Collabits Journal
Vol. 1 No. 3 September 2024 : 152-163
E-ISSN : 3046-6709, P-ISSN : 3062-8601
https://publikasi.mercubuana.ac.id/index.php/collabits

Design of Web-Based Customer Menu Ordering Information System Case Study: Coffee Luck Coffee Shop

Dudun Abdurohman[1*], Eggi Firman Saputra[2], Harris Munahar[3]

[1,2,3] Information Systems Study Program, Universitas Raharja, Indonesia

*Coressponden Author: dudun@raharja.info

Abstract - Technological developments make it easier for users to carry out all tasks such as food and beverage ordering process. One of the technologies used is a computer. Computer is one of the machines in charge to help human work so that the job can be done quickly. Kedai COFFEE LUCK is one of the places engaged in the culinary that is located in Jl. Ks. Tubun No. 86, RT. 004/RW. 003, Koang Jaya, Pasar Baru, Karawaci, Tangerang City, Banten 15112. Currently the food and beverage ordering process at COFFEE LUCK Store still uses paper for the recording of its order transactions. Conventional system like this is less effective, which raises the problem that the food ordering paper often disappear because the paper ordering only stored in the cashier table, took a long time to build a food ordering report because it must recap the booking paper in one by one, there is still a difference in the amount of payment with the amount of food ordering due to calculation errors. Based on the problems that occur at this time, in need of food and beverage ordering system that can help the cashier in managing the food and beverage order data. This research uses Object Oriented Analysis (OOA) analysis methods and system design using Unified Modelling Language (UML) tools. Collection through observation stages, interviews with stakeholders and library studies. This food and beverage ordering system is made using the PHP programming language and Mysql database. This research resulted in a Web-based customer menu ordering system with the aim to increase time efficiency in managing food and beverage order data and to reduce the difference in payment and booking amounts.

Keywords :
Ordering;
Shops;
Food;
Object Oriented Analysis (OOA);
PHP;

Article History:
Received: 07-07-2024
Revised: 25-08-2024
Accepted: 29-09-2024

Article DOI : 10.22441/collabits.v1i3.30262

1. INTRODUCTION

The rapid development of technology has permeated various fields, including education, banking, industry, and culinary businesses. The competition within the culinary business sector continues to intensify each year, prompting companies to enhance their customer engagement strategies. Technological advancements have made it easier for users to perform tasks, such as ordering food and beverages. One such technology is the computer, a machine designed to assist human tasks, allowing for quicker completion of work.

COFFEE LUCK is a culinary business located at Jl. Ks. Tubun No.86, RT.004/RW.003, Koang Jaya, Pasar Baru, Karawaci, Tangerang City, Banten 15112. Currently, COFFEE LUCK faces challenges in its order management process, as it still uses paper-based methods to record food and beverage orders. This manual system is inefficient and presents several problems, such as misplaced order slips that are typically stored in the cashier's drawer, a lengthy report generation process due to the manual compilation of order slips, and discrepancies between payment amounts and order totals caused by calculation errors. Given these issues, there is a need for a food and beverage ordering system that can assist cashiers in managing order data more effectively. Therefore, this research focuses on developing a web-based information system for customer menu orders at COFFEE LUCK. The proposed system aims to streamline the order management process, improving overall efficiency and reducing the likelihood of errors. This study employs the Object-Oriented Analysis (OOA) method and utilizes Unified Modeling Language (UML) tools for system design. Data collection was conducted through observation, interviews with stakeholders, and literature reviews. The goal of this research is to design an information system that addresses the identified issues and meets the needs of COFFEE LUCK.

Based on the background described, the problems identified include the current use of paper-based methods for recording customer orders, which is inefficient, and the manual report generation process, which involves compiling order slips one by one, making it time-consuming. These reports are essential for tracking the revenue earned by the shop.From the identified problems, the main research question is formulated as follows: How can an information system be developed to facilitate the management of order data by administrators, cashiers, and shop owners, making it more effective?

The scope of this research is defined as follows: the study focuses solely on developing a food and beverage

ordering information system for COFFEE LUCK. The programming language used is PHP, with MySQL as the database. System design is conducted using Unified Modeling Language (UML) tools. The information system developed includes features such as menu data for food and beverages, employee data management, customer data management, order transaction processing, payment receipts, and order report generation. This research aims to create an efficient and reliable system to support COFFEE LUCK in managing its customer orders, thereby enhancing operational effectiveness and reducing errors in the order management process.

2. CONCEPTUAL STAGE

2.1 General Theory

Is the main theories needed by the author in making this application and also as a foundation for making applications. From making this application, the author chose to use several books as the basis for making the application.

2.1.1 Understanding Information Systems

According to Tata Sutabri (2012:13-38), an information system is a system within an organization that integrates the needs of daily transaction processing with managerial operations and strategic activities to provide necessary reports for specific external parties. Yakub (2012) defines an information system as an organized combination of people, hardware, software, communication networks, and resources that collect, transform, and disseminate information within an organization. From these definitions, the author concludes that an information system is a series of procedures involving a combination of people, hardware, software, communication networks, and data sources that collect, process, store, and distribute information for planning, decision-making, and control purposes.

2.1.2 Food Ordering Process

According to Simorangkir (2010), as cited in the journal by Rachmad Aulia and Ahmad Zakir (2018), the processing of food and beverage orders is a critical aspect of restaurant business operations. The ordering process can be carried out either directly within the restaurant or indirectly from the customer's location. Direct order recording at the restaurant typically involves manual tools such as pens and paper or computers, as seen in several fast-food restaurants. Meanwhile, indirect ordering outside the restaurant usually utilizes telephones, requiring customers to manually state their orders.

2.2 Specialized Theory

2.2.1 Unified Modelling Language (UML)

According to Adi Nugroho (2010:6-7), UML (Unified Modeling Language) is a modeling language specifically designed for object-oriented systems or software. UML serves as a standard way to visualize the design of a system, providing a set of graphical notations to create abstract models of a system's structure and behavior. The primary purpose of modeling is to simplify complex systems and problems, making them more comprehensible and easier to analyze. By employing UML, developers and stakeholders can effectively communicate system requirements, design decisions, and operational workflows, ultimately improving the planning, implementation, and maintenance of software systems. This approach not only enhances understanding but also supports better decision-making throughout the software development lifecycle.

2.2.2 PHP (Framework Codeigniter)

A framework is a collection of instructions organized within classes and functions, each serving specific purposes to assist developers in their implementation without the need to repeatedly write the same program syntax. This approach helps save time and prevents redundant code, resulting in clean and structured source code. One popular example is CodeIgniter, an open-source PHP framework that employs the Model-View-Controller (MVC) method. CodeIgniter is free to use and, like other frameworks, is designed to simplify the work of developers by enabling them to build web-based applications without starting from scratch.

The MVC (Model-View-Controller) architecture is a design pattern that separates the main components of an application into three distinct layers:

1. Model: The Model handles operations related to database processing and manipulation, such as retrieving data, inserting new records, and other database-related functions. All instructions for database management are contained within the Model, allowing developers to keep the data handling logic separate from other components.
2. View: The View manages the user interface and the visual elements presented to the user. By separating the user interface logic into the View, web designers can focus on the website's appearance without needing to interact with the business logic contained in the Model and Controller, facilitating easier and more flexible design development.
3. Controller: The Controller acts as a bridge between the Model and the View. It contains the action instructions that connect the data processing in the Model with the display logic in the View, ensuring that the user interacts indirectly with the Model through the Controller. For instance, when an application displays data using the MVC concept, the Controller retrieves data from the Model and then forwards it to the View for display. This separation of concerns makes it easier to manage tasks, as front-end developers can work on the View, focusing solely on design, while back-end developers handle the Controller and Model.

In summary, MVC simplifies application development by dividing responsibilities, allowing for rapid and organized development. Web designers or front-end developers work with the View to craft the application's appearance, while back-end developers manage the underlying logic within the Controller and Model. This division of labor promotes efficient teamwork, streamlined workflows, and faster application development.

2.2.3 MySQL

A database is often defined as a collection of related data. Technically, a database consists of a set of tables or other

objects such as indexes, views, and more. The primary purpose of creating a database is to facilitate data access, allowing data to be added, modified, deleted, or read easily and quickly. Currently, various software tools are available to manage databases, commonly known as Database Management Systems (DBMS). Examples of popular DBMS products include Access, MS SQL Server, and MySQL. MySQL is a database server software that is open-source and cross-platform, meaning it can run on various operating systems. MySQL offers several features, including portability, allowing it to operate stably on multiple systems such as Windows, Linux, FreeBSD, Mac OS X Server, Solaris, and Amiga. As open-source software, MySQL is distributed under the GPL license, making it free to use.

MySQL supports multiple users simultaneously without conflicts, and it exhibits impressive performance, particularly in handling simple queries, allowing it to process more SQL commands per time unit. It supports a wide range of data types, including signed/unsigned integers, floats, doubles, chars, text, dates, timestamps, and more. MySQL provides a comprehensive set of operators and functions to support SQL commands like SELECT and WHERE. In terms of security, MySQL features multiple security layers, including subnet mask levels, hostnames, and detailed user access permissions with encrypted passwords. MySQL also boasts scalability, capable of managing large-scale databases with over 50 million records, 60,000 tables, and up to 5 billion rows, and it can handle up to 32 indexes per table.

MySQL supports connectivity with clients using TCP/IP, UNIX sockets (UNIX), or named pipes (NT), and it can detect error messages in over 20 languages, though Indonesian is not yet included. Its interface supports integration with various applications and programming languages through API functions (Application Programming Interface). Additionally, MySQL is equipped with multiple tools for database administration, complete with online guidance for each tool. MySQL's table structure is more flexible in handling ALTER TABLE commands compared to other databases like PostgreSQL or Oracle, making it a robust choice for managing complex and large-scale data environments.

2.2.4 APP Server

AppServ is an application designed to facilitate the installation of several programs, including Apache, PHP, and MySQL, in a short amount of time. Many users have expressed difficulties in manually installing Apache, PHP, and MySQL separately. AppServ simplifies the process, making it easier for users to set up a web server and database environment. AppServ is available in several versions, such as 2.5.9, 2.5.10, and the latest version 2.6.0.

The primary purpose of AppServ is to streamline the creation of databases and web servers. Typically, MySQL is used for database creation, while PHP is employed for scripting. PHP and MySQL are interconnected because PHP scripts appear on web pages and are linked to MySQL databases. When creating a website, HTML scripts are used for display purposes, while PHP scripts handle the processing of HTML. When data is entered into a web page connected to a database, it is processed by PHP and subsequently stored in the database. This integrated approach greatly simplifies the development and management of web applications, providing a unified environment for creating and managing web servers and databases efficiently.

2.2.5 Client Server

According to Galih (2013) as cited in the journal by Deni Utama et al. (2016), a client is defined as any system or process that requests data or services from a server. Conversely, a server is a system or process that provides the requested data or services to the client. For a client to access a server, both must be interconnected within a network. Thus, the client-server architecture represents an application design in which clients and servers communicate with each other to access services over a network.

2.2.6 Waterfall Method

According to Pressman (2015:42), the Waterfall model is a classical, systematic, and sequential approach to software development. Officially known as the "Linear Sequential Model," it is frequently referred to as the "classic life cycle" or the "waterfall method." This model, which is among the generic models in software engineering, was first introduced by Winston Royce around 1970, and although it is often considered outdated, it remains one of the most widely used models in Software Engineering (SE). The Waterfall model adopts a systematic and sequential approach, where each phase must be completed before the next one begins, hence the term "waterfall" which implies a cascading effect from one phase to the next. According to Pressman, the phases of the Waterfall model are as follows:

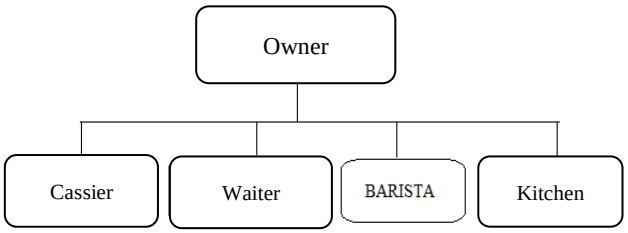
1. Communication (Project Initiation & Requirements Gathering): Prior to engaging in technical work, it is crucial to communicate with the customer to understand and achieve the desired objectives. This phase involves project initiation, such as analyzing issues, gathering necessary data, and helping to define the software's features and functions. Additional data can also be sourced from journals, articles, and the internet.
2. Planning (Estimating, Scheduling, Tracking): The next phase is planning, which involves estimating the technical tasks, identifying potential risks, determining the resources needed, outlining the deliverables, scheduling the tasks, and tracking the progress of the system development.
3. Modeling (Analysis & Design): This stage focuses on the design and modeling of the system architecture, including data structure, software architecture, user interface design, and program algorithms. The goal is to develop a comprehensive understanding of what needs to be accomplished.
4. Construction (Code & Test): The Construction phase involves translating the design into code that is machine-readable. Following coding, the system and code undergo testing to identify and rectify any errors.
5. Deployment (Delivery, Support, Feedback): The Deployment phase encompasses the implementation

of the software to the customer, regular maintenance, software updates, evaluation, and development based on feedback to ensure that the system remains functional and evolves as needed (Pressman, 2015:17).

4. Identification of Problems and System Needs

The full name of the establishment is Coffee Luck and Garage, which translates to a Vespa workshop complemented by a coffee shop. The origins of Coffee Luck and Garage trace back to a Vespa community that sought to create a space where members could gather, leading to the development of a combined coffee shop and workshop. This unique concept distinguishes Coffee Luck and Garage from other establishments; despite its seemingly simple design, it features a distinctive garage or workshop aesthetic, allowing patrons to enjoy coffee in an environment reminiscent of a Vespa garage. The company envisions enhancing the lifestyle of contemporary youth, fostering creativity and independence within an expansive context. It aims to impart a unique character to its surroundings. The mission of the company includes cultivating distinctive characteristics in its clientele and organizing creative initiatives within the automotive and coffee industries.

Figure 1. Organization Structure

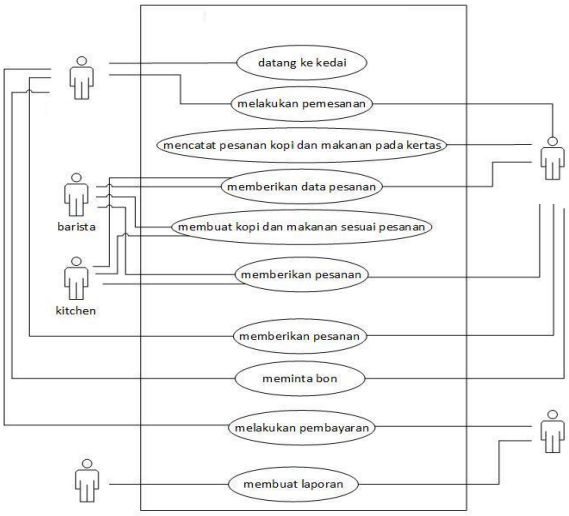


The organizational structure of Coffee Luck delineates the following authorities and responsibilities:

- Owner: The owner holds the highest authority and bears significant responsibilities, including setting the overarching policies, overseeing financial outcomes such as profits and losses, hiring and dismissing employees, and managing the company's assets. The owner is also responsible for effective leadership and guidance, representing the company in agreements, planning, and supervising employee tasks. Additionally, the owner is tasked with developing and implementing general policies and determining dividend amounts.
- Server: The server is responsible for providing excellent customer service, maintaining positive relationships with customers, managing employee needs such as leave requests, and creating employee shift schedules. The server also handles employee attendance.
- Cashier: The cashier manages financial transactions and activities, ensuring that sales targets are met.
- Barista: The barista prepares coffee for customers, fosters good communication with patrons, and strives to retain existing customers.
- Kitchen Staff: The kitchen staff prepares necessary kitchen ingredients and prepares food for customers at Coffee Luck.

3.1 Current System Description

A. Definition of Use Cases for the Running System



Based on Figure 2, the following explanations are provided:

1. Use Case Name: Arriving at the Café
   Actor: Customer
   Description: The customer arrives at the café.
2. Use Case Name: Placing an Order
   Actor: Customer and Server
   Description: The customer places an order with the server, who receives the customer's order.
3. Use Case Name: Recording Orders on Paper
   Actor: Server
   Description: The server records the customer's food and coffee orders on paper.
4. Use Case Name: Providing Order Data
   Actor: Server, Barista, and Kitchen
   Description: The server provides the order data to the barista and kitchen.
5. Use Case Name: Preparing Coffee and Food According to Order
   Actor: Barista and Kitchen
   Description: The barista and kitchen prepare the coffee and food according to the order.
6. Use Case Name: Delivering Orders
   Actor: Barista, Kitchen, and Server
   Description: The barista and kitchen deliver the prepared orders to the server.
7. Use Case Name: Receiving Orders
   Actor: Server and Customer
   Description: The server delivers the order to the customer, who receives the previously ordered items.
8. Use Case Name: Requesting a Receipt
   Actor: Customer and Server
   Description: The customer requests a receipt from the

server.

9. Use Case Name: Making Payment
   Actor: Customer and Cashier
   Description: The customer makes a payment to the cashier.
10. Use Case Name: Generating Report
    Actor: Cashier and Owner
    Description: The cashier generates a report and provides it to the owner, who reviews, checks, and signs the report.

B. Definition of Running System Actors

Table 1. Current System Actor Definition

No.	Actor Name	Description
1.	Customers	People who visit and make purchase transactions for food and drinks sold in the store *coffee luck*.
2.	Waiter	The person on duty serves every customer who visits the store *coffee luck* starting from ordering food and drinks until the customer receives the order.
3.	Barista	The person on duty makes drinks (coffee) ordered by customers.
4.	Kitchen	The person on duty makes the food and prepares the food ingredients that are needed.
5.	Cashier	The person in charge of serving payment transactions and providing proof of payment to customers. Make daily sales reports which will later be reported to the shop owner.

3.2 Current System Analysis

In the current system analysis for the Coffee Luck café, various use cases are defined to outline the interactions between different actors within the café's operations. The system begins with the customer arriving at the café, initiating their experience. Once at the café, the customer places an order with the server, who then records the order on paper. The server subsequently provides the order data to the barista and kitchen, who are responsible for preparing the coffee and food according to the order. After preparation, the barista and kitchen deliver the completed orders to the server, who then delivers the order to the customer. The customer may then request a receipt, which the server provides. Following this, the customer makes the payment to the cashier. Lastly, the cashier generates a report detailing the day's transactions and provides it to the owner, who reviews, checks, and signs off on the report. This structured sequence of interactions ensures smooth operation within the café, with clearly defined responsibilities and processes for each role involved.

3.3 Analysis Result

A. Problems Encountered

Based on the research analysis at SMK Bina Amma'mur, the current helpdesk service system has several shortcomings. Currently, the system involves reporting computer damage by directly meeting with the IT helpdesk to inform them of the issue. The Head of IT then records the problem in a repair request form, after which a technician team is dispatched to address the reported problem. Upon completion, the repair is reported back to the Head of IT, who updates the repair request status. However, the problem-handling process at SMK Bina Amma'mur remains inefficient due to the manual nature of damage recording, which negatively impacts performance. Issues such as redundant data entry, lengthy data retrieval times from repair request logs, frequent errors in report generation, and discrepancies between the reported data and the actual computer repair process are common, highlighting the need for an improved system.

B. Identification of Problem Solving

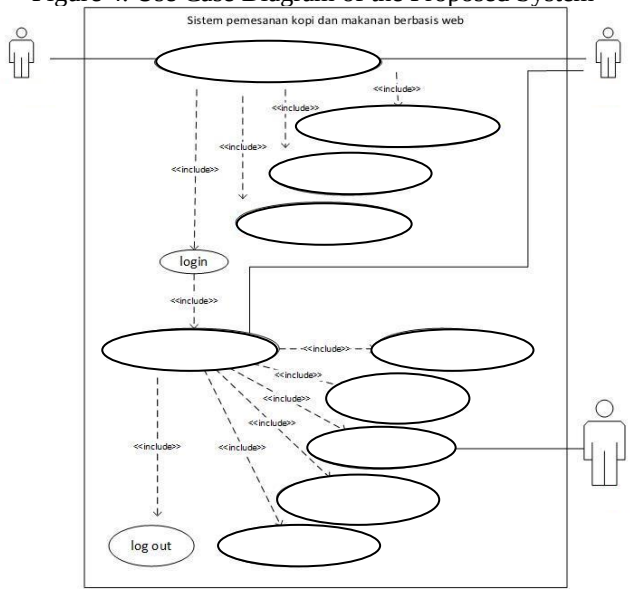
Based on observations at SMK Bina Amma'mur, it was identified that the Helpdesk management system requires further development to prevent delays in submitting damage complaint information and to facilitate the IT department in managing computer repair data. Based on the analysis of the issues, the researcher chose the alternative of enhancing the existing application due to its various advantages, such as simplifying the process for teachers and staff to report damage complaints, thereby minimizing errors in information delivery, and aiding the IT department in managing damage reports, ensuring that reports are completed in a timely manner. This development represents the final stage of the elicitation process, serving as a reference for further enhancement of the IT Helpdesk system, with the goal of streamlining data processing within the Helpdesk system.

The final draft elicitation identifies several functional requirements of the system, including login menus for staff, head technicians, and technicians, login verification, dashboards, input and management of damage data, and damage report features. The non-functional requirements encompass displaying the school logo, utilizing MySQL as the database, employing PHP as the programming language, using Apache as the web server, restricting access rights to specific users, and ensuring data security. This system development is expected to improve the efficiency and effectiveness of computer repair data management at SMK Bina Am Ma'mur.

4. System Implementation and Process

4.1 Use Case

Figure 4. Use Case Diagram of the Proposed System



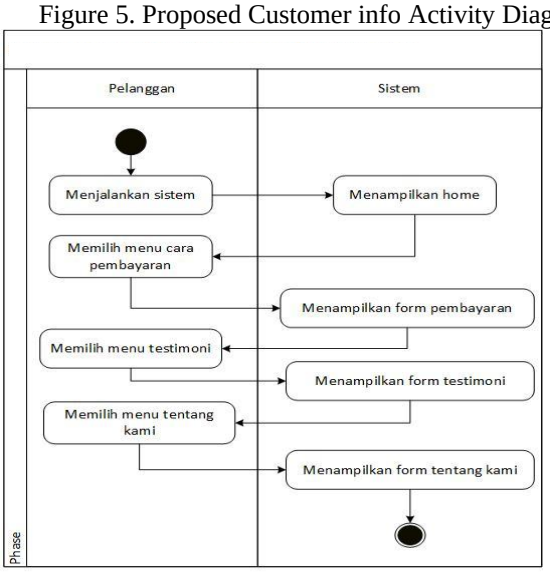
Based on figure 4. *Use Case Diagram* The proposed system contains the following explanation:

1. Definition *Use Case* proposal system
   a) No *Use Case* : Displays the menu *home*
      Actor: Customers and Cashiers
      Information : Customers and cashiers can display the menu *home*
   b) Nama Use Case: Displays the payment method
      Actor: Customers and Cashiers
      Information : Customers and cashiers can display a payment methods menu.
   c) Nama Use Case: Displaying Testimonials
      Actor: Customers and Cashiers
      Information: Customers and cashiers can display a testimonial menu
   d) Nama Use Case : Showing About Us.
      Actor : Customers and Cashiers
      Information: Customers and Cashiers can display About Us.
   e) Nama Use Case: *Login*
      Actor: Admin
      Information: Admin must input *username* and *password* first before using the program.
   f) Nama Use Case: Product Management
      Actor: Cashier
      Information: The cashier can input product data
   g) Nama Use Case : Management *Brand*
      Actor: Cashier
      Information : The cashier can input product data
   h) Nama Use Case : Transaction Management
      Actor: Cashier
      Information: The cashier can input order transaction data
   i) Nama Use Case: Customer Management
      Actor: Cashier
      Information : The cashier can input customer data
   j) Nama Use Case : Testimonial Management
      Actor : Cashier
      Information: The cashier can input testimonial data
   k) Nama Use Case : *Logout*
      Actor: Cashier
      Information: The cashier did *logout* when it is *close order*

4.2 Activity Diagram

1. Customer Activity Diagram

Figure 5. Proposed Customer info Activity Diagram

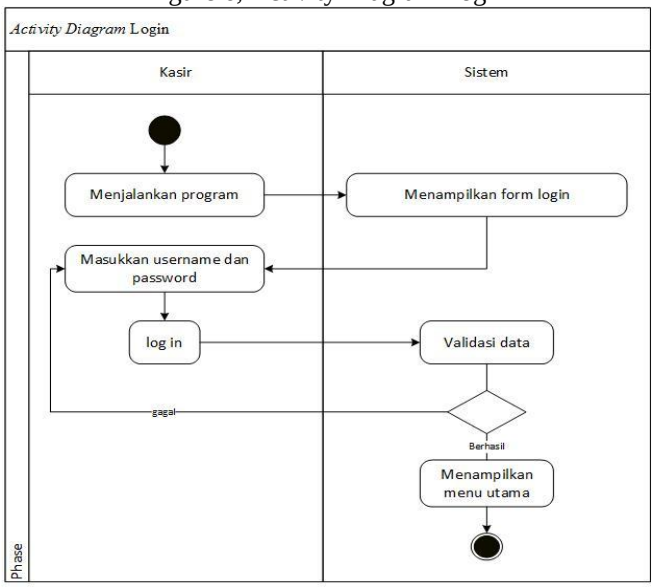


Based on the proposed picture 3.6 there are:

1. 2 Vertical Swimlane which is used as actor and system.
2. 1 Initial Node as an initialized object.
3. 8 Activity from a system that reflects the execution of an action including:
   a. Run the program.
   b. Displays menus home.
   c. Select the payment method menu.
   d. Displaying form payment method.
   e. Select the testimonials menu.
   f. Displaying form testimony.
   g. Select the about us menu.
   h. Displaying form about Us.
4. Final State as a fin

2. Activity Diagram Login

Figure 6, Activity Diagram Login

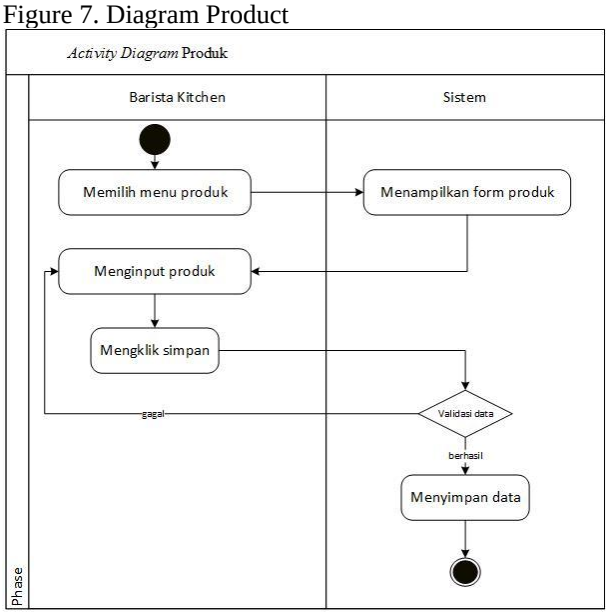


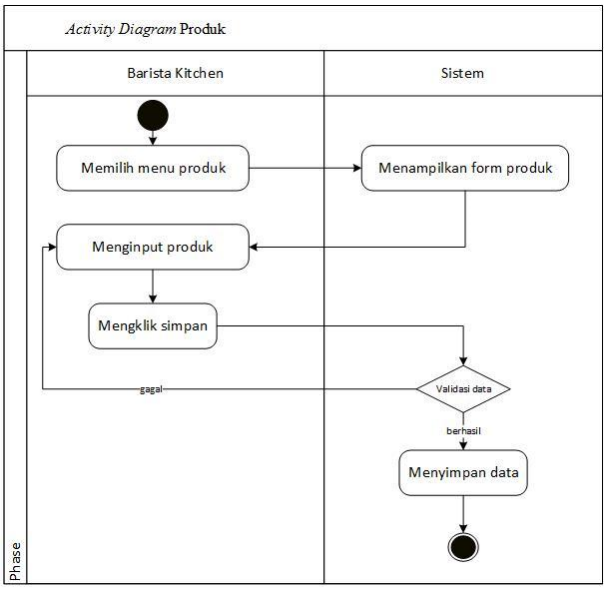
Based on the proposed Figurer 6 there are:

1. 2 *Vertical Swimlane* which is used as *actor* and *system.*
2. 1 *Initial Node* as an initialized object.
3. 2 *Activity* from a system that reflects the execution of an action including:
   a. Run the program
   b. Displaying *form login*
   c. Enter *username* and *password*
   d. *Click login*
   e. Data validation
   f. Displays the main menu
4. 1 *Final State* as a final object

3. Activity Diagram Product Management

Figure 7. Diagram Product



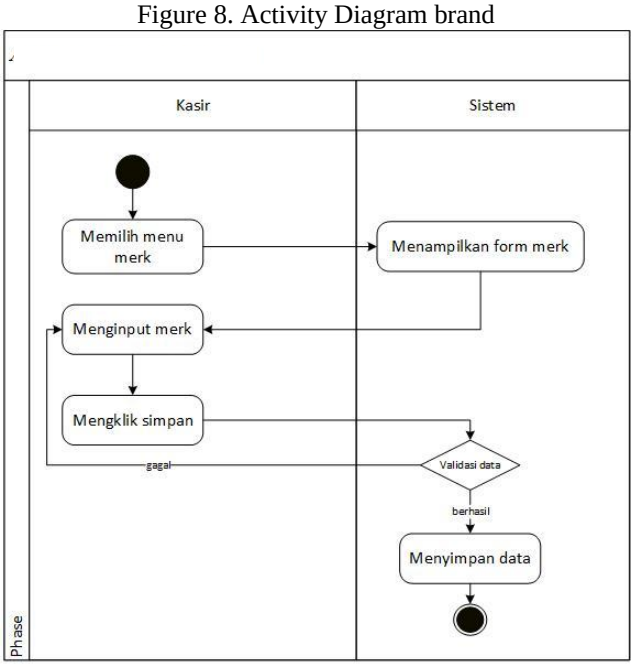


Based on the proposed Figure 7 there are:

1. 2 Vertical Swimlane which is used as an actor and system.
2. 1 Initial Node as an initialized object.
3. 6 Activity of a system that reflects the execution of an action including:
   a. Select the product form menu
   b. Displays the product form
   c. Input data

4. Activity Diagram Management Brand

Figure 8. Activity Diagram brand



Based on Figure 8, the following explanations are provided:

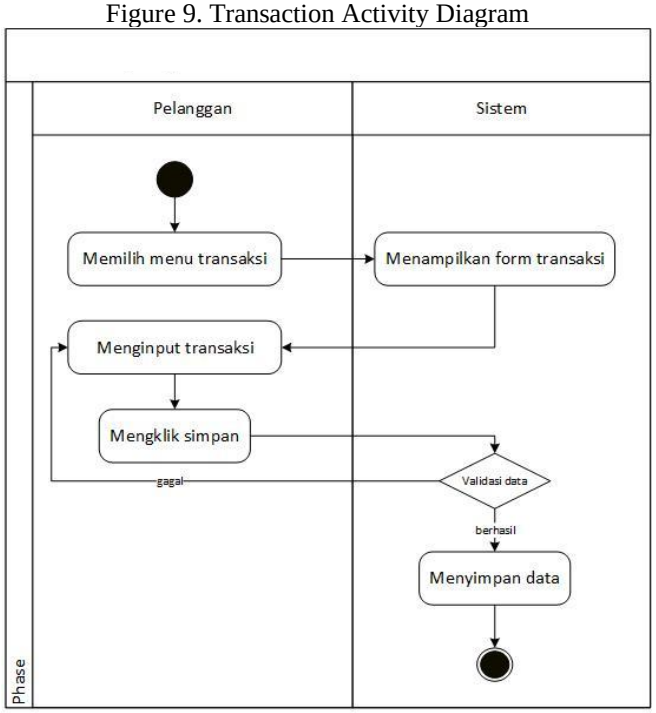
1. *2 Vertical Swimlane* which is used as *actor* and *system.*
2. 1 *Initial Node* as an initialized object.
3. 6 *Activity* from a system that reflects the execution of an action including:

a. Select the manage brand menu
b. Displays brand form
c. Input data
d. Click save
e. Data validation
f. Data is saved

4. 1 *Final State* as a final object.

5. Activity Diagram Transaction Management

Figure 9. Transaction Activity Diagram



Based on Figure 9, the following explanations are provided:

1. *2 Vertical Swimlane* which is used as *actor* and *system.*
2. 1 *Initial Node* as an initialized object.
3. 5 *Activity* from a system that reflects the execution of an action including:
    a. Select the transaction menu
    b. Displays the transaction form
    c. *Input* transaction
    d. Click save
    e. Data validation
    f. Data is saved.
4. 1 *Final State* as a final object.

6. Activity Diagram Customer Management

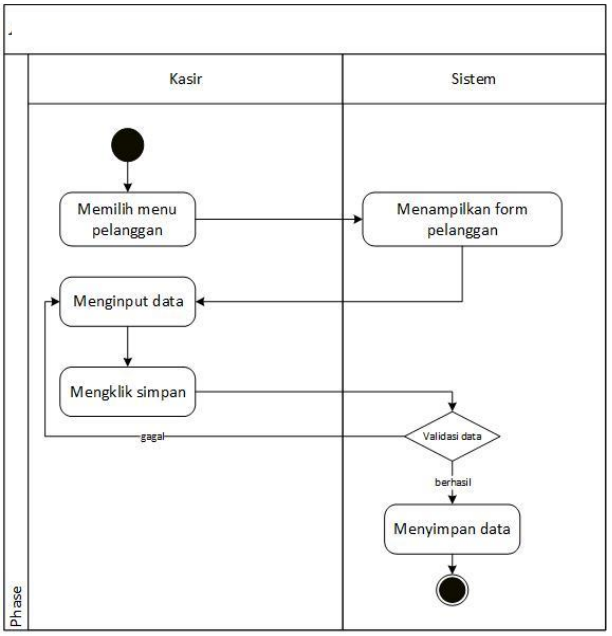
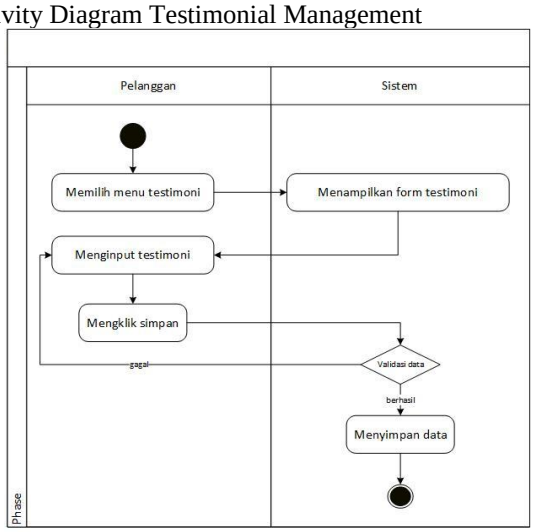


Figure 10. Customer Activity Diagram

Based on Figure 10, the following explanations are provided:

1. *2 Vertical Swimlane* which is used as *actor* and *system.*
2. 1 *Initial Node* as an initialized object.
3. 5 *Activity* from a system that reflects the execution of an action including:
    a. Select the customer menu.
    b. Displays the customer form.
    c. Input data.
    d. Click save.
    e. Data validation.
    f. Data is saved.
    g. 1 *Final State* as a final object.
4. 5 *Activity* of a system that reflects the execution of an action.

7. Activity Diagram Testimonial Management

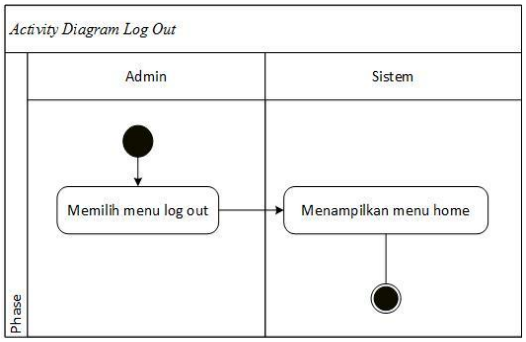


Based on Figure 11, the following explanations are provided:

1. *Vertical Swimlane* which is used as *actor* and *system.*
2. 1 *Initial Node* as an initialized object.
3. 5 *Activity* of a system that reflects the execution of an action including:
    a. Select the testimonials menu.
    b. Displays a testimonial form.
    c. Input data.
    d. Click save.
    e. Data validation.
    f. Data is saved.
4. 1 *Final State* as a final object.

8. Activity Diagram Log Out

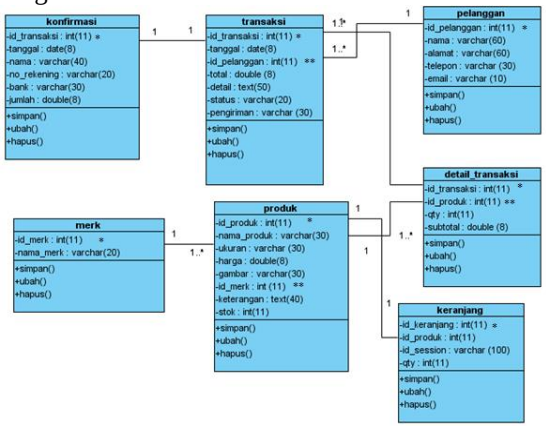
Figure 11. Logout Activity Diagram



Based on Figure 11, the following explanations are provided:

1. *2 Vertical Swimlane* which is used as *actor* and *system.*
2. 1 *Initial Node* as an initialized object.
3. 2 *Activity* of a system that reflects the execution of an action including:
    a. Select menu menu *Log out*
    b. Displaying *form* menu *home*
4. 1 *Final Node* as a final object.

9. Class Diagram



Based on figure 3.21 *Class Diagram* proposed there are:

1. Has 7 *classes* namely as a table in which there are attributes.
2. Has 6 *associations* namely as a relationship between attribute tables in *class* with the same operation.

4.3 Proposed System Configuration

Hardware Specifications:

- Processor: Intel® Core™ i7-5005U Processor (2.00 GHz, 3M cache)
- RAM: 12.00 GB
- Hard Disk: 1 TB
- Mouse: Logitech M100r
- Monitor: (Not specified in the provided text)

Software Specifications:

- Operating System: Windows 7 (64-bit)
- Browsers: Google Chrome and Mozilla Firefox
- XAMPP Version: 5.6
- MariaDB: Version from October 19, 2015, starting with XAMPP versions 5.5.30 and 5.6.14

User Roles:

1. Employee: Individual who submits repair requests.
2. Technician: Person responsible for inspecting and repairing issues.
3. Head Technician: Person accountable for overseeing maintenance requests.

4.4 Proposed System View

a. Main Menu Page Display

Figure 13, Main Menu Display

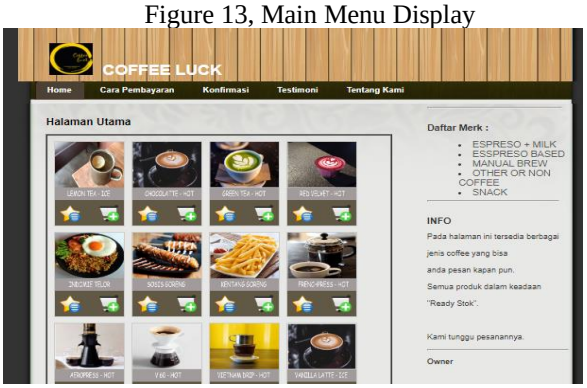


Figure 13 shows the main menu which is the initial menu of the menu ordering system after the system is run.

b. Payment Method Menu Display

Figure 14, Payment Method Menu Display

Figure 14 informs how to pay after placing an order

c. Menu Display About Us

Figure 15, About Us Menu Display

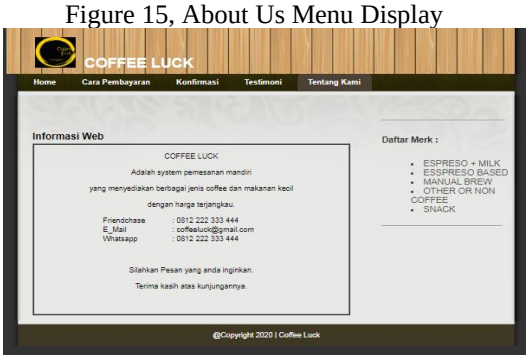


Figure 15 is a display of the about us menu which contains store profile information *coffee Luck.*

d. Shopping Cart Form Menu Display

Figure 16, Shopping Cart Form Display

Figure 16 is a shopping cart form that serves to accommodate products purchased by customers.

e. Login Form Menu Display

Figure 17, View Form Login

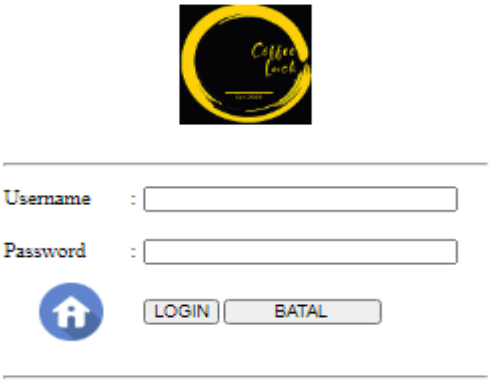
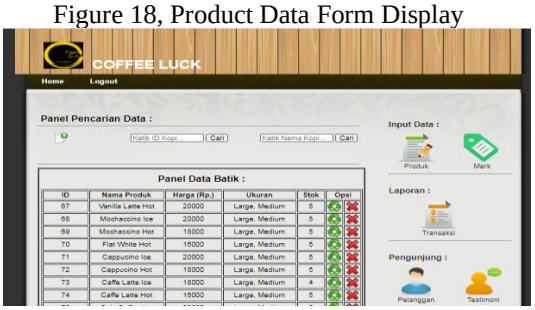


Figure 17 shows *form login* which functions to maintain the security of ordering data from people who are not registered to use the system.

f. Product Data Form Menu Display

Figure 18, Product Data Form Display



Picture 18 product data form that works to add, change, delete product data.

g. Brand Data Form Menu Display

Figure 19, Brand Data Form Display

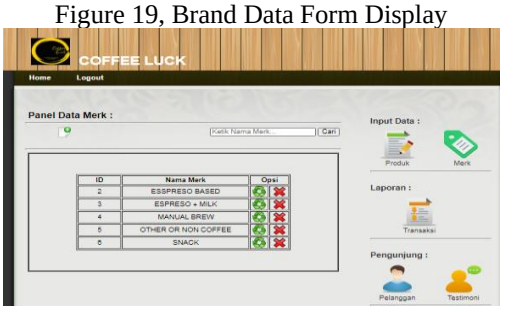


Figure 19 The brand data form has the function to add, change, and delete brand data.

h. **Transaction Data Form Menu Display**

Figure 20. Transaction Data Form Display

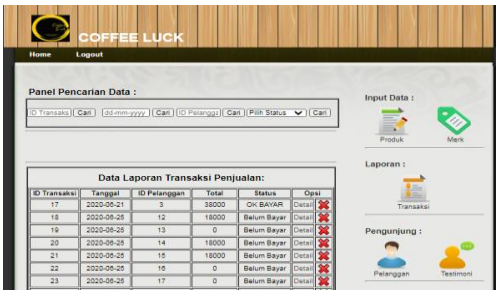


Figure 20 is a transaction data form which functions to record menu order transactions.

4.4 Blackbox Testing

The black box testing for the proposed system includes evaluating the login functionality for cashiers and the process of adding products. For the cashier login module, testing scenarios involve entering various combinations of username and password to ensure system responses are as expected. Specifically, if neither the username nor password is entered and the login button is pressed, the system should deny access and display a failure message, which has been validated as correct. Conversely, when valid credentials are provided, the system should successfully navigate to the main page or dashboard, which was also confirmed as accurate. In the context of product management, testing focuses on the addition of new products. The first scenario tests the system's response when the product data fields are left blank but the "add data" button is pressed. The expected result is that the system will deny the addition of the product and present a failure message, which was validated. The second scenario involves entering complete product data. The system's expected behavior is to display a confirmation that the product has been added successfully, which was also confirmed. These tests ensure that the system handles both valid and invalid inputs appropriately, contributing to overall system reliability and user experience.

4.5 System Specification

A. Hardware Specifications (Hardware)
The hardware required by an application or system is a *unit* personal computer. The proposed hardware is created based on current system needs and anticipated future needs. The requirements specifications are as follows:
1. Hardware Server
   a. Processor : Core I5
   b. RAM : 64 GB.
   c. Hard drive : 500GB SSD.
2. Hardware Client
   a. Processor : Intel Core i3-4005U, 1.7Ghz.
   b. RAM : 32 GB.
   c. Hard drive : 500 GB.
   d. Monitor : 14,5 Inches.

B. Software Specifications (Software)
Software needed in the process of creating an information system:
   A. Operating system *Windows*.
   B. *XAMPP Control Panel v3.2.1* (*Apache* 80,443, *PHP* 5.6.23, *MySQL* 1.8.3).
   C. *Browser* (*Google Chrome*, *Mozilla Firefox*, and others).

C. User Specifications
User (*Brainware*) is the person who uses and maintains the system to be implemented, therefore resources are needed that can fulfill these needs. In this case, users are divided into two, namely:
   a. Cashier (Store Employee), namely the person in charge of managing sales data
   b. Customers, namely people who buy food and coffee from the shop
   c. Admin, namely the person who manages food ordering data

3. CONCLUSION

Based on the discussion previously described, the following conclusions are obtained:
1. The design of the customer menu ordering information system can assist the cashier in managing food and beverage orders so as to provide the value of menu order effectiveness for customers.
2. The designed food and beverage ordering information system can assist cashiers in recapitulating order reports.
3. With the food and beverage ordering information system, the cashier can ensure the amount of payment in accordance with the amount of the order.

4. SUGGESTION

Based on the analysis provided in the preceding chapters and the observations made concerning the research problem statements, the following conclusions can be drawn:
1. The current IT helpdesk service system at SMK Bina Amma'mur operates manually, where employees must visit the IT department in person to report issues with their computers or laptops due to the absence of an automated helpdesk system. This manual process is inefficient as it relies on paper-based record-keeping, which is prone to errors and delays. Consequently, the existing helpdesk service system fails to operate effectively and struggles to address issues in a timely manner.
2. There is a pressing need for an IT helpdesk service system that allows for easier submission of repair complaints, facilitating quicker resolution of reported issues. An automated helpdesk system would significantly enhance the efficiency of complaint management, streamline communication, and improve the overall effectiveness of IT support services at the institution

5. ACKNOWLEDGEMENT

The author wishes to express sincere gratitude to everyone who contributed to the success of this research. Special appreciation is extended to the Cosmos development team for their exceptional framework, which has greatly facilitated the development of the operating system. The selection of Visual Studio 2022 as our primary development environment has proven to be an excellent choice, with its robust features significantly enhancing our project's productivity.

We are also deeply thankful to VMware for their outstanding virtualization platform, which has provided a reliable and efficient environment for testing and evaluation. Their support has been crucial in advancing our research efforts.Our heartfelt thanks go to all colleagues, mentors, and contributors who provided valuable feedback and insights throughout the development process. Your input has been instrumental in refining and completing our operating system. We deeply appreciate your unwavering support and contributions to this project.

REFERENCE

[1] Acmad, Amirullah Kusaini Alkakim, dkk. 2018. Pembangunan Sistem Pemesanan Koki dan Makanan Berbasis Web. Malang: Universitas Brawijaya. Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, Vol.2, No.9, E-ISSN: 2548-964x. Sumber: https://www.neliti.com/publications/246722/none. Diakses: 28 April 2020.

[2] Al-Bahra. 2013. Analisis dan Desain Sistem Informasi. Yogyakarta : Graha Ilmu

[3] Aulia, Rachmat dan Ahmad Zakir. 2018. Pemanfaatan Teknologi Mobile untuk Pemesanan dan Pemrosesan Menu di Restaurant. Medan: Universitas Prima Indonesia (UNPRI). Jurnal Penelitian Teknik Informatika. Sumber: http://jurnal.unprimdn.ac.id/index.php/JUTIKOMP /article/download/328/211. Diakses: 30 April 2020.

[4] Darmawan, Deni. 2012. Pendidikan Teknologi Informasi dan Komunikasi. Bandung: PT Remaja Rosdakarya.

[5] Diana, Anastasia, dan Lilis Setiawati. 2011. Sistem Informasi Akuntansi. Yogyakarta: CV Andi Offset.

[6] Sucipto, Toto dkk.2006.Akuntansi 1A (Siklus Akuntansi Perusahaan Jasa) Untuk Tingkat 1 SMK. Jakarta: Yudhistira.

[7] Hartono,Bambang.2013. Sistem Informasi Manajemen B Jakarta: PT . Rineka Cipta.

[8] Inayati, Immah , dkk. 2015. Aplikasi Pemesanan Makanan Berbasis web (Studi Kasus RM Lesehan Berkah Ilahi Gresik). Surabaya: Universitas Narotama Surabaya. Sumber:https://jurnal.narotama.ac.id/index.php/narodroid/article/download/71/61/. Diakses: 30 April 2020.

[9] Krismiaji. 2010. Sistem Informasi Akuntasi. Yogyakarta : UPP AMP YKPN.

[10] Nugroho, Adi. 2010. Analisa dan Perancangan Sistem Informasi dengan Metodologi Berorientasi Objek. Bandung: Informatika.

[11] Rahman, Arif. 2013. Panduan Akuntansi dan Perpajakan. Jakarta: TransMedia.

[12] Rudianto. 2012. Konsep dan Teknik Penyusunan Laporan Keuangan. Jakarta

[13] Erlangga.Sutabri, Tata. 2012. Konsep Sistem Informasi.Yogyakarta: CV. Andi Offset.

[14] Utama, Deni. 2016. Aplikasi Pemesanan Makanan dan Minuman Restaurant Berbasis Client Server Dengan Platform Android dan PHP Menggunakan Metode Centralized DBMS Architecture (Studi Kasus: Café Cempakoe Kota Bengkulu). Bengkulu: Univeritas

[15] Bengkulu. Jurnal Rekursif Vol.4 No.3 ISSN:2303-0755. Sumber: https://ejournal.unib.ac.id/index.php/rekursif/article /download/852/1215. Diakses: 28 April 2020.

[16] Wahyono, Teguh. 2003. Sistem Informasi. Yogyakarta: Graha Ilmu.