

# Introduction

CKF, a multinational e-commerce company commonly known as CKF Group Malaysia, has made a name for itself with its best-selling categories such as Baby products, Accessories, Home Appliances, and Mobile phones through its e-commerce platform. However, CKF has realized that its current Product Delivery System (PDS) is not equipped to handle the increasing demand for products and lacks user-friendliness. Therefore, CKF is looking for a new PDS that can overcome these challenges, enabling them to take control of delivery operations, increase revenue, and provide customers with a seamless, user-friendly delivery experience on a single platform.

## Objectives

* + - To develop an improved PDS to handle high demand, increase efficiency, and provide seamless delivery for customers.
    - To optimize delivery operations, reduce delivery times, and expand to new markets to increase revenue.
    - To improve customer satisfaction with a user-friendly and reliable PDS that meets their expectations.
    - To use technology and data analytics to streamline delivery logistics, reduce costs, and improve resource allocation.

## Scope

* + - Create and implement an advanced Product Delivery System (PDS).
    - Analyze the current delivery system to identify challenges and opportunities for improvement.
    - Design a user-friendly and reliable PDS to meet customer needs.
    - Use technology and data analytics to optimize delivery logistics and reduce costs.
    - Collaborate with stakeholders from various teams.
    - Execute the project in phases, testing and validating each functionality.

## Problem Background and Requirement

The system is not easy to use and is unable to handle the increasing demand for products. Therefore, they require a new PDS that can address these issues and take control of delivery operations, increase revenue, and provide a user-friendly experience for their customers. The team has been assigned to design, develop and implement a new PDS using RMI, a distributed computing technology. The system should allow customers to register an account, order items, and generate reports while ensuring secure communication between the customer and the PDS (McEwen, 2004).

## Requirement

One of the requirements for the new Product Delivery System (PDS) for CKF's e-commerce platform is that it should be user-friendly and easy to use for customers. This includes allowing customers to register an account with their first name, last name, and providing clear instructions for ordering items, and generating reports. Additionally, the system should be designed to handle increasing demand for products and provide a seamless experience for customers throughout the entire delivery process (Alan Faisandier, 2020).. Finally, the system should ensure secure communication between the customer and the PDS to protect sensitive information.

### Customer Requirement

The customer requirements for the new Product Delivery System (PDS) for CKF's e-commerce platform include the following:

* + - * Secure login functionality
      * Personal details management
      * Easy order placement
      * Order log tracking
      * Secure and easy payment process
      * Secure logout functionality

# Research

Research is a crucial aspect of developing a new Product Delivery System (PDS) for CKF's e- commerce platform. It involves investigating various technologies and tools that can be used to create a secure and reliable system that meets the requirements of the organization and its customers. The research phase may include evaluating existing PDS systems to identify their strengths and weaknesses, exploring different distributed computing technologies, selecting appropriate programming languages, and choosing a suitable database or file system (Warlina, 2018).

# Design

Design is the process of creating a detailed plan for the construction of a system or product. In the case of CKF's new Product Delivery System, the design involved selecting appropriate tools and technologies, defining system requirements, and creating a system architecture. Proper design minimizes potential issues and errors during development, resulting in an effective and efficient system.

## Schema Diagram

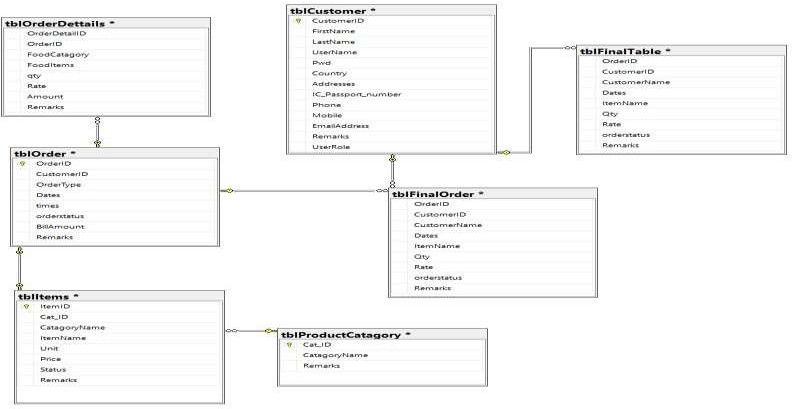


Figure 1 Schema diagram

The database diagram presented above illustrates how various tables within the system are interlinked. By examining the diagram, one can gain insights into the relationships between different tables. Specifically, the OrderDettails table is connected to the Order table through the use of OrderID. Therefore, the database diagram provides an overview of how tables in the database are interconnected and the keys used to link them.

* 1. **ER Diagram**

An ERD is a graphical depiction of data models that outlines the relationships between entities in a database or system. It employs symbols and lines to represent the different elements of a system and demonstrate how they are interrelated. ERDs are an essential tool in database design that enable a better comprehension of how data flows and how entities relate to each other. They are a widely adopted practice in database design and development.

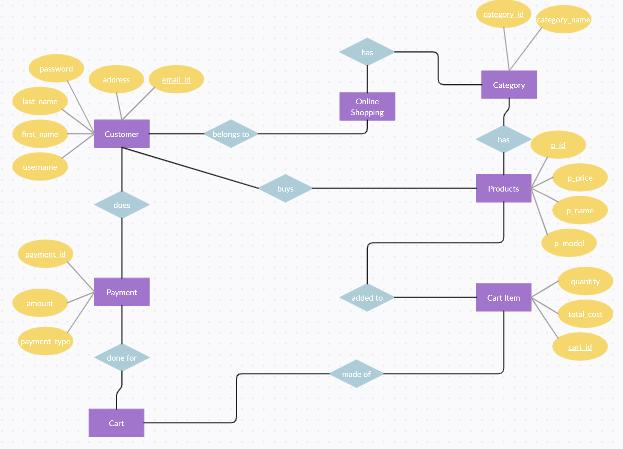
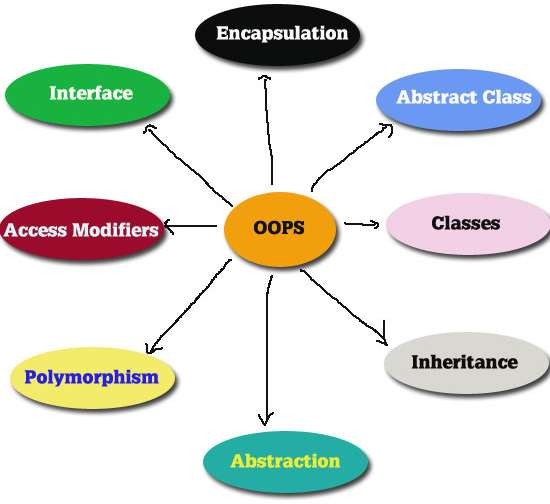


Fig: Entity relationship diagram

# Employing OOP concepts to resolve difficulties in distributed systems.

OOP is a programming methodology that emphasizes the use of objects as building blocks to create robust and scalable software. These objects encapsulate data and related functions in a modular and reusable fashion. The four core principles of OOP - inheritance, polymorphism, encapsulation, and abstraction - provide a foundation for designing software that is easy to maintain and extend (tutorialspoint, 2020).



## Abstraction

Figure 2 oops concept (Bogan, 2020)

Abstraction in OOP simplifies complex systems by separating behavior from implementation, exposing only relevant information to users. Abstract classes and interfaces define methods

without specifying how they should be implemented, enabling modular and flexible software that is easy to maintain and extend (Joyce, 2019).

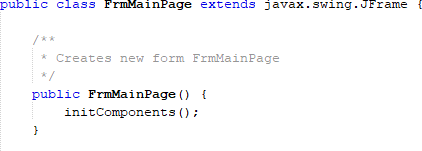


Figure 3 Example of Abstraction used in system

## Encapsulation

In OOP, encapsulation entails obscuring an object's internal workings and offering a constrained interface for dealing with it. This safeguards data integrity and makes it possible to alter how an object is implemented without having an impact on the rest of the program. Encapsulation enhances the security, upkeep, and adaptability of software systems (Joyce, 2019).

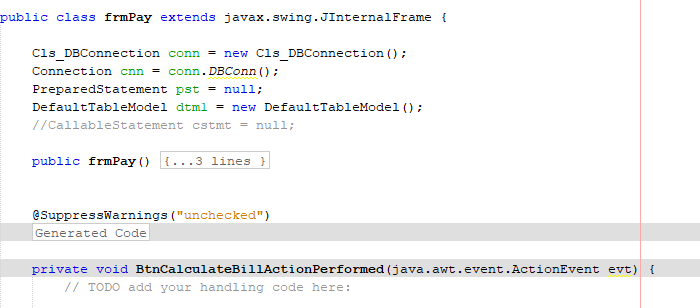


Figure 4 Example of encapsulation in the system.

## Inheritance

Inheritance makes it possible to build new classes out of older ones, preventing duplication of code and encouraging reuse. Subclasses introduce new methods and attributes while retaining those of

their parent classes. Classes may be organized hierarchically thanks to inheritance, which also makes it possible to design modular, extendable software systems (Joyce, 2019).

## Polymorphism

It alludes to the capacity of various things to react in various ways to the same message. Software systems are more adaptable and expandable because to polymorphism, which makes it possible to write code that can operate with objects of various types (Joyce, 2019).

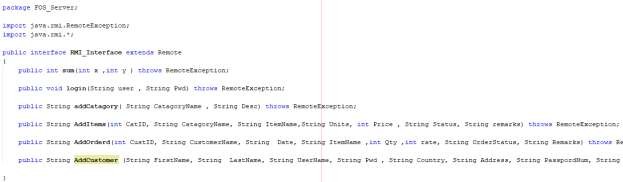


Figure 5 Example of polymorphism in code

# Implementation

## System Screenshots

### Product Order page

Figure 6 Screenshot of the product order page

The photograph provided above depicts the product order page displaying a comprehensive list of customer orders along with product availability and pricing information. Additionally, this page offers a detailed overview of each product for your convenience.

### Login page

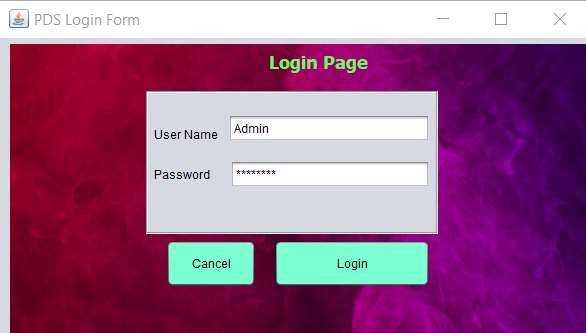


Figure 7 Screenshot of the login page

The photograph above displays a snapshot of the login page for the PDS system. Within the image, an administrator can be observed logging in by providing their username and password. Additionally, the login and cancel buttons are visible on this page.

### Home page

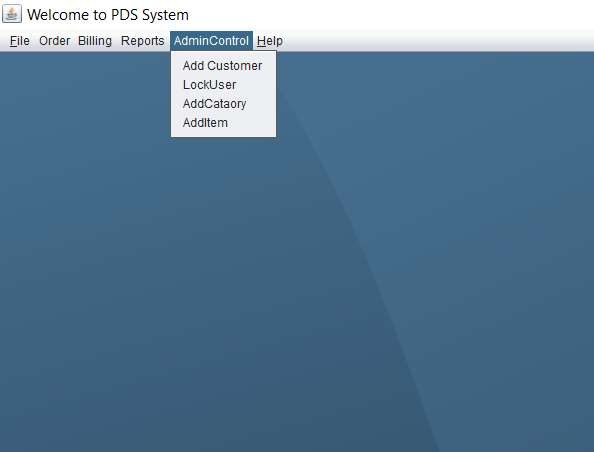


Figure 8 screenshot of the homepage

The photograph provided above presents a snapshot of the system's homepage, which offers an array of options such as files, orders, billing, reports, admin control, and help. Furthermore, this page allows access to various administrative controls including the ability to add customers, lock users, add categories, and include items.

### View users

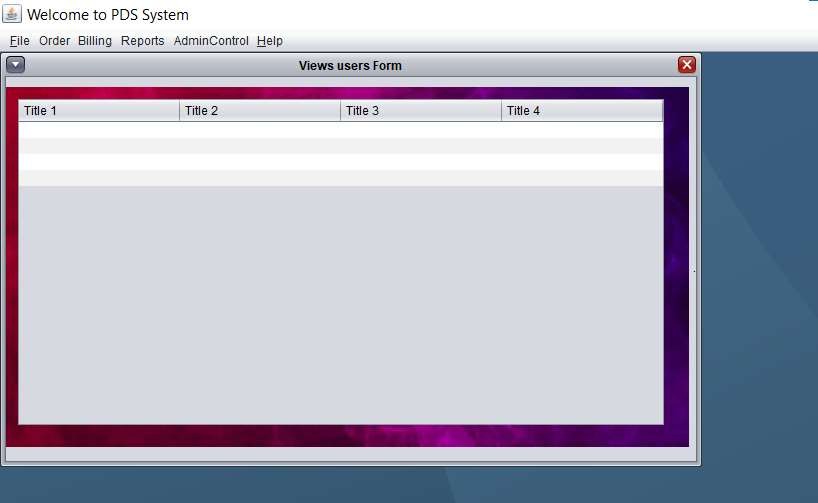


Figure 9 Screenshot of the view users

The image provided above displays a snapshot of the "View Users" form within the system. This particular page allows users to view a comprehensive list of all users in the system, along with their associated information.

### Change password



Figure 10 Screenshot of password change

Displayed above is a screenshot of the password modification page for the system. Here, the user is prompted to input their current password, along with their desired new password. To confirm the change, the user must re-enter the new password. Finally, at the bottom of the page, the options to either implement or cancel the password change are available.

### Ordered product with bill payment:

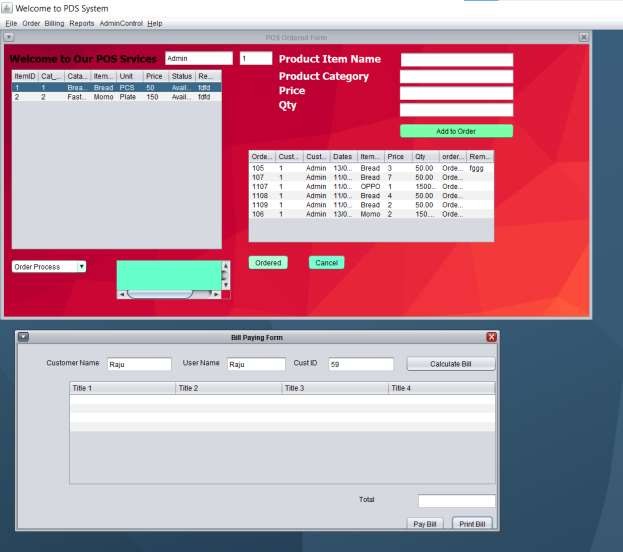


Figure 11 Screenshot of view order with payment information

The image displayed above captures a screenshot of the order viewing page that features payment information and a comprehensive list of all ordered items. This page offers a detailed overview of each payment made, providing the user with an accurate depiction of their transactions.

### All orders:

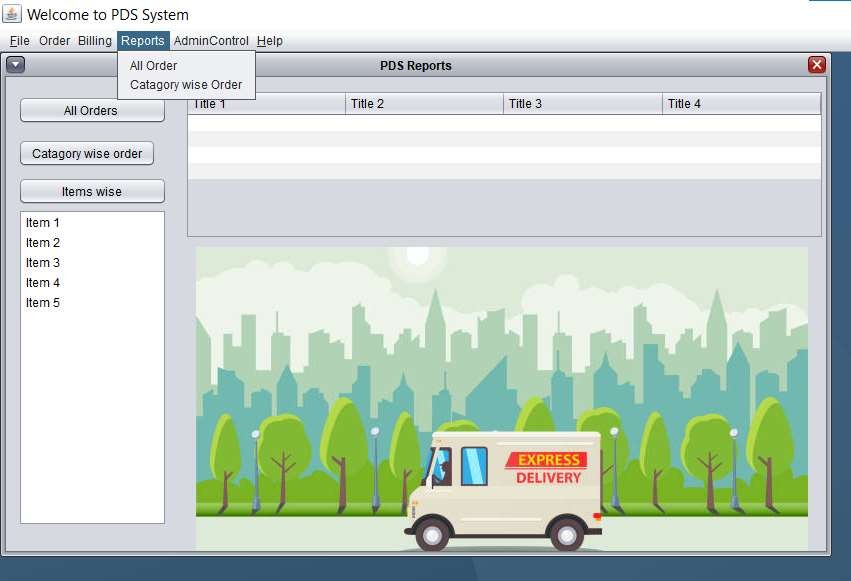


Figure 12 Screenshot of view all orders

Displayed above is an image of the order viewing page, where the user has the option to view their orders categorized by item or category. This page presents the entirety of the user's orders on a single page, simplifying the task of monitoring orders.

### Add customer Page:

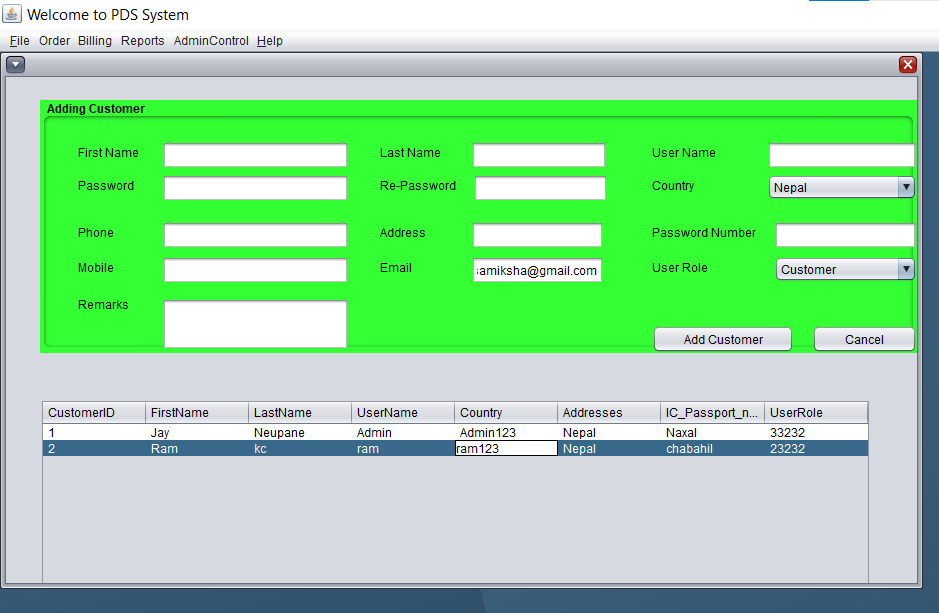


Figure 13 Screenshot of add customers page

The photograph displayed above shows the "add customer" page of the system, which enables you to view the list of current users and add new users to the system by supplying the required information as prompted.

### Add category page:



Figure 14 Screenshot of add category page

The photograph above is a screenshot of the "add category" page in the system, which allows you to add products to the system.

### Add products page:

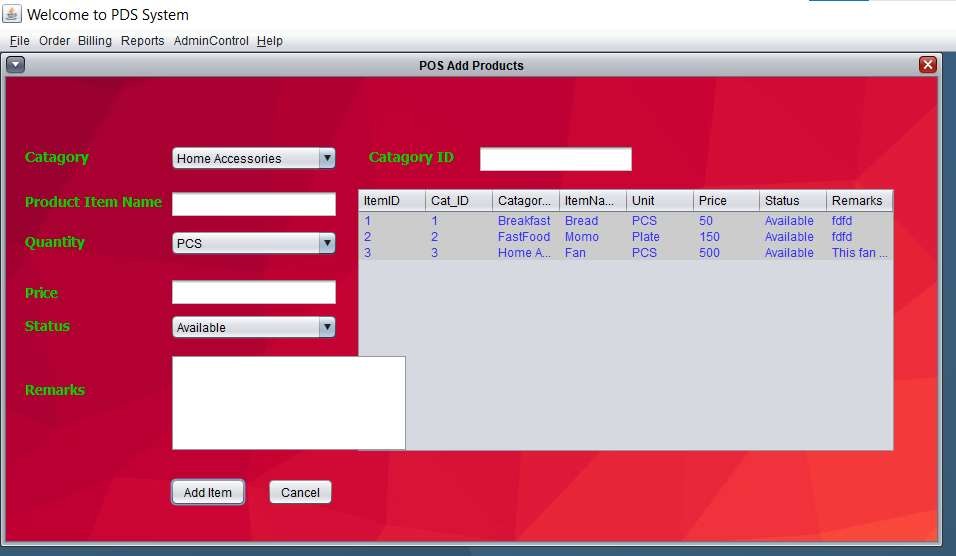


Figure 15 Screenshot of add product page

The photograph above depicts the "add product" page of the system, which facilitates the addition of new products. This page enables you to specify the quantity, status, and price of the product being added.

## Code snippet

Implementation of RMI Architecture:

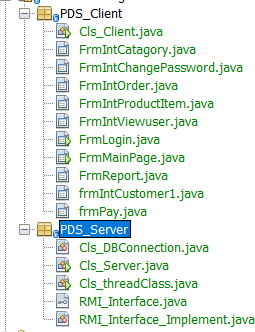


Figure 16 Screenshot of the implementation of RMI Architecture.



Figure 17 Screenshot for the add code

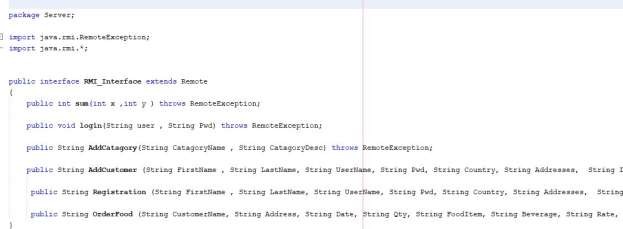


Figure 18 screenshot of the RMI Interface code

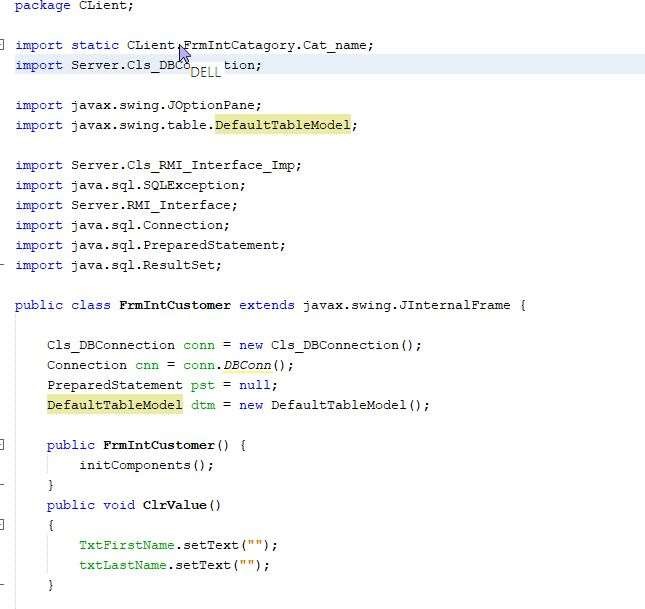


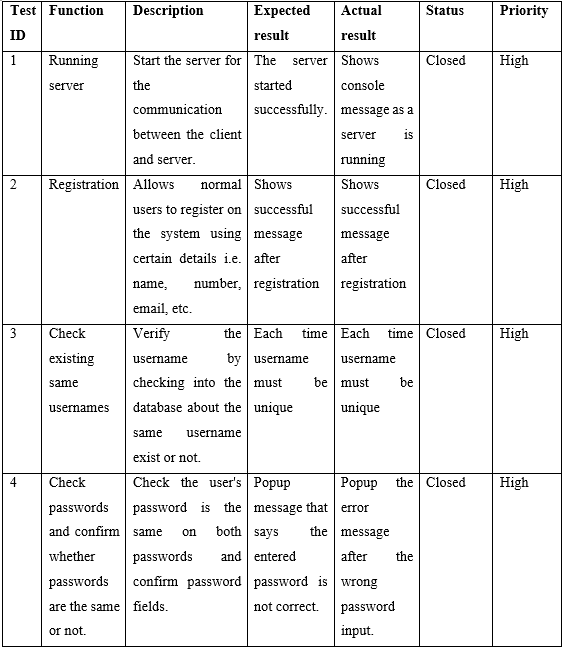
Figure 19 Screenshot of the RMI implementation code

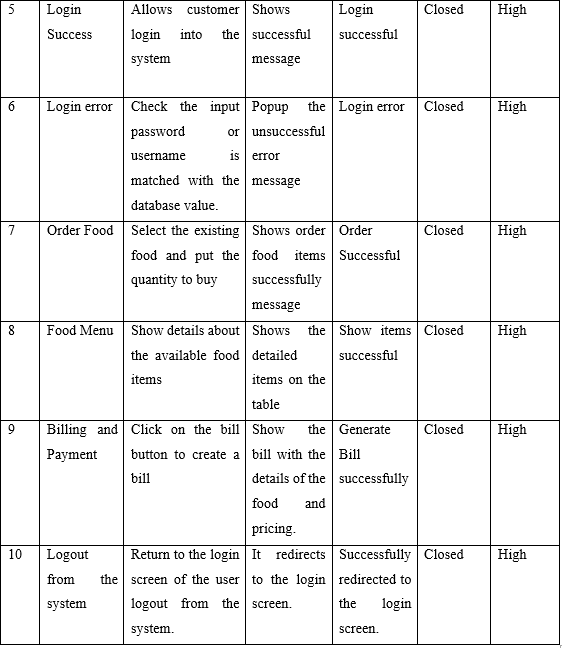


Figure 20 Screenshot of the code for login page

# Testing

The crucial process of software testing include assessing the usability, performance, security, and functioning of a software program. Software testing's main goal is to find any flaws, problems, or mistakes in the program before it is made available to end users. This procedure aids in making ensuring the program functions as planned, satisfies user needs, and is dependable and safe. To find problems in software applications, a variety of software testing approaches are used, including unit testing, integration testing, system testing, and acceptance testing (Hamilton, 2021).. By avoiding the delivery of flawed software, proper software testing techniques may help businesses save time, money, and reputation.





# RMI

RMI (Remote Method Invocation) is a Java technology that enables the development of distributed applications. It allows objects in one JVM to invoke methods on objects in another JVM, even if they are running on different physical machines or are located remotely. RMI provides an efficient and straightforward mechanism for building distributed applications, enabling various components to communicate with each other transparently and securely.

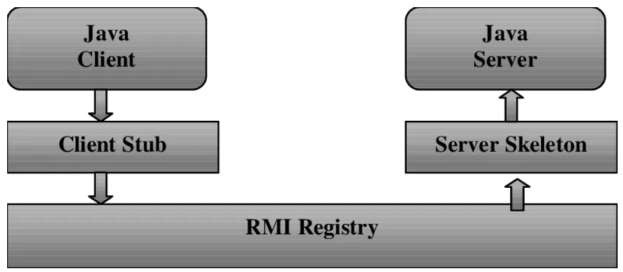


Figure 21 RMI

Following are the various technology involved in the development of RMI application:-

Interface Definition: The RMI interface defines the methods that the client can call remotely on the server. The server-side implementation of the interface is responsible for handling the method calls and returning the appropriate response.

Remote Object Implementation: The remote object implementation provides the actual implementation of the methods defined in the RMI interface. This implementation runs on the server-side and can be accessed by the client remotely.

Stub and Skeleton: The stub and skeleton are used to enable the communication between the client and server. The stub on the client-side marshals the parameters passed to the server and sends them over the network. The skeleton on the server-side marshals the parameters and invokes the appropriate method implementation.

Registry: The RMI registry is used to bind the remote object implementation to a name that can be accessed by the client. The client can then look up the remote object by the name and obtain a reference to it.

## Distributed system

A distributed system is a network of independent computers that work together as a single system, providing services to users. They communicate by passing messages over a network, and each has its own memory and processors. The goal is to improve performance, scalability, and fault tolerance, and to enable resource and information sharing. Applications and data can be spread across multiple nodes, which can be geographically dispersed.

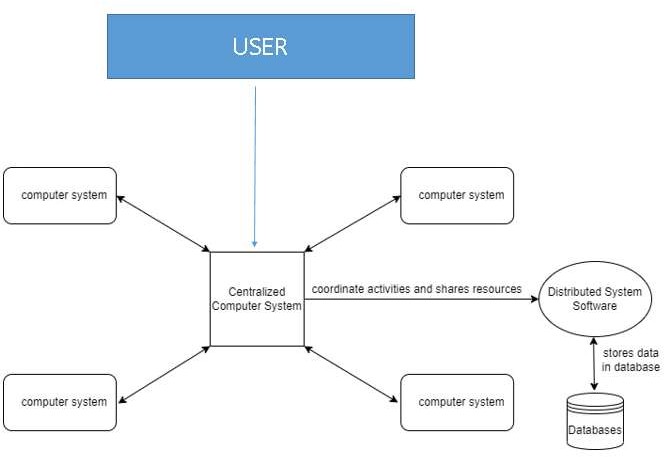


Figure 22 Diagram of distributed system (Ahamed, 2020)

## Blockchain

Blockchain is a tamper-proof and transparent digital ledger that records transactions across a distributed network using cryptographic techniques. It allows for secure and decentralized storage of data beyond just financial transactions.

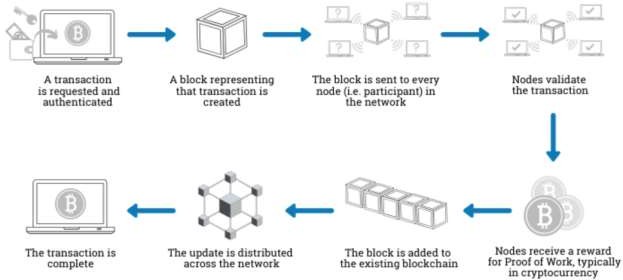


Figure 23 Diagram showing block chain (Gupta, 2022)

## Relationship between DS and Blockchain

Blockchain and distributed systems are connected in that blockchain is a form of distributed system. A distributed system is made up of a number of independent computers that cooperate to accomplish a particular task. A decentralized, impenetrable ledger of transactions may be created using the distributed ledger technology known as blockchain. Data and computation are divided among several network nodes in a distributed system, which enhances scalability, fault-tolerance, and resilience. Similar to this, the ledger in Blockchain is duplicated over several network nodes, making it impervious to manipulation and guaranteeing that the ledger stays accurate and up-to- date. A particular kind of distributed system called a blockchain employs cryptographic methods to guarantee the reliability and veracity of transactions. Blockchain’s decentralized structure also makes it immune to assaults and offers a high level of transparency and trust. Therefore, Blockchain and distributed systems are related in that Blockchain is a particular class of distributed system that has distinctive traits and advantages.

# Conclusion

To overcome the limitations of CKF's current Product Delivery System (PDS) and enhance the customer experience, a new PDS has been proposed. The new PDS must ensure secure login, easy management of personal details, order placement, delivery tracking, secure payment, and secure logout. The proposed solution will utilize RMI as a distributed computing technology, Java programming language, and a suitable database or file system. The system should be fault-tolerant and meet quality standards like usability, maintainability, and heterogeneity. Cloud computing and virtualization could be leveraged for future improvements. To ensure the efficacy and functionality of the new system, thorough testing should be conducted.

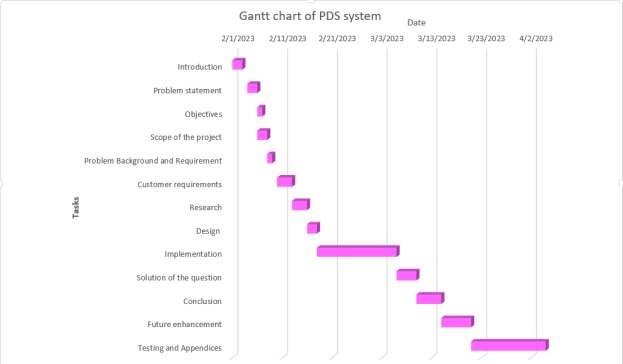
# Future enhancement

One future enhancement of the CKF e-commerce platform's new Product Delivery System (PDS) could be the integration of cloud computing and virtualization technologies. This would enable the system to scale more efficiently, handle increased traffic and orders, and provide more flexibility and agility in managing resources. Additionally, it could improve the system's fault tolerance, disaster recovery, and security capabilities. Another potential enhancement could be the use of machine learning algorithms to optimize the delivery process. By analyzing delivery data, such as delivery times, traffic patterns, and driver availability, the system could make more informed decisions about delivery routes and schedules. This could lead to faster delivery times and more efficient use of resources, ultimately improving the customer experience.

Overall, these enhancements could help to further improve the effectiveness and efficiency of CKF's e-commerce platform's new PDS, and ensure that it continues to meet the evolving needs and expectations of customers.

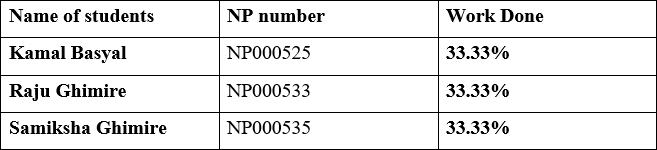
# Appendices and Workload Matrix

## Gantt chart



**Workload matrix**

Figure 24 Gantt chart of PDS system



# References

Ahamed, A. (2020). *Remote method invocation.* Retrieved from https://aashikahamed.medium.com/remote-method-invocation-56e669fc3cfe

Alan Faisandier, G. R. (2020, August 22). *sebokwiki*. Retrieved january 19, 2121, from [www.sebokwiki.org:](http://www.sebokwiki.org/) https://[www.sebokwiki.org/wiki/System\_Requirements](http://www.sebokwiki.org/wiki/System_Requirements)

Bogan, K. (2020). *blogspot*. Retrieved January 12, 2021, from [http://key-to-](http://key-to-/) programming.blogspot.com/2015/09/what-is-oop.html

Gupta, R. (2022). *Goals of distributed system: Distributed Systems.* Blogger. Retrieved from https://pywix.blogspot.com/2020/04/goals-of-distributed-system.html

Hamilton, T. (2021, Oct 8). *Software testing*. Retrieved from Guru99.com: https:/[/www.guru99.com/software](http://www.guru99.com/software-testing-introduction-importance.html)-[testing-introduction-importance.html](http://www.guru99.com/software-testing-introduction-importance.html)

Joyce, F. (2019). *Java Programming.* Cengage Learning.

McEwen, S. (2004, April 16). *ibm*. Retrieved January 14, 2121, from [www.ibm.com:](http://www.ibm.com/) https:/[/www.ibm.com/developerworks/r](http://www.ibm.com/developerworks/rational/library/4166.html)a[tional/library/4166.html](http://www.ibm.com/developerworks/rational/library/4166.html)

tutorialspoint. (2020). *tutorialspoint*. Retrieved January 13, 2021, from https:/[/www.tutorialspoint.com/Wha](http://www.tutorialspoint.com/What-is-object-oriented-programming-)t[-is-object-oriented-programming-](http://www.tutorialspoint.com/What-is-object-oriented-programming-) OOP#:~:text=Object- oriented%20programming%20%28OOP%29%20is%20a%20programming%20paradigm

%20based,certain%20properties%20such%20as%20height%2C%20gender%2C%20age

%2C%20etc.

Warlina, L. (2018, September ). *researchgate*. Retrieved january 19, 2121, from [www.researchgate.net:](http://www.researchgate.net/) https:/[/www.rese](http://www.researchgate.net/publication/327897483_Designing_Web-)a[rchgate.net/publication/327897483\_Designing\_Web-](http://www.researchgate.net/publication/327897483_Designing_Web-) based\_Food\_Ordering\_Information\_System\_in\_Restaurant