can help developers better understand the overall flow of a system and make necessary adjustments to improve performance and functionality. To create a sequence diagram, we must first identify the objects or components within the system and the messages they send to each other. Each object is represented as a lifeline, and the messages are depicted as arrows connecting the lifelines. Additional information, such as the order and duration of messages, can be added to the diagram to provide more detailed insight into the system's operation. Overall, sequence diagrams are a valuable tool for designing, testing, and maintaining complex applications and systems.

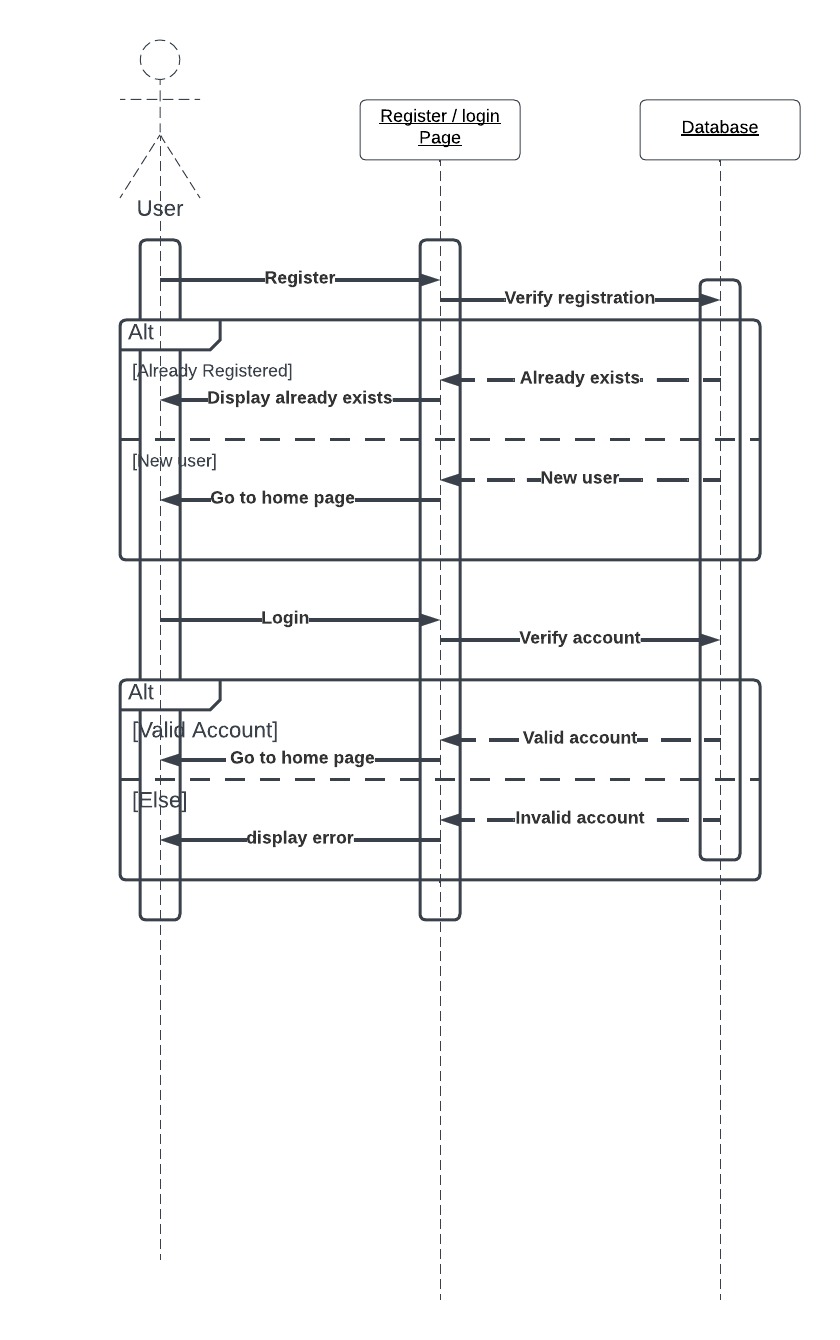


Figure ‑: all users sequence diagram

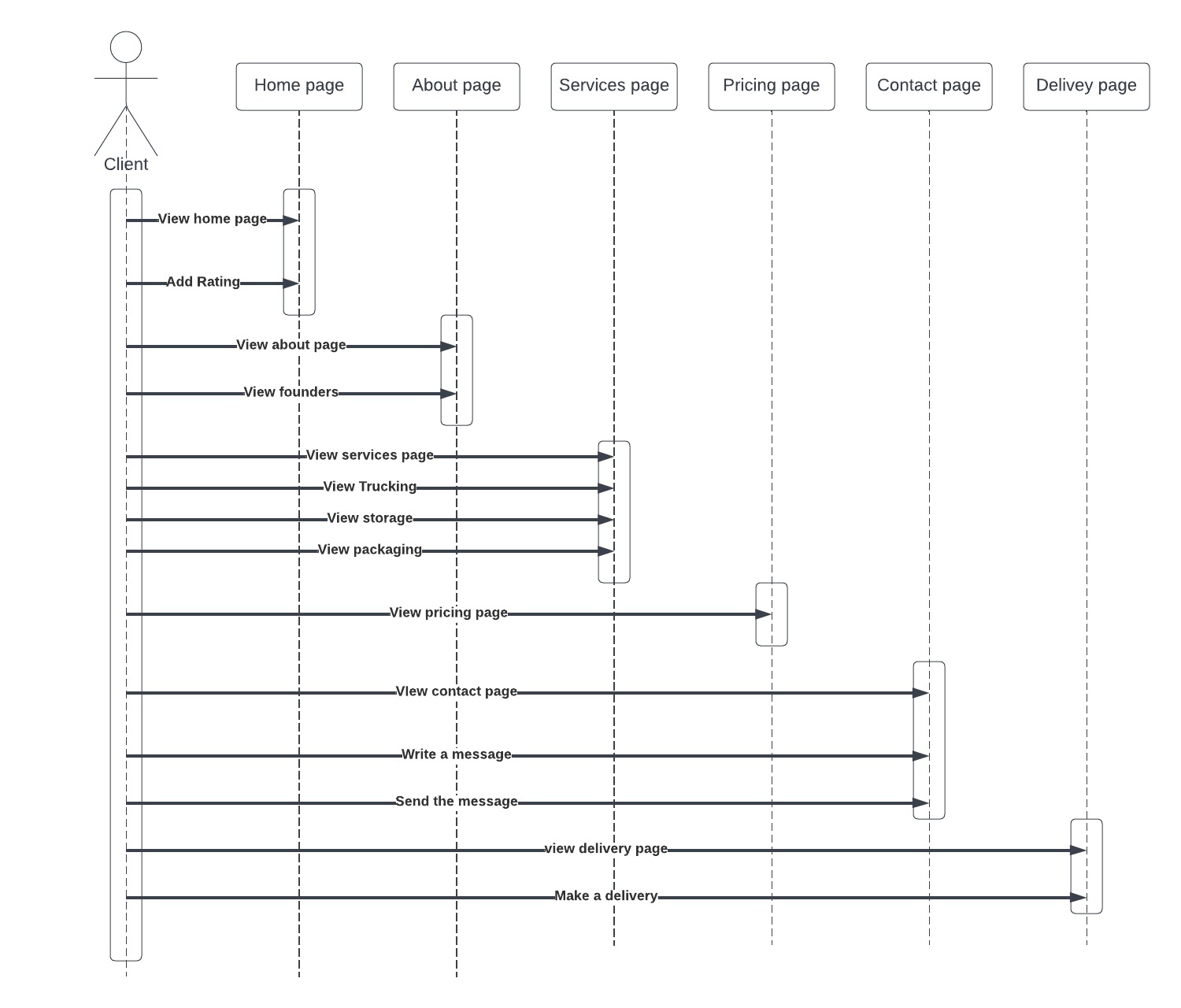
To access their accounts, all users have the option to navigate to the login page from the home screen. Once on the login page, the user is prompted to either sign up or log in to their existing account. For new users, signing up requires providing their personal information, such as their name, address, and contact information, which will be stored securely in the system's database. After successful registration, the user is directed to the login page where they enter their email address and password. Upon submission, the system verifies that the email and password are both correct and correspond to an existing user account in the database. If successful, the user is redirected to their account dashboard. If unsuccessful, the user is prompted to try again or reset their password if needed.

Figure ‑: client sequence diagram

The client sequence diagram outlines the steps that a client would take while interacting with the system. Upon launching the application, the client is prompted to login using their email and password. The application then verifies the login information by checking if it exists in the database and correctly relates to one another. If the information is valid, the client is then logged in to their page, where they are able to view the services provided by the company, including the prices and three main tabs: track deliveries, order delivery, and history. If the login information is invalid, the client remains on the login page until further action is taken. Alternatively, the client also has the option to sign up for an account, which involves checking if their account already exists in the database. If the account already exists, the registration process is aborted. If the account does not exist, the client is registered and then automatically logged in.

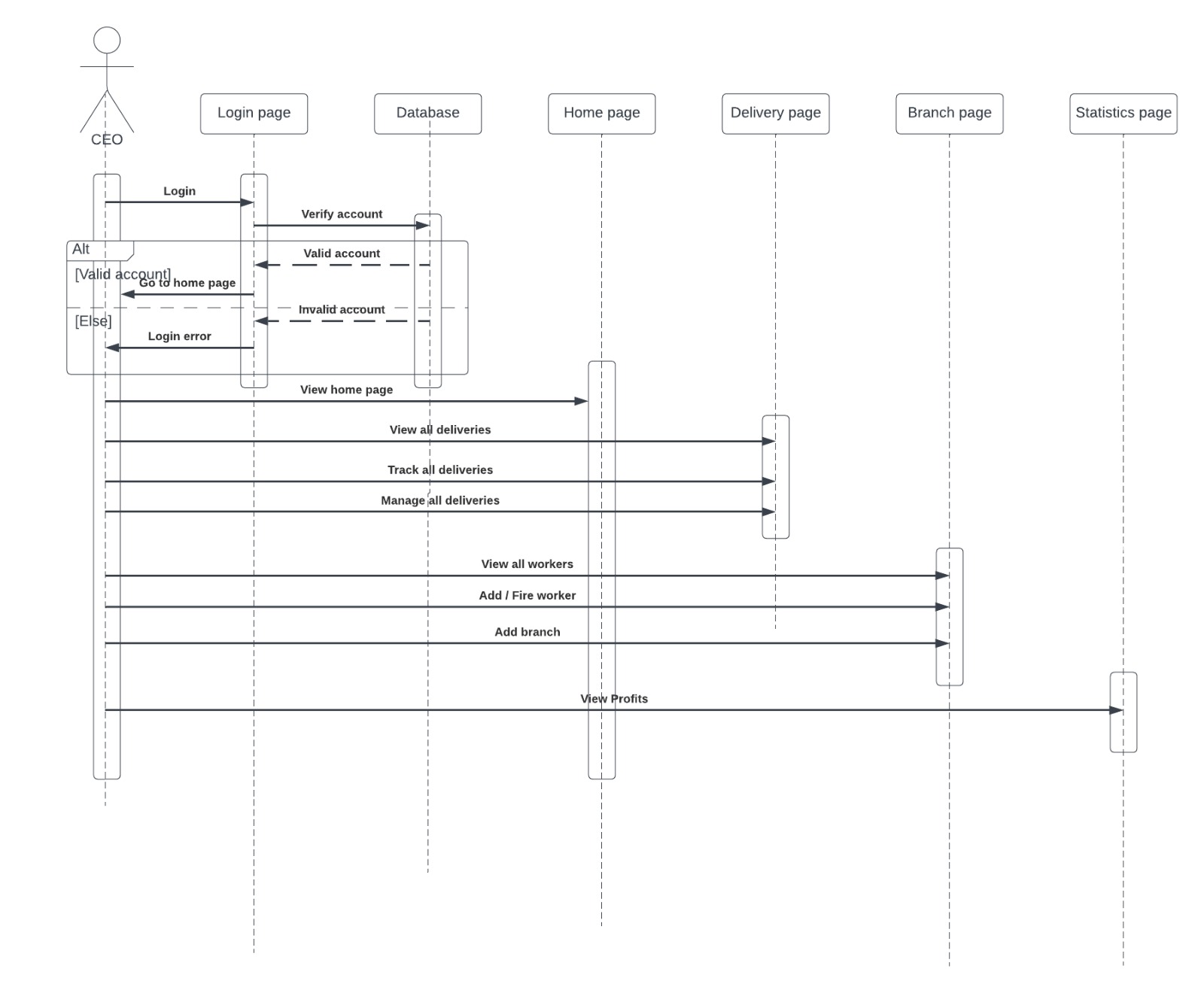


Figure ‑: CEO sequence diagram

The CEO logs in to the system by providing his or her credentials. The system then verifies the credentials and grants access to the CEO. Once the CEO is logged in, he or she can perform various actions. For example, the CEO can view the list of deliveries, which involves sending a request to the system to retrieve the relevant data. The system responds by returning the list of deliveries to the CEO. The CEO can also manage the list of deliveries by adding new deliveries, updating existing ones, or deleting old ones. To do this, the CEO sends a request to the system with the appropriate parameters. The system processes the request and updates the list of deliveries accordingly. Another task that the CEO can perform is managing the branches of the company. This involves sending a request to the system to retrieve the list of branches. The system responds by returning the list of branches to the CEO. The CEO can then select a specific branch and manage it by adding new workers, updating their information, or removing them from the system.

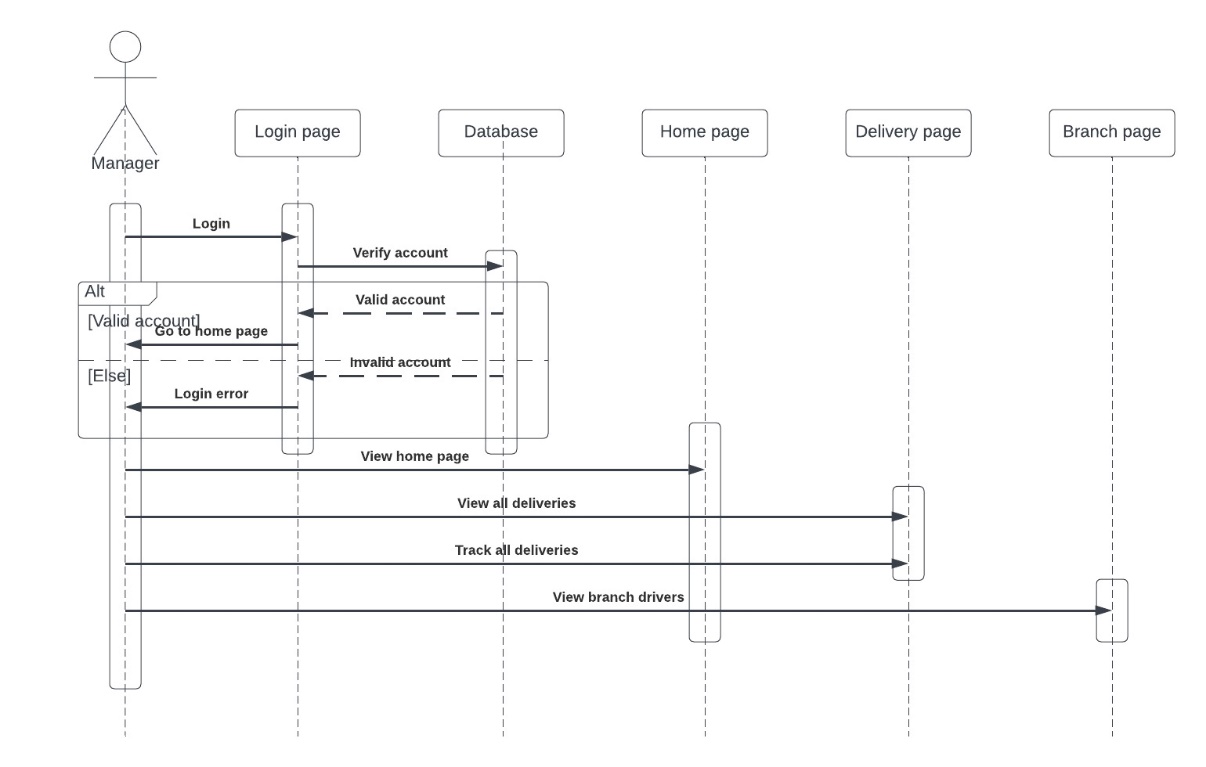


Figure ‑: Branch Manager sequence diagram

The sequence diagram for the branch manager begins with the manager logging into the system using their unique login credentials. Once authenticated, the manager is directed to their personal dashboard, where they can view their branch's delivery information and monitor worker activity. The manager can choose to view their branch's delivery list or worker list, where they can add or remove workers as needed. They can also update the status of each delivery by marking it as "delivered" or "in transit." Additionally, the manager has the ability to assign a worker to a specific delivery and track their progress using the system's GPS capabilities. If there are any issues or problems with a delivery, the manager can send a message to the worker or contact customer support for further assistance. Finally, when the manager is finished with their tasks, they can choose to log out of the system to ensure the security of their account.

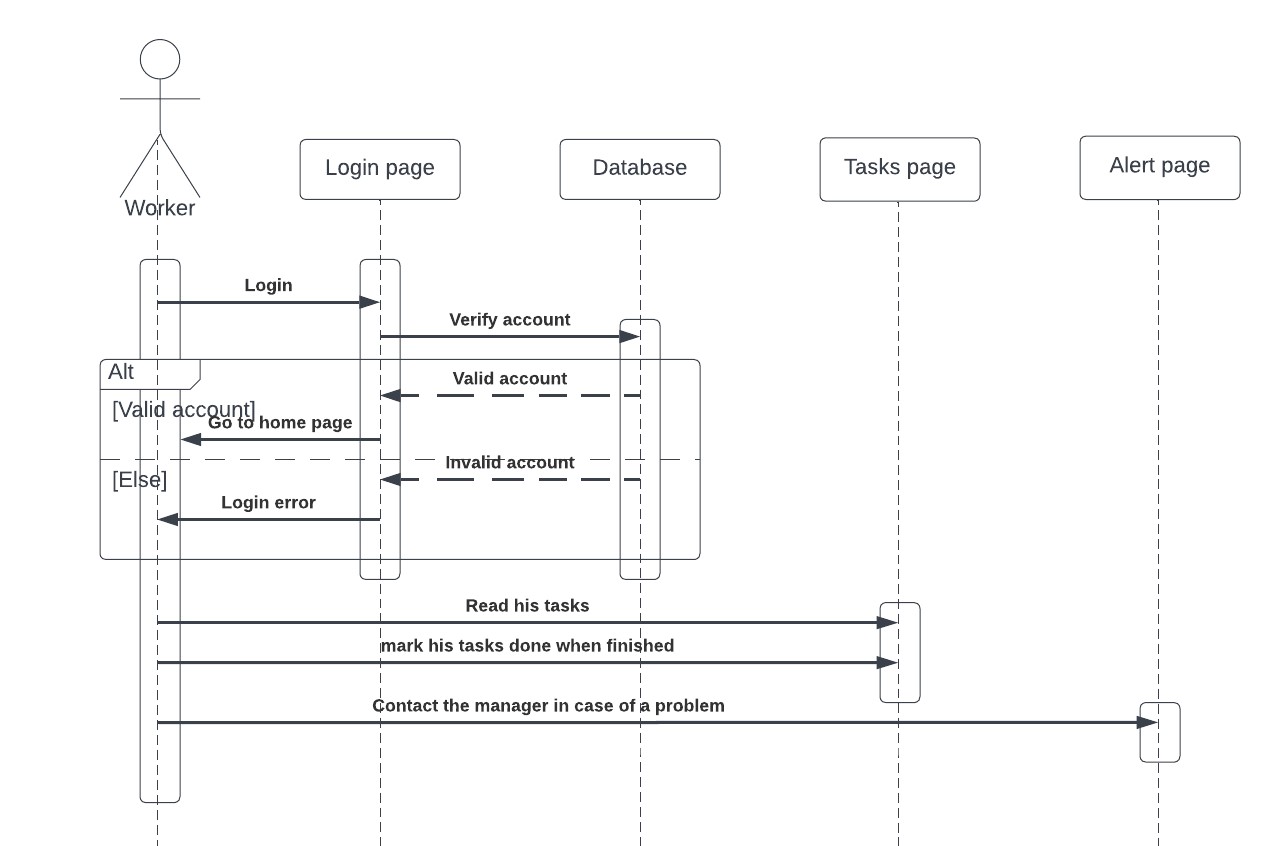


Figure ‑: worker sequence diagram

The sequence diagram for the worker begins with the worker logging in to the application by providing their login credentials. Once logged in, the worker is directed to their dashboard where they can view their assigned deliveries for the day. The worker can then select a specific delivery and mark its current location in the system. If the worker encounters any issues or problems during the delivery process, they can report it through the application by sending a message to their manager. The manager will receive the message and can take appropriate action to resolve the issue. Once the delivery has been successfully completed, the worker marks it as delivered in the system (mobile application). The delivery status is then updated in the database, allowing the client to track the package's progress. Finally, the worker can log out of the system to end their session.

## Activity Diagrams

An activity diagram is a type of UML (Unified Modeling Language) diagram that shows the flow of activities within a system or process. It depicts the sequence of actions or steps involved in completing a task, and can be used to model a wide range of scenarios, including business workflows, software processes, and system interactions.

Activity diagrams are useful for visualizing complex processes, breaking them down into simpler steps, and identifying potential bottlenecks or inefficiencies. They can help stakeholders understand the flow of a process, and can also be used to communicate the design of a system to developers. An activity diagram consists of nodes and edges. The nodes represent activities, decisions, and initial or final states, while the edges represent the flow of control between them. Each activity node can contain a description of the action it represents, as well as other details such as inputs, outputs, and conditions.

Activity diagrams are particularly useful for modeling use cases, as they can provide a clear and concise overview of the steps involved in completing a particular task or process. They can also be used to document business processes, system interactions, and software workflows. Overall, activity diagrams are a valuable tool for system analysts, designers, and developers, helping them to better understand and communicate the flow of activities within a system or process.

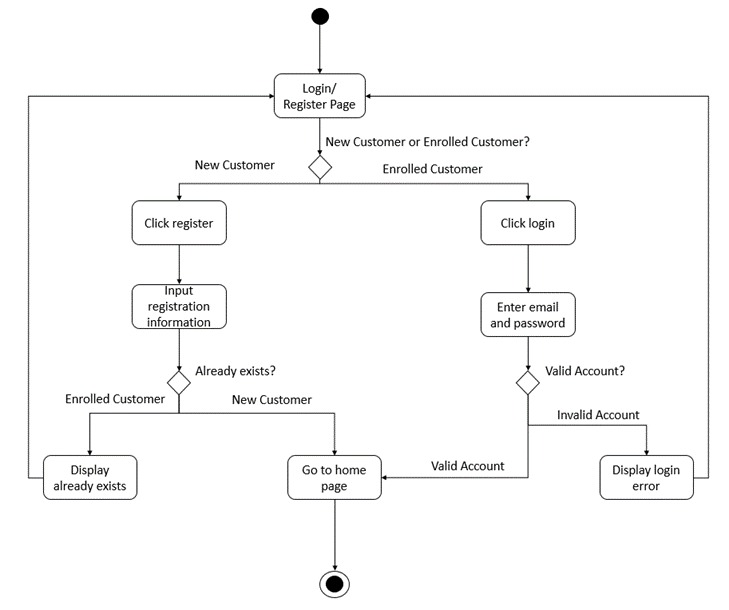


Figure ‑: activity diagram

When a user attempts to log in to the application, their email and password are verified against the database to ensure that they correspond to an existing account. If the login credentials are validated, the user is granted access to their personal page. However, if the verification fails, the user is unable to proceed beyond the login page until they take further action. Additionally, the application provides users with the option to register for an account. During the registration process, the system checks to see if the user's account already exists in the database. If the account exists, the registration process is terminated. On the other hand, if the account is not found, the user is successfully registered and automatically logged in.

## Entity-Relationship (ER) Diagrams

An ER diagram, also known as an entity relationship diagram, is a visual representation of the entities, attributes, and relationships within a database or information system. This diagram helps in the conceptual design of databases, providing a clear view of how data is organized and how different entities and attributes relate to each other. An ER diagram consists of various elements, including entities, attributes, relationships, and cardinalities. Entities are the objects or concepts within the system, such as customers or orders, while attributes are the properties or characteristics of these entities, such as a customer's name or address. Relationships represent how different entities are related to each other, such as how an order is related to a customer. Cardinalities specify the nature of these relationships, such as how many orders a customer can have. Drawing an ER diagram is an essential step in the database design process, allowing designers to create a blueprint for the data model and identify potential issues before implementation. ER diagrams are also useful in communicating database design concepts to stakeholders and developers, ensuring everyone has a shared understanding of the data model.

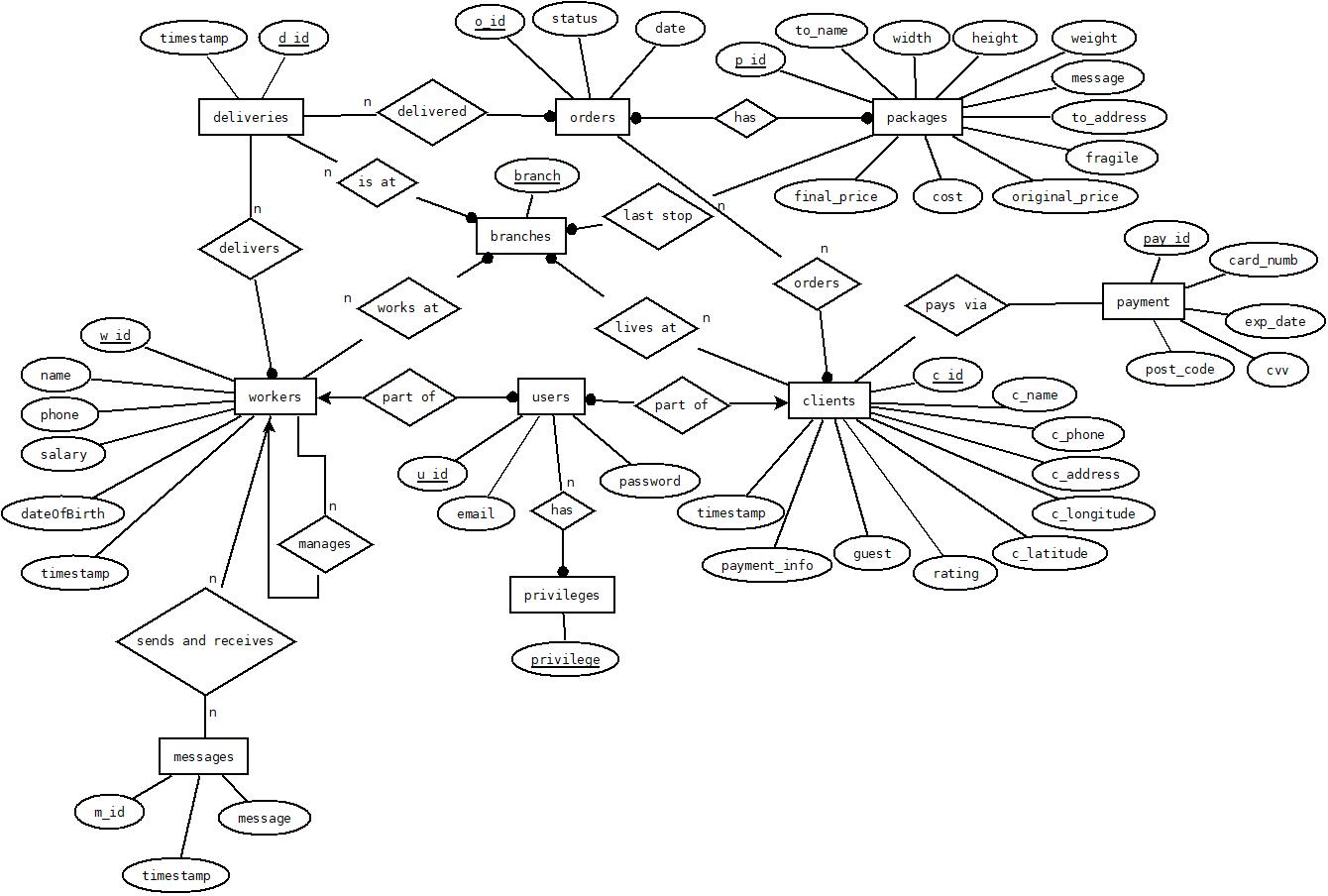


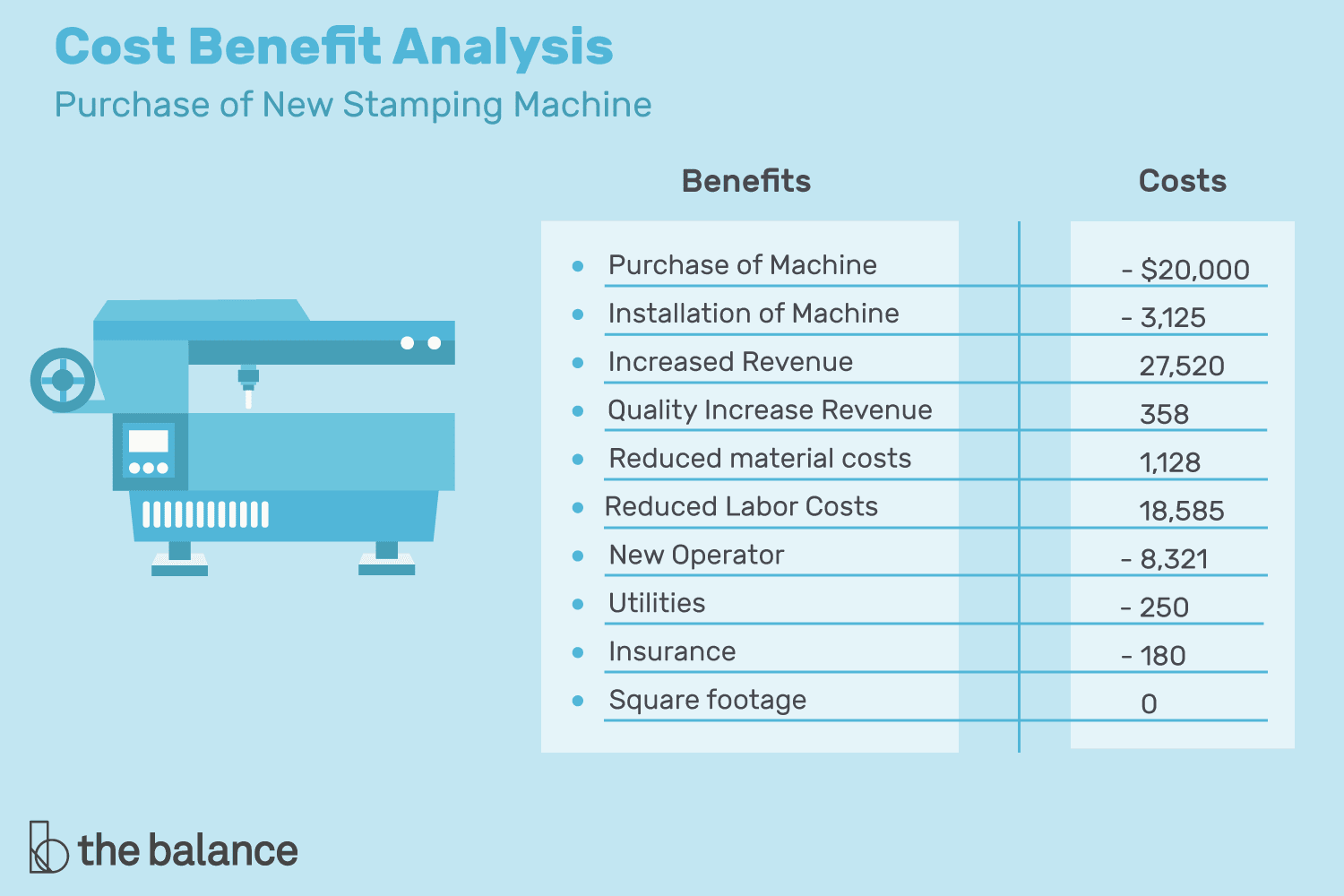
Figure ‑: ER diagram

In the ER diagram provided, clients and workers are categorized as users and granted appropriate privileges based on their respective roles, which include Client, CEO, BranchManager, IT, and worker. Each client is associated with a specific district near a branch, while each worker is assigned to a particular branch within the company. The deliveries table is designed to track the location of deliveries and which worker delivered them. When a client places an order, the specific details are stored in the packages table. The order table serves as a link between the client, their package, and the delivery. The client and worker tables provide further information about these major user types. The CEO and IT have the ability to manage other workers, further highlighting their roles within the system. As well as the CEO has the ability to send messages to other workers via the application.

## Non-Technical Aspects

### Financial Viability

A cost benefit analysis of the project. Here is an example:



**Figure ‎3-2- Cost Benefit Analysis**

### Stakeholders

The stakeholders of this project can be divided into different groups, including:

* Customers: They are the primary users of the application, and will benefit from the convenience of placing orders, tracking deliveries, and managing their accounts. They may be harmed if the application fails to function properly, or if their personal information is compromised.
* Company management: This includes the CEO, branch managers, and IT personnel, who will benefit from the efficient management of deliveries and workers, as well as the ability to monitor the company's profits. They may be harmed if the application fails to meet their needs or if it causes more problems than it solves.
* Workers: They will benefit from the ability to easily track deliveries and manage their work schedules, but may also be harmed if the application adds unnecessary burdens to their work or if it fails to work properly.
* Third-party service providers: This may include payment processors, and other service providers that are integrated into the application. They will benefit from increased business, but may be harmed if the application fails to properly integrate with their services or if it causes problems for their own operations.
* Regulators and legal authorities: They may have a say in how the application works in order to ensure compliance with laws and regulations related to data privacy, consumer protection, and other relevant issues.

Ultimately, the stakeholders who should have a say in how the project works include all of the groups mentioned above, as well as any others who may be affected by the application's operations. This may include representatives from customer advocacy groups, worker organizations, and other relevant stakeholders who can provide valuable input on how the application can be improved and how potential harms can be mitigated.

### Scope

The scope of this project includes the development of a web-based application for package delivery management. The application should allow clients to register and log in to their accounts, view available services, and place delivery orders. The application should also allow for the tracking of deliveries and the viewing of order history. The CEO, branch managers, and workers should have access to the application with varying levels of privileges depending on their roles. The application should be secure and scalable, able to handle a potentially large number of users and packages. The project should be completed within a set timeline and budget.

### Risks

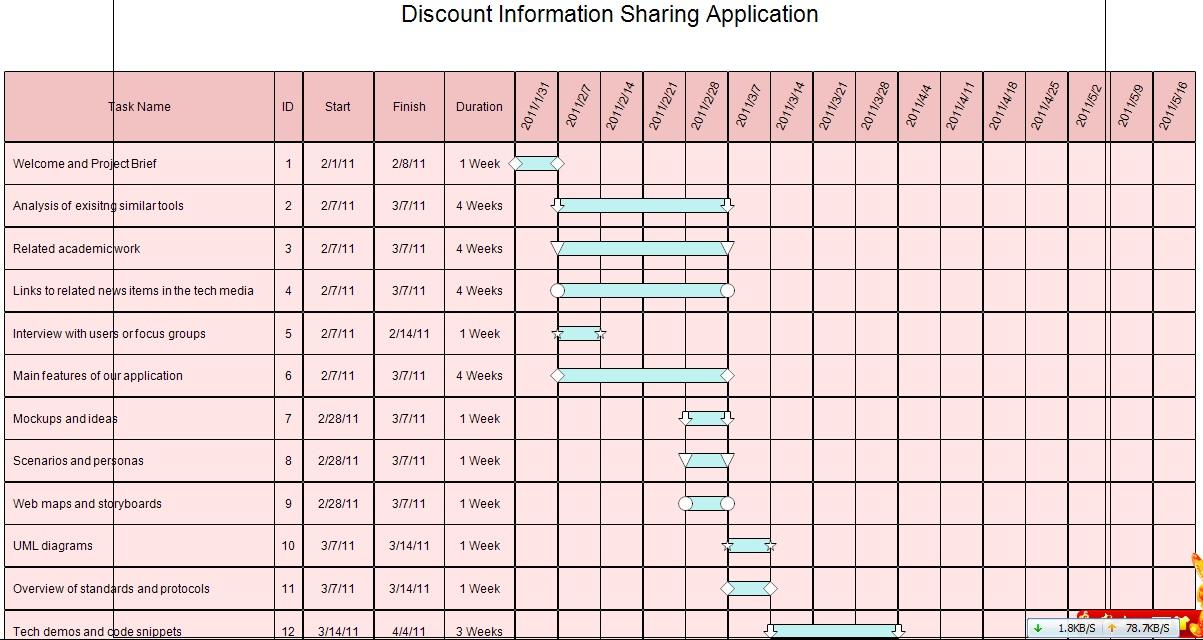
There are several risks that could potentially impact the success of this project. One major risk is technical challenges that may arise during the development process, such as compatibility issues or system malfunctions. Another risk is the possibility of security breaches or data leaks, which could lead to the loss of sensitive information or damage to the company's reputation. Additionally, the project may face financial risks if costs exceed the allocated budget or if there are unexpected expenses that arise.

Other factors that may impact the project's success include changes in regulations or market conditions, stakeholder disagreements or conflicts, and unforeseen external events such as natural disasters or global crises.

To mitigate these risks, the project team should conduct thorough testing and quality assurance procedures, implement robust security measures, and regularly review and adjust the project plan as needed to stay on track and within budget. Additionally, open communication and collaboration with stakeholders can help to identify and address potential issues before they become major problems.

### Schedule and Milestones

What will be completed and when. Here is an example:



**Figure ‎3-3- Scheduling Tasks and Milestones**

### Ethical and Social Considerations

There are several ethical issues that should be taken into consideration when completing the design of the application. One of the primary concerns is data privacy and security. As the application will store personal information of clients and workers, it is important to ensure that this data is securely stored and protected from unauthorized access.

Another ethical issue to consider is fairness and transparency in the application's decision-making processes. For example, if the application is used to assign work to individual workers, it is important to ensure that this process is fair and unbiased, and that workers are not discriminated against based on factors such as age, gender, or race.

Additionally, it is important to consider the potential impact of the application on the environment. For example, if the application is used to manage deliveries, it is important to consider how this may impact carbon emissions from transportation and explore ways to reduce the environmental impact.

Finally, it is important to ensure that the application does not perpetuate any harmful stereotypes or biases, particularly with regards to the representation of different groups of people such as clients and workers.

### Environmental and Sustainability Considerations

As an application-based project, it is unlikely that there will be significant environmental impact associated with its development and use. Therefore, it can be stated that this project has no major environmental considerations or impact. However, if the project is to be implemented in such a way that involves the use of physical resources such as server infrastructure, energy consumption and disposal of hardware should be considered in order to minimize any potential environmental impact.

### Relevant Standards

* ISO 28000: This standard provides guidelines for implementing a security management system in the supply chain.
* ISO 31000: This standard provides guidelines for implementing a risk management system in an organization.
* Electronic Product Code (EPC): This standard provides a framework for identifying, tracking, and managing goods in the supply chain using RFID technology.
* Electronic Data Interchange (EDI): This standard provides a format for exchanging business documents electronically, such as purchase orders, invoices, and shipping notices.
* Application Programming Interface (API): This standard provides a way for different software systems to communicate with each other, enabling data sharing and integration.
* Global Positioning System (GPS): This standard provides a way to determine the precise location of a vehicle or shipment using satellite technology.
* Internet Protocol (IP): This standard provides a way to transmit data over the internet and other networks.
* Secure Sockets Layer (SSL): This standard provides a way to secure data transmissions over the internet using encryption.
* Transport Layer Security (TLS): This standard provides a way to secure data transmissions over the internet using encryption and authentication
* PHP: a server-side scripting language commonly used for web development. There are several PHP standards organizations, such as the PHP Standards Group, that provide coding standards and best practices for PHP development.
* JavaScript: a scripting language used primarily for client-side web development. The ECMAScript specification provides the standard for JavaScript syntax and behavior.
* HTML/CSS: the markup and styling languages used to create web pages. The World Wide Web Consortium (W3C) provides standards for HTML and CSS development.
* MySQL: the programming language used to manage relational databases. The ANSI SQL standard defines the syntax and behavior of SQL.

In addition to these technical standards, there may be non-technical standards that are relevant to the project, such as organizational policies and procedures, industry regulations, or legal requirements.

## Conclusion

The cargo tracking application is designed to provide users with a seamless experience in tracking their packages. The application caters to multiple user types, including clients, workers, branch managers, IT, and CEO, each with different levels of access and privileges. The ER diagram outlines the relationships and dependencies between different tables, while the sequence diagrams depict the step-by-step interactions between users and the system.

During the design process, various ethical and social considerations were taken into account, such as data privacy and security, as well as the potential impact of the application on stakeholders, including clients and workers.

To ensure the smooth functioning of the application, various technical standards, such as programming languages and database management systems, were used. However, there are also risks involved in the project, such as technical failures or unforeseen ethical issues.

Overall, the scope of the project was clearly defined, with a focus on delivering a user-friendly and efficient package tracking system.

# Implementation/Simulation and Testing

## Introduction

This paragraph in meant to introduce the topics to be covered in this chapter.

## Implementation Tools

List the tools used for implementing the system. This includes hardware, compilers, IDE, frameworks, CASE tools, etc...

## Implementation Summary

Description of detailed implementation steps. Demonstrate the typical code fragments (details of implementation, e.g. source code listings must be included in an appendix and saved on an accompanying CD/DVD)

## Test Cases and Acceptance Criteria

Describe the test cases used and the acceptance criteria.

## Conclusion

This paragraph in meant to draw conclusions highlighting the main ideas in this chapter.

# Conclusion and Future Work

## Conclusion

The cargo/delivery application project aims to develop a solution that addresses the challenges faced by logistics providers and customers, such as lack of visibility, inefficient processes, limited communication, security concerns, competition, and increasing demand. By providing real-time tracking, process automation, improved communication, enhanced security measures, a unique value proposition, and valuable data insights, the cargo/delivery application will help logistics providers optimize their operations, reduce costs, increase revenue, and improve customer satisfaction.

## Future Work

Describe the opportunities for expanding the work done in this thesis.

**APPENDIX A:   
Implementation Details**

Any details not fit in chapter 5: e.g. detailed calculation, complex algorithms, etc…

**APPENDIXB:  
 USER Manual**

Fill in the instruction manual for using the application

**APPENDIXC:   
deployment and configuration Manual**

Outline the deployment and configuration details in addition to any know troubleshooting techniques.

**REFERENCES**

|  |  |
| --- | --- |
| [1] | X. Y. a. X. C. J. Huang, "E-commerce logistics in China: current status, issues and future prospects," in *Journal of Industrial Engineering and Management Science*, 2019. |
| [2] | L. Wang, "The Impact of the Internet on Logistics Management," Journal of Applied Sciences, [Online]. Available: https://thescipub.com/pdf/10.3844/jassp.2014.1507.1514. [Accessed 2023]. |
| [3] | M. Nuce, "The Evolution of Logistics: What's Changed Since the 90s?," Inbound Logistics, [Online]. Available: https://www.inboundlogistics.com/cms/article/the-evolution-of-logistics-whats-changed-since-the-90s/. [Accessed 2023]. |
| [4] | K. Page, "The Impact of Technology on Logistics and Supply Chain Management," Google Books, [Online]. Available: https://books.google.com/books?id=J0fGDwAAQBAJ&printsec=frontcover#v=onepage&q&f=false. [Accessed 2023]. |
| [5] | [Online]. Available: https://pixabay.com/images/search/shipment/. |
| [6] | S. T. a. H. Huang, "The Role of Internet in Logistics: A Case Study," in *Journal of Industrial Engineering and Management Science*, 2019. |
| [7] | Y. Z. a. Z. Yang, "The Impact of Internet on Logistics Management: A Literature Review," in *International Journal of Trade, Economics and Finance*, 2017. |
| [8] | "Wakilni," Wakilni, [Online]. Available: https://wakilni.com/. [Accessed 2023]. |
| [9] | "aramex solutions and services," Aramex, [Online]. Available: https://www.aramex.com/lb/en/aramex-home?utm\_source=google&utm\_medium=cpc&utm\_campaign=AMX-CONV-WW-SC-IP-20210506&utm\_term=general&gclid=CjwKCAiA3pugBhAwEiwAWFzwdTlc3N3GRcFIX8FkntH0epGAOly105NUqFkEGTn-Opi57y01-2TpjRoC5mIQAvD\_BwE. [Accessed 2023]. |
| [10] | "Go share," GoShare, [Online]. Available: https://goshare.co/. [Accessed 2023]. |
| [11] | "techtarget," techtarget, [Online]. Available: https://www.techtarget.com/searchcontentmanagement/definition/content-management-system-CMS. [Accessed 2023]. |
| [12] | "TechnoPedia," TechnoPedia, [Online]. Available: https://www.techopedia.com/definition/5435/graphical-user-interface-gui. [Accessed 2023]. |