PAPERAZZI: A Coin-Operated Printing Machine Using Raspberry Pi 3B

A Capstone Project
Submitted to the Faculty of
the Department of Computer Studies
Cavite State University
Imus, Campus

In partial fulfillment
of the requirements for the degree of
Bachelor of Science in Information Technology

Joy Mica M. Nillos
Kimberly Louise M. Ravalo
Justine Jade A. Sulo
February 2025

BIOGRAPHICAL DATA

Joy Mica M. Nillos was born on May 30, 2002 in Dasmariñas Cavite, she is the 4th child of Mr. Jerry Nillos and Mrs. Marisa Nillos and currently lives in Brgy. Salawag Dasmariñas Cavite.

She finished her primary education at Salawag Elementary School in 2015. Her secondary education was at City of Bacoor National High School - Georgetown Annex in 2019, upper secondary education at Asian Institute of Computer Studies - Bacoor in 2021 and in tertiary education in Cavite State University - Imus present.

She is Joy Mica expected to get her degree in Bachelor of Science in Information Technology in 2025.

BIOGRAPHICAL DATA

Kimberly Louise M. Ravalo was born on February 1, 2001, in Dasmariñas City. She is the second child of Louis E. Ravalo Jr., a house husband and Geraldine Manzano, a caregiver. She currently lives in Blk 25 Lot 17 Phase 2 Winward Hills Subdivision Burol 1, Dasmariñas, Cavite with their father and siblings.

At the age of 3 years old, she started his pre-kindergarten education. Furthermore, she finished her primary education at Mary Mediatrix of All Grace School. Her secondary education was at Dasmariñas Integrated Highschool – SCC.

Kimberly Louise is a student under the program Bachelor of Science in Information Technology in Cavite State University – Imus Campus.

BIOGRAPHICAL DATA

Justine Jade A. Sulo was born on April 23, 2003, in Caramoan, Catanduanes. She is the second child of Dolor A. Sulo and Joey R. Sulo. She currently resides at Blk 6 Lot 30 Silang St. Windsor Homes, Dasmariñas, Cavite.

She began her early education at Sta. Cristina Elementary School (SCES), where she completed her primary studies. For her secondary education, she attended Dasmariñas East Integrated School (DEIS). Continuing her academic journey, she pursued her senior high school education at the Asian Institute if Science and Technology (AISAT).

Currently, Justine Jade is a college student at Cavite State University – Imus Campus (CvSU), taking Bachelor of Scince in Information Technology.

ACKNOWLEDGEMENT

The researchers of Paperrazzi: A Coin-Operated Printing Machine Using Raspberry Pi 3B express their heartfelt gratitude to everyone who contributed to the success of this project through their guidance, support, and encouragement.

Special thanks go to Mr. Carlo Malabanan, the group’s capstone adviser, for his expertise and constructive feedback, which greatly shaped the project's direction. The group also appreciates Ms. Sherilyn Fajutagana, the technical critic, for her thorough analysis and recommendations, enhancing the project's quality.

The researchers extend their gratitude to Ms. Aileen Aranda, the project's client, for her trust, collaboration, and inspiring vision. Lastly, the team thanks their families, friends, and peers for their unwavering support, which motivated them to overcome challenges and achieve their goals.

THE AUTHORS

ABSTRACT

NILLOS, JOY MICA M., RAVALO, KIMBERLY LOUSIE M., SULO, JUSTINE JADE A. Paperazzi: A Coin Operated Printing Machine Using Raspberry Pi 3B for Cavite State University Imus Campus. Undergraduate Capstone Project Bachelor of Science in Information Technology. Cavite State University, Imus Campus, Imus, Cavite. February 2025. Adviser: Mr. Carlo P. Malabanan.

The Paperrazzi machine is an innovative self-service system designed to streamline document printing for students and professionals. Developed from July 2024 to February 2025, this project integrates a coin-operated mechanism, providing a user-friendly and secure method for quick and reliable printing services.

This study addresses the challenges faced by students in accessing affordable and efficient printing solutions on campus. The absence of convenient and automated printing services has highlighted the need for a system that combines accessibility, cost-effectiveness, and ease of use.

The system was developed using Raspberry Pi as the primary processor, programmed with Python to manage operations. Additional tools, including MySQL for database management and Wi-Fi modules for file transfer, were employed to ensure seamless functionality and convenience for users.

A survey conducted among fifty (50) students at CvSU Imus Campus revealed a strong demand for an on-campus, coin-operated printing system. The respondents emphasized the importance of affordability, accessibility, and reliability, validating the project's relevance and importance.

The Paperrazzi machine exclusively supports A4-sized bond paper, features real-time sales reporting, and enables file transfers via Wi-Fi. By addressing common printing challenges, the system offers a practical and efficient solution for modern printing needs.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figures **Page**

LIST OF APPENDIX TABLES

LIST OF APPENDIX FIGURES

LIST OF APPENDICES

PAPERAZZI: A COIN OPERATED PRINTING MACHINE USING RASPBERRY PI 3B

Joy Mica M. Nillos
Kimberly Louise M. Ravalo
Justin Jade A. Sulo

An undergraduate capstone project manuscript submitted to the faculty of the Department of Computer Studies, Cavite State University, Imus Campus, Cavite in partial fulfillment of the requirements for the degree of Bachelor of Science in Information Technology with Contribution No. _____. Prepared under the supervision of Mr. Carlo Malabanan.

INTRODUCTION

The coin-operated printing machine project aimed to transform on-the-go printing by offering a convenient and accessible solution. In today's fast-paced world, individuals often needed to print documents, assignments, or presentations quickly and affordably. However, traditional printing methods were inconvenient and expensive, involving trips to dedicated printing shops or reliance on personal printers that might not always have been readily available. This project sought to address these challenges by providing a user-friendly and cost-effective alternative.

The coin-operated printing machine integrated hardware and software components to create a smooth printing experience. Users simply inserted coins into the machine, selected their desired printing options through a user-friendly interface, and received their printed documents in a matter of moments. This innovative solution targeted a wide range of users, including students, professionals, and the general public, who often required printing services on the go. By bridging the gap between traditional printing methods and the evolving needs of modern users, this project had.

the potential to transform the way people accessed and utilized printing services, making it more convenient, affordable, and accessible to all.

Project Context

Access to printing services was essential for many sectors, yet traditional printing facilities often presented barriers such as limited availability, high costs, and inaccessibility. This lack of access posed significant challenges for individuals and organizations who relied on obtaining physical copies of documents for educational, business, or personal purposes. This research aimed to investigate the potential of coin-operated printing machine as a solution to these issues. By examining the technical feasibility of using Raspberry pi assessing user acceptance, analyzing cost-effectiveness compared to traditional options, evaluating reliability and performance, and examining the broader social impact, the study sought to determine if this innovative approach could effectively improve access to printing services while remaining affordable and user-friendly.

Objectives of the Study

Purpose and Des The primary objective of this study is to develop and deploy a coin operated printing machine to provide a reliable, user-friendly, and efficient solution for coin-operated printing services.

Specifically, it aimed to:

1. design a Coin Operated Printing Machine that:
    a. provides an intuitive user interface through a touchscreen monitor for selecting and printing documents;
    b. accepts exact coins only through a coin acceptor;
    c. allows users to send files wirelessly via Wi-Fi for printing;
    d. Uses a database system to store transaction logs; and
    e. Generates sale reports that can be viewed online by the client;
2. develop the machine using the following:

a. python for programming the system logic;
b. raspberry pi 3b as the main computing unit;
c. touchscreen monitor for user interface and operation;
d. coin acceptor for validating payments;
e. SQL for database managements;
f. power supply to ensure stable machine operation; and
g. jumper wires for electrical connections between components.

3. Evaluate the machine using the adapted ISO 25010 evaluation instrument; and
4. Prepare and implementation plan for deployment and maintenance.

Purpose and Description

This study aimed to design and develop a user-friendly, coin-operated printing machine to provide convenient and affordable printing services. The project desired to address the evolving needs of modern users who required on-demand access to printing resources without the need for dedicated facilities or technical expertise. By combining convenience, affordability, and accessibility, this solution catered to the printing needs of students, professionals, and the general public in various locations. The suggested system possesses the following capabilities.

1. The coin-operated printing machine features a user-friendly interface with a clear display and straightforward instructions, making it accessible even to non-technical users.
2. It provides real-time feedback on both payment status and printing progress, ensuring users have a seamless and informed experience.
3. Its compact and durable design makes it portable and robust, suitable for deployment in locations such as schools, libraries, offices, and other public spaces.
4. The machine integrates hardware and software, including a Raspberry Pi and a printer module, ensuring smooth and efficient operations.

5. It is designed for ease of installation and maintenance, reducing operational costs and minimizing downtime.
6. The machine offers customizable printing options, catering to the diverse needs of users by allowing them to select various print settings.

This coin-operated printing machine provides practical benefits to both clients and users by addressing their specific needs for convenient, affordable, and efficient

printing services. Below are its key significances to clients and users and future researcher

1. **Client.** The development of the Paperrazzi Coin-Operated Machine offers a user-friendly, automated solution for printing needs in public spaces. The integration of an intuitive interface, along with coin acceptors, allows for an efficient, self-sustaining business model that provides a valuable service to customers in libraries, schools, internet cafes, and similar establishments. This system will enable clients to minimize operational overhead while offering customers seamless access to printing services.
2. **Users.** This study provides users with an easy and efficient means to print documents without requiring assistance from a service attendant. By simply sending a file via QR code, users can pay through cash ensuring a convenient and hassle-free experience. The Paperrazzi machine offers flexibility, and ease of access, making it ideal for individuals who need immediate printing services in locations with high foot traffic.
3. **Future Researchers.** Researchers can use this study as a reference for improving coin-operated systems and integrating more advanced features. The use of Raspberry Pi for real-time communication and processing provides a solid foundation for similar systems. Researchers may explore areas such as enhancing the payment integration, expanding device compatibility, improving security features, and developing new methods for customer interaction with coin-operated machines in public environments. Additionally, this project may inspire further studies into the environmental impact of such machines or ways to reduce energy consumption.

Time and Place of the Study

The study was conducted at the Cavite State University - Imus Campus over two semesters, starting from July 2024 until February 2025. The development, testing, and initial deployment of the coin-operated printing machine took place within a designated project space at the Cavite State University - Imus Campus to ensure a focused and secure environment for the research.

Scope and Limitation of the Study

The coin-operated printing machine is designed and accept one (1) peso, five (5) peso, ten (10) peso, and twenty (20) peso coins (new or old) for document printing. It uses a coin acceptor to detect and validate these coins. The system is built around a Raspberry Pi 3B model and features a touchscreen interface for user interaction. The machine supports a single printer and prints exclusively on A4-sized paper. Users can upload their documents via a website using the Wi-Fi option, which is accessed by scanning a QR code displayed on the machine's screen. All print job data is stored in a MySQL database to ensure efficient management and tracking of print jobs. The software comprises several key modules, including the coin acceptor system for payment validation, the printer management system for handling prints jobs, and the user interface with Wi-Fi functionality. The website for Wi-Fi printing allows users to upload documents directly for printing, while the database module ensures accurate tracking and updating of print job statuses. The system efficiently manages print job statuses, ensuring jobs are processed as soon as they are uploaded.

The machine is limited to using a single printer for document printing, and it does not support cashless payment methods, focusing solely on coin-operated transactions. It is also constrained by the use of the Raspberry Pi 3B model, which may limit the system's performance for more complex tasks or higher volume printing. The touchscreen interface is the only method for user interaction, and while it provides a simple and intuitive way to interact with the machine, it may not be as flexible as

other input methods. Additionally, the machine is housed in a compact 4’6” casing, which may limit the space available for additional features. The Wi-Fi option relies on users being able to scan a QR code and upload documents via a website, which may pose difficulties if the user is unfamiliar with the process or lacks access to the internet.

Conceptual Framework

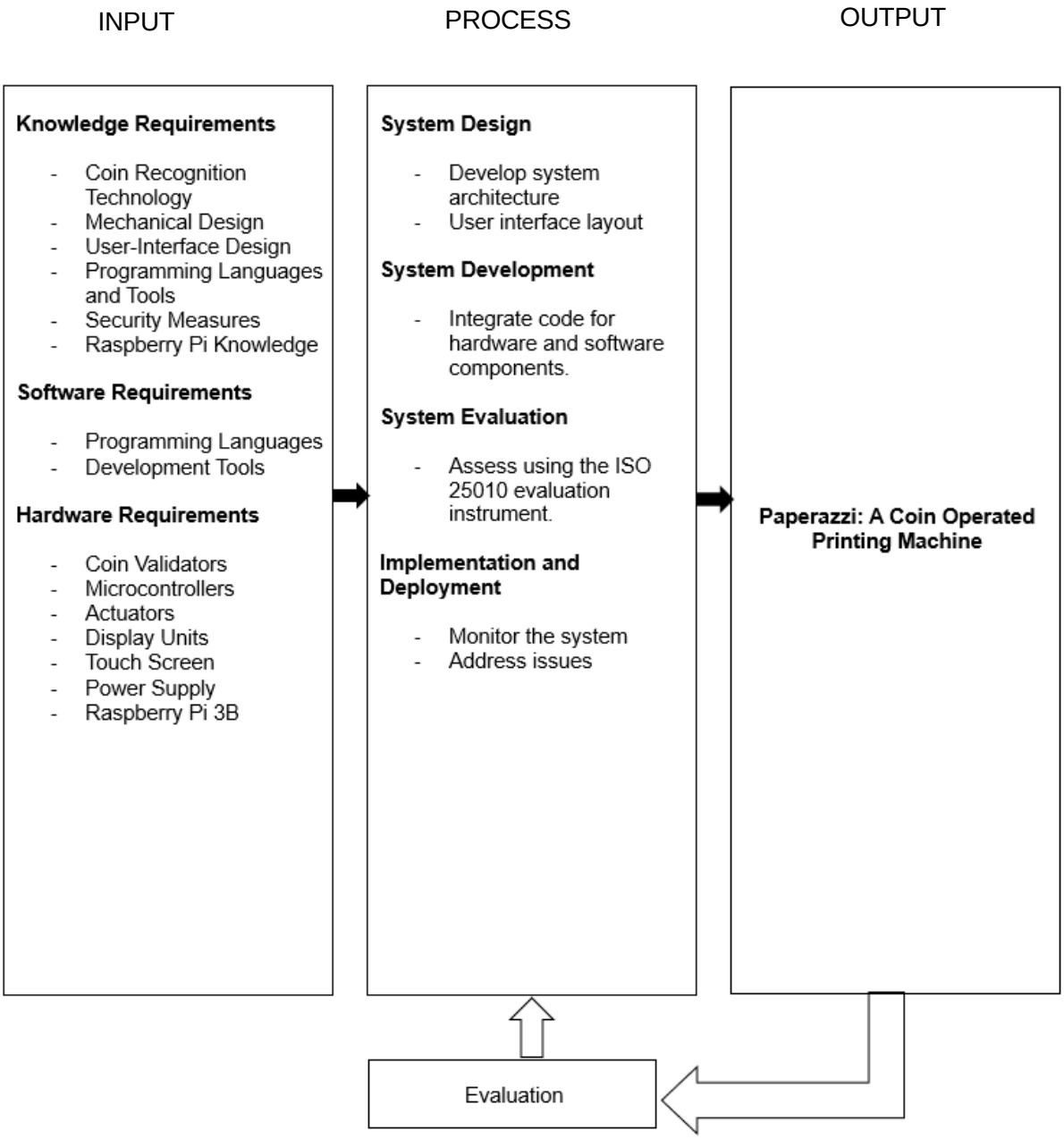


Figure 1. Conceptual Framework of the Study

The conceptual framework for a coin vending printing machine follows the Input- Process-Output model. The input consists of essential knowledge, software, and hardware requirements, including coin recognition technology, programming, tools, Raspberry Pi 3B, and coin validators. The process involves system design, development, evaluation using the ISO 25010 standard, and implementation through monitoring and issue resolution. Finally, the output is a fully functional Paperazzi coin-operated printing machine, ensuring efficient and user-friendly printing services.

Definition of terms

The following terms are operationally defined in this study:

Coin-Operated Printing: In this study, coin-operated printing is a system that enables users to initiate and pay for printing services by inserting coins into a designated mechanism integrated into the machine.

Coin Vendo: A technical mechanism incorporated into the printing machine that facilitates the conversion of banknotes into their equivalent value in coins, ensuring compatibility with the coin-operated system.

Deployment: In this study, deployment refers to the systematic installation, configuration, and activation of the coin-operated printing machine in its operational location to ensure readiness for public use.

Hardware Components: The physical elements of the coin-operated printing machine, including the Raspberry pi 3B, printer module, coin acceptor, display screen, and power supply, which are essential for system functionality.

Internet of Things (IoT)-based Machines: Devices equipped with connectivity technologies that enable data transmission, interaction with users, and integration with digital platforms, as applied in this study to enhance the printing machine's functionality.

Maintenance: In this study, maintenance refers to the ongoing technical procedures necessary to sustain the operational efficiency of the coin-operated printing machine, including troubleshooting, replenishing consumables, cleaning, and updating software systems.

Prototype Development: The iterative process conducted in this study to design, construct, and optimize a functional model of the coin-operated printing machine for evaluation and testing purposes.

Raspberry Pi 3B: A compact and powerful single-board computer used in this study to serve as the main processing unit for the printing machine, managing the system's user interface, file handling, and IoT-related functionalities.

Software Components: In this study, software components refer to the programs and algorithms developed to manage the machine's operations, such as coin validation, payment processing, user interface management, and document printing.

System Architecture: The organized structure of the coin-operated printing machine, encompassing the technical arrangement and interaction between hardware and software components, as defined in this study.

Technical Feasibility: In this study, technical feasibility refers to the assessment of technological resources, expertise, and infrastructure required to design, develop, and implement the coin-operated printing machine effectively.

User-Centric Design Approach: A development methodology utilized in this study that prioritizes the functional requirements, preferences, and ease of use for end- users during the design and implementation phases of the printing machine.

User Interface (UI): The interactive and visual elements of the printing machine, designed in this study to facilitate user navigation, payment processing, and print job execution through an intuitive and accessible interface.

Waterfall Model: This study follows a linear and systematic approach, comprising distinct phases: requirements analysis, system design, implementation, testing, deployment, and maintenance.

REVIEW OF RELATED LITERATURE

This chapter discusses the different related literature and studies for the mobile application in the sign language tutorial for Filipinos. This literature and study review aided the researchers in gaining familiarity with and understanding of the topic.

Related Literature

Paper Automation

Automating paper vending, as seen in university print shops, offers a more efficient and reliable way to sell paper compared to manual methods. This technology streamlines the process, reduces errors, and minimizes the risk of dishonest transactions, leading to a more trustworthy retail environment. By adopting coin-operated paper vending machines, universities can improve the overall experience for students and enhance resource management within their print shops.

Vending Machines

Vending Machines are automated machines that dispense selling products such as snacks, beverages, lottery tickets, and etc. It is vital to save time and reduce human energy. These vending machines are developed in the way of Non IoT based and IoT based methods. These Non IoT based machines are not smart and are not operated in real-time data, which are functioned when giving cash or card and inputs (vending things) of the machine. It is controlled by a microcontroller and distributed to the given inputs. IoT- based machines are computerized, which have cashless payment facilities, order facility before going to the vending machine to order things, and can be identified the location of machines by the customer

Counting and interpreting currency coins

Counting and interpreting currency coins with accuracy and high speed is a challenging problem for banks and stores, and even consumers. The systems of the machine that are available in the market are not good enough because sometimes it

makes a mistake when it detects the value of a coin and causes the error calculation in the result. In this proposed work, a coin operated machine which can differentiate between coins accurately and automatically is implemented. An automatic coin counting and sorting machine prototype is developed by using Raspberry Pi as the main controller.

Vending Machine with Cash and Cashless Payment Support

A Vending machine is a device that is used to disperse a product to user once a certain amount of money is deposited into it. Vending machines were once used just to sell food, drinks and other small items but nowadays they also sell products like cameras, mobiles etc. Vending machines are easily found in tourist spots, airports, railway stations etc and Vending machines are very popular in developed countries like Japan, US, UK. Conventionally people used cash as standard payment mode for vending machines.

Intelligent Sanitary Napkin Coin Operated Dispensing System

There are various vending machines operating now such as the Intelligent Sanitary Napkin Coin-Operated Dispensing System which is a vending machine that offers sanitary napkin dispensers to users once a predetermined amount is inserted into the device. The features of the microcontroller are broken down into a more accessible packaging by Raspberry Pi. Their vending machine can store up to 24 napkins and has an LED display to indicate the availability of stock. Installing vending machines in public facilities can provide convenient access to napkins when needed, avoiding potential awkward situations. (Anitha & Chaithanya, 2020)

Vending Machine with Cash and Cashless Payment Support

Wi-Fi connection using a vending machine can be the cheapest Internet connection because a vending machine is coin-operated. “Coin-operated” means that one only needs to insert a coin on the machine to access the Internet through a Wi-Fi connection. The idea of Piso-net largely influenced the creators of the J2KLC Wi-Fi Vendo machine to establish this kind of business. The word J2KLC came from the

acronym of each member of the researchers. The researchers provided an affordable and easy way to have an internet connection to the community and also give profit to possible customers. (Alburo et al., 2020)

Smart Charging Vending Machine

The goal of Smart charging vending machines is to design and implement a vending machine in which the user has to plug the mobile phone into one of the slot and insert a coin. Charging is done depending on the type of the coins inserted by the users. This machine can be quickly and easily installed outside any business premises e.g.: Railway Station, Airports, Hospital, shopping mall etc.

Vending Machine Technologies: A Review Article

Vending Machines are automated machines that dispense selling products such as snacks, beverages, lottery tickets, and etc. It is vital to save time and reduce human energy. These vending machines are developed in the way of Non IoT-based and IoT based methods. These Non IoT based machines are not smart and are not operated in real-time data, which are functioned when giving cash or card and inputs (vending things) of the machine.

Newspaper Vending Machine

A newspaper vending machine is an automatic machine for selling newspapers and customers operate it by themselves. The advantages of vending machine method are meet the concept selling anywhere, closer to customer and cost down in sales. This vending machine uses some coins or money to initiate selling process. The aim of this research is to applied microcontroller to control newspaper vending machine, Programmable Logic Device (PLD) to count coins, light sensor to detect the coin, sensor TCS3200 is used to detect color of money, dc motor to eject newspaper from vending machine and LCD show instruction to purchase the newspaper.

Coins Operated Pizza Vending Machine

Pizza is the fastest food ordered and Due to the small number of pizza restaurants and Most people don't know how make it and the high demand on it by

people. So, we decided to design and implement an automatic machine with dimensions (3m long,1.5m width), make to produce pizza with high quality without waiting in markets. The prototype contains a six main parts: Coins Detector, Refrigerator, sauce distribution, Cheese distribution, Vegetable distribution and Baking stage, with experiments as follows. is Activated the Coins Detector by pay 5 (₪), The advantages of using this machine are for save the man power, money and time, this machine is located in public places.

Systems Technical Background

Coin-operated printing vending machines integrated a combination of mechanical, electrical, electronic, and software systems to automate the process of document printing and payment. This section was divided into the technical background that enabled these machines to function accordingly.

Mechanical Components

- **Coin Mechanism**: The coin mechanism remains a core component, responsible for accepting and validating coins based on their physical attributes and electromagnetic properties. Rejected coins are returned, ensuring accurate payment processing.
- **Paper Handling Mechanism**: This mechanism is specialized for managing paper. It includes a paper feed system that draws paper from a supply, a printing mechanism (often inkjet or laser), and an output tray for delivering printed documents.
- **User Interface**: This involves physical buttons or a touchscreen for user interaction, allowing them to select document types, printing options, and initiate the printing process.

Electrical and Electronic Components

- **Printing Technology**: The printing technology employed (inkjet, laser) significantly impacts print quality, speed, and cost-efficiency. The choice depends on the target user base and desired printing capabilities.
- **Display**: A display, either LED or LCD, provides visual feedback to the user, displaying instructions, printing progress, error messages, and potentially even previews of documents.

Software Components

- **Document Processing Software**: This software may be integrated into the firmware or reside on a separate module. It handles document formatting, scaling, and conversion to ensure compatibility with the printing mechanism.
- **User Interface Software**: This software governs the interaction between the user and the machine, ensuring a user-friendly experience through navigation and clear instructions.

Synthesis

This study focused on automating paper vending in university print shops to enhance efficiency, reliability, and the overall user experience while reducing errors and dishonest transactions. Inspired by the design and functionality of various vending machine technologies, the system incorporated a coin-operated mechanism to provide a straightforward and accessible solution for students. By utilizing an accurate coin recognition system, the project ensured precise and reliable transactions, addressing the common challenges associated with manual processes.

The study drew insights from both IoT and non-IoT vending machines to create a robust, adaptable, and cost-effective solution tailored specifically for paper vending. Key inspirations included coin-operated Wi-Fi vending machines, which demonstrated affordability and practicality, and sanitary napkin dispensers, which highlighted the

importance of user-friendly designs. Additionally, systems like newspaper vending machines and pizza vending machines showcased the effectiveness of automation for dispensing complex products efficiently, which informed the design of this project.

By integrating proven concepts from existing vending technologies, the study aimed to streamline paper vending processes in university settings. This automation not only simplified resource management for print shops but also offered students a seamless, reliable, and convenient experience when accessing printing materials.

METHODOLOGY

This chapter includes the designs of software, systems, products, or processes, system development, system testing and evaluation, data analysis, and implementation of the study.

Design of Software, Systems Product, and/or Processes

PAPERAZZI is a self-service, coin-operated printing powered by a Raspberry Pi. It allows users to upload documents via cloud services and print them by inserting coins. The system ensures an easy-to-use experience while focusing on seamless payment and printing processes.

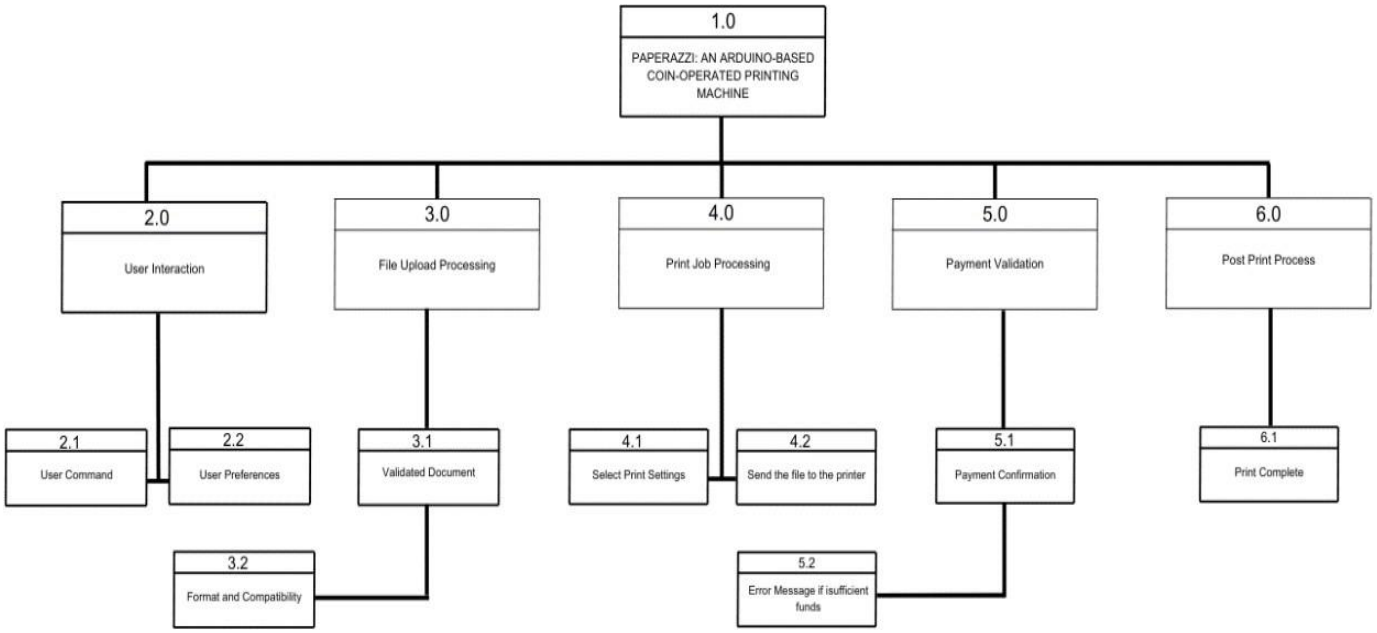


Figure 2. HIPO diagram

The HIPO (Hierarchy plus Input-Process-Output) diagram for PAPERAZZI breaks down the system into five key components: User Interaction, Payment Validation, File Upload Processing, Print Job Processing, and Post-Print Process. The

User Interaction component allows users to insert coins, upload documents, and select print settings. The Payment Validation process confirms the user's payment by checking if enough coins have been inserted. The File Upload Processing component validates the uploaded file for format compatibility and prepares it for printing. The Print Job Processing system applies the selected print settings (such as color and number of pages) and sends the file to the printer. The Post-Print Process notifies the user of completion and resets the system for the next user.

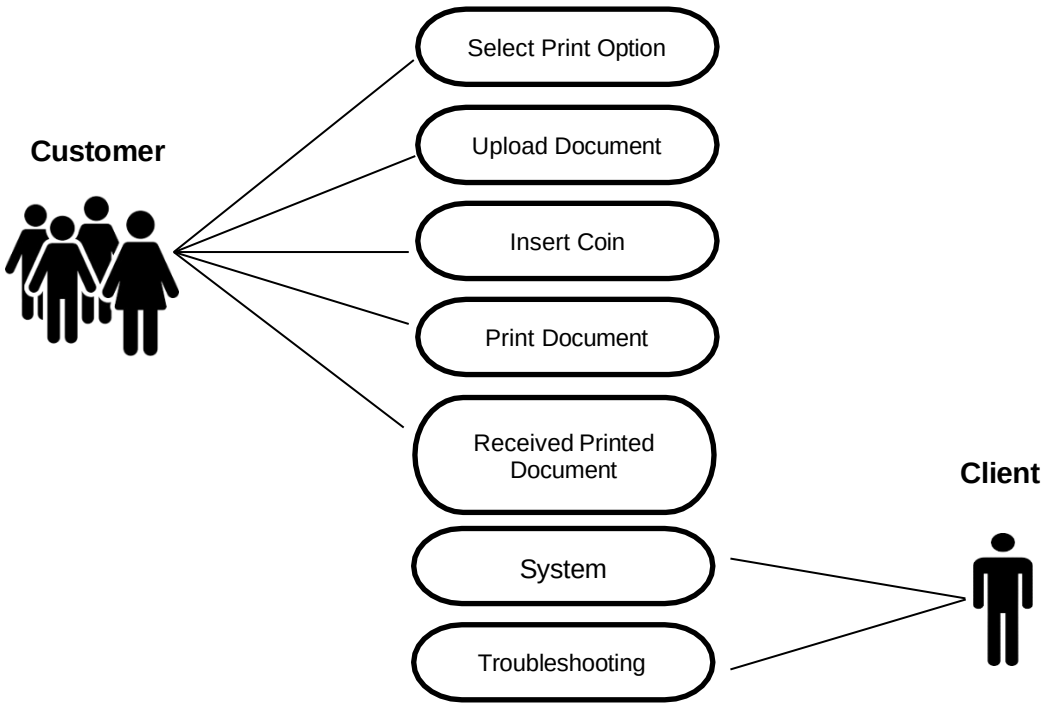


Figure 3. Use Case Diagram of Paperazzi

The use case diagram for Paperazzi illustrates the interactions between the customer and the client, highlighting key functionalities. The user can insert coins, upload documents, select print settings, and initiate printing. The system validates payment, processes the uploaded file, and completes the print job. After printing, the system notifies the user and resets for the next session.

System Development

The development of the coin-vending printing machine adapted the waterfall model with six stages namely…

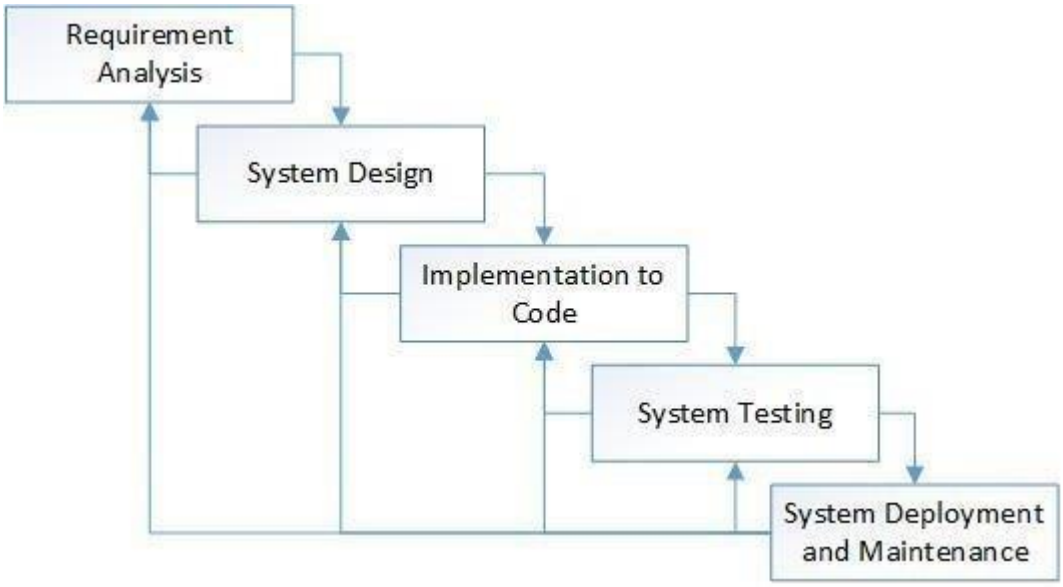


Figure 4. Modified Waterfall Model (Utama, 2018)

The development of a coin-vendo printing machine followed the modified waterfall model, a software development lifecycle approach that allows for some iteration and feedback between phases. This model's flexibility proved valuable in adapting to the evolving needs and constraints encountered during the project.

Requirement Analysis. The initial phase involved gathering detailed information from potential users and stakeholders to understand their specific needs and preferences.

During this phase, extensive surveys and interviews were conducted with potential users to identify their specific printing needs (e.g., paper sizes, color options), payment preferences (coins, digital), and overall expectations regarding the machine's functionality and user interface.

System Design. Based on the requirements gathered in the previous phase, the system's architecture, both hardware and software, was designed in detail. This included selecting appropriate components, defining the system's structure and modules, and designing the user interface (UI) and user experience (UX).

The hardware architecture of the coin-operated printing machine was created, incorporating a Raspberry Pi 3b as the central control unit, along with a printer module, coin acceptor, display screen, buttons for user input, and a power supply. The software architecture was also outlined, detailing the logic for handling printing operations, coin acceptance and validation, payment processing, user interface interactions, and potential remote monitoring features.

Implementation. The design was translated into a working product during this phase. The hardware components were assembled and integrated, connecting the Raspberry pi with the printer, coin acceptor, display screen, and other peripherals. The outcome was a functional prototype ready for testing and refinement.

Testing. This phase focused on verifying that the implemented system met the defined requirements and functioned as expected. Various testing methodologies, such as unit testing, integration testing, and system testing, were employed to identify and address defects or issues before deployment.

Rigorous testing was conducted to ensure the machine's performance, reliability, and adherence to the specified requirements.

Deployment and Maintenance. After completing testing and making necessary refinements, the machine was deployed to its intended location.

This process included installation, configuration, and user training. Maintenance became an ongoing activity to monitor, troubleshoot, update, and enhance the system to ensure its continued operation and performance.

System Testing

The "Paperrazzi: A Coin Operated Printing Machine" project aimed to combine hardware (a Raspberry Pi) with software to manage printing services. The system featured wireless file transfer options, to enhance user convenience. To ensure the system's functionality, reliability, and quality, a thorough testing strategy was implemented:

Unit Testing: Individual components or modules, such as Raspberry Pi code functions, coin validation logic, print queue management, and user interface elements, were tested in isolation to ensure they performed as intended. Test cases were developed based on the specifications and expected behaviors of each unit.

System Testing: The entire system was tested to ensure it met the defined requirements. End-to-end testing simulated real-world usage scenarios, verifying that the integrated system functioned as intended. Performance testing was conducted to assess system behavior under different loads and stress conditions, while security testing ensured protection against common threats.

System Evaluation

Upon completing the system, the researchers carried out a survey to evaluate the proposed project. They developed the survey questionnaire using the Likert Scale with descriptive interpretation as a basis. The questionnaire was organized into six key categories: functionality, reliability, usability, performance efficiency, security, and maintainability.

RESPONDENT	FREQUENCY	PERCENTAGE
Students	50	100%

TOTAL	**50**	**100%**

Table 1. The respondent of the Study

Table 1 shows the demographic distribution of respondents who evaluated the *Paperazzi* coin-operated printing machine. All fifty (50) respondents were students, accounting for 100% of the total sample size. The researchers selected students as the primary respondents because they represent the machine's intended end users-students of CvSU-Imus Campus, who frequently require accessible and convenient printing services. By gathering feedback exclusively from students, the researchers ensured that the evaluation reflects the real-world experience and expectations of its primary users. The 100% response rate also indicates that all selected participants actively engaged in providing feedback, ensuring a reliable and comprehensive assessment of the system's performance, functionality, and usability.

The researchers used a point-based system to determine the level of agreement from respondents on the survey questionnaire items. The assigned points were:

NUMERICAL SCALE	INTERPRETATION
5	Excellent
4	Very Good
3	Good
2	Fair
1	Poor

Table 2. Option in each item of the Questionnaire

NUMERICAL SCALE	INTERRETATION
4.51 - 5.00	Excellent
3.51 - 4.50	Very Good
2.51 - 3.50	Good
1.51 - 2.50	Fair
1.50 - below	Poor

Table 3. Descriptive interpretation of the mean

The following statistical procedures will be used to analyze the data to be gathered from System Evaluation Questionnaire. The results of the statistical procedure will determine the general perception of the respondent on the system. Sample mean is the average score of a sample on a given variable.

Formula:

$$\bar{x} = \frac{\sum_{i=1}^{n} X_i}{n}$$

Where:

$\bar{x}$= mean

Xi = representation of each observation from respondents

n = total number of respondents

Sample standard deviation is a measure of the spread (variability) of the scores in the sample on a given variable.

Formula:

$$s = \sqrt{\frac{\sum_{i=1}^{n} (X_i - \bar{x})^2}{n-1}}$$

Where:

$\bar{x}$= mean

Xi = representation of each observation from respondents

n = total number of respondents

s = sample standard deviation

Percentage determines the frequency counts and percentage distribution of personal related variables of the respondents.

Formula:

$$\% = \frac{f}{n} \cdot 100$$

Where:

% = percentage

f = frequency

n = total number of respondents

Implementation Plan

The implementation of the "Paperazzi" coin-operated printing machine followed a structured, step-by-step approach to ensure a smooth transition from concept to deployment. First, the project team finalized the system design and components, including the Raspberry Pi 3B model as the central processing unit, along with a compatible coin acceptor mechanism, printer module (A4 compatible), and touchscreen display. Next, the hardware was assembled, integrating all components into a functional printing machine prototype. Meanwhile, the software development team programmed the necessary features, such as coin validation, print job processing, and user interface functionality for the touchscreen.

Following initial development, the team conducted rigorous unit testing on both hardware and software to identify and resolve any issues. Once testing was complete, a pilot deployment took place at the Cavite State University Imus Campus, allowing for real-world testing and user feedback. During this phase, user training was conducted

to ensure effective use of the machine. After gathering feedback and making necessary adjustments, the system was refined and fully deployed at the Cavite State University Imus Campus.

Ongoing support and maintenance were established to address any future issues and facilitate continuous improvements based on user experiences and technological advancements. This phased implementation plan ensured a successful launch and integration of "Paperazzi," ultimately providing a convenient and reliable printing solution for users at the Cavite State University Imus Campus.

STRATEGY	ACTIVITIES	PERSONS INVOLVED	DURATION
Approval from the campus administrator	Letters	Researchers, Administrators	September-October
System's Installation	Installation of the system required software and hardware	Researchers, Administrators	October-November
Train the Users (Admin and End Users)	Teach the users about the manuals of how to use the system and the machine	Researchers, Administrators, End Users	November
Fix Errors	Quickly solve the error might users encounter	Quickly solve the error might users encounter	November-December

Table 4. Shows the Implementation Plan

RESULT AND DISCUSSION

This chapter details the statistical results of the study's evaluation. The researchers used a modified waterfall model for system development and gathered feedback from 50 student respondents at CvSU-Imus Campus to assess the system's performance.

System Design

The researchers of Paperazzi: A Coin-Operated Printing Machine presented an innovative solution for self-service printing. The system was designed to streamline the printing process by automating payment, file transfer, and printing, all while providing a simple and user-friendly interface. The system integrated several essential components to ensure seamless operation:

1. Raspberry Pi 3B Controller: Served as the central processing unit, managing file handling, printer control, and payment processing.

2. Coin Acceptor Integration: Functioned as the primary payment system, allowing users to pay exclusively with coins. The system validated and processed payments before initiating the printing process.

3. Printer Management: The system was equipped with a single printer, dedicated to printing documents on A4 paper size, ensuring consistent and high-quality output for every print job.

4. File Transfer System: Supported wireless file transfer from mobile devices via Wi-Fi, enabling users to upload documents easily.

5. User Interface: A touchscreen guided users through the process, from file selection to payment confirmation and printing.

6. System Monitoring: Continuously tracked system usage, error logs, and printing statistics to ensure reliable operation and prompt maintenance.

The primary function of the Paperazzi system was to offer a convenient, self-service printing solution that minimized the need for human assistance while providing a reliable and efficient user experience.

System Development

The system development process used the modified waterfall model. This model provided a structured yet flexible approach by breaking the project into clear phases. It allowed the team to revisit earlier stages when necessary, ensuring that all components worked seamlessly together. By using this approach, the team managed risks effectively, set clear milestones, and delivered a reliable and well-integrated coin-operated printing machine.

System Testing

A self-service, coin-operated printing system for PAPERAZZI was strategically focused on adhering to the ISO 25010 software quality standard. This comprehensive framework ensured that crucial quality features, such as functionality, usability, reliability, and security, were organized and integrated into the system to fulfill specific user needs and operational requirements.

System Evaluation

Table 5. Rating on the Paperazzi's Functionality

	FUNCTIONALITY	
INDICATOR	**MEAN**	**INTERPRETATION**
Functional completeness. Degree to which the set of functions covers all the specified tasks and user objectives.	4.9	EXCELLENT
Functional correctness. Degree to which a product or system provides the correct results with the needed degree of precision.	4.58	EXCELLENT
Functional appropriateness. Degree to which the functions facilitate the accomplishment of specified tasks and objectives	4.3	GOOD
WEIGHTED MEAN	**4.6**	**GOOD**

The weighted average for overall functionality of the system is 4.6, which is considered "Good." The highest-rated feature is Functional Completeness, with a mean score of 4.9, indicating that the system offers nearly all the necessary functions to meet the specified tasks and user objectives. Functional Correctness received a slightly lower score of 4.58, suggesting that while the system provides accurate results, there may still be room for improvement in precision or consistency. However, the overall assessment of Functional Appropriateness is rated as "Good," with a mean score of 4.3, indicating that the system's features are generally suitable for accomplishing the specified tasks and objectives, though some adjustments may be needed to optimize task facilitation.

Table 6. Rating on the Paperazzi's Reliability

RELIABILITY		
INDICATOR	**MEAN**	**INTERPRETATION**
Maturity. Degree to which a system, product or component meets needs for reliability under normal operation.	3.96	GOOD
Availability. Degree to which a system, product or component is operational and accessible when required for use.	4.12	GOOD
Fault tolerance. Degree to which a system, product or component operates as intended despite the presence of hardware or software faults.	3.94	GOOD
Recoverability. Degree to which, in the event of an interruption or a failure, a product or system can recover the data directly affected and re-establish the desired state of the system.	4.14	GOOD
WEIGHTED MEAN	**4.04**	**GOOD**

The overall reliability score is 4.04, rated as "Good." Availability scored the highest at 4.12, indicating the system is generally accessible when needed. Recoverability follows closely with 4.14, showing strong data recovery and system restoration capabilities. Maturity and Fault Tolerance both scored 3.96 and 3.94 respectively, reflecting good reliability, though some improvements in handling faults and normal operation consistency could be made

Table 7. Rating on the Paperazzi's Usability

USABILITY		
INDICATOR	**MEAN**	**INTERPRETATION**
Appropriateness recognizability. Degree to which users can recognize whether a product or system is appropriate for their needs.	4.3	GOOD
Learnability. degree to which a product or system can be used by specified users to achieve specified goals of learning to use the product or system with effectiveness, efficiency, freedom from risk and satisfaction in a specified context of use.	4.04	GOOD
Operability. Degree to which a product or system has attributes that make it easy to operate and control.	3.96	GOOD
User error protection. Degree to which a system protects users against making errors	4.16	GOOD
User interface aesthetics. Degree to which a user interface enables pleasing and satisfying interaction for the user.	4.02	GOOD
Accessibility. Degree to which a product or system can be used by people with the widest range of characteristics and capabilities to achieve a specified goal in a specified context of use.	3.98	GOOD
WEIGHTED MEAN	**4.07**	**GOOD**

The overall usability score is 4.07, rated as "Good." User Error Protection has the highest score at 4.16, indicating strong safeguards against user

mistakes. Appropriateness Recognizability follows with 4.3, meaning users can easily recognize if the system meets their needs. Learnability and Accessibility scored 4.04 and 3.98 respectively, reflecting a generally easy-to-learn system that is accessible to a wide range of users. Operability scored 3.96, suggesting the system is easy to operate but could benefit from further improvements in control and usability.

Table 8. Rating on the Paperazzi's Performance Efficiency

	PERFORMANCE	
INDICATOR	**MEAN**	**INTERPRETATION**
Time Behaviour. Degree to which the response and processing times and throughput rates of a product or system, when performing its functions, meet requirements.	4.56	EXCELLENT
Resource utilization. Degree to which the amounts and types of resources used by a product or system, when performing its functions, meet requirements.	4.22	GOOD
Capacity. Degree to which the maximum limits of a product or system parameter meet requirements.	3.96	GOOD
WEIGHTED MEAN	**4.24**	**GOOD**

The overall Performance Efficiency score is 4.24, rated as "Good." Time Behavior received the highest rating, indicating excellent response and processing times that meet the required standards. Resource Utilization follows with a good score, suggesting efficient use of resources. Capacity scored slightly lower, showing that

while the system generally meets capacity requirements, there may be room for improvement in handling maximum limits.

Table 9. Rating on the Paperazzi's Compatibility

	COMPATIBILITY	
INDICATOR	**MEAN**	**INTERPRETATION**
Co-existence. Degree to which a product can perform its required functions efficiently while sharing a common environment and resources with other products, without detrimental impact on any other product.	4.16	GOOD
Interoperability. Degree to which two or more systems, products or components can exchange information and use the information that has been exchanged.	4.16	GOOD
WEIGHTED MEAN	**4.16**	**GOOD**

The overall co-existence and interoperability score is 4.16, rated as "Good." Both Co-existence and Interoperability received the same score, indicating that the system performs efficiently while sharing resources with other products and is effective in exchanging and using information with other systems**.**

Table 10. Rating on the Paperazzi's Security

SECURITY		
INDICATOR	**MEAN**	**INTERPRETATION**
Confidentiality. Degree to which a product or system ensures that data are accessible only to those authorized to have access.	3.86	GOOD
Integrity. Degree to which a system, product or component prevents unauthorized access to, or modification of, computer programs or data.	4.2	GOOD
Non-repudiation. Degree to which actions or events can be proven to have taken place, so that the events or actions cannot be repudiated later.	4.14	GOOD
Accountability. Degree to which the actions of an entity can be traced uniquely to the entity.	3.86	GOOD
Authenticity. Degree to which the identity of a subject or resource can be proved to be the one claimed.	4.04	GOOD
WEIGHTED MEAN	**4.02**	**GOOD**

The overall security score is 4.02, rated as "Good." Integrity and non-repudiation received the highest scores, reflecting strong protection against unauthorized access and modification of data, as well as the ability to prove actions or events. Confidentiality and Accountability scored lower, indicating room for improvement in ensuring authorized access and uniquely tracing actions. Authenticity was rated moderately, suggesting that the system generally proves the identity of subjects and resources effectively.

Table 11. Rating on the Paperazzi's Maintainability

MAINTAINABILITY		
INDICATOR	**MEAN**	**INTERPRETATION**
Modularity. Degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components.	4.1	GOOD
Reusability. Degree to which an asset can be used in more than one system, or in building other assets.	4.0	GOOD
Analysability. Degree of effectiveness and efficiency with which it is possible to assess the impact on a product or system of an intended change to one or more of its parts, or to diagnose a product for deficiencies or causes of failures, or to identify parts to be modified.	4.0	GOOD
Modifiability. Degree to which a product or system can be effectively and efficiently modified without introducing defects or degrading existing product quality.	3.86	GOOD
Testability. Degree of effectiveness and efficiency with which test criteria can be established for a system, product or component and tests can be performed to determine whether those criteria have been met.	4.16	GOOD
WEIGHTED MEAN	**4.02**	**GOOD**

The overall maintainability score is 4.02, rated as "Good." Testability received the highest rating, indicating that the system is effective in establishing and performing tests to meet criteria. Modularity and Reusability scored well, showing that the system

is composed of discrete components that can be reused in other systems. Analyzability and Modifiability were rated slightly lower, suggesting that while the system is generally easy to analyze and modify, there is room for improvement in making changes without affecting quality.

Table 12. Rating on the Paperazzi's Portability

PORTABILITY		
INDICATOR	**MEAN**	**INTERPRETATION**
Adaptability. Degree to which a product or system can effectively and efficiently be adapted for different or evolving hardware, software or other operational or usage environments.	3.88	GOOD
Installability. Degree of effectiveness and efficiency with which a product or system can be successfully installed and/or uninstalled in a specified environment.	3.94	GOOD
Replaceability. Degree to which a product can replace another specified software product for the same purpose in the same environment.	4.02	GOOD
WEIGHTED MEAN	**3.94**	**GOOD**

The overall portability score is 3.94, rated as "Good." Replaceability received the highest rating, indicating that the system can effectively replace another product for the same purpose. Installability follows closely, suggesting the system can be efficiently installed or uninstalled. Adaptability received a slightly lower score, reflecting that while the system can be adapted to different environments, there may be room for improvement in handling evolving hardware or software.

Table 13. Overall Rating of Paperazzi

CRITERIA	WEIGHTED MEAN	INTERPRETATION
FUNCTIONALITY	**4.6**	**EXCELLENT**
RELIABILITY	**4.04**	**GOOD**
USABILITY	**4.07**	**GOOD**
PERFORMANCE EFFIENCY	**4.24**	**GOOD**
COMPATIBILITY	**4.16**	**GOOD**
SECURITY	**4.02**	**GOOD**
MAITAINABILITY	**4.02**	**GOOD**
PORTABILITY	**3.94**	**GOOD**
GRAND MEAN	**4.13**	**GOOD**

The system’s overall performance is rated as "Good" with a Grand Mean of 4.13. Functionality stands out with an excellent score of 4.6, indicating that the system effectively meets the core tasks of document uploading, payment processing, and printing. Reliability (4.04), Usability (4.07), and Performance Efficiency (4.24) are rated as good, demonstrating stable operation and efficient use of resources, though there is potential for improvement in areas like system stability under heavy usage or optimizing user interface responsiveness.Security is well-rated, ensuring safe handling of payment and user data, but there could be enhancements in areas like Adaptability, where the system could better handle varying user environments or conditions.

Maintainability and Portability also received strong ratings, but Modifiability might benefit from more flexible configurations or easier updates. Overall, PAPERAZZI performs strongly, but improvements in these specific areas could further enhance the system's efficiency and user experience.

Implementation Result

The researchers started the implementation plan of Paperazzi for CvSU-Imus Campus. The researchers conducted all of the activities from July 2024 to Febuary 2025.

STRATEGY	ACTIVITIES	PERSON INVOLVED	DURATIONS	REMARKS
Approval from the campus administrator	Letters	Researchers, Administrators	September-October	Done
System's Installation	Installation of the system required software and hardware	Researchers, Administrators	October-November	Done
Train the Users (Admin and End Users)	Teach the users about the manuals of how to use the system and the machine	Researchers, Administrators, Users	November-December	Done
Fix Errors	Quickly solve the error might users encounter	Quickly solve the error might users encounter	November-December	Done

Table 14. Summary of implementation

SUMMARY, CONCLUSION, AND RECOMMENDATION

Summary

The *Paperazzi* project is an innovative coin-operated printing machine designed to cater to the printing needs of students at CvSU-Imus Campus. This machine exclusively supports A4-sized prints and accepts only exact coins for payment, with no option for dispensing change. It provides users with a convenient way to send their files for printing through Wi-Fi, eliminating the need for physical connections. The machine allows users to customize their printing preferences, such as selecting specific page ranges, choosing the number of copies, and opting for grayscale or color mode, all through a simple and intuitive interface.

To ensure efficient management and monitoring, the system includes an online platform where the client or admin can access detailed sales reports, offering valuable insights into usage and revenue. The entire system is powered by a Raspberry Pi 3B, chosen for its versatility and capability to handle the processing needs of the machine. The project was developed using a modified waterfall model, allowing for a structured yet flexible approach to development, ensuring high-quality results at each stage. Overall, *Paperazzi* is a cost-effective and user-friendly solution that simplifies printing for students while providing administrators with easy access to operational data.

Conclusion

The researchers designed and implemented a hardware and software system that provides significant benefits for users of the PAPERAZZI: A Coin-Operated Printing Machine at Cavite State University (CVSU) Imus. This system enhances the convenience and efficiency of self-service printing in public spaces, transforming the traditional printing experience into a more streamlined, user-friendly process. The integration of cloud-based document upload options, coupled with a coin-operated payment system, addresses the common challenges faced by users in accessing

printing services within the university. According to the findings of the researchers, the hardware system proved beneficial based on feedback collected through user surveys and testing. The PAPERAZZI system improves the overall user experience by providing a secure, efficient, and self-sufficient printing solution, making it easier for users at Cavite State University (CVSU) Imus to access printing services without the need for a staff member. The system's performance in handling payment and printing processes effectively contributes to its positive reception.

Recommendation

To further enhance the *Paperazzi* coin-operated printing machine, the following recommendations are suggested:

1. Expand Payment Options: Incorporate additional payment methods, such as digital wallets (e.g., GCash or PayMaya) or QR code scanning, to offer more flexibility and convenience for users.
2. Introduce a Change-Dispensing Mechanism: Adding a feature to dispense change can improve user satisfaction by accommodating customers who may not have exact coins.
3. Add Multi-Paper Size Compatibility: Extend the system to support other common paper sizes, such as Letter or Legal, to cater to a broader range of printing needs.
4. Enhance File Transfer Options: Include support for USB drives or cloud-based file uploads, such as Google Drive or Dropbox, for greater accessibility and user convenience.
5. Energy Efficiency Features: Introduce an automatic power-saving mode when the machine is idle for extended periods to reduce energy consumption.
6. User Feedback Mechanism: Include a simple feedback interface for users to report issues or suggest improvements, helping maintain service quality.

7. Enhanced Data Analytics: Upgrade the admin portal with detailed data visualization tools to analyze sales trends, user preferences, and machine performance, aiding in decision-making.

By implementing these recommendations, *Paperazzi* can provide an even more seamless and user-friendly experience while improving operational efficiency and meeting the evolving needs of its users.

REFERENCES

Abad, E. P. (2017). Design and implementation of paper vending machine for retail of common usable papers for unstop students. International Journal for Research in Applied Science & Engineering Technology, 5(11). https://www.ijraset.com

Alburo, M. C., Tindugan, L., & Dacuno, A. M. (2020, April 1). Feasibility study at J2KLC Wifi Vendo Machine in 591 Bagbag St., Quirino Highway, Novaliches, Quezon City. *AASGB Conference Proceedings Journal for Management and Research Application*, *1*(1). https://ojs.aaresearchindex.com/index.php/aasgbcpjmra/article/view/1355

Irmansyah, M., & Madona, E. (2020). Newspaper Vending Machine. *WMA-2 EAI.* https://eudl.eu/doi/10.4108/eai.24-1-2018.2292389

Kavale, A., Shukla, S., & Bramhe, P. (2019, November). Coin counting and sorting machine. In 2019 9th International Conference on Emerging Trends in Engineering and Technology-Signal and Information Processing (ICETET-SIP-19) (pp. 1-4). IEEE.

Mahajan, G., Phale, V., Mane, S., & Patil, A.B. (2020). Vending machine with cash and cashless payment support. *International Research Journal of Engineering and Technology (IRJET)*, *7*(6), 62-65.

Mapari, N., Ansari, K., Ansari, T., & Shaikh, A. (2019). Smart charging vending machine. In *Sumit Das's Publication*. https://www.researchgate.net/profile/Sumit-Das-13/publication/346969142_Raw_paper/data/5fd5e068a6fdccdcb8c076d3/mapari2019.pdf

Raed, A. (2019, December 1). Coins operated pizza vending machine [Master's thesis, Palestine Polytechnic University]. https://scholar.ppu.edu/handle/123456789/7078

Ratnasri, N., & Sharmilan, T. (2021). Vending machine technologies: A review article. *ResearchGate*.https://www.researchgate.net/publication/357577286_Vending_Machine_Technologies_A_Review_Article

S, A., & J, C. D. (2020). Intelligent sanitary napkin coin operated dispensing system. *Social Science Research Network (SSRN)*. https://doi.org/10.2139/ssrn.3614231

Utama, S. R., & Guntur, R. F. (2018). Implementation of Google's Technology in Android Mobile App: Kediri City Fasum and Fasos Information System. *ResearchGate*. https://www.researchgate.net/publication/323046269_Implementation_of_Google's_Technology_in_Android_Mobile_App_Kediri_City_Fasum_and_Fasos_Information_System

APPENDICES

Appendix 1. Relevant Source Code

```
# frame 1,2,3 import tkinter as tk from tkinter import messagebox from PIL import Image, ImageTk import subprocess import os import mysql.connector # Define the close_application function before binding it def close_application(event=None): root.quit() # Function to open printingoptions.py on the Desktop def open_printing_options(): try: desktop_path = os.path.expanduser("~/Desktop") # Get user's Desktop path script_path = os.path.join(desktop_path, "printingoptions.py") subprocess.run(["python", script_path], check=True) # Open printingoptions.py except Exception as e: print(f"Error opening printingoptions.py: {e}") # Database connection function def connect_to_database(): try: connection = mysql.connector.connect( host="paperazzi.cre40o0wmfru.ap-southeast-2.rds.amazonaws.com", # Replace with your DB endpoint user="admin", # Replace with your DB username password="paperazzi", # Replace with your DB password database="paperazzi" # Replace with your database name ) return connection except mysql.connector.Error as err: print(f"Database connection error: {err}") return None # Function to check if there's a new print job def check_for_new_print_job(): try: connection = connect_to_database() if not connection: return None cursor = connection.cursor() query = """ SELECT job_id, document_name, status FROM print_jobs WHERE status = 'uploaded' # Checking for newly uploaded jobs ORDER BY created_at DESC LIMIT 1 """ cursor.execute(query) job = cursor.fetchone() if job: # A new print job is found print("New print job detected!") return True else: return False except mysql.connector.Error as err: print(f"Error fetching data: {err}") return False finally: if connection and connection.is_connected(): cursor.close() connection.close() # Function to show Wi-Fi frame def go_to_wifi(): print_options_frame.pack_forget() # Hide the print options frame wifi_frame.pack(pady=20) # Show the Wi-Fi frame # Create the main window root = tk.Tk() root.title("Paperazzi") root.config(bg="white") root.attributes('-fullscreen', True) root.bind("<Escape>", close_application) # Now works since close_application is defined # Main frame with logo and Start Printing button main_frame = tk.Frame(root, bg="white") main_frame.pack() logo_frame = tk.Frame(main_frame, bg="white") logo_frame.pack(pady=(50, 10)) logo_image = Image.open("logo.jpg") logo_photo = ImageTk.PhotoImage(logo_image) logo_label = tk.Label(logo_frame, image=logo_photo, bg="white") logo_label.pack() start_button_frame = tk.Frame(main_frame, bg="white") start_button_frame.pack(pady=20) start_button = tk.Button( start_button_frame, text="Start Printing", font=("Bebas Neue", 20), command=lambda: start_printing() ) start_button.pack(pady=10) def start_printing(): main_frame.pack_forget() print_options_frame.pack(pady=20) # Print Options frame print_options_frame = tk.Frame(root, bg="white") logo_label_print_options = tk.Label(print_options_frame, image=logo_photo, bg="white") logo_label_print_options.pack(pady=(10, 10)) print_options_label = tk.Label( print_options_frame, text="How would you like to print?", font=("Bebas Neue", 30),
```

```
bg="white", fg="black" )
print_options_label.pack(pady=20)
bluetooth_image =
Image.open("bluetooth.png").resize((120,
120)) bluetooth_photo =
ImageTk.PhotoImage(bluetooth_image)
bluetooth_label =
tk.Label(print_options_frame,
image=bluetooth_photo, bg="white",
cursor="hand2")
bluetooth_label.pack(side="left", padx=40)
bluetooth_label.bind("<Button-1>", lambda
e: go_to_bluetooth()) wifi_image =
Image.open("wifi.png").resize((120, 120))
wifi_photo =
ImageTk.PhotoImage(wifi_image)
wifi_label = tk.Label(print_options_frame,
image=wifi_photo, bg="white",
cursor="hand2")
wifi_label.pack(side="right", padx=40)
wifi_label.bind("<Button-1>", lambda e:
go_to_wifi()) # Bluetooth Frame
bluetooth_frame = tk.Frame(root,
bg="white") bluetooth_frame_label =
tk.Label( bluetooth_frame, text="Bluetooth
Printing Options", font=("Bebas Neue",
30), bg="white", fg="black" )
bluetooth_frame_label.pack(pady=20)
bluetooth_instruction_label = tk.Label(
bluetooth_frame, text="Please ensure your
Bluetooth device is connected.",
font=("Arial", 16), bg="white", fg="black" )
bluetooth_instruction_label.pack(pady=10)
bluetooth_back_button = tk.Button(
bluetooth_frame, text="Back",
font=("Arial", 16), bg="#FF294F",
fg="white", command=lambda:
print_options_frame.pack(pady=20) )
bluetooth_back_button.pack(pady=20) #
Wi-Fi Frame wifi_frame = tk.Frame(root,
bg="white") # Inner frame for centering
content wifi_inner_frame =
tk.Frame(wifi_frame, bg="white")
wifi_inner_frame.pack(expand=True,
fill="both", pady=(50, 0)) # Add padding to
push downward wifi_frame_label =
tk.Label( wifi_inner_frame, text="Wi-Fi
Printing Options", font=("Bebas Neue",
50), bg="white", fg="black" )
wifi_frame_label.pack(pady=20)
wifi_instruction_label = tk.Label(
wifi_inner_frame, text="Connect to Wi-Fi
and scan the QR code:", font=("Arial", 36),
bg="white", fg="black" )
wifi_instruction_label.pack(pady=10) #
Display QR Code qr_image =
Image.open("qr_code.png").resize((500,
500)) qr_photo =
ImageTk.PhotoImage(qr_image) qr_label
= tk.Label(wifi_inner_frame,
image=qr_photo, bg="white")
qr_label.pack(pady=10) # Website link
under the QR code website_link_label =
tk.Label( wifi_inner_frame, text="or visit
https://paperazzi.onrender.com",
font=("Arial", 14), bg="white", fg="blue" )
website_link_label.pack(pady=5) # Back
button for Wi-Fi frame wifi_back_button =
tk.Button( wifi_inner_frame, text="Back",
font=("Arial", 16), bg="#FF294F",
fg="white", command=lambda:
print_options_frame.pack(pady=20) )
wifi_back_button.pack(pady=20) # Start
the GUI root.mainloop()
#Frame 4 import tkinter as tk from PIL
import Image, ImageTk import qrcode #
Function to generate QR code for the
website def generate_qr_code(url,
file_path="qr_code.png"): qr =
qrcode.QRCode( version=1,
error_correction=qrcode.constants.ERRO
R_CORRECT_H, box_size=15, # Larger
size for the QR code border=2, )
qr.add_data(url) qr.make(fit=True)
qr_image =
qr.make_image(fill_color="black",
back_color="white")
qr_image.save(file_path)
generate_qr_code("https://paperazzi.onre
nder.com/") # Function to go to Wi-Fi
frame def go_to_wifi():
main_frame.pack_forget()
print_options_frame.pack_forget()
wifi_frame.pack(pady=20) # Function to go
back to the print options frame def
go_back_to_print_options():
wifi_frame.pack_forget()
print_options_frame.pack(pady=20) #
Function to go to Bluetooth (if needed) def
go_to_bluetooth():
messagebox.showinfo("Bluetooth",
"Bluetooth selected. Proceeding with
Bluetooth printing...") def
close_application(event=None): root.quit()
```

```
# Create the main window root = tk.Tk() root.title("Paperazzi") root.config(bg="white") root.attributes('-fullscreen', True) root.bind("<Escape>", close_application) # Main frame with logo and Start Printing button main_frame = tk.Frame(root, bg="white") main_frame.pack() logo_frame = tk.Frame(main_frame, bg="white") logo_frame.pack(pady=(50, 10)) logo_image = Image.open("logo.jpg") logo_photo = ImageTk.PhotoImage(logo_image) logo_label = tk.Label(logo_frame, image=logo_photo, bg="white") logo_label.pack() start_button_frame = tk.Frame(main_frame, bg="white") start_button_frame.pack(pady=20) start_button = tk.Button( start_button_frame, text="Start Printing", font=("Bebas Neue", 20), command=lambda: start_printing() ) start_button.pack(pady=10) def start_printing(): main_frame.pack_forget() print_options_frame.pack(pady=20) # Print Options frame print_options_frame = tk.Frame(root, bg="white") print_options_label = tk.Label( print_options_frame, text="How would you like to print?", font=("Bebas Neue", 30), bg="white", fg="black" ) print_options_label.pack(pady=20) # Bluetooth Button bluetooth_image = Image.open("bluetooth.png").resize((120, 120)) bluetooth_photo = ImageTk.PhotoImage(bluetooth_image) bluetooth_label = tk.Label(print_options_frame, image=bluetooth_photo, bg="white", cursor="hand2") bluetooth_label.pack(side="left", padx=40) bluetooth_label.bind("<Button-1>", lambda e: go_to_bluetooth()) # Wi-Fi Button wifi_image = Image.open("wifi.png").resize((120, 120)) wifi_photo = ImageTk.PhotoImage(wifi_image) wifi_label = tk.Label(print_options_frame, image=wifi_photo, bg="white", cursor="hand2") wifi_label.pack(side="right", padx=40) wifi_label.bind("<Button-1>", lambda e: go_to_wifi()) # Wi-Fi Frame to Display QR Code and Instructions wifi_frame = tk.Frame(root, bg="white") # Instruction Label Above the QR Code wifi_instructions = tk.Label( wifi_frame, text="Connect to Wi-Fi and scan the QR code", font=("Bebas Neue", 24), # Larger font size bg="white", fg="black", wraplength=800, justify="center" ) wifi_instructions.pack(pady=20) # Load the larger QR code image qr_image = Image.open("qr_code.png").resize((300, 300)) # Larger QR code size qr_photo = ImageTk.PhotoImage(qr_image) qr_label = tk.Label(wifi_frame, image=qr_photo, bg="white") qr_label.pack(pady=10) # Website link text under the QR code website_link_label = tk.Label( wifi_frame, text="or visit https://paperazzi.onrender.com/", font=("Bebas Neue", 18), # Slightly smaller font size bg="white", fg="#007BFF", # Blue color for the link cursor="hand2" ) website_link_label.pack() # Open the website when clicking the text website_link_label.bind("<Button-1>", lambda e: open_website()) # Back Button back_button = tk.Button( wifi_frame, text="Back", font=("Bebas Neue", 20), command=go_back_to_print_options, bg="#FF294F", fg="white" ) back_button.pack(pady=20) # Function to open the website def open_website(): import webbrowser webbrowser.open("https://paperazzi.onrender.com/") root.mainloop()

# App.py from flask import Flask, render_template, request, redirect, url_for import os import mysql.connector import fitz # PyMuPDF library for PDF processing from docx import Document # For handling Word documents from flask_socketio import SocketIO, emit app = Flask(__name__) # Initialize SocketIO socketio = SocketIO(app) # Database configuration from environment variables db_config = { 'host': os.getenv('DB_HOST'), 'user': os.getenv('DB_USER'), 'password': os.getenv('DB_PASSWORD'), 'database': os.getenv('DB_NAME') } # Allowed file
```

```
extensions ALLOWED_EXTENSIONS = {'doc', 'docx', 'pdf'} # Function to check allowed file extensions def allowed_file(filename): return '.' in filename and filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS # Function to calculate total pages for supported file types def get_total_pages(file_path): try: # Handle PDF files if file_path.lower().endswith('.pdf'): with fitz.open(file_path) as pdf: if pdf.is_encrypted: pdf.authenticate('') # Try opening with an empty password return pdf.page_count # Handle DOCX and DOC files elif file_path.lower().endswith(('.doc', '.docx')): doc = Document(file_path) total_characters = sum(len(p.text) for p in doc.paragraphs) average_chars_per_page = 1500 # Assumes ~1500 characters per page estimated_pages = max(1, total_characters // average_chars_per_page) return estimated_pages # Unsupported file type return "N/A" except Exception as e: print(f"Error processing {file_path}: {e}") return None # Function to get or reconnect the MySQL connection and cursor def get_db_connection(): if not hasattr(app, 'db_connection') or not app.db_connection.is_connected(): app.db_connection = mysql.connector.connect(**db_config) app.db_cursor = app.db_connection.cursor(dictionary=True) return app.db_connection, app.db_cursor # Make sure uploads directory exists if not os.path.exists('uploads'): os.makedirs('uploads') @app.route('/') def index(): return render_template('index.html') @app.route('/upload', methods=['POST']) def upload_file(): if 'file' not in request.files: return redirect(request.url) file = request.files['file'] if file and allowed_file(file.filename): filename = file.filename file_path = os.path.join('uploads', filename) # Save the file first file.save(file_path) # Calculate file size after saving file_size = os.path.getsize(file_path) # Calculate total pages for any supported file type total_pages = get_total_pages(file_path) if total_pages is None: os.remove(file_path) return f"Error processing the file: {filename}", 500 try: # Read file binary data after saving with open(file_path, 'rb') as f: file_data = f.read() # Get the MySQL connection and cursor db_connection, db_cursor = get_db_connection() # Insert the file into MySQL database query = """ INSERT INTO print_jobs (document_name, document_size, file_data, status, total_pages) VALUES (%s, %s, %s, %s, %s) """ values = (filename, file_size, file_data, 'uploaded', total_pages) # Mark as uploaded db_cursor.execute(query, values) db_connection.commit() print(f"File {filename} inserted into DB with total pages: {total_pages}") socketio.emit('new_print_job', {'status': 'new_job', 'document_name': filename}) return render_template('uploaded_file.html', filename=filename, file_size=file_size, total_pages=total_pages) except mysql.connector.Error as err: print(f"Error: {err}") return f"Error uploading file to the database: {err}", 500 finally: # Clean up the file after processing os.remove(file_path) return 'Invalid file type, please upload a valid file.' @app.teardown_appcontext def close_db_connection(exception): if hasattr(app, 'db_connection') and app.db_connection.is_connected(): app.db_cursor.close() app.db_connection.close() if __name__ == '__main__': socketio.run(app, host='0.0.0.0', port=5000)
#Uploaded_file <!DOCTYPE html> <html lang="en"> <head> <meta charset="UTF-8"> <meta name="viewport" content="width=device-width, initial-scale=1.0"> <title>Print Job Submitted</title> <style> body { font-family: Arial, sans-serif; background-color: #f0f0f0; margin: 0; padding: 0; display: flex; justify-content: center; align-items: center; height: 100vh; } .container { width: 80%; max-width: 600px; background-color: white; padding: 30px; border-radius: 8px; box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1); text-align: center; } h1 { color: #FA2A55; }
```

```
h2 { color: #333; } ul { list-style: none; padding: 0; } li { margin: 10px 0; font-size: 18px; color: #555; } a { text-decoration: none; color: white; background-color: #FA2A55; padding: 10px 20px; border-radius: 5px; transition: background-color 0.3s; } a:hover { background-color: #e0284b; } .message { margin-top: 20px; font-size: 18px; color: #555; } .back-link { margin-top: 20px; display: inline-block; padding: 8px 15px; background-color: #2196F3; color: white; border-radius: 5px; text-decoration: none; } .back-link:hover { background-color: #1976D2; } .logo { width: 150px; margin-bottom: 20px; } </style> </head> <body> <div class="container"> <!-- Logo --> <img src="{{ url_for('static', filename='logo.jpg') }}" alt="Logo" class="logo"> <h1>Print Job Submitted</h1> <p class="message"> Your file <strong>{{ filename }}</strong> has been successfully uploaded with <strong>{{ total_pages if total_pages is not none else 'N/A' }}</strong> pages. Please wait for your printing preference on the screen. </p> <p><a class="back-link" href="/">Go back to the home page</a></p> </div> </body> </html>
# Index.html <!DOCTYPE html> <html lang="en"> <head> <meta charset="UTF-8"> <meta name="viewport" content="width=device-width, initial-scale=1.0"> <title>Paperazzi: Upload your file</title> <style> body { font-family: Arial, sans-serif; margin: 0; padding: 0; background-color: #f4f4f9; display: flex; justify-content: center; align-items: center; height: 100vh; } .container { width: 100%; max-width: 600px; text-align: center; background-color: white; padding: 20px; box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1); border-radius: 8px; } header { margin-bottom: 20px; } header img { max-width: 150px; height: auto; } h1 { color: #333; font-size: 24px; margin-bottom: 15px; } form { display: flex; flex-direction: column; align-items: center; } label { font-size: 18px; margin-bottom: 10px; color: #555; } input[type="file"] { margin-bottom: 15px; padding: 10px; border: 1px solid #ddd; border-radius: 5px; background-color: #fafafa; } button { padding: 10px 20px; background-color: #007bff; color: white; border: none; border-radius: 5px; font-size: 16px; cursor: pointer; transition: background-color 0.3s; } button:hover { background-color: #0056b3; } </style> </head> <body> <div class="container"> <header> <img src="{{ url_for('static', filename='logo.jpg') }}" alt="Paperazzi Logo"> </header> <h1>Upload a file for printing</h1> <form action="/upload" method="POST" enctype="multipart/form-data"> <label for="file">Choose file:</label> <input type="file" name="file" required> <button type="submit">Upload</button> </form> </div> </body> </html>
#Frame 5 import tkinter as tk from PIL import Image, ImageTk import subprocess import RPi.GPIO as GPIO import threading # Needed for threading in coin detection import time # Required for sleep functions class PrintingSummaryApp: def __init__(self, root, summary_details, coin_details): self.root = root self.root.title("Printing Summary") self.price_details_frame = None # Initialize the price_details_frame here # Make the window fullscreen self.root.attributes('-fullscreen', True) self.root.configure(bg='white') # Set background color to white # Bind Escape key to close the app self.root.bind("<Escape>", self.close_app) # Extract summary details self.page_from = summary_details["page_from"] self.page_to = summary_details["page_to"] self.copies = summary_details["copies"] self.is_colored = summary_details["is_colored"] self.total_price = summary_details["total_price"] # Extract coin details self.coin_count = coin_details["coin_count"] self.payment_goal = coin_details["payment_goal"]  self.coin_var = tk.StringVar(value=f"{self.coin_count:.2f} PHP") # Printer details self.printer_name = coin_details["printer_name"] self.file_to_print = coin_details["file_to_print"] # Create the layout self.create_widgets() def create_widgets(self): # Create main
```

```
frames for the left (logo) and right (summary) left_frame = tk.Frame(self.root, bg='white') left_frame.grid(row=0, column=0, padx=10, pady=(50, 10), sticky="nsew") right_frame = tk.Frame(self.root, bg='white') right_frame.grid(row=0, column=1, padx=10, pady=(50, 10), sticky="nsew") # Configure grid to expand proportionally self.root.grid_rowconfigure(0, weight=1) self.root.grid_columnconfigure(0, weight=1) self.root.grid_columnconfigure(1, weight=3) # Display the logo on the left side self.display_logo(left_frame) # Display the summary details on the right side self.display_summary(right_frame) def display_logo(self, frame): try: # Get screen dimensions for dynamic resizing screen_width = self.root.winfo_screenwidth() screen_height = self.root.winfo_screenheight() # Resize logo based on screen size logo_image = Image.open("logo.jpg") new_width = int(screen_width * 0.5) new_height = int(screen_height * 0.😎 logo_image_resized = logo_image.resize((new_width, new_height), Image.Resampling.LANCZOS)  logo_photo = ImageTk.PhotoImage(logo_image_resized ) logo_label = tk.Label(frame, image=logo_photo, bg='white') logo_label.image = logo_photo # Keep a reference to avoid garbage collection logo_label.grid(row=0, column=0, pady=(20, 0)) except Exception as e: print(f"Error loading logo: {e}") def display_summary(self, frame): # Use dynamic font size based on screen size screen_width = self.root.winfo_screenwidth() font_size = int(screen_width * 0.02) # Create the "Summary" label at the top summary_label = tk.Label(frame, text="SUMMARY", font=("Arial", font_size, "bold"), fg="black", bg='white') summary_label.grid(row=0, column=0, pady=15, columnspan=2) # Display summary details tk.Label(frame, text=f"Pages: {self.page_from} to {self.page_to}", font=("Arial", font_size), bg='white').grid(row=1, column=0, pady=5, sticky="w") tk.Label(frame, text=f"Copies: {self.copies}", font=("Arial", font_size), bg='white').grid(row=2, column=0, pady=5, sticky="w") tk.Label(frame, text=f"Print Type: {'Colored' if self.is_colored else 'Grayscale'}", font=("Arial", font_size), bg='white').grid(row=3, column=0, pady=5, sticky="w") # Add more vertical space between SUMMARY and Total Price/Coins Accepted self.display_total_price_section(frame, pady_top=50) self.display_coins_accepted_section(fram e, pady_top=20) # Create the "Price Details" section (toggle) self.price_details_label = tk.Label(frame, text="Price Details", font=("Arial", 14, "underline"), fg="blue", bg='white', cursor="hand2") self.price_details_label.grid(row=4, column=0, columnspan=2, pady=5, sticky="w") self.price_details_label.bind("<Button-1>", self.toggle_price_details) def display_total_price_section(self, frame, pady_top=10): """Create the Total Price section with adjustable margin on top.""" total_price_frame = tk.Frame(frame, bg='white', bd=2, relief="solid", width=500, height=100) total_price_frame.grid(row=6, column=0, columnspan=2, pady=(pady_top, 10), sticky="w", padx=10) # TOTAL PRICE label total_price_label = tk.Label(total_price_frame, text="TOTAL PRICE:", font=("Arial", 20), bg='white') total_price_label.grid(row=0, column=0, padx=15, pady=15, sticky="w") # Total price value (this will be updated dynamically) self.total_price_value_label = tk.Label(total_price_frame, text=f"{self.total_price:.2f} PHP", font=("Arial", 26, "bold"), fg="red", bg='white') self.total_price_value_label.grid(row=0, column=1, padx=15, pady=15, sticky="e") def display_coins_accepted_section(self, frame, pady_top=10): """Create the COINS ACCEPTED section with adjustable margin on top.""" coins_frame =
```

```
tk.Frame(frame, bg='white', bd=2, relief="solid", width=500, height=100) coins_frame.grid(row=7, column=0, columnspan=2, pady=(pady_top, 10), sticky="w", padx=10) # COINS ACCEPTED label coins_label = tk.Label(coins_frame, text="COIN/S ACCEPTED:", font=("Arial", 20), bg='white') coins_label.grid(row=0, column=0, padx=15, pady=15, sticky="w") # Coins accepted value self.coins_value_label = tk.Label(coins_frame, text="00.00 PHP", font=("Arial", 26, "bold"), fg="red", bg='white') self.coins_value_label.grid(row=0, column=1, padx=15, pady=15, sticky="e") def toggle_price_details(self, event): if self.price_details_frame and self.price_details_frame.winfo_ismapped() : # If price details are visible, hide them self.price_details_frame.grid_forget() else: # If price details are not visible, show them if self.price_details_frame is None: # Create the price details frame only if it doesn't exist self.create_price_details_frame() # Place the price details frame on the window self.price_details_frame.grid(row=5, column=0, columnspan=2, pady=5, sticky="w") # Place under "Print Type:" def create_price_details_frame(self): """Create the frame that displays price details.""" self.price_details_frame = tk.Frame(self.price_details_label.master, bg='white') # Attach to the same parent as the "Print Type:" label # Calculate the price breakdown price_per_page = 5 if self.is_colored else 3 num_pages = self.page_to - self.page_from + 1 total_price = num_pages * self.copies * price_per_page self.total_price = total_price # Update total price for display # Price details content details_text = (f"Price per page: {'5 pesos (Colored)' if self.is_colored else '3 pesos (Grayscale)'}\n" f"Number of Pages: {num_pages}\n" f"Number of Copies: {self.copies}\n" f"Total Calculation: {price_per_page} pesos * {num_pages} pages * {self.copies} copies = {total_price} pesos") # Use a smaller, italicized font tk.Label(self.price_details_frame, text=details_text, font=("Arial", 12, "italic"), bg='white', justify="left").grid(row=0, column=0, pady=5, sticky="w") def update_coin_count(self, coin_count): """Update the coin count and trigger print if goal is reached.""" self.coin_count = coin_count self.coin_var.set(f"{self.coin_count:.2f} PHP") self.coins_value_label.config(text=f"{self.coin_count:.2f} PHP") if self.coin_count >= self.payment_goal: print("Payment goal reached! Starting print job...") self.trigger_print_job() def trigger_print_job(self): """Trigger the print job in a separate thread.""" def print_job(): try: subprocess.run(["lp", "-d", self.printer_name, self.file_to_print], check=True) print("Print job sent successfully!") except subprocess.CalledProcessError as e: print(f"Print job failed: {e}") self.reset_payment() # Run print job in a separate thread to avoid blocking the GUI threading.Thread(target=print_job).start() def reset_payment(self): self.coin_count = 0 self.coin_var.set(f"{self.coin_count:.2f} PHP") def close_app(self, event=None): """Close the application.""" self.root.quit() def coin_detection(app): COIN_PIN = 4 # GPIO pin connected to the coin mechanism GPIO.setmode(GPIO.BCM) GPIO.setup(COIN_PIN, GPIO.IN, pull_up_down=GPIO.PUD_UP) # Define pulse_count in the coin_detection function's scope pulse_count = 0 pulse_window_time = 1 # Time to count pulses pulse_to_peso = {1: 1, 5: 5, 10: 10, 20: 20} # Pulses to pesos mapping def coin_inserted(channel): nonlocal pulse_count # Use nonlocal to modify pulse_count from the enclosing function pulse_count += 1 print(f"Coin inserted, pulse count: {pulse_count}") # Debugging line def process_coin(): time.sleep(pulse_window_time) coin_value = pulse_to_peso.get(pulse_count, 0) print(f"Coin value detected: {coin_value} PHP") # Debugging line if coin_value > 0: new_coin_count = app.coin_count +
```

```
coin_value print(f"New coin count: {new_coin_count}") # Debugging line app.root.after(0, app.update_coin_count, new_coin_count) # Safely update coin count pulse_count = 0 # Reset pulse count after processing threading.Thread(target=process_coin).start() GPIO.add_event_detect(COIN_PIN, GPIO.RISING, callback=coin_inserted, bouncetime=50) try: while True: time.sleep(1) except KeyboardInterrupt: GPIO.cleanup() if _name_ == "_main_": root = tk.Tk() summary_details = { "page_from": 1, "page_to": 5, "copies": 2, "is_colored": True, "total_price": 50.00 } coin_details = { "coin_count": 0, "payment_goal": 50.00, "printer_name": "Canon_TS200_series_USB", # Replace with your printer name "file_to_print": "/home/pi/Desktop/test_print.txt" # Path to file to print } app = PrintingSummaryApp(root, summary_details, coin_details) threading.Thread(target=coin_detection, args=(app,), daemon=True).start() root.mainloop()
```

Appendix 2. Evaluation Instrument

General Direction: Please accomplish this questionnaire very carefully and honestly. Please rest assured that any information that you supply will be treated with the greatest confidentiality and anonymity.

Name (optional): _____Company: _____
Sex: ____Male ____Female Sector: _____
Industry others (please specify) _____

Kindly put a check mark at the right of the software evaluation characteristics under the proper heading to indicate your assessment on the software product based on the specified factor.

Numerical Rating	Equivalent
5	Excellent
4	Very Good
3	Good
2	Fair
1	Poor

SOFTWARE EVALAUTION CHARACTERISTICS	Excellent	Very Good	Good	Fair	Poor
FUNCTIONALITY					
Functional completeness. Degree to which the set of functions covers all the specified tasks and user objectives.					
Functional correctness. Degree to which a product or system provides the correct results with the needed degree of precision.					
Functional appropriateness. Degree to which the functions facilitate the accomplishment of specified tasks and objectives					
RELIABILITY					
Maturity. Degree to which a system, product or component meets needs for reliability under normal operation.					
Availability. Degree to which a system, product or component is operational and accessible when required for use.					
Fault tolerance. Degree to which a system, product or component operates as intended despite the presence of hardware or software faults.					
Recoverability. Degree to which, in the event of an interruption or a failure, a product or system can recover the data directly affected and re-establish the desired state of the system.					
USABILITY					
Appropriateness recognizability. Degree to which users can recognize whether a product or system is appropriate for their needs.					

Learnability. degree to which a product or system can be used by specified users to achieve specified goals of learning to use the product or system with effectiveness, efficiency, freedom from risk and satisfaction in a specified context of use.					
Operability. Degree to which a product or system has attributes that make it easy to operate and control.					
User error protection. Degree to which a system protects users against making errors					
User interface aesthetics. Degree to which a user interface enables pleasing and satisfying interaction for the user.					
Accessibility. Degree to which a product or system can be used by people with the widest range of characteristics and capabilities to achieve a specified goal in a specified context of use.					
PERFORMANCE EFFICIENCY					
Time Behaviour. Degree to which the response and processing times and throughput rates of a product or system, when performing its functions, meet requirements.					
Resource utilization. Degree to which the amounts and types of resources used by a product or system, when performing its functions, meet requirements.					
Capacity. Degree to which the maximum limits of a product or system parameter meet requirements.					
COMPATIBILITY					
Co-existence. Degree to which a product can perform its required functions efficiently while sharing a common environment and resources with other products, without detrimental impact on any other product.					
Interoperability. Degree to which two or more systems, products or components can exchange information and use the information that has been exchanged.					
SECURITY					
Confidentiality. Degree to which a product or system ensures that data are accessible only to those authorized to have access.					
Integrity. Degree to which a system, product or component prevents unauthorized access to, or modification of, computer programs or data.					
Non-repudiation. Degree to which actions or events can be proven to have taken place, so that the events or actions cannot be repudiated later.					
Accountability. Degree to which the actions of an entity can be traced uniquely to the entity.					

Authenticity. Degree to which the identity of a subject or resource can be proved to be the one claimed.					
MAINTAINABILITY					
Modularity. Degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components.					
Reusability. Degree to which an asset can be used in more than one system, or in building other assets.					
Analysability. Degree of effectiveness and efficiency with which it is possible to assess the impact on a product or system of an intended change to one or more of its parts, or to diagnose a product for deficiencies or causes of failures, or to identify parts to be modified.					
Modifiability. Degree to which a product or system can be effectively and efficiently modified without introducing defects or degrading existing product quality.					
Testability. Degree of effectiveness and efficiency with which test criteria can be established for a system, product or component and tests can be performed to determine whether those criteria have been met.					
PORTABILITY					
Adaptability. Degree to which a product or system can effectively and efficiently be adapted for different or evolving hardware, software or other operational or usage environments.					
Installability. Degree of effectiveness and efficiency with which a product or system can be successfully installed and/or uninstalled in a specified environment.					
Replaceability. Degree to which a product can replace another specified software product for the same purpose in the same environment.					

Appendix 3. Sample Input/Output/Reports

Appendix Table 1. Sample Input/Output/Reports for User

Input	Output	Remark
User sends a file and selects page range	Machine displays “Page range selected: 1-5” and calculates the total cost.	Notify the user of the cost before payment.
User inserts exact coins	Machine displays “Payment accepted. Printing in progress.”	Ensure coins are inserted properly.
User inserts insufficient coins	Machine displays “Insufficient payment. Please add the remaining amount.”	User must add exact coins to proceed.

Appendix Table 2. Sample Input/Output/Reports for Client/Admin

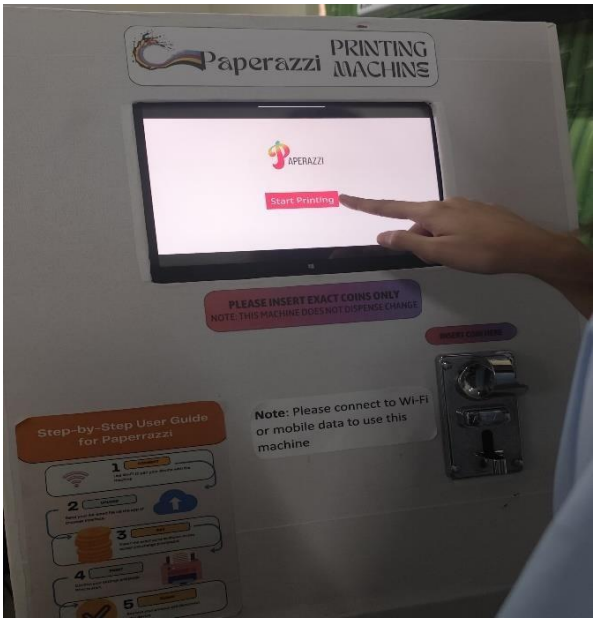
Input	Output	Remark
Admin logs into the online portal	Dashboard displays total sales, transaction logs, and machine status.	Admin should use correct credentials to access the portal.
Admin requests sales report for a specific date	System generates and downloads a detailed sales report in PDF.	Reports include coin count and usage statistics.

Appendix 4. User's Guide

USER MANUAL OF PAPEREAZZ

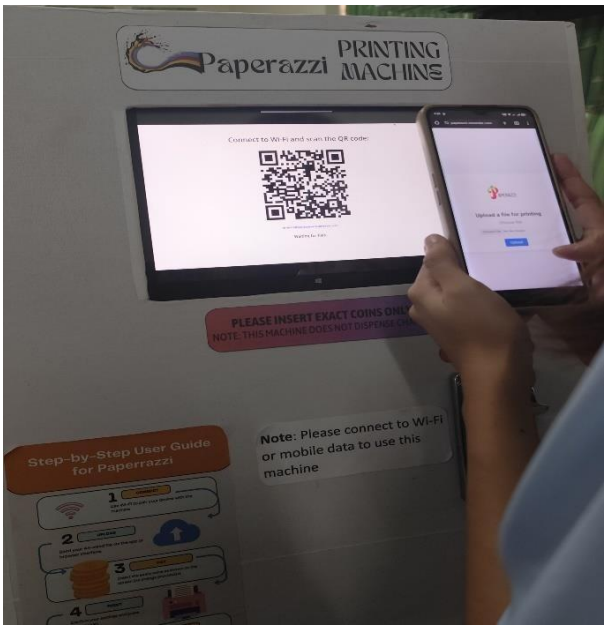
HARDWARE MANUAL

1. Enable Wi-Fi on your phone and connect to the machine.

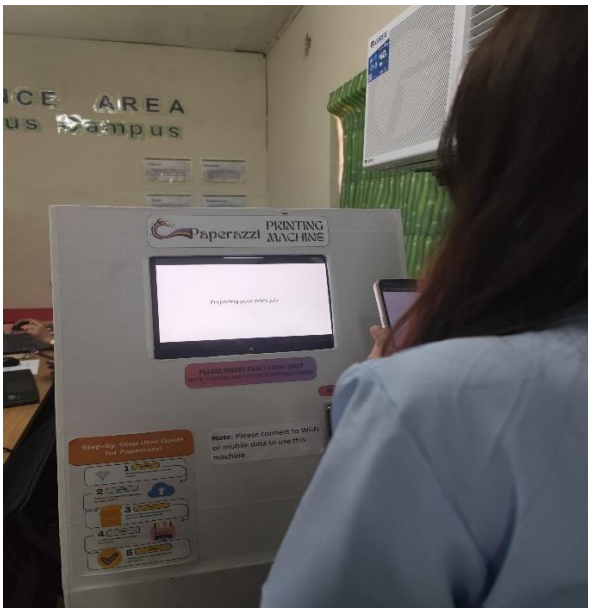


2. Scan the QR code displayed on the screen.

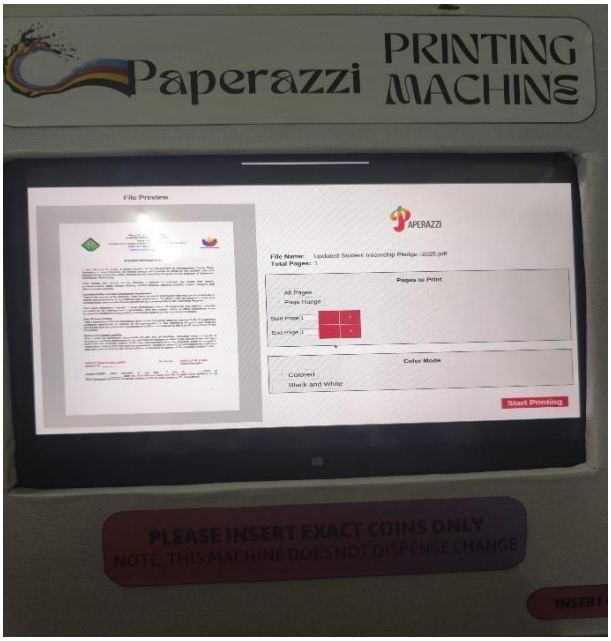
3. After scanning, the upload page will appear.



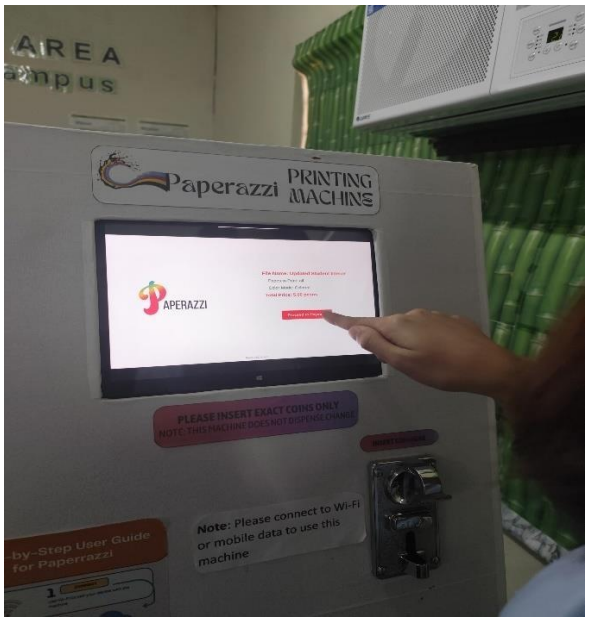
4. Upload the File



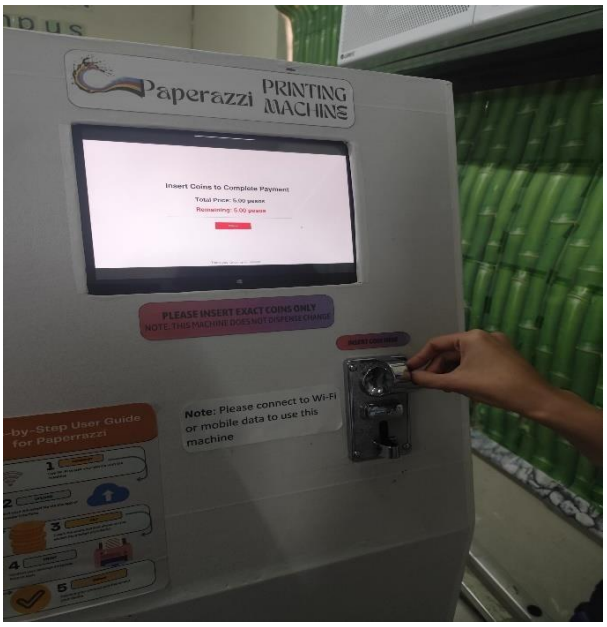
5. File preview before printing. The system displays a preview of the uploaded document, allowing the user to verify the file before confirming the print job.



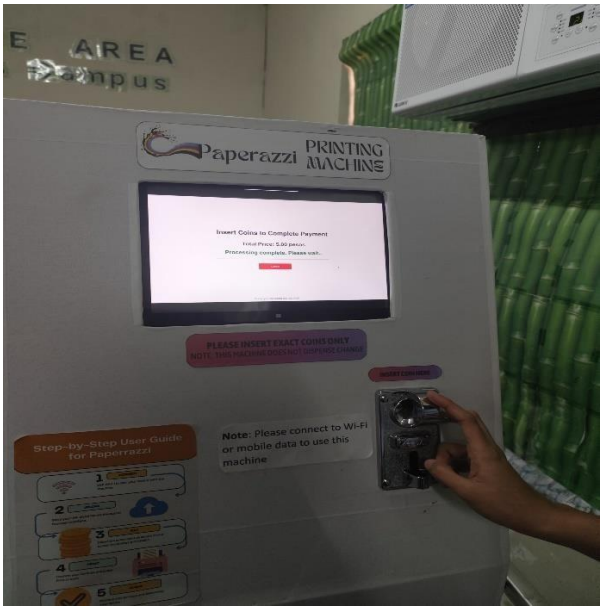
6. The user is prompted to proceed with payment after ensuring the document is correct.



7. The user inserts the exact coin amount into the machine’s coin slot for payment, as the machine does not provide change.



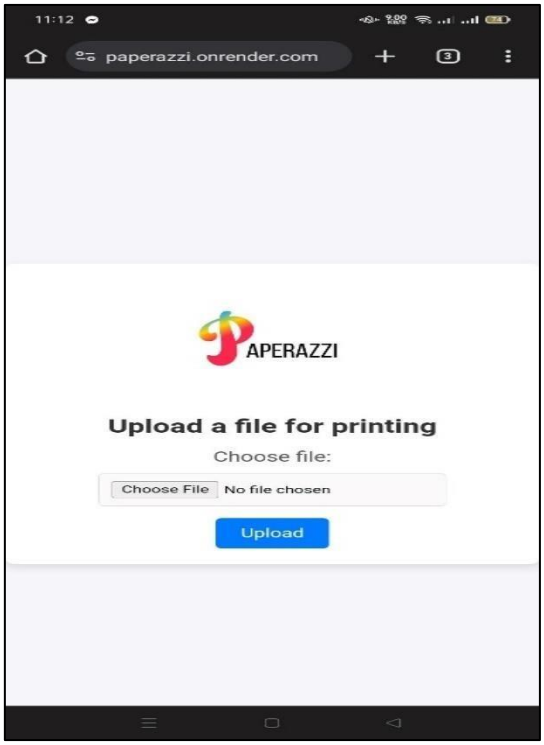
8. The machine verifies the inserted coins and processes the payment before initiating the print job.



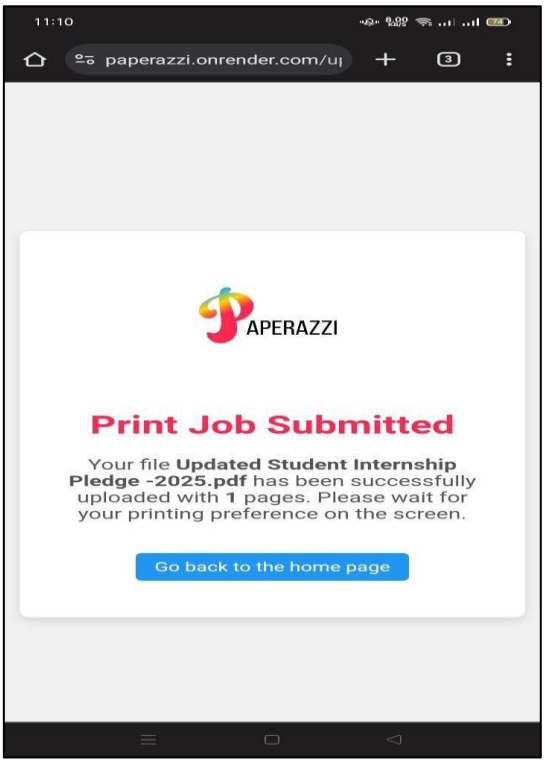
9. The user collects the printed A4-sized document from the output tray located at the buttom of the vending printing machine.

SOFTWARE MANUAL

1. This screen allows users to upload their documents for printing. Click the "Choose File" button to select a file from your device, then press "Upload" to submit it.

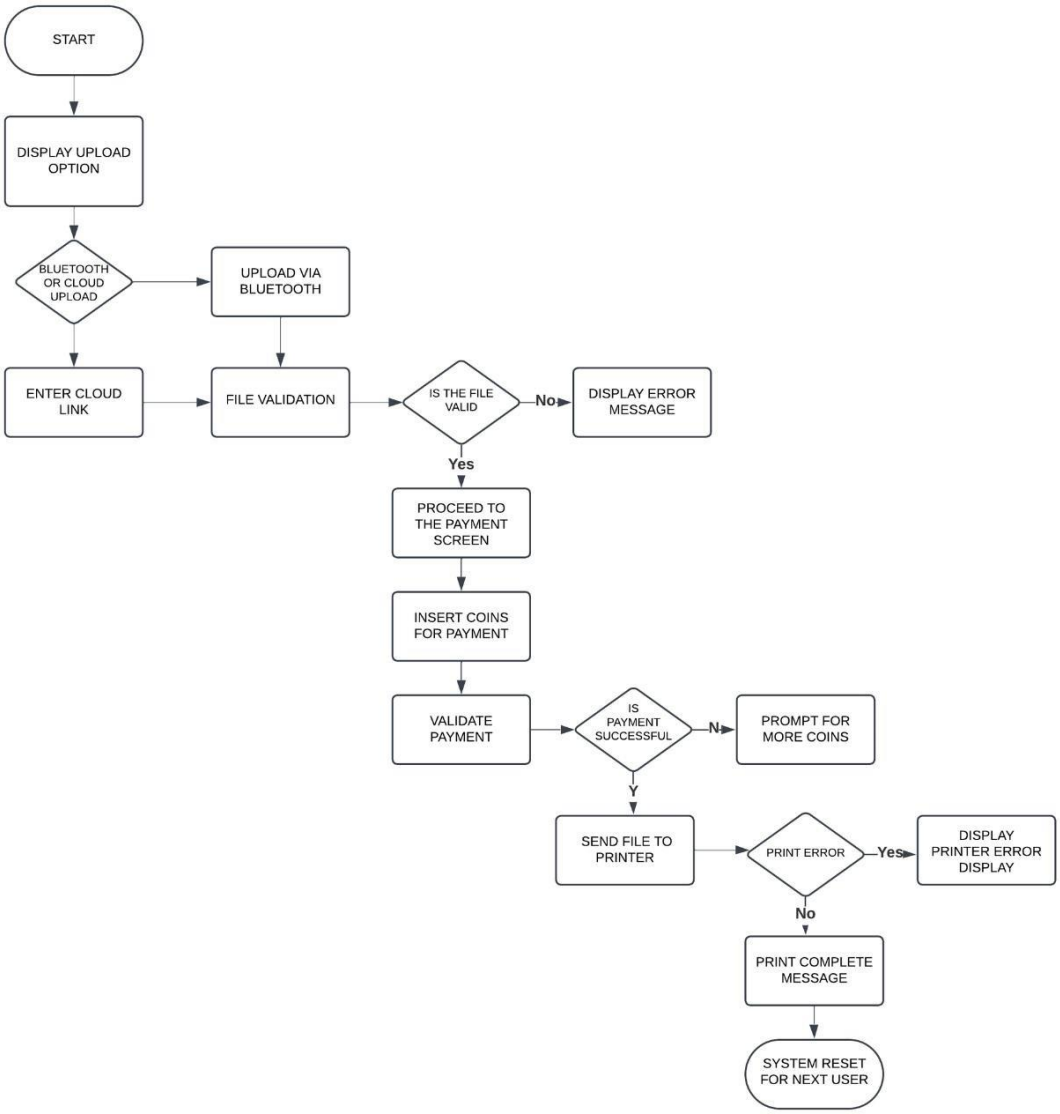


2. After successfully uploading a file, this screen confirms that the document has been submitted for printing. The file name and page count are displayed. Users should wait for the next step to select printing preferences.

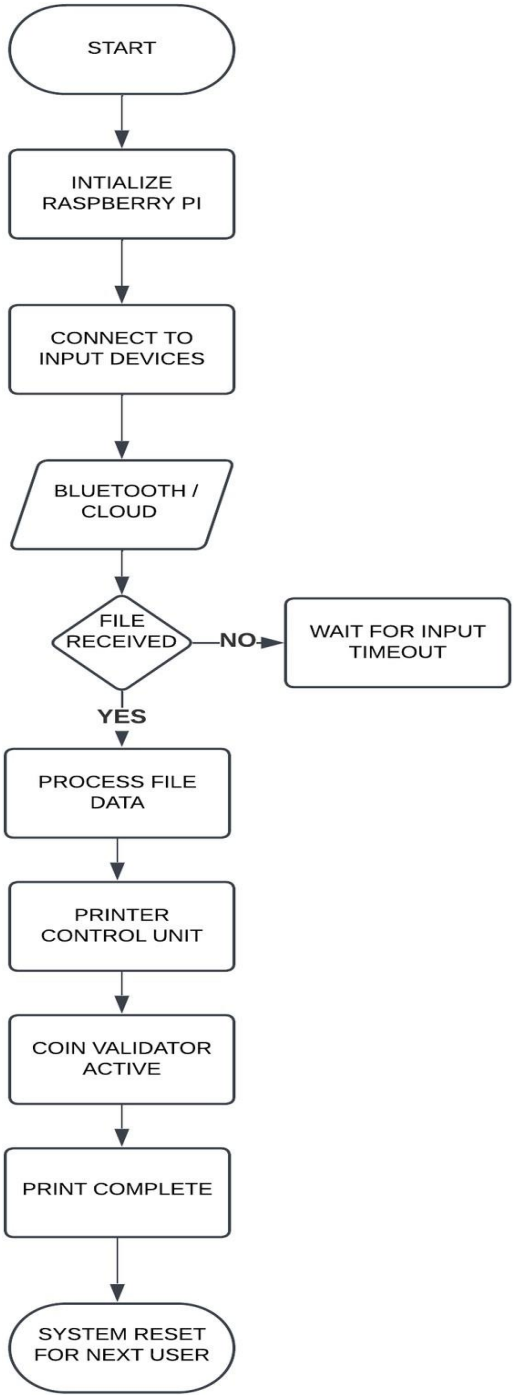


Appendix 5. Process /Data/Information/Flow

Appendix Figure 1. Context Diagram



Appendix Figure 2. System Software Flowchart

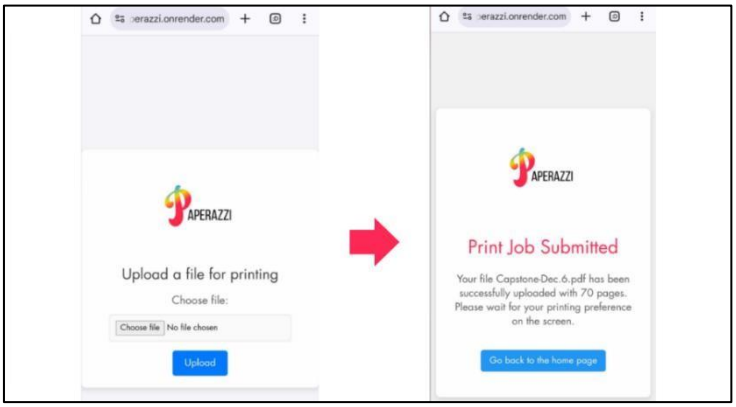


Appendix Figure 3. System Hardware Flowchart

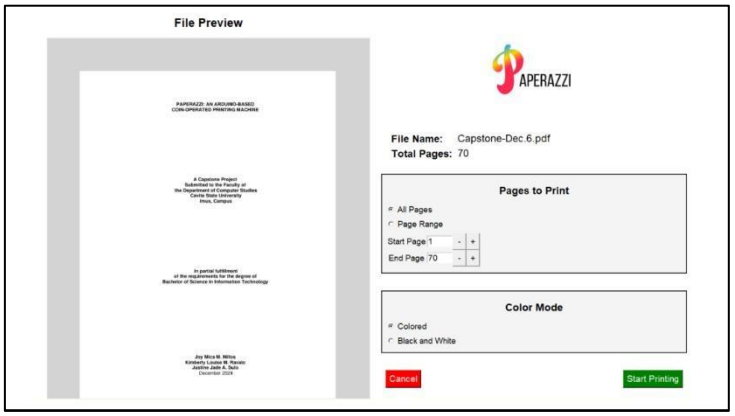
Appendix 6. Screen Layout

Appendix Figure 4. Starting Process

Appendix Figure 5. QR Scanning

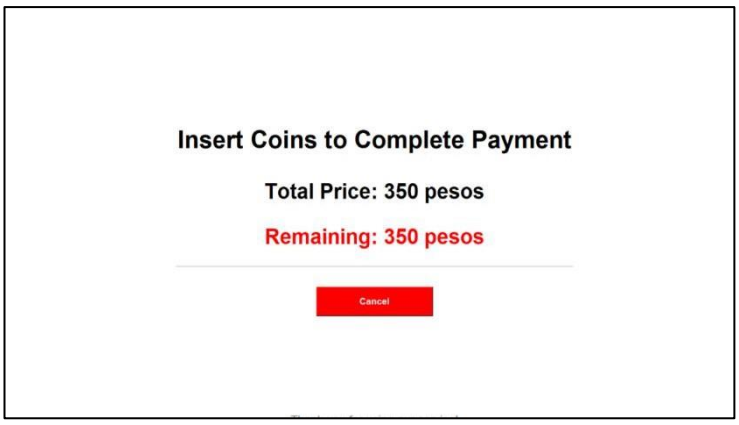


Appendix Figure 6 Uploading the File



Appendix Figure 7. File Preview

Appendix Figure 8. Process Page



Appendix Figure 9 Summary of the Printing Paper

Appendix 7. Gantt Chart

Republic of the Philippines
CAVITE STATE UNIVERSITY
Imus Campus
Cavite Civic Center Palico IV, Imus, Cavite
(046) 471-66-07 / (046) 471-67-70/ (046) 686-2349
www.cvsu.edu.ph

DEPARTMENT OF COMPUTER STUDIES

(GANTT CHART)

Name of Researcher(s): _____

Degree or Course : _____
Title of the Study : _____

ACTIVITY	2023									2024	
	Mar	Apr	May	Jun	Jul	Sept	Oct	Nov	Dec	Jan	Feb
Proposal drafting											
Proposal defense											
System design											
System Development											
System Testing											
System evaluation											
System Implementation											
Manuscript Preparation											
Final oral defense											
Manuscript review											

Appendix 8. Budgetary Requirements

Republic of the Philippines
CAVITE STATE UNIVERSITY
Imus Campus
Cavite Civic Center Palico IV, Imus, Cavite
046) 471-66-07 / (046) 471-67-70/ (046) 686-2349
www.cvsu.edu.ph

BAGONG PILIPINAS

DEPARTMENT OF COMPUTER STUDIES

BUDGETARY REQUIREMENTS

Name of Researcher(s) : _____
_____
_____

Degree or Course : _____
Title of the Study : _____

PARTICULARS	AMOUNT (P)
1. ICHICO All in One Mini Desktop	3,539
2. Raspberry Pi 3B	3,182
3. Universal Coin Slot	399
4. Buck Converter 12v to 5v	100
5. Jumper Wires	300
6. Machine's Case w/ Labor	1,600
7. Other Wirings	100
8. Connectors and Cables	500
9. FlashDrive 16gb & 32gb	500
10. A4 Size Bond Paper and Wall Sticker	400
TOTAL	**10,620**

Appendix 9. Request for Proposal Defense

Republic of the Philippines
CAVITE STATE UNIVERSITY
Imus Campus
Cavite Civic Center Palico IV, Imus, Cavite
(046) 471-66-07 / (046) 471-67-70/ (046) 686-2349
www.cvsu.edu.ph

DEPARTMENT OF COMPUTER STUDIES

REQUEST FOR PROPOSAL DEFENSE

Name of Researcher(s): _____
_____
_____
_____

Degree or Course : _____
Date of Intended
Proposal Defense : _____
Title of the Study : _____
_____
_____

Endorsed by:

_____ _____
Adviser Technical Critic

Noted:

_____ _____
Department Chairperson Campus Research Coordinator

Approved:

_____
Campus Administrator