

**CARD SWIPE TIME-KEEPING AND ATTENDANCE SYSTEM (CSTAS) FOR
PHILIPPINE SCIENCE HIGH SCHOOL WESTERN VISAYAS**

A Research Paper
Presented to
The Faculty of Philippine Science High School Western Visayas
Bito-on, Jaro, Iloilo City

In Partial Fulfillment
Of the Requirements for
SCIENCE RESEARCH 2

by

Noreen Marian C. Bautista
China May Y. Molina
Patrick S. Quezon
Fourth Year - Graviton

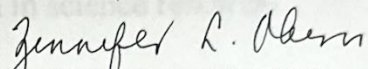
February 2006

APPROVAL SHEET

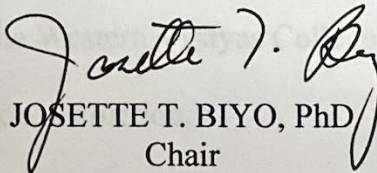
This Research Paper Hereto Entitled:

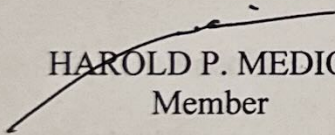
“Card Swipe Time Keeping and Attendance System for Philippine Science High School Western Visayas”

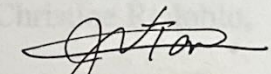
Prepared and submitted by Bautista, Noreen Marian C., Molina, China May Y., and Quezon, Patrick S. in partial fulfillment of the requirements in Science Research 2, has been examined and is recommended for acceptance and approval.


ZENNIFER L. OBERIO
Science Research 2 Adviser

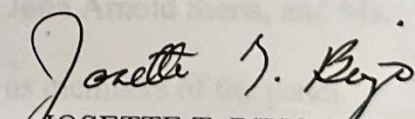
Approved by the committee on oral examination with a grade of PASSED on February 2006.


JOSETTE T. BIYO, PhD
Chair


HAROLD P. MEDIODIA
Member


JOSE VICENTE R. TAN
Member

Accepted in partial fulfillment of the requirements for Science Research 2.


JOSETTE T. BIYO, PhD
OIC, Office of the Director

ACKNOWLEDGEMENT

We express our gratitude to our ever-supportive parents, Rene and Marissa Quezon, Resurrecion and Editha Molina, and Norlito and Nilda Bautista and to our other family members who have supported us in every way.

For our adviser, Ms. Zennifer Oberio and OIC Directress Dr. Josette Biyo, thank you for the guidance you have always shown. Excellence unfolds within us because of you.

For Pfizer Philippines, thank you for believing in us. May you never cease your benevolence to give once in a lifetime opportunities to the youth in science research.

For our industrious and dedicated programming mentor and friend, Mr. Llowen Miraflor of Central Philippine University, we salute your patience with us. Our gratitude also goes to Engr. Calibjo, Dean of the Computer Studies Department in the Central Philippine University.

To Julie Vi Cabrias of the Western Visayas College of Science and Technology for his services with the log box construction.

To the PSHS-WV Computer Science teachers namely, Mrs. Christine Redoblo, Mr. Leo Redoblo and Mr. Vicente Tan, thank you for the valuable information and inspiration.

Thank you also to the PSHS-WV Technology teachers, Mr. Alberto Tanoy and Mr. Lawrence Calambro.

For Mr. Harold Mediodia, Mr. Edgar Babiera, Mr. John Arnold Siena, and Ms. Resa Joyce Yandog, who had been witnesses of our study as members of the panel.

To the one and only Ruchaligra class Batch 2006. You have been the best classmates ever. Thank you to each and every one of you for in a way you have all made this possible for us.

And most of all, this research study is offered to our Almighty Father, without Whom we would accomplish nothing! Thank You, Lord, for the graces, for the trials, for every thing we learned during our journey! We praise Your Name forever!

NOREEN MARIAN C. BAUTISTA
CHINA MAY Y. MOLINA
PATRICK S. QUEZON
Researchers

March 2006

II. REVIEW OF RELATED LITERATURE

A. Chapter Overview	10
B. Time In/Time Out and Identification System	11
C. Bar Coding	13
D. Code 39	16
E. Light Sensors	17
F. Robotics Command System	17
G. Firmware	18
H. Infrared Tower	19
I. Java	19
J. LeJOS	36
K. Java Program	20
L. PC to RCX Communication	21
M. Graphical User Interface	23

III. METHODOLOGY

A. Overview of the Methodology	23
B. Methods	25
C. Programming Tutorial	31

TABLE OF CONTENTS

	PAGE
Approval Sheet	i
Acknowledgement	ii
Abstract	iv
List of Figures	vii
List of Plates	viii
List of Appendices	ix
Chapter	
I. INTRODUCTION	
A. Background of the Study	1
B. Statement of the Problem and Objectives of the Study	3
C. Significance of the Study	4
D. Scope and Limitations of the Study	5
E. Definitions of Terms	7
II. REVIEW OF RELATED LITERATURE	
A. Chapter Overview	10
B. Time In/Time Out and Identification System	11
C. Bar Coding	15
D. Code 39	16
E. Light Sensors	17
F. Robotics Command System	17
G. Firmware	18
H. Infrared Tower	19
I. Java	19
J. LeJOS	20
K. Java Program	20
L. PC to RCX Communication	21
M. Graphical User Interface	22
III. METHODOLOGY	
A. Overview of the Methodology	23
B. Methods	26
C. Programming Tutorial	31

LIST OF FIGURES

IV.	RESULTS AND DISCUSSIONS	
	A. Results	32
	B. Discussions	46
		24
V.	SUMMARY AND RECOMMENDATIONS	25
	A. Summary	47
	B. Recommendations	47
		36
		40
		40
		41
		41
		42
		42
		43
		43

BIBLIOGRAPHY

APPENDICES

LIST OF FIGURES

Figure	PAGE
1. The flow chart of methodology	24
2. The desired flow of the process	25
3. An example of a bar-coded card	32
4. The log box	33
5. The flow chart of the card reader program	36
6. Login Prompt	40
7. The Log In Log Out System showing two different options	40
8. The Data Administration Module window	41
9. The Log in Log out window	41
10. Log in Log out window where users can log	42
11. Prompt telling the user whether he has logged in or not	42
12. Table in Microsoft Access showing the employees' logs	43
13. Names, codes, and pins of the employees	43

LIST OF PLATES

Plate

1. RCX
2. Light Sensor
3. IR Tower

LIST OF APPENDICES

Appendix

- A. Pictures of Devices
- B. Programs

ABSTRACT

Time-keeping and Attendance systems are of vital importance to institutions. The Buddy Clock is one of the basic means of recording the employees' time in or out, although more sophisticated technologies are available in the market.

This study designs an alternative computer-based system that uses a light sensor to read bar-coded cards as a means of establishing employee identity.

The devices used were housed in a log box made of wood. The log box has a slit for the cards to be swiped through. The cards contain different barcode combinations that correspond to different employees. For the devices to be workable, a Java-based program consisting of three parts – the card reader, the log in or out and the RCX-to-PC communication was needed.

The cards and the log box were successfully made. The program on the other hand is still not workable for it lacks the RCX-to-PC communication part. However, the log-in/log-out program can act as a stand-alone program to record the times of logging in/out even without the devices, provided it is a manual method of computer log-in.

Bautista, Noreen Marian C., Molina, China May Y., Quezon, Patrick S. "Card Swipe Time-Keeping and Attendance System for Philippine Science High School Western Visayas." Unpublished Research. Philippine Science High School Western Visayas. Bito-on, Jaro, Iloilo City. February 2006.

ABSTRACT

Time-Keeping and Attendance systems are of vital importance to institutions. The Bundy Clock is one of the basic means of recording the employees' time in or out; although more sophisticated technologies are available in the market.

This study designs an alternative computer-based system that uses a light sensor to read bar-coded cards as a means of establishing employee identity.

The devices used were housed in a log box made of wood. The log box has a slit for the cards to be swiped through. The cards contain different barcode combinations that correspond to different employees. For the devices to be workable, a Java-based program consisting of three parts – the card reader, the log in or out and the RCX-to-PC communication was needed.

The cards and the log box were successfully made. The program on the other hand is still not workable for it lacks the RCX-to-PC communication part. However, the log-in/log-out program can act as a stand-alone program to record the times of logging in/out even without the devices, provided it is a manual method of computer log-in.

CHAPTER I

INTRODUCTION

A. Background of the Study

With the advent of modern technology, man has focused on the use of equipment and systems to make his life more comfortable and his work easier. In the workplace, mechanical and electronic equipment are used more and more to replace or supplement human labor. For example, the system for monitoring the working hours of employees depends on such gadgets.

The most common system nowadays uses a Bundy clock. The Bundy clock is used to stamp the times when the employees log in or out of their workplaces. The Bundy clock system, however, has evolved into more efficient and technologically advanced systems.

One example of an advanced system uses the finger scanner. To establish his identity, an employee presses his thumb against a small reader surface (optical or silicon) usually up to two inches square size. The data gathered by the reader is checked against a roster containing a list of the company's employees and their corresponding thumb marks. The time in or out is stored in the system until it is transferred to a computer. Its security feature is based on the uniqueness of a person's thumb mark.

If the system uses a magnetic reader, employees swipe cards with magnetic strips to establish their identities. A magnetic reader reads the information stored in the strips. The time when the swiping was done is stored in a diskette. The transfer of data from the system diskette to a computer is done periodically. In case a person requests another to

log in or out for him, a security guard or a built-in camera is employed to double-check the procedure.

There are other ways of establishing employee identities--retinal scanners, speaker recognition (it is used to "read" unique sound waves in the voices of employees), hand geometry (much like a finger scan but reads the whole hand instead of a thumb), and DNA scanners.

A problem arises due to the cost of such expensive technologies. Biometric devices start at under \$ 100 for a basic microphone or digital camera. Fingerprint scanners cost about the same. More specialized iris or retina scanners cost several hundred dollars per unit. Some of the other devices have prices that range from \$ 250 to up to \$ 2,000 or more. Add to this the fee for a programmer or a software designer, many small- to medium-sized companies in the country cannot afford to purchase such systems. Instead, they settle for the Bundy clock. A Bundy clock may cost from Php 20,000 or higher.

The Philippine Science High School-Western Visayas uses the Bundy clock, an Iwata Model/GWTD. It costs Php 22,500. It uses a ribbon that costs from Php 250 to Php 300. The ribbon has to be changed every three years.

Philippine Science High School-Western Visayas employees are required to punch a Daily Time Record (DTR) in the Bundy clock and to sign in and out on a logbook. A security guard is present to make a duplicate record of the time stamped on the DTR for double-checking. The time required to complete one log depends upon the person logging—whether he is in a hurry or not. One log usually takes 10 seconds to up to 25 seconds.

During the middle and end of the month, the times recorded on the DTRs are copied manually to prepare the payroll. To ensure accuracy, the times on the DTRs are checked against the logbook. Sometimes though, when they are in a hurry, some employees no longer make an entry in the logbook. Thus, the accuracy of the records is not guaranteed.

On the whole, the current system takes up a considerable amount of man-hours for recording, checking and transcribing of information.

Experience in working with the light sensor in the LEGO Mindstorms kit in Robotics class gave the researchers the idea to explore its use in such a system.

B. Statement of the Problem and Objectives of the Study

B.1 Statement of the Problem:

This study aims to design a prototype of a computer-based Card Swipe Time-Keeping and Attendance System for Philippine Science High School-Western Visayas using a light sensor that will read bar-coded cards as a means of establishing employee identity.

B.2 Objectives of the Study:

This study has the following objectives:

1. To design a log box that will use a light sensor to read bar-coded cards.
2. To design cards with different bar code combinations—each combination will correspond to a different individual.
3. To create a computer program that will display the employees' name, date, time in or out and store them in a database.

C. Significance of the Study

The Card Swipe Time-Keeping and Attendance System was designed to use a light sensor inside the log box to read bar-coded cards. The use of the bar-coded cards in this system is an alternative to magnetic strips and readers, finger scans, palm scans, retina scans, and voice readers and DNA scans.

Designing the system was an application of the things learned in Robotics and Computer classes. The researchers used equipment and devices that they have been trained to use in these classes.

The CSTAS will help eliminate the need for manual recording since the data gathered by the system will be automatically transferred to a PC. In the event that a written record is required, the softcopy will simply be printed into a hardcopy.

Employees of PSHS-WV will benefit from this study. The system will eliminate the necessity for an employee to sign in and out of a logbook.

If an alternative system to the Bundy clock is successfully designed, it can be adapted for the use of small- to medium-sized offices. The data stored in the computer will also be integrated into a database that will be useful for other applications such as payroll and pay slip preparation. The bar-coded cards will be incorporated with the present identification cards (IDs) of the employees.

Various offices in establishing employee identity use Barcode scanners. However, this study aimed to use materials and devices that were already available in the school. Instead of using barcode scanners, a light sensor was used to read bar-coded cards.

D. Scope and Limitations of the Study

This study aimed to design a prototype of a computer-based Card Swipe Time-Keeping and Attendance System for Philippine Science High School-Western Visayas as an alternative to the current Bundy clock. The design made use of a light sensor that read bar-coded cards as a means of establishing employee identity. The light sensor was connected to the Robotics Command Explorer (RCX) brick that sent the data it generated to the Personal Computer (PC) with the use of an Infrared (IR) tower.

The study included the design of the log box, the bar-coded cards and the program by which the data generated by the log box will be transferred to a computer.

The light sensor quantified the reflected intensity of light it perceived although it did not record its readings.

The computer at the receiving end contained a program that stored and identified different identities corresponding to the different readings made by the light sensor. It also recorded the employees' time in and out. The program was not able to compute payrolls. It did make reports such as SSS, PhilHealth, and Pag-Ibig contributions. Java language was used for the program of both the log box and the PC. This is because a program for RCX-to-PC communication already existed and the language used is Java.

The design parameters were based on Philippine Science High School-Western Visayas conditions and management system requirements. Thus, without modifications, the system may not be readily suited to other offices or management set-ups.

The system made use of an existing model of a light sensor. This study did not include the design of a new light sensor. The system made use of the components of the LEGO MINDSTORMS Robotics Invention kit with devices equipped with a protocol for

communicating with a PC. This feature was integrated to the design of CSTAS, and thus was a delimitation to the capabilities of the designed system.

System contingency in case of a power failure was not included.

E. Definition of Terms

Accurate Time - corresponds to the time displayed on the CSTAS program that coincides with the time found on the taskbar.

Color-coded cards – cards, each bearing a different color combination used for establishing different identities

Compiler - A program to translate source code into code to be executed by a computer
(docs.sun.com/db/doc/805-4368/6j450e60c)

Firmware - coded instructions that are stored permanently in read-only memory
(WordNet ® 2.0, © 2003 Princeton University)

Infrared (IR) tower - connects to the computer, establishing the wireless link between the computer and the RCX brick
(http://www.mooreed.com.au/products/robolab/keyelements.htm)

Java - a programming language expressly designed for use in the distributed environment of the Internet. It was designed to have the "look and feel" of the C++ language, but it is simpler to use than C++ and enforces an object-oriented programming model. Java can be used to create complete applications that may run on a single computer or be distributed among servers and clients in a network
(http://searchwebservices.techtarget.com/sDefinition/0,,sid26_gci212415,00.html).

LEGO MINDSTORMS Robotics Invention System – a system that enables you to design and program real robots that move, act, and think on their own. The heart of the system is the RCX, an autonomous LEGO microcomputer that can be programmed using a PC. The RCX serves as the "brain" of the inventions -- using sensors to take input from the environment, to process data, and to signal output motors to turn on and off

(http://www.ideafinder.com/resource/tools/featured_tools/rpt101.htm)

LeJOS - a Java based replacement firmware for the Lego Mindstorms RCX

microcontroller (lejos.sourceforge.net/).

Light sensor - a sensor that measures the amount of light that it sees. It reports the amount of light to the RCX as a number between 0 (total darkness) and 100 (very bright)

(<http://www.rec.ri.cmu.edu/education/multimedia/lightsensor.shtml>)

Log box - a time keeping device that is used for your employees' time in and time out.

This device replaces the function of a Bundy clock

(<http://www.esprint.com/new/rushour-faq.htm#What%20is%20a%20log%20box?>)

PC - Personal Computer. Strictly speaking, this refers to the IBM PC, which is any IBM-manufactured personal computer made prior to the PS/2 Series. Usually PC refers to any personal computer compatible with IBMs. It is also used to refer to any personal computer (ccs.uchicago.edu/technotes/misc/Glossary/gloss3.html). In this study this refers to an IBM computer used for the testing and evaluation of the system.

Protocol – the format of the data that the devices agree upon in order to communicate with each other

(http://www.webopedia.com/TERM/C/communications_protocol.htm)

RCX (Robotics Command System) - The RCX is a programmable, microcontroller-based brick that can simultaneously operate three motors, three sensors, and an infrared serial communications interface. The brick is one of approximately 727 pieces of LEGO® set 9719 Robotics Invention System

(<http://graphics.stanford.edu/~kekoa/rcx/>)

System clock - displays or sets the system time on the computer screen (Microsoft XP Professional Help and Support Center).

Time-Keeping/Attendance Keeping - refers to the process of recording the employees' attendances in their different times of arrival and departure.

This chapter contains the various summaries of the articles needed during the conduct of this study.

The history of the Identification System was discussed in order to establish the its continuous growth and development. From this the researchers derived the idea of creating a system having a purpose similar to that of the gadgets' discussed in this chapter.

Barcodes have been used for identification because of its design. The uniqueness of each code contributes to the efficiency of the system that it is involved with. This study uses a particular code called the Code 39. This code consists of nine alternating black and white stripes, each of different width.

The devices involved in this study are also discussed. These are the light sensor, the Robotics Commands System, and the IR Tower. These devices are included in the Lego Mindstorms Robotics Invention Kit. These devices have unique characteristics. The RCX is run by embedded software called the firmware.

A program written in the Java programming language runs the system. A specific kind of programming language called the LeJOS (Lego Java Operating System) is required for use by the Lego devices involved. A program for the communication between the devices and the PC is also written in Java. Data streaming plays a major part in this study.

CHAPTER II

REVIEW OF RELATED LITERATURE

A. Chapter Overview

This chapter contains the various summaries of the articles needed during the conduct of this study.

The history of the Identification System was discussed in order to establish its continuous growth and development. From this the researchers derived the idea of creating a system having a purpose similar to that of the gadgets' discussed in this chapter.

Barcodes have been used for identification because of its design. The uniqueness of each code contributes to the efficiency of the system that it is involved with. This study uses a particular code called the Code 39. This code consists of nine alternating black and white stripes, each of different width.

The devices involved in this study are also discussed. These are the light sensor, the Robotics Commands System, and the IR Tower. These devices are included in the Lego Mindstorms Robotics Invention kit. These devices have unique characteristics. The RCX is run by embedded software called the firmware.

A program written in the Java programming language runs the system. A specific kind of programming language called the LeJOS (Lego Java Operating System) is required for use by the Lego devices involved. A program for the communication between the devices and the PC is also written in Java. Data streaming plays a major part in this study.

In the PC side of the program, an object-oriented display is used. It is called the Graphical User Interface (GUI). It uses a window style display where users can have an easy time of accessing functionalities available in the program.

B. Time-In/Time-Out and Identification System

Time In/Time Out Systems were manufactured over a hundred and ten years ago. The first time clock was developed by Harlow Bundy in the year 1889 and was called the Bundy Clock. It acts as a log box or a time-keeping device used for employee's Daily Time Record Module.

Today, new advancements are developed. A computerized time clock is designed by Bundy Time Machines, which allows employees to clock-in and clock-out on an electronic device. The downside of this system is that this clock does not store information on a disk. Instead, manual recording and tabulation is still required. The software that is used for this new system includes the employee's ID number, time logged-in, time logged-out, and time deduction for breaks and pay rates (<http://metrontimeclock.com/story1.html> and <http://metrontimeclock.com/story2.html>).

In Sunderland, a new retinal eye scanning technology was developed. The purpose of this system is to identify school children when they borrow books from the library. The advantage of this system is that the technology would make more efficient use of the lunch hours because it could scan ten to twelve pupils in a minute. It is also cost-effective since the expenses that would be invested are just the same as in the swipe card system (Bundy Time Clocks). However, unlike a swipe system where you have to buy new cards every year, this system does not need any accessories.

The technology uses a low light beam that possesses minimal or virtually no health risk, unlike a high-powered laser beam that has a high potential for damage (http://prisonplanet.com/eye_scannerrrs_for_school_childre.html. January 8, 2003, Wednesday).

Fingerprint scanning is also a technology that was developed originally for identifying policemen. However, some schools and other companies use the technology for the clocking-in and clocking-out of the faculty and staff.

One of the main purposes of the fingerprint scanning is a secure entry device for door locks and computer network access. Some banks have begun using fingerprint readers for authorization on Automated Teller Machines. Grocery stores are also experimenting with the scan checkout that automatically recognizes and bills a registered user's card or credit account.

The system is accurate and powerful because of the large amount of data that can be drawn from the fingerprints. Given the number of information contained in a fingerprint, it is unlikely that two fingers would be identical. The chance that two fingerprints are identical is one in sixty-four billion.

Since there are some hackers or unauthorized fingerprint readers, companies pushed through the idea of creating a human-sensing device that can differentiate living human fingers and beast replicas to unreal fingerprints (<http://ctl.dni.us/biomet%20web/BMFingerprint.html>).

Facial recognition is a verification system that analyzes the characteristics of a person's face images through a digital video camera. It measures the over-all facial structure, including the distances between eyes, nose, and mouth and jaw edges. These

measurements and information are stored in a database and used as a comparison when a user stands before the camera.

The system will locate the user's face and perform matches to the identity saved in the database. The system usually comes to a decision in less than five seconds. It is possible for a person to fake a face so to prevent it, many systems now require the user to smile, blink, or move in a way that can be verified by the camera database.

Facial recognition is now implemented in many law enforcements areas as a potential tool for the verification and identification of terrorists and wanted criminals. Software has also been developed for an ATM's user identification purposes (<http://ctl.dni.us/biomet%20web/BMFacial.html>).

Hand geometry is the ancestor of modern technology of identification systems. Hand geometry involves the measurement and the analysis of the shape of one's hand. It requires special hardware to use and can easily be integrated into other devices or systems for the database. Since the human hand is not as unique as the fingerprints, features of the hand may not be very descriptive enough to identify a user. However, it is possible to devise a method by combining various individual features and measurements of finger and hand for verification purposes.

The user places the palm of his hand on the metal surface, which has guidance pegs on it. The pegs properly align the hand so the device can read the hand attributes. The device then checks its database for verification and comparison.

There are also some disadvantages to this technology. One constraint is the hardware cost and size. The system is not cost-effective and the metal surface is very big. Also, while injuries to hands can cause difficulty in using the reader effectively; the lack

of accuracy in general requires that the system will just be used for verification and not in identification. Since only a small amount of information is measured, it is possible to have duplicate readings if more people are processed by the system (<http://ctl.dni.us/biomet%/20web/BMHand.html>).

Iris scanning analyzes the features that exist in the colored tissue surrounding the pupil of the eye, which has more than two hundred (200) points that can be used for the comparison, including rings, furrows, and freckles. The scan uses a regular video camera style and can be done further away than the retinal scan. It can work through glasses and has the ability to create accurate measurements that can be used for identification purposes, not just verification. The user places himself in front of the scanner to see his own eye's reflection on the device. Identification process may take five seconds.

The uniqueness of the eyes, even between the left and the right eye of the same person, makes iris scanning very powerful for identification purposes. The likelihood of a false result is extremely low and its relative speed and ease makes it a good system. The only drawbacks are the potential difficulty of getting someone to hold their head in the right spot if they are not doing the scan willingly (<http://ctl.dni.us/biomet%/20web/BMIris>).

Another existing time-in/time-out system is the Acroprint Time Q Badge Swipe Time and Payroll Recorders-Standalone Time and Attendance System (<http://clock/system/acroprint.html>). The new system tracks an employee's arrival and departure times. It can calculate hours on weekly and monthly pay periods. These functions eliminate the need to buy time cards and ink ribbons for printing the time on the cards. There will also be no need to allocate man-hours for the preparation and the

recording in the time cards. Employees just swipe their badge cards to clock-in or clock-out, as well as seeing their hours worked for the day and the pay period. The system has the capacity to store data for a minimum of fifty employee badges. The problem for this system is its ability to track down the one who punches in and goes out without working. This problem however, is the same for all other existing systems.

Local applications in the country (Philippines) include the biometric fingerprint scanning. Such a system is now commonly used in big companies and universities. The University of San Agustin and Central Philippine University, both found in Iloilo, are some of the institutions noted to use the fingerprint scanner. The Provincial Capitol of Iloilo uses an index finger scanner, which scans the user's fingerprint, and asks for a pin code edited by the user for reliability and security. The secured identity of the user is stored in a database and sent in a host computer for payroll purposes.

C. Bar Coding

The industrial use of barcodes can be traced back as far as the 1960s. Some of these early implementations were used to identify railroad card. Common barcodes started appearing on grocery shelves in the early 1970s

(<http://www.idautomation.com/barcoding4beginners.html#Why>). The representation of parallel alphanumeric characters in varying thickness and width contains the information, product identification and price, defines the typical barcode symbol (Webster's Encyclopedic Unabridged Dictionary of the English Language, 1983). This symbol, called the Universal Product Code (UPC), is the standard code for the grocery industry and has been in existence since 1973.

The device known as the barcode reader or the barcode scanner reads the barcode symbol. It is an optical scanning device that reads texts, which have been converted into a special set of parallel lines (McGraw-Hill Dictionary of Scientific and Technical Terms, 5th Edition 1994).

One of the primary advantages of the barcodes over other technologies is its insusceptibility to errors in data input. Barcodes can have built-in safeguards (check digits) to prevent incorrect scans from being entered, minimizing the possibility of errors.

Other advantages of bar coding are speed, timeliness and cost-efficiency.

Scanning a barcode is faster than manually recording information or keying data into a terminal. Bar-coded information is often immediately transferred to a host computer.

Real-time data collection enables timely information to be accessed almost instantly, when the data is still current.

In addition, improved efficiency can be realized by substituting barcode systems in place of manual systems, resulting in increased productivity and reduced labor cost (<http://www.webermaking.com/html/barcodebook.html#Anchor-In-45470>).

D. Code 39

Barcodes come in different types and faces that were made to suit the functionalities of companies and their barcode scanners. Code 39 is one of these types. Each character in the barcode consists of nine black and white stripes in either thick or thin widths. The code will always start and end with a black stripe. The uniqueness of each character depends on the sequence of thick and thin widths.

E. Light Sensors

A light sensor is an analog device that measures the intensity of the light it sees (<http://ec.cmu.edu/eduaction/multimedia/lightsensor.html>). In the LEGO Light Sensor, the amount of light it perceives is reported to the Robotics Command System (RCX) as a value corresponding to the intensity of the light transmitted. The light sensor uses its own light source, a red Light Emitting Diode (LED) to illuminate a small area. A phototransistor then receives the reflected light and it converts the light into voltages. (<http://elecdesign.com/Articles/Print.cmf?ArticleID=6339>). This voltage is transported to the RCX, which interprets the data in terms of its own values. These values are seen on the display window of the RCX (<http://philohome.com/sensors/lightensors.htm>).

F. Robotics Command System

The Robotics Command System (RCX; also known as the Robotics Command Explorer) is part of the LEGO Mindstorms kit designed by LEGO Inc. that comes packaged with two touch sensors, two small electric motors, a light sensor, a software (RCX Code), and seven hundred (700) plus LEGO pieces that allows a hobbyist to control an intelligent machine or explore robot parts.

The heart of the RCX brick is a small 8-bit microcontroller that acts as the system's brain or processing unit. The Hitachi H8 microcontroller is a small processing unit for the RCX brick that manages input data collection and data processing from the light receiver of the light sensor. It is the microcontroller that converts analog data to digital and shows the value of light intensity on the Liquid Crystal Display (LCD) screen. Six 1.5-volt alkaline batteries (rechargeable or non-rechargeable) power the RCX. The

version 1.0 of the RCX also contains an Alternating Current (AC) port, where an electrical cable can be attached.

There are several ways to make the RCX communicate with the computer. One way would be via the Infrared (IR) port of the RCX. Instead of directly downloading the software program in the RCX, the computer uses its infrared power to send control commands to the RCX; the computer can read the sensor data values that are sent by the RCX. One problem in this method is the over flooding of the data. If there are too many data to be sent to the computer, the processing between the RCX and the IR tower is very time consuming.

Another mode of transmission is via cable connections. A disadvantage of this method, however, is the effect on the circuits of the RCX. Taking apart the RCX brick and putting in a cable connection through soldering may destroy some of the circuits. It will reduce the sensitivity and accuracy of the RCX brick in receiving and displaying the values from the light sensor. The RCX runs programs through its operating system, called the firmware.

G. Firmware

The firmware is the software that is embedded in a hardware device that allows reading and executing of the software, but does not allow modification, e.g., writing or deleting data by an end user (exact words and marks from www.bandwidthmarket.com/resources/glossary/F3.html).

In the RCX, it serves as an operating system that enables the execution of programs downloaded to it. The LEGO firmware for the RCX can be downloaded from the LEGO Mindstorms Robotics Inventions System Software that comes packaged with

the Robotics Inventions Kit. The firmware is downloaded through an Infrared Tower to a computer, to the infrared port of the RCX.

H. Infrared (IR) Tower

The LEGO Mindstorms kit has an Infrared (IR) Tower that provides the interface between the RCX (Robotics Command System) and the computer. Through an IR Tower, programs from the computer are downloaded to the RCX that are responsible for the running of a LEGO Robot. The IR Tower is also the one that initiates computer- RCX communication through its light beams, which transmit the data or receive the information. The tower is connected to a Universal Serial Bus (USB) port of the computer.

I. Java

Java is the programming language created by Sun Microsystems which allows the user to create programs which will run well in a networked environment (such as the World Wide Web). Its structure enables web page commands and is equipped with several functions. One function is the Superclock (<http://javaapp/superclock/time.ktml>) and JavaCounter Module that enables the time indicators. The clock function indicates the time in analog format.

Java is also one of many programming languages, which can be used to write programs for the Robotics Command System. Like all programming languages, Java consist of (1) an IDE or the Integrated Development Environment, which is a programming environment integrated into an application (<http://ipi.apit.edu/glossary.html>), and (2) a compiler that translated source code (the

program written by the programmer) written in Java programming language into and the bytecode or "machine language" for the Java virtual machine (<http://docs.sun.com>).

J. LeJOS

LeJOS is the replacement firmware when Java is used for programming in the RCX. It is easily downloaded from the Internet to the RCX. The website where LeJOS can be downloaded is <http://lejos.sourceforge.com/download.html>. LeJOS allows a user to read and execute Java programs.

K. Java Program

For a system to process data, software is the key. In the interaction between the PC and the RCX, there is already communication with them every time one downloads or sends the program (in robotics), which means that the RCX is already preprogrammed to communicate with different devices.

Java programs use streams to communicate. A stream consists of an output device that sends and an input device that will receive the data. Streams are based upon connections between two devices (PC and RCX) and they are classified according to their direction. An output is the one that initiates the sending data and the input is the one that receives the data. (<http://java/datastreaming/comm/program.html>).

Computers use byte streams also known as data streams. Byte is the smallest primitive data type in Java and it is not the one that makes the RCX and the PC communicate. If one needs to use streams for PC communication there is a need to make a set of protocols. A protocol is a set of rules for communication that a sender and the

receiver agree to follow. These rules will include the signal for the arrival of data and the signal for the departure of the information.

The protocol is contained in the program that will run the system. In preparing the program, commands using Java are used. Examples of these commands are PcRecv.java and RCXSend.java. The RCXSend.java is the program that will enable the RCX to send values or the data to the computer.

L. PC to RCX Communication

To facilitate communication between the RCX and the PC, there are three things that are needed: (1) an output stream to the sender's program, (2) an input stream to the receiver's program and (3) establish a protocol for the communication

(<http://java/datastreaming/comm/program.html>).

The output stream would be the RCXSend.java. The program uses the RCXBean to manage the data stream with the RCX. The RCXBean specifies the port where the Infrared is connected and it directs the entrance and the exit of the data. This program is to be compiled in the standard JDK compiler.

The input stream is the command PcRecv.java. The program does not require RCXBean so it works with the received data immediately. It has an RCXPort that is the physical Infrared port of the RCX. It can read all primitive types of data. PcRecv.java uses a LeJOS compiler. A compiler is the one that translated source code written in Java programmed language into bytecode or "machine language" for the Java virtual machine (<http://docs.sun.com>).

M. Graphical User Interface

A graphical user interface (or GUI, pronounced as “gooey”) is a method of interacting with a computer through a metaphor of direct manipulation of graphical images and widgets in addition to text

(http://en.wikipedia.org/wiki/Graphical_use_interface).

The term came into existence because the first interactive user interfaces to computers were not graphical; they were text-and-keyboard oriented and usually consisted of commands that had to be remembered and computer responses that were infamously brief. The command interface of the DOS operating system (which one can still get to from Windows operating system) is an example of the typical user-computer interface before GUIs arrived. An intermediate step in user interfaces between the command line interface and the GUI was the non-graphical *menu-based interface*.

Applications typically use the elements of the GUI that come with the operating system and add their own graphical user interface elements and ideas. A GUI sometimes uses one or more metaphors for objects familiar in real life, such as the desktop, the view through a window, or the physical layout of a building. Elements of a GUI include such things as: windows, pull-down menus, buttons, scroll bars, iconic images, wizards, and the mouse.

When creating an application, many object-oriented tools exist that facilitate writing a graphical user interface. Each GUI element is defined as a class widget from which one can create object instances for the application desired. Prepackaged methods can be decoded and modified to make the object declared respond to user stimulus (http://searchwebservices.techtarget.com/sDefinition/0,,sid26_gci213989,00.htm).

CHAPTER III

METHODOLOGY

A. Overview of Methodology

The study aimed to design a prototype of a computer-based Card Swipe Time-Keeping and Attendance System for Philippine Science High School-Western Visayas (PSHS-WV) that used a light sensor that read bar-coded cards as means of establishing employee identity. The design used the devices from the LEGO Mindstorms Robotics Inventions System, a kit used by the PSHS-WV students in the Robotics class.

Specifically there were three key design aspects: (1) the bar-coded card, (2) the log box, and (3) the computer program.

A card was made from cardboard and had a Code 39 type of barcode that served as the unique identification of a certain employee and staff. A LEGO Light Sensor was used to read the barcode in terms of varying widths of white and black. The sensor was contained in a log box designed by the researchers. The log box had a slot for the card to be swiped through. From the log box, the light sensor was connected to a LEGO Robotics Command System (RCX) brick. This microcomputer LEGO brick acted as an integration unit and assigned its own set of values based on the readings of the light sensor.

Programming on the PC (Personal Computer) and the RCX was then done. On the PC side, a Graphical User Interface (GUI) was created to serve as the window for logging in and out. On the RCX side, a card reader program and a communication program was encoded. Testing of the program was then conducted to see if the functionality had taken effect.

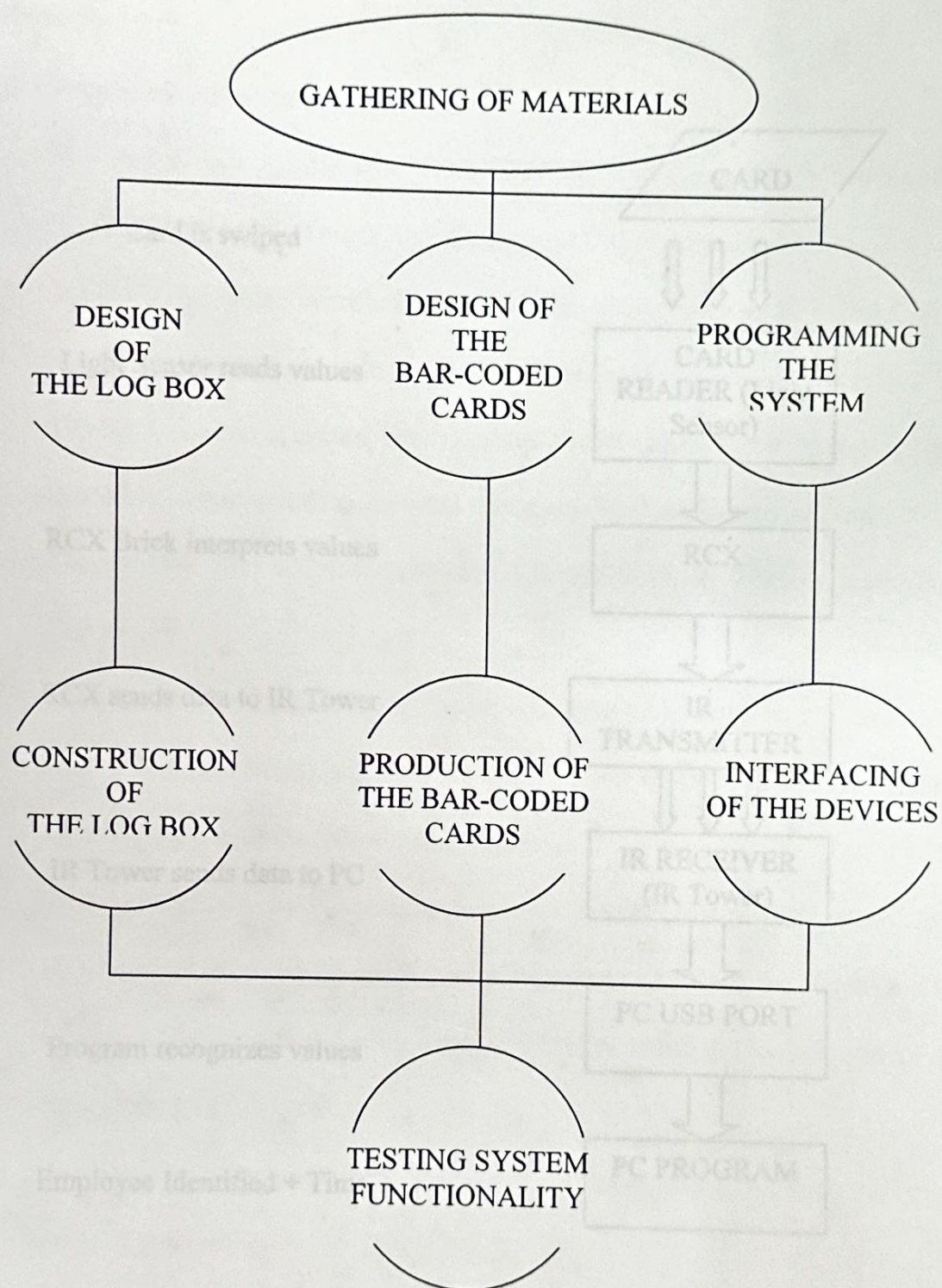


Figure 2. The desired flow of the process.
 Figure1. The flow chart of methodology.

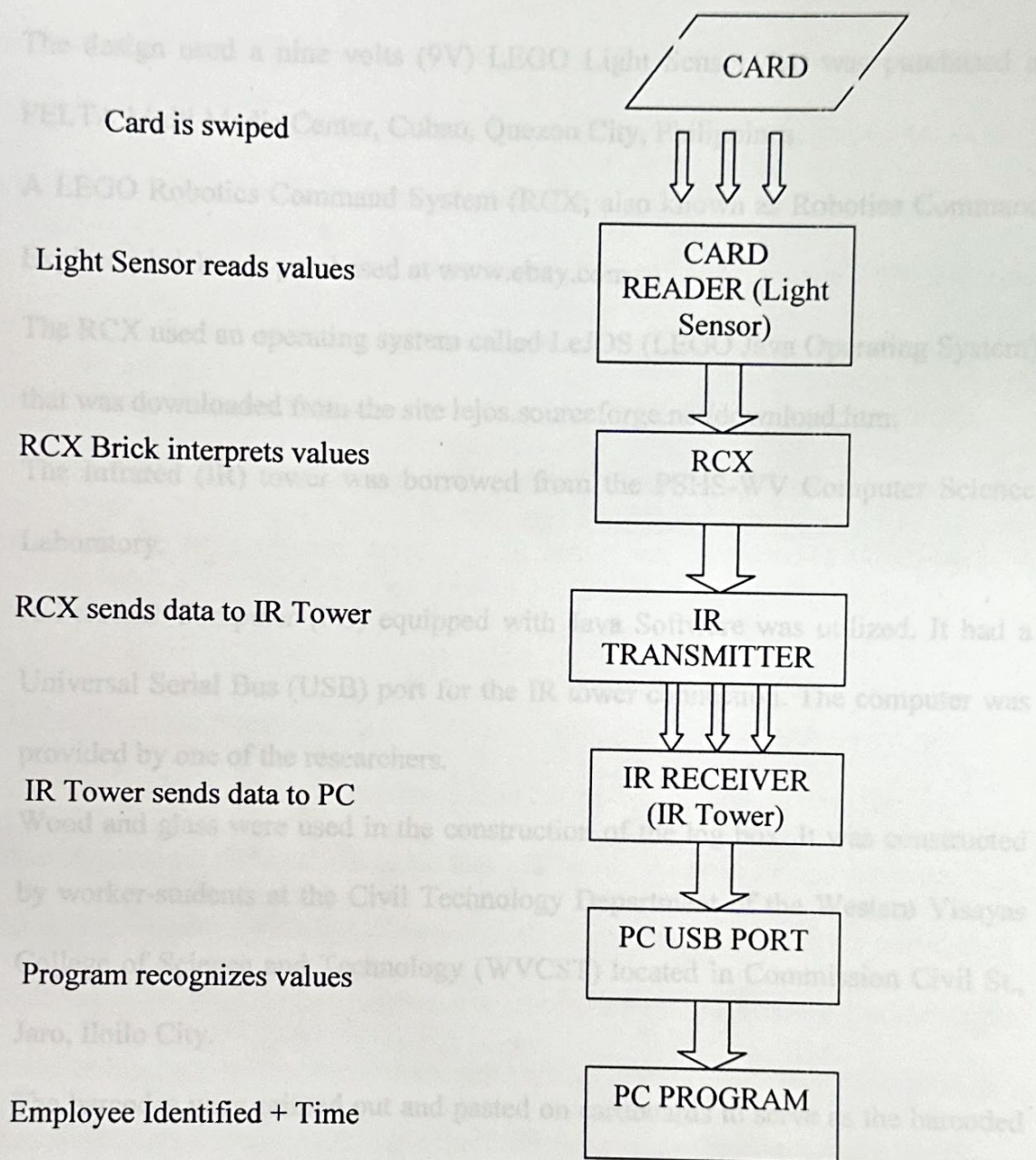


Figure 2. The desired flow of the process.

B. Methods

B.1 Procurement of Materials

- a. The design used a nine volts (9V) LEGO Light Sensor that was purchased at FELTA Multi Media Center, Cubao, Quezon City, Philippines
- b. A LEGO Robotics Command System (RCX; also known as Robotics Command Explorer) brick was purchased at www.ebay.com.
- c. The RCX used an operating system called LeJOS (LEGO Java Operating System) that was downloaded from the site lejos.sourceforge.net/download.htm.
- d. The Infrared (IR) tower was borrowed from the PSHS-WV Computer Science Laboratory.
- e. A Personal Computer (PC) equipped with Java Software was utilized. It had a Universal Serial Bus (USB) port for the IR tower connection. The computer was provided by one of the researchers.
- f. Wood and glass were used in the construction of the log box. It was constructed by worker-students at the Civil Technology Department of the Western Visayas College of Science and Technology (WVCST) located in Commission Civil St., Jaro, Iloilo City.
- g. The barcodes were printed out and pasted on cardboards to serve as the barcoded cards of the system.

B.2 Assembly of the Log Box

The log box was designed by the researchers, and the design was sent to the Western Visayas College of Science and Technology (WVCST) Civil Technology Department for construction. It was made of wood and had a slot that allowed the card insertion. It was designed so that external factors that can affect the barcode reading process were limited, which include the amount of ambient light and other environmental or man-made interference such as temperature, glare from the sun, and human contact.

The sensor in the log box was connected to the RCX through a simple LEGO attachment to one of its ports. Once attached, the RCX faced the IR Tower in such a way that infrared signals were not interrupted. The IR Tower was connected to a USB port found in the PC. (Figure 2)

B.3 Identification Card

A Code 39 Barcode Font was downloaded at www.downloads.com. The researchers figured out a bar-code scheme that will be sufficient for an approximate population of 50 employees. Codes consisted of three numbers, ranging from one to six. Each character in the barcode font is composed of nine alternating black and white stripes of different widths. The permutation of six numbers or barcode characters taken three at a time gave one hundred twenty (120) different barcodes—more than enough for PSHS-WV employees.

B.4 Program

The program had three features:

- (1) The Card Reading Program, (2) the protocol of the RCX and PC, and (3) the actual Time and Attendance program.

B.4.1 Card Reading Program

This program is basically the barcode reader of the system. LeJOS programming was used to build the program in the PC and was later downloaded to the RCX and stored in its memory. LeJOS (LEGO Java Operating System) is a Java-based Operating System specialized for LEGO Robotics and is the replacement firmware (the standard operating system of the RCX) of the brick and holds its memory. The Card Reading program decoded the black and white stripes of the barcode to its corresponding numerical form, which was a three-digit number. It was composed of three subprograms namely:

- 1) CodeGetter – determined the thickness of each stripe that it read and noted the sequence in an array.
- 2) DecodeNumber- translated the array into a single digit
- 3) CommBaggage- stands for “Communication Baggage.” This is an array that stores the digits sent by DecodeNumber. If there are already three (3) digits in the array (i.e. if the employee’s code was decoded), the RCX-to-PC communication or Protocol program will start.

B.4.2 Protocol Programming

In order for the RCX to send data to the PC, a protocol must be established. A protocol is a set of rules laid down for devices to enable them to communicate. The protocol in this case was embedded in both programs in the RCX and the PC.

The Java and LEJOS Languages were used for the protocol program of both devices. This is because a program for RCX-to-PC communication already exists and the language use is Java.

Configuration included setting the Infrared port of the RCX and the USB IR Tower of the PC. It was for the purpose of opening a line of communication between the two devices—creating an input stream in the PC and an output stream in the RCX. The two are necessary to synchronize the communication and establish the sender and the receiver of the data; specifying the type of data to be transmitted, which is the light intensity.

B.4.3 Time and Attendance Program

The program was able to perform two functions:

- 1) Time Keeping (including date)
- 2) Establishing Identity and Recording

Java was the programming language used, not only because it was compatible with the LEGO devices, but also because it had a clock function already embedded in its development. A four-digit pin code was asked from the cardholder after the code was inputted. This provided security actions to ensure that no other individual aside from the card's owner can log in or out with the card. Using the clock function, the name of the person, the date, the time, and the indicator for logging in or out was displayed in the

additional table for number of hours work per employee was added to the database. The data can also be printed out for documentation, recording or other purposes.

As a mode for data example, ten names of PSHS-WV scholars were randomly chosen for the prototypal list of employees.

C. Programming Tutorial

The researchers underwent tutorial sessions on Java programming from the Informatics Computer Institute in SM City Iloilo, and from the Central Philippine University Computer Studies Department. The purpose of their lessons was to develop familiarity and competence in using the command of the language in order to create the program appropriate in the indicated design.

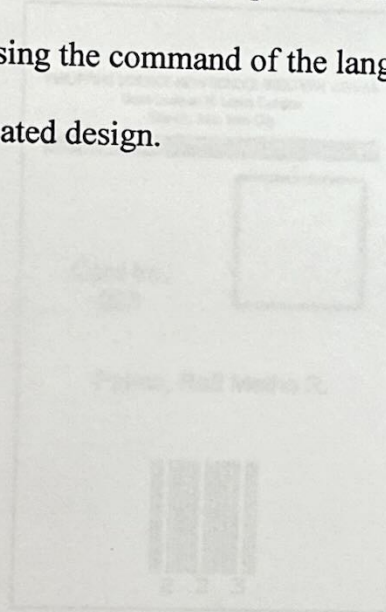


Figure 3. Example of a bar-coded card.

By a fixed sequence of three numbers from one to six were determined

representing the employees.

CHAPTER IV

RESULTS AND DISCUSSIONS

A. Results

A.1 Bar-coded Cards

Bar-coded cards were patterned after the current identification cards of PSHS-WV employees. They were presented as a regular vertical ID card except for the additional barcode at the bottom (Figure 3).

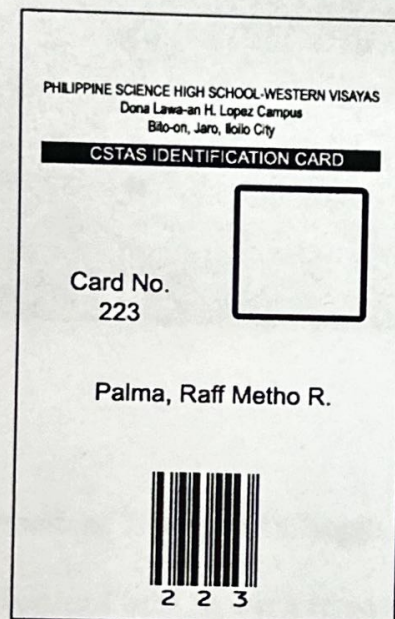


Figure 3. An example of a bar-coded card.

Ten different sequences of three numbers from one to six were determined representing ten employees.

A.2 Log Box

The log box resembled an L-shaped structure (Figure 4).



Figure 4. The log box.

It has a height of 14 cm, width of 21 cm, and a length of 26.5 cm. The Infrared Tower faced the RCX as it was supposed to. There is a removable cabinet where the RCX was placed in case this device needed to use batteries. The light sensor is connected to one of the RCX's sensor ports, and it also has a compartment that allows it to face the card-swiping area. The card slot is thin enough for cards to be swiped through. The design ensures that the infrared transmission would not be interrupted. The RCX cabinet is of the exact length so that the IR port of the RCX would be directly in front of the IR tower during the transmissions.

A.3 Program

The PC used for this study has the following specifications:

OS Name:	Microsoft Windows XP Home Edition
Version:	5.1.2600 Service Pack 2 Build 2600
OS Manufacturer:	Microsoft Corporation
System Manufacturer:	TOSHIBA
System Model Satellite:	A75
Total Physical Memory:	512.00 MB
RAM:	448 MB
Processor:	Mobile Intel® Pentium® 4 CPU 3.20 GHz

The operating system required by the programming language is Windows.

The PC should have at least 500 kilobytes of memory for the source codes alone.

The Lego IR tower was installed to the PC. This enabled the IR tower to be connected to the PC.

In order for the program to run, the PC used was equipped with a JCreator Application. This application provided the environment for the program-writing stage. Once completed, the source code was first compiled. After it has been compiled, it was run. The JCreator is equipped with these options.

The Card Reading Program and the Time and Attendance Program were both workable. Using LeJOS programming, barcodes were successfully read by the light sensor. This was affirmed when a Code 39 bar-coded card was swiped in front of the sensor, and the RCX displayed the numerical value of the barcode on its Liquid Crystal Display (LCD).

The Protocol Program designed to initiate and maintain data streaming between the RCX, the IR Tower, and the PC failed to work after various configurations. However, the Time and Attendance program in the PC was successful as an alternative to RCX inputs and had a functional database that stored the information desired (time, employee's name, date, and log status).

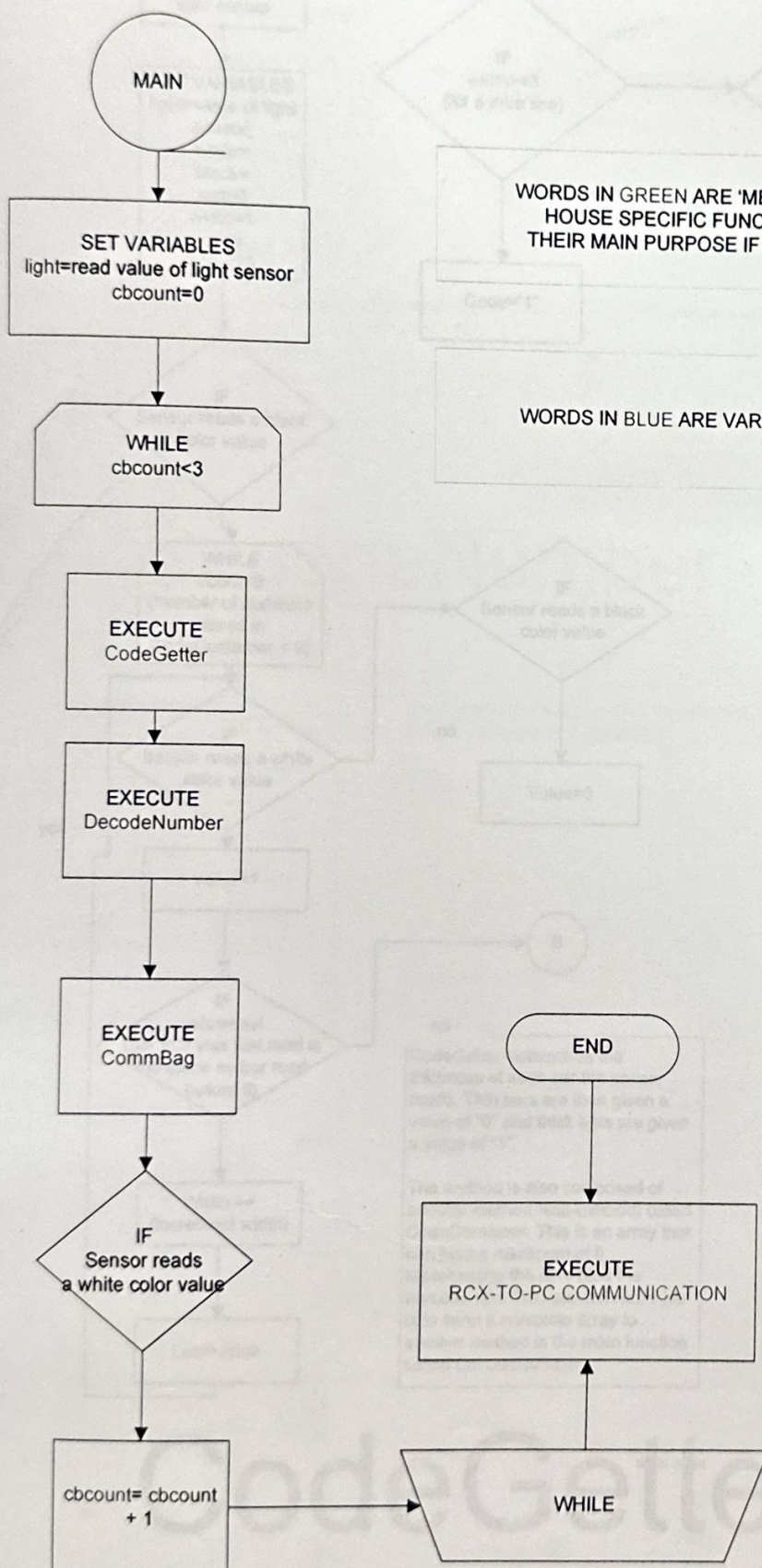
It presented the login window for the administrator upon first access of the program. As initial administrators, all of the researchers had administrative accounts to the program. The login of the administrators was successful. The two window options of 'Data Administration' and 'Log In and Out Program' appeared after. In the Data Administration option, the administrators were able to view the database and add new employees as well as delete existing ones.

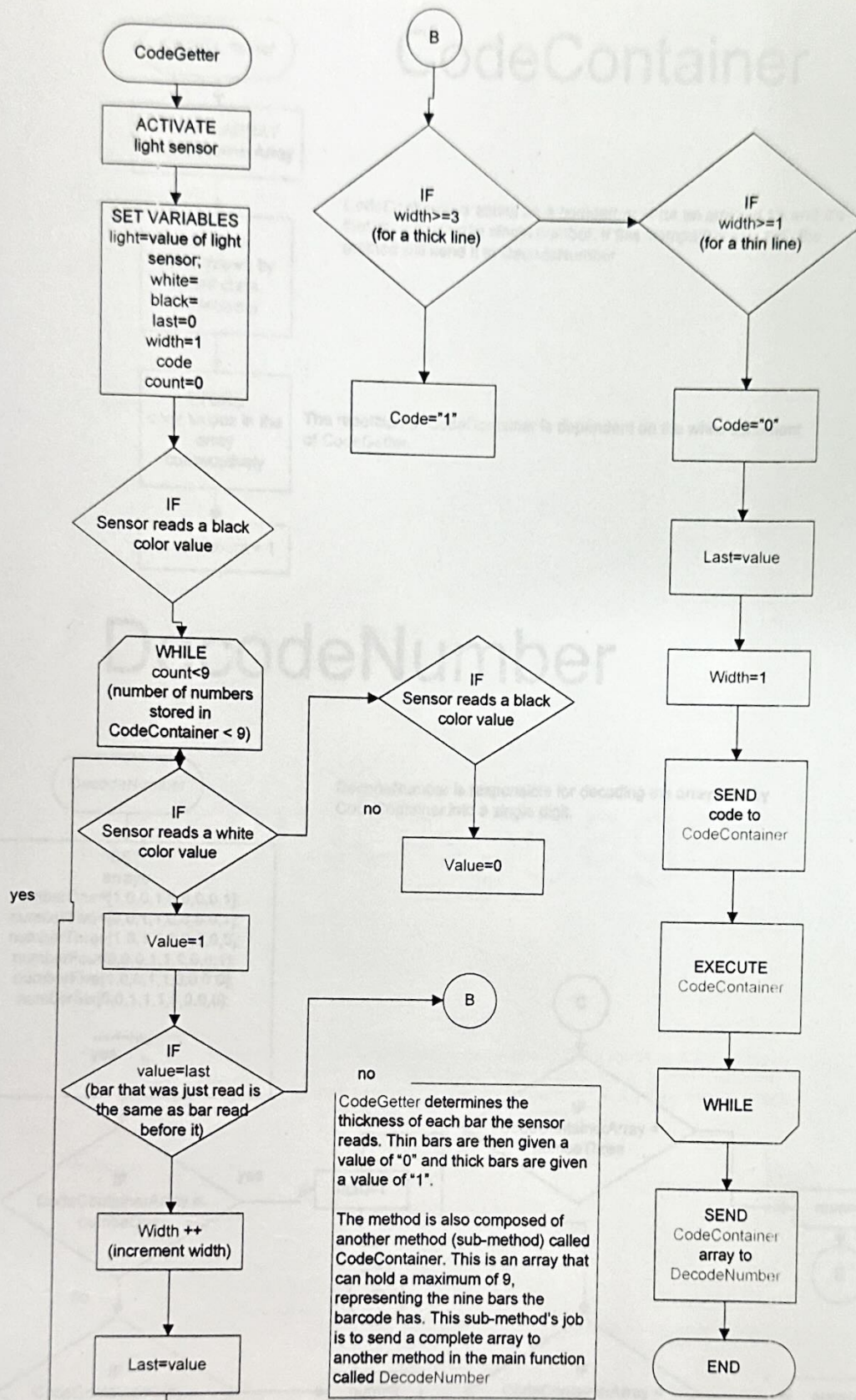
The Login and Logout option successfully opened the environment where the employees will log. By manually inputting their codes and assigned pin numbers, the names of the employees appeared with the indicator of logging in or out. The data was also successfully passed to the database.

A.4 Results in Pictures and Graphics

The following four pages show the flow chart of the Card Reader Program that was downloaded to the RCX (Figure 5). The pages after that will show the various screenshots of the PC-based Login System of the Time and Attendance Program. The orthographic sketches of the log box are also included.

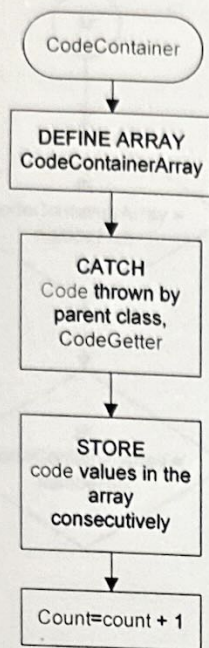
CARD READER PROGRAM





CodeGetter

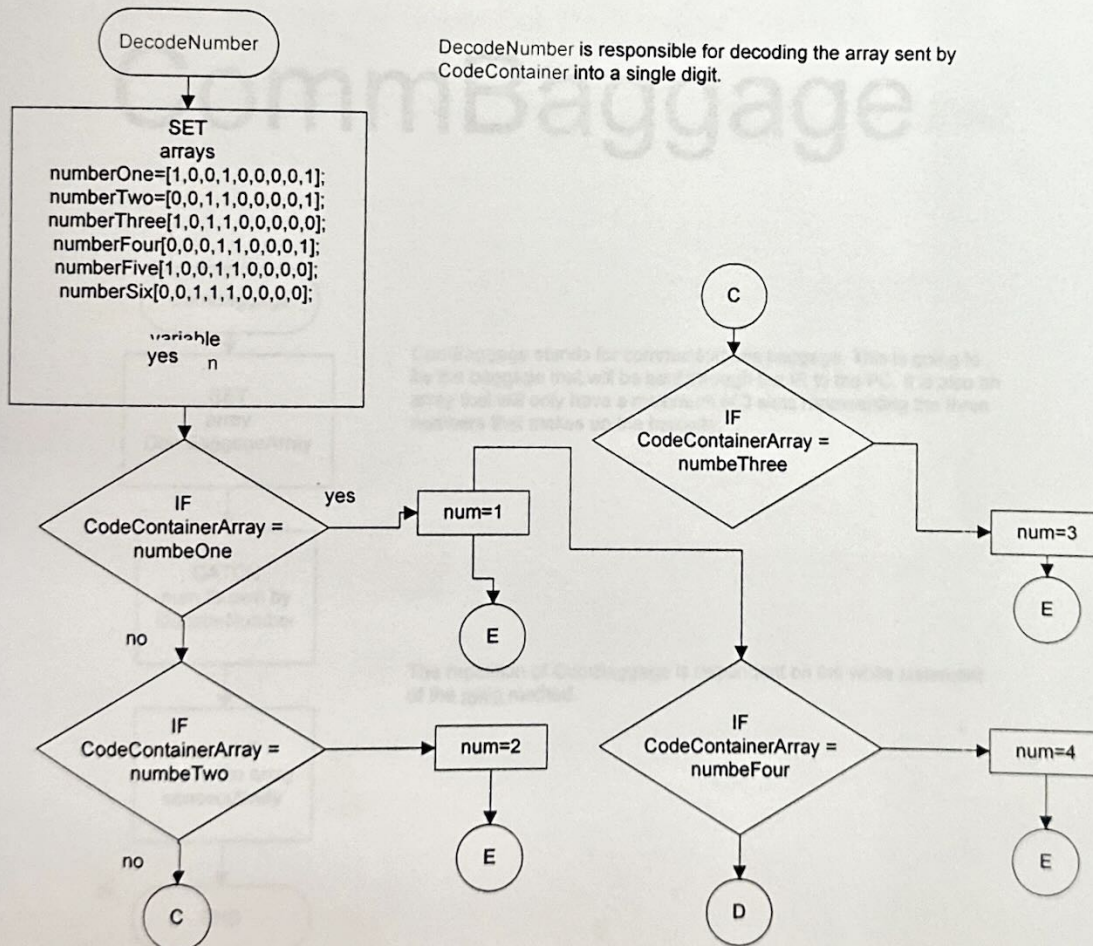
CodeContainer



CodeContainer is acting as a compartment for an array of 1's and 0's that represents one single number. If this 'compartment is full', the method will send it to DecodeNumber

The repetition of CodeContainer is dependent on the while statement of CodeGetter.

DecodeNumber



DecodeNumber is responsible for decoding the array sent by CodeContainer into a single digit.

Figure 5. The flow chart of the card reader program.

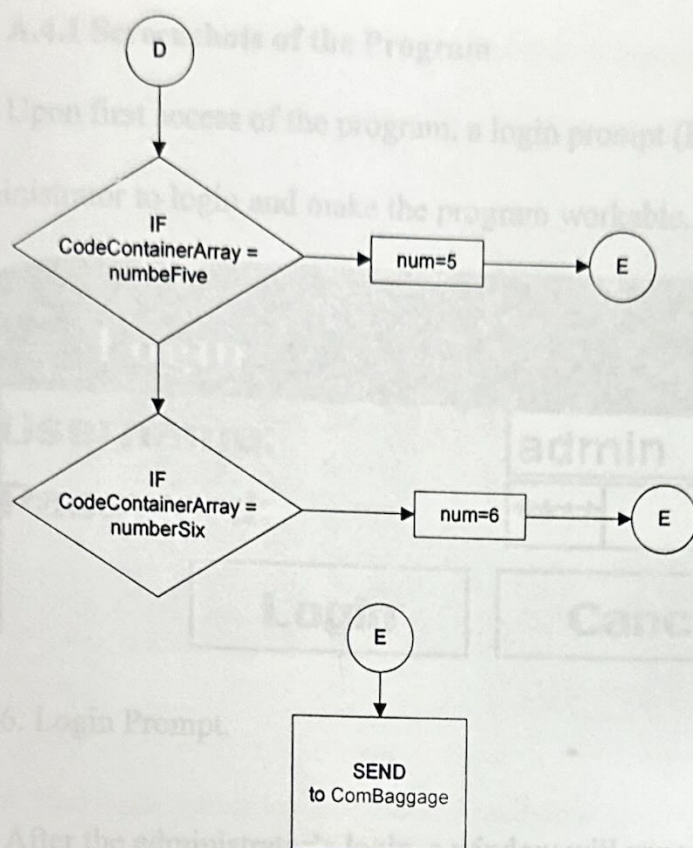
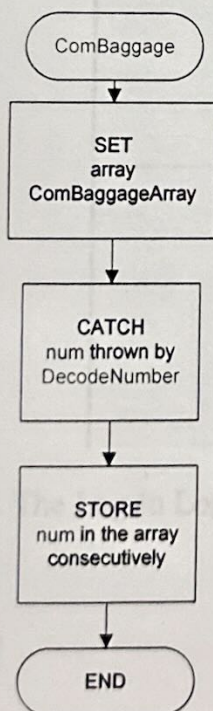


Figure 6: Login Prompt.

CommBaggage



ComBaggage stands for communications baggage. This is going to be the baggage that will be sent through the IR to the PC. It is also an array that will only have a maximum of 3 slots representing the three numbers that makes up the barcode.

The repetition of ComBaggage is dependent on the while statement of the main method.

Figure 5. The flow chart of the card reader program.

A.4.1 Screenshots of the Program

Upon first access of the program, a login prompt (Figure 6) will appear prompting an administrator to login and make the program workable.

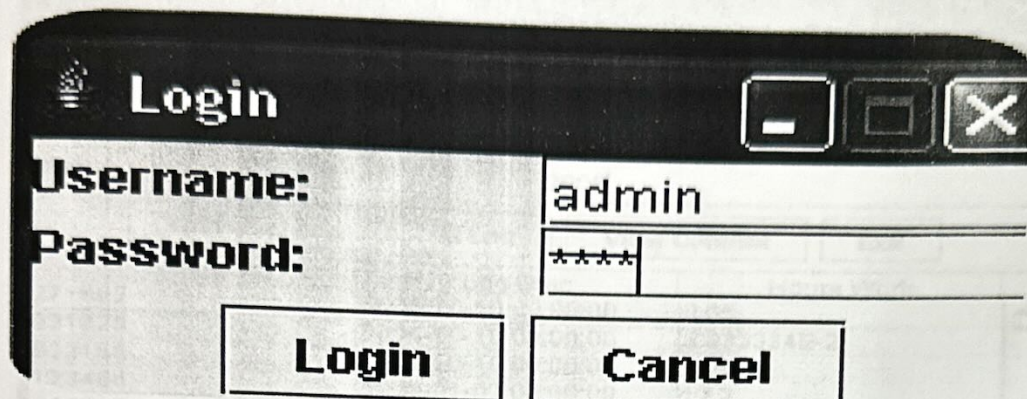


Figure 6. Login Prompt.

After the administrator's login, a window will appear presenting the options of Data Administration and the Log in Log Out Program (Figure 7).

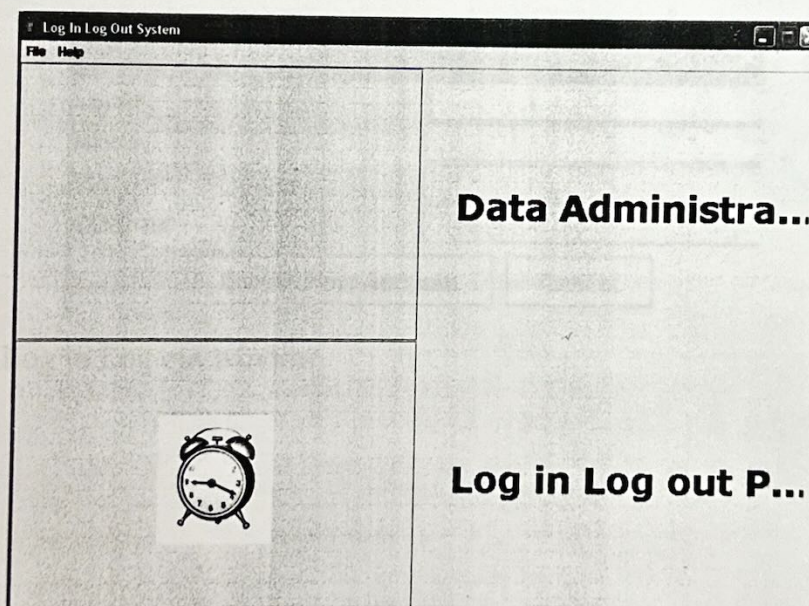


Figure 7. The Log In Log Out System showing two different options.

A window will appear when the Data Administration Option is chosen (Figure 8). Administrators can view the content of the database through this option.

Code	Log Date	Hours Work
271692	2005-10-10 00:00:00	0.05
931825	2005-10-10 00:00:00	3.333334E-2
932165	2005-10-10 00:00:00	0.05
123456	2006-01-02 00:00:00	12.2
245316	2006-01-04 00:00:00	0.0
123456	2006-01-24 00:00:00	0.0

Figure 8. The Data Administration Module window.

The administrator can add a new account using this window:

Figure 9. The Log in Log out window.

This is the GUI for the Login and Logout Program:

The screenshot shows a window titled "Log in Log out". It contains the following elements:

- Time:** 11:35:48 PM
- Date:** Feb 23, 2006
- Code:** A text input field.
- Pin:** A text input field.
- At the bottom, there are three buttons: "Log in", "Log out", and "Cancel".

Figure 10. Log in Log out window where users can log.

Figure 12. Table in Microsoft Access showing the employees' logs.

After the employee enters his correct code and pin code, his name will appear on the screen (Figure 11).

The screenshot shows the "Log in Log out" window with the following elements:

- Time:** 11:40:46 PM
- Date:** Feb 23, 2006
- Code:** A text input field.
- Pin:** A text input field.
- Billones, Irene** is displayed in the bottom left area.
- og out at 11:40 PM** is displayed in the bottom right area.
- At the bottom, there are three buttons: "Log in", "Log out", and "Cancel".

A small dialog box titled "Logged out" is overlaid on the main window. It contains an information icon, the text "Billones, Irene has logged out", and an "OK" button.

Figure 11. Prompt telling the user whether he has logged in or not.

These are the data stored in the database:

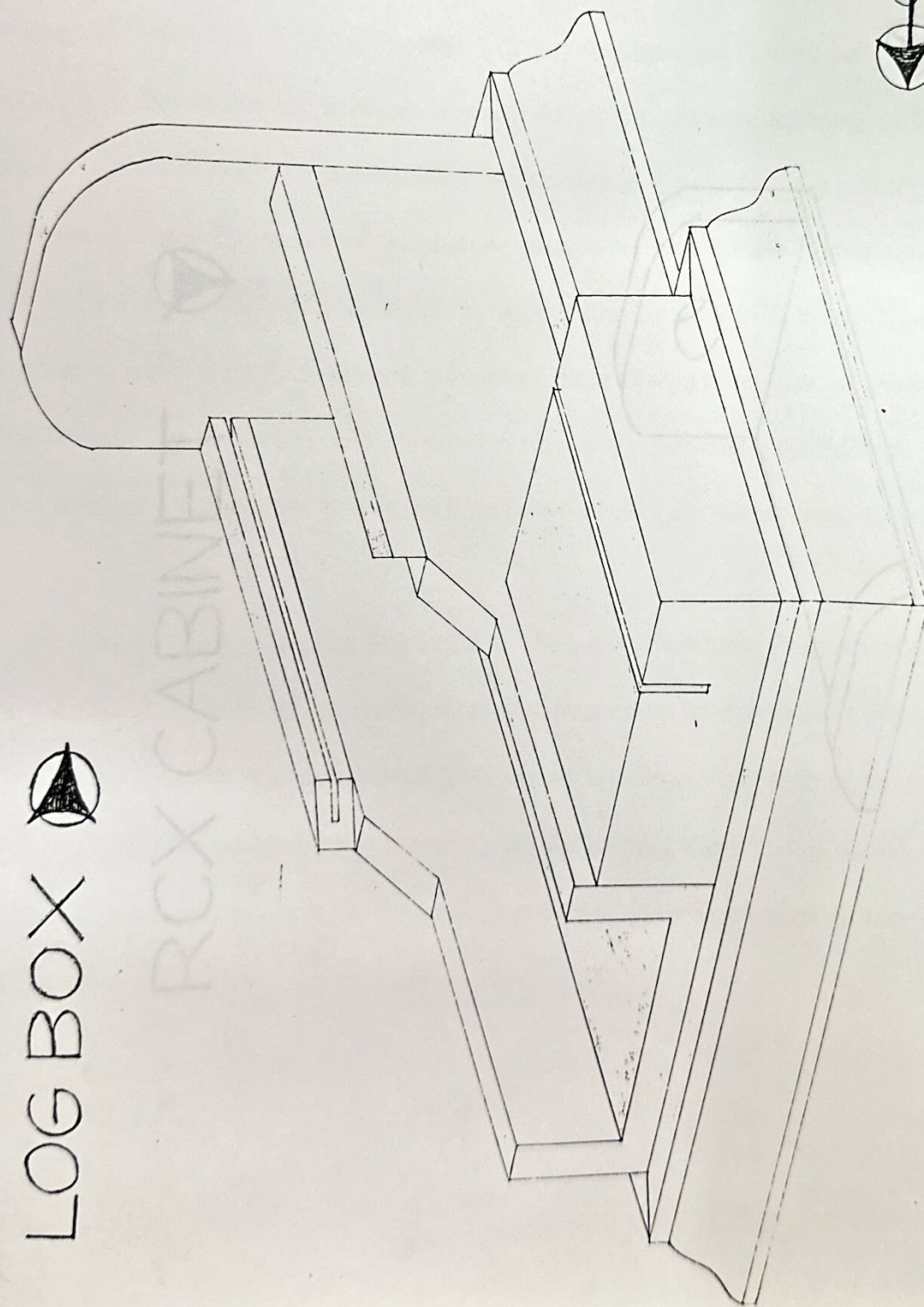
Dailylog						
code	log_date	in_hour	in_minute	out_hour	out_minute	hours_work
652	12/9/2005	8	21	17	0	9.183333
111	12/13/2005	13	9	0	0	0
554	12/9/2005	7	30	16	31	9.016666
612	12/9/2005	17	42	17	0	8.333334E-02
146	12/9/2005	17	47	17	0	8.333334E-02
325	12/9/2005	17	55	17	0	8.333334E-02
411	12/13/2005	12	28	12	28	0
226	12/9/2005	7	30	17	0	10.03333
525	12/9/2005	7	30	17	0	10.38333
334	12/13/2005	13	5	0	0	0

Figure 12. Table in Microsoft Access showing the employees' logs.

Masterfile		
code	name	pin
652	Lebron, James	1111
111	Billones, Irene	1234
222	Juanico, Claire	222
554	Uy, Chicki Florette	2453
612	Villacastin, Anne Jinky	2716
146	Palma, Raff Metho	2835
325	Dulla, Yevgeny Aster	3942
411	Chua, John Bryan	7486
226	Ferolino, William Patrick	8731
525	Vefnegas, Mcgyver	9183
334	Ramirez, Jennifer	9318

Figure 13. Names, codes, and pins of the employees.

LOG BOX



SCALE 1:0.5cm

B. Discussions

The Protocol Programming side of the system posed as a difficult stage as it requires configuring both the computer and the RCX to transmit specific data required in the system. This is the reason why the researchers had to explore ways on applying settings in the program of the RCX and the computer to suit the system design.

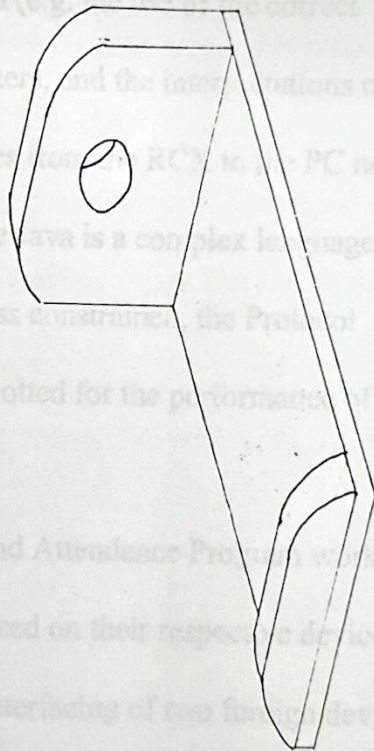
The failure of the Protocol Program to run was due to the factors of limited knowledge and resources of the researchers. The required parameters were not the appropriate commands and methods were not found, therefore the accuracy of the protocol was not achieved. An appropriate configuration (e.g. choice of the correct methods and commands, the use of the accurate parameters and the interpositions of arrays) that will allow the transmission of the light values from the RCX to the PC need to be designed and written in Java form. However, since Java is a complex language and the time to learn it was a more comprehensive scope was constrained, the Protocol Program was not completed within the time frame allotted for the performance of this study.

On the other hand, the Card Reading and Time and Attendance Program worked successfully because the program development was focused on their respective devices only. Unlike the Protocol Program, which involves the interfacing of two firewire devices, and thus requires advanced programming skills, the two programs that were accomplished were within the researchers' possession of a higher level of programming proficiency.

SCALE 1:0.5cm



RCX CABINET



B. Discussions

The Protocol Programming side of the system posed as a difficult stage since it requires configuring both the computer and the RCX to transmit specific data required in the system. This is the reason why the researchers had to explore ways on applying settings in the program of the RCX and the computer to suit the system design.

The failure of the Protocol Program to run was due to the factors of limited time, knowledge and resources of the researchers. The required parameters were not set, and the appropriate commands and methods were not found, therefore the accuracy of the protocol was not achieved. An appropriate configuration (e.g. the use of the correct methods and commands, the use of the accurate parameters, and the interpretations of arrays) that will allow the transmission of the light values from the RCX to the PC need to be designed and written in Java form. However, since Java is a complex language and the time to learn Java in a more comprehensive scope was constrained, the Protocol Program was not accomplished within the time frame allotted for the performance of this study.

On the other hand, the Card Reading and Time and Attendance Program worked successfully because the program development was focused on their respective devices only. Unlike the Protocol Program, which involves the interfacing of two foreign devices, and thus requires more advanced programming skills, the two programs that were accomplished were ones wherein the researchers possess a higher level of programming familiarity.

CHAPTER V

SUMMARY AND RECOMMENDATIONS

A. Summary

The Card Swipe mechanism was not realized after this study was conducted. Although the Card Reader and the Time and Attendance Programs were successfully made, the Protocol Program that connects the two aforementioned programs was not. The lack of this program did not allow the transfer of data between devices. The values read by the light sensor were not transmitted to the PC. However the following objectives were successfully met:

1. To design a log box that will use a light sensor to read bar-coded cards.
2. To design cards with different bar code combinations—each combination will correspond to a different individual.
3. To create a computer program that will display the employees' name, date, time in or out and store them in a database.

B. Recommendations

Follow-up studies may be conducted to continue the development of the Card Swipe Time-Keeping and Attendance System. Future researchers could accomplish the RCX-to-PC communication or the Protocol Program. To aid in the understanding of this topic, one must obtain sufficient knowledge and background on Java and LeJOS especially about the communication between the RCX and the PC. There must be at least a basic background on Lego Devices specifically those of the Lego Mindstorms Robotics

Inventions Kit. Sources in the Internet are always abundant and information and contact persons can always be found there.

A contingency plan in case of a power failure and security measures that ensure the accuracy of the employees' logs may also be incorporated to the system.

BIBLIOGRAPHY

- Bagnall, Brian. "Lego Mindstorms Programming: RCX Communication". Retrieved last December 2004. (Internet Article) At <http://informit.com/articles/article.asp?p=26439>
- Bagnall, Brian. "Lego Mindstorms Programming: RCX Communication". Retrieved last December 2004. (Internet Article) At <http://phptr.com/articles/article.asp?p=26439>
- Bar-coding. (Internet Article) At <http://idautomation.com/barcoding4beginners.html>
- Bar-coding. (Internet Article) At <http://webermaking.com/html/barcodebook.html#Anchor-In-45470>
- Biometrics. "Facial Recognition". (Internet Article) At <http://ctl.dni.us/biomet%20web/BMFacial.html>
- Biometrics. "Fingerprint Scanning". (Internet Article) At <http://ctl.dni.us/biomet%20web/BMFingerprint.html>
- Biometrics. "Hand Geometry". (Internet Article) At <http://ctl.dni.us/biomet%20web/BMHand.html>
- Biometrics. "Iris Scanning". (Internet Article) At <http://ctl.dni.us/biomet%20web/BMIris.html>
- Chow, Ming. "Teleroobotics Using Lego Mindstorms and Java". Retrieved Last May 8, 2005. (Internet Article) At <http://www.oreillynet.com/pub/wlg/5930>.
- Ferguson. Webster's Illustrated Contemporary Dictionary, Encyclopedic Edition. "Bar codes"
- Firmware. (Internet Article) At <http://www.bandwidthmarket.com/resources/glossary/F3.html>

Frequently Asked Questions. "Rush Hour". (Internet Article) At
<http://esprint.com/new/rushhour-faq.html#what%20is%20a%20log%20box?>

Gockley, Rachel. "Infrared Communication". Retrieved last December 6, 2004. (Internet Article) At <http://contrib.andrew.cmu.edu/~rgockley/legos/ir.html>

Gramercy. Webster's Encyclopedic Unabridged Dictionary of the English Language. "Bar codes". (Random House Incorporated, 1983)

Java Applications. "Superclock". (Internet Article) At
<http://javaapp/superclock/time.ktml>

Java Programming Language. "Compiler". (Internet Article) At
<http://docs.sun.com/db/doc/805-4368/6j450e60>

Java Programming Language. "Data Streaming". (Internet Article) At
<http://java/datastreaming/comm/program.html>

Java Programming Language. "IDE". (Internet Article) At
<http://ipi.apit.edu/glossary.html>

Java Programming Language. "Protocols in Java". (Internet Article) At
<http://webopedia.com/TERM/c/communication-protocol.html>

LEGO Mindstorms RIS. (Internet Article) At
http://www.ideafinder.com/resource/tools/featured_tools/rpt101.htm

LeJOS. (Internet Article) At <http://lejos.sourceforge.com/download.html>

Light Sensor. (Internet Article) At
<http://ec.cmu.edu/education/multimedia/lightsensor.html>

Light Sensor. (Internet Article) At
<http://elecdesign.com/Articles/Print.cmf/ArticleID=6339>

Light Sensor. (Internet Article) At <http://philohome.com/sensor/lightensors.htm>

Light Sensor. (Internet Article) At
http://process_equipment.globalspec.com/learnmore/manufacturing_process_equipment/inspection_tools_instruments/light/color_sensors

McDaniel, George. IBM Dictionary of Computing. "Barcodes." (McGraw-Hill, 1994)

Microsoft XP Professional Help and Support Center

Parker. McGraw Dictionary of Scientific and Technical Terms, 5th edition. "Barcodes." (McGraw-Hill, 1994)

Personal Computer. (Internet Article) At
ccs.uchicago.edu/technotes/misc/Glossary/gloss3.html

Retina Scanner and Biometrics, January 8, 2003, Wednesday. (Internet Article) At
http://prisonplanet.com.eye_scanners_for_school_children.html

Robotics Command Explorer. "Way to Invent." (Internet Article) At
<http://graphics.stanford.edu/~kekoa/rcx.html>

The American Heritage Dictionary, 3rd edition. "Barcode." (Houghton Mifflin Company, 1992)

Time Clocks. "Acroprint Time Q Badge Swipe Time and Payroll Recorders-Standalone Time and Attendance System." At <http://clock/system/acroprint.html>

Time Clocks. (Internet Article) At <http://metrontimeclock.com/story1.html>

Time Clocks. (Internet Article) At <http://metrontimeclock.com/story1.html>

Pictures of Design

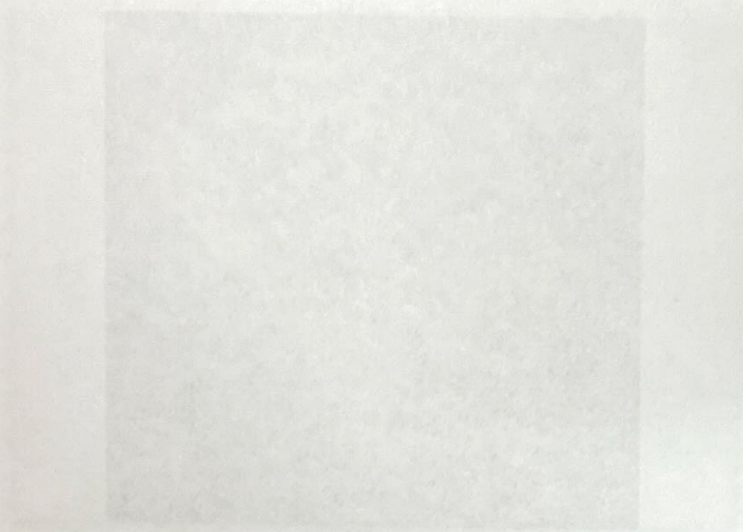


Plate 1. RCX

APPENDIX A

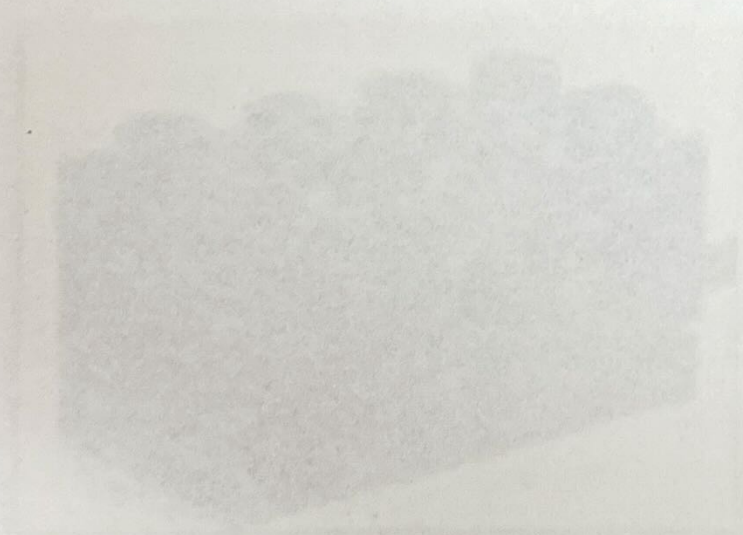


Plate 2. Light Series

Pictures of Devices



Plate 3. IR Tower.

Plate 1. RCX.



Plate 2. Light Sensor.



Plate 3. IR Tower.

PROGRAMS

1) LogInLogout.java

This was the main program that was accessed in order to run the system used in the PC. It consisted of different subprograms that make the structure of the functions.

Subprograms:

a) CSTAS.java

This program created the window that presented two functions:

The Data Administration and the Login and Logout Program (Fig. 1).

b) DataAdmin.java

This program contains the instructions for the Data Administration option. The functions include adding new users, deleting existing users, and changing passwords.

c) Login.java

This program created the login window for the administrators that shows up when the program is first accessed (Fig. 1).

2) Card Reader Program

This program read the cards in terms of the variety of the widths. It was encoded in the computer and downloaded to the RCX.

APPENDIX B

PROGRAMS

1) Loginlogout.java

This was the main program that was accessed in order to run the system based in the PC. It consisted of different subprograms that make the structure of its functions

Subprograms:

a) CSTAS.java

This program created the window that presented two functions:

The Data Administration and the Login and Logout Program (Fig.).

b) DataAdmin.java

This program contains the instructions for the Data Administration option. The functions include adding new users, deleting existing users, and changing passwords.

c) Login.java

This program created the login window for the administrators that shows up when the program is first accessed (Fig.).

2) Card Reader Program

This program read the cards in terms of the variety of the widths. It was encoded in the computer and downloaded to the RCX.

Loginlogout.java

This program is the main program for the system. This should be accessed if the whole system is to be started

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;
import java.sql.*;
import java.io.*;
import java.text.*;
import java.util.Date;
import java.util.Calendar;
```

```
public class Loginlogout extends JFrame
{
    private final int WIDTH = 1024, HEIGHT = 768, FONT_SIZE = 42;
```

```
    JTextField txtCode = new JTextField(""),
        txtName = new JTextField(""),
        txtMess = new JTextField("");
```

```
    JPasswordField pinText = new JPasswordField(10);
```

```
    JLabel lblTime = new JLabel("Time: "),
        lblCurTime = new JLabel(""),
        lblDate = new JLabel("Date: "),
        lblCurDate = new JLabel(""),
        lblCode = new JLabel("Code: "),
        lblPin = new JLabel("Pin: "),
        lblName = new JLabel("Name: ");
```

setting the variables

```
    JButton btnLogin = new JButton("Log in"),
        btnLogout = new JButton("Log out"),
        btnCancel = new JButton("Cancel");
```

```
    JPanel fieldPanel = new JPanel(),
        inputPanel = new JPanel(),
        btnPanel = new JPanel(),
        firstPanel = new JPanel(),
        secondPanel = new JPanel();
```

```
int rows;
Connection myConnection;
Statement myStatement;
ResultSet myResultSet;
String CurrentDate="", getDate="";
```

syntax for connection to the database

```
public Loginlogout()
{
```

```
    setResizable(true);
    setLocation(200,200);
```

```
    txtName.setEditable(false);
    txtMess.setEditable(false);
```

Creates the structure for the display. This include the buttons, text field, table and window size.

```
    openConnection();
    fieldPanel.setLayout(new GridLayout(5,2));
    Date CurrentDate = new Date();
    getDate = DateFormat.getDateInstance().format(CurrentDate);
    lblCurDate = new JLabel(getDate);
```

```
    lblTime.setFont (new Font ("Verdana", Font.BOLD, FONT_SIZE));
    lblDate.setFont (new Font ("Verdana", Font.BOLD, FONT_SIZE));
```



```

lblCurTime.setFont (new Font ("Verdana", Font.BOLD, FONT_SIZE));
lblCurDate.setFont (new Font ("Verdana", Font.BOLD, FONT_SIZE));

lblCode.setFont (new Font ("Verdana", Font.BOLD, 32));
lblPin.setFont (new Font ("Verdana", Font.BOLD, 32));
txtCode.setFont (new Font ("Verdana", Font.BOLD, 32));
pinText.setFont (new Font ("Verdana", Font.BOLD, 32));
txtName.setFont (new Font ("Verdana", Font.BOLD, 32));
txtMess.setFont (new Font ("Verdana", Font.BOLD, 32));

new TimerTime(lblCurTime);
fieldPanel.add(lblTime);
fieldPanel.add(lblCurTime);
fieldPanel.add(lblDate);
fieldPanel.add(lblCurDate);
fieldPanel.add(lblCode);
fieldPanel.add(txtCode);
fieldPanel.add(lblPin);
fieldPanel.add(pinText);
fieldPanel.add(txtName);
fieldPanel.add(txtMess);

inputPanel.setLayout(new BorderLayout());
inputPanel.add(fieldPanel, BorderLayout.PAGE_START);
getContentPane().add(inputPanel, BorderLayout.NORTH);

btnPanel.setLayout(new FlowLayout(FlowLayout.CENTER));
btnPanel.add(btnLogin);
btnPanel.add(btnLogout);
btnPanel.add(btnCancel);
getContentPane().add(btnPanel, BorderLayout.SOUTH);

pack();
setVisible(true);

```

Instructions for actions

```

btnLogin.addActionListener(new ActionListener()

```

```

{
    public void actionPerformed(ActionEvent e)
    {

```

```

        if(txtCode.getText().equals("") || (pinText.getText().equals("")))
        {

```

If no pin or code is
required, "error"
will be inputted

```

            JOptionPane.showMessageDialog(null, "Code and Pin is  
", JOptionPane.INFORMATION_MESSAGE);
        }
    }

```

```

        else
        {

```

If pin or code has
been inputted,
program will get
employee name from
database

```

        /* String.copyValueOf(pinText.getPassword());
        try
        Code.getText();
        String sqlString ="SELECT NAME, PIN from M
        asterfile where code='"+code+"'";

```

```

        Statement p1 =

```

```

myConnection.createStatement();

```

```

        p1.clearBatch();
        ResultSet myResultSet =

```

```

p1.executeQuery(sqlString);

```

```

        if(myResultSet.next())

```



```

        {
myResultSet.getString(1);
String name =
if(pin.equals(myResultSet.getString(2)))
{
String sqlString4 = "SELECT *
from DailyLog where code='"+code+"' AND hours_work=0";
Statement p4 =
myConnection.createStatement();
p4.clearBatch();
ResultSet myResultSet4 =
p4.executeQuery(sqlString4);
if(!myResultSet4.next())
{
Calendar c =
Calendar.getInstance();
int in_hr =
c.get(Calendar.HOUR_OF_DAY);
int in_min =
c.get(Calendar.MINUTE);
System.out.println("Hour:"+in_hr);
System.out.println("Minute:"+in_min);
int tin_hr=0,tin_min=0;
if(in_hr < 7)
{
tin_hr=7;
tin_min=30;
}
else if (in_hr == 7 && in_min
<=30)
{
tin_hr=7;
tin_min=30;
}
else
{
tin_hr=in_hr;
tin_min=in_min;
}
int out_hr =0, out_min=0;
float hrs_work =0.0f;
String sqlString0 = "INSERT into DailyLog
VALUES('"+code+"','"+getDate+"','"+tin_hr+"','"+tin_min+"','"+out_hr+"','"+out_
min+"','"+hrs_work+"')";
int rows = p1.executeUpdate(sqlString0);
String dis_zero="";
txtName.setText(name);
if(in_min <10)
{
dis_zero="0";
}
}
}

```



```

String AM_PM=" AM";
if(in_hr >=12)
{
    AM_PM=" PM";
}
if(in_hr > 12)
{
    in_hr-=12;
}

txtMess.setText(" has log in at
"+in_hr+": "+dis_zero+in_min+AM_PM);
JOptionPane.showMessageDialog(null, name+" has
logged in", "Logged in", JOptionPane.INFORMATION_MESSAGE);
txtCode.setText("");
pinText.setText("");
    }
    else
    {
        JOptionPane.showMessageDialog(null, name+" you have logged in already", "Logged
in", JOptionPane.INFORMATION_MESSAGE);

        txtCode.setText("");
        pinText.setText("");
    }
    else
    {
        System.out.println("Incorrect
Pin");

        JOptionPane.showMessageDialog(null, name+" check your pin", "Login Failed",
JOptionPane.INFORMATION_MESSAGE);

        pinText.setText("");
    }
    else
    {
        JOptionPane.showMessageDialog(null,
"Code Not Found", "Login Failed", JOptionPane.INFORMATION_MESSAGE);
        txtCode.setText("");
        pinText.setText("");
    }
    catch (Exception e1)
    {
        System.out.println(e1.getMessage());
    }
}

});
btnLogout.addActionListener(new ActionListener()
{
    public void actionPerformed(ActionEvent e)
    {
        if(txtCode.getText().equals("") || (pinText.getText().equals(""))
        {
            JOptionPane.showMessageDialog(null, "Code and
Pin is Required", "Error", JOptionPane.INFORMATION_MESSAGE);
        }
        else

```

Displays the
log status of
the employee


```

        {
            String pin =
String.copyValueOf(pinText.getPassword());
            try
            {
                String code = txtCode.getText();
                String sqlString = "SELECT NAME,PIN
from Masterfile where code='"+code+"'";

                Statement p1 =
myConnection.createStatement();

                p1.clearBatch();
                ResultSet myResultSet =
p1.executeQuery(sqlString);

                if(myResultSet.next())
                {
                    String name =
myResultSet.getString(1);

                    if(pin.equals(myResultSet.getString(2)))
                    {
                        String sqlString1 = "SELECT * from DailyLog
where code='"+code+"' AND hours_work=0";

                        Statement p2 =
myConnection.createStatement();

                        p2.clearBatch();
                        ResultSet myResultSet1 =
p2.executeQuery(sqlString1);

                        if(myResultSet1.next())
                        {
                            System.out.println("found");
                            int in_hr =
myResultSet1.getInt(3);

                            int in_min =
myResultSet1.getInt(4);

                            Date CurrDate = new
Date();

                            String date_today =
DateFormat.getDateInstance().format(CurrDate);

                            System.out.println("Hour:"+in_hr);

                            System.out.println("Minute:"+in_min);

                            Calendar c = Calendar.getInstance();
                            int out_hr =
c.get(Calendar.HOUR_OF_DAY);

                            int out_min =
c.get(Calendar.MINUTE);

                            System.out.println("Hour:"+out_hr);

                            System.out.println("Minute:"+out_min);

                            float hrs_work=8.5f;
                            float res_min;
                            float res_hour;

```



```

        int tout_hr=out_hr;
        int lout_hr=out_hr,lout_min=out_min;
        if(out_hr >= 17)
        {
            lout_hr=17;
            lout_min=0;
        }

if(out_min < in_m
{
    res_min = (60+out_min) - in_min;
    tout_hr--;
    res_hour = tout_hr - in_hr;
    hrs_work = res_hour + (res_min/60);
}
else
{
    res_min = out_min - in_min;
    res_hour = out_hr - in_hr;
    hrs_work = res_hour + (res_min/60);
}

System.out.println("Hours work: "+hrs_work);

//String sqlString2 = "UPDATE DailyLog SET
out_hour='"+out_hr+"',out_minute='"+out_min+"',hours_work='"+hrs_work+"' WHERE
Code='"+code+"'";

String sqlString2 = "UPDATE DailyLog SET
out_hour='"+lout_hr+"',out_minute='"+lout_min+"',hours_work='"+hrs_work+"'
WHERE Code='"+code+"' AND hours_work=0";

Statement p3 =
myConnection.createStatement();

p3.executeUpdate(sqlString2);

p2.clearBatch();
myConnection.commit();
txtName.setText(name);
String dis_zero="";
if(out_min <10)
{
    dis_zero="0";
}
String AM_PM=" AM";
if(out_hr >=12)
{
    AM_PM=" PM";
}
if(out_hr > 12)
{
    out_hr-=12;
}

txtMess.setText(" has log
out at "+out_hr+": "+dis_zero+out_min+AM_PM);
JOptionPane.showMessageDialog(null, name+"
has logged out", "Logged out", JOptionPane.INFORMATION_MESSAGE);
txtCode.setText("");
pinText.setText("");
}

```



```

        else
        {
            JOptionPane.showMessageDialog(null, name+" has not logged in", "Logged
out", JOptionPane.INFORMATION_MESSAGE);
            txtCode.setText("");
            pinText.setText("");
        }
        else
        {
            System.out.println("Incorrect
Pin");
            JOptionPane.showMessageDialog(null, name+" check your pin", "Log Out Failed",
JOptionPane.INFORMATION_MESSAGE);
            pinText.setText("");
        }
        else
        {
            JOptionPane.showMessageDialog(null,
"Code Not Found", "Log Out Failed", JOptionPane.INFORMATION_MESSAGE);
            txtCode.setText("");
            pinText.setText("");
        }
        catch (Exception e1)
        {
            System.out.println(e1.getMessage());
        }
    }
});
addWindowListener(new WindowAdapter()
{
    public void windowClosing(WindowEvent e)
    {
        dispose();
    }
});
btnCancel.addActionListener(new ActionListener()
{
    public void actionPerformed(ActionEvent e)
    {
        dispose();
    }
});
}

public void openConnection()
{
    String theSource;
    theSource = "jdbc:odbc:CSTAS";
    try
    {
        Class.forName ("sun.jdbc.odbc.JdbcOdbcDriver");
        myConnection =
DriverManager.getConnection(theSource);
        System.out.println("Connected");
    }
}

```

Closes the connection
to the database


```

        catch (Exception e)
        {
            System.out.println(e.getMessage());
        }
    }

    public void closeConnection()
    {
        try
        {
            myConnection.close();
        }
        catch (Exception e1)
        {
        }
    }
}

public static void main(String[] args)
{
    new Login();
}
}

```

CSTAS.java

```

import javax.swing.*;
import java.awt.event.*;
import java.awt.*;

public class CSTAS extends JWindow
{
    private final int WIDTH = 1024, HEIGHT = 768, FONT_SIZE = 36;
    JMenuBar mainMenu = new JMenuBar();
    JMenu
    fileMenu = new JMenu("File"),
    helpMenu = new JMenu("Help"),
    newMenu = new JMenu("New");
    JMenuItem changeLoginItem = new JMenuItem("Change Login"),
    useractItem = new JMenuItem("User Account"),
    adminAccountItem = new JMenuItem("Admin Account"),
    helpMenuItem = new JMenuItem("About"),
    exitItem = new JMenuItem("Exit");
    JFrame openScreen = new JFrame(),
    mainScreen = new JFrame("Log In Log Out System"),
    newAcctFram = new JFrame();
    JLabel adminLbl = new JLabel("    Data Administration"),
    userLbl = new JLabel("    Log in Log out Program");
    JButton adminBtn = new JButton(new ImageIcon("logoPSHS.gif")),
    userBtn = new JButton(new ImageIcon("relo.gif"));

    JPanel mainPanel = new JPanel();

    String accessLevel;

    public CSTAS(String access)
    {
        accessLevel=access;
        if(!accessLevel.equals("Admin"))
        {

```

//defining variables


```

        useractItem.setEnabled(false);
        adminAccountItem.setEnabled(false);
        adminBtn.setEnabled(false);
    }
    mainProg();
}

public void mainProg()
{
    changeLoginItem.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            mainScreen.dispose();
            new Login();
        }
    });
    mainScreen.addWindowListener(new WindowAdapter()
    {
        public void windowClosing(WindowEvent e)
        {
            System.exit(0);
        }
    });
    exitItem.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            System.exit(0);
        }
    });
    adminBtn.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            System.out.println("asdasd");
            new DataAdmin();
        }
    });
    userBtn.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            new Loginlogout();
        }
    });
    useractItem.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            new NewUserAcct();
        }
    });
    helpMenuItem.addActionListener(new ActionListener()
    {

```

//instructions


```

        public void actionPerformed(ActionEvent e)
        {
            JOptionPane.showMessageDialog(null,
"CREATED BY:\nNoreen Marian Bautista\nChina May Molina\nPatrick Quezon\n",
"Research Project", JOptionPane.INFORMATION_MESSAGE);
        }
    });
    //////////
    adminAccountItem.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            new NewAdminAcct();
        }
    });

    adminLbl.setFont (new Font ("Verdana", Font.BOLD, FONT_SIZE));
    userLbl.setFont (new Font ("Verdana", Font.BOLD, FONT_SIZE));
    mainScreen.setResizable(false);
    mainScreen.setJMenuBar(mainMenu);
    mainMenu.add(fileMenu);
    mainMenu.add(helpMenu);    //////////
    fileMenu.add(newMenu);
    helpMenu.add(helpMenuItem);
    newMenu.add(useractItem);
    newMenu.add(adminAccountItem);
    fileMenu.add(changeLoginItem);
    fileMenu.add(exitItem);
    mainPanel.setPreferredSize(new Dimension(1024,768));
    mainPanel.setLayout(new GridLayout(2,2));
    mainPanel.add(adminBtn);
    mainPanel.add(adminLbl);
    mainPanel.add(userBtn);
    mainPanel.add(userLbl);
    mainScreen.getContentPane().add(mainPanel);
    mainScreen.pack();
    //mainScreen.setLocation(350,200);
    mainScreen.setVisible(true);
    System.out.println(accessLevel);
}
}

```

DataAdmin.java

```

import java.sql.*;
import java.lang.*;
import javax.swing.*;
import java.awt.event.*;
import java.awt.*;
import javax.swing.table.DefaultTableModel;

public class DataAdmin extends JWindow {

    public DataAdmin()
    {
        launch();
    }

    boolean isempty = false;
    String[] columnNames = {"Code","Log Date","Hours Work"};
    String[] rowData = {"","","",""};
}

```



```

DefaultTableModel myTable;
JFrame frame = new JFrame("Data Administration Module");
JPanel  textPanel = new JPanel(),
        inputPanel = new JPanel(),
        buttonPanel = new JPanel();
//defining variables

JTable QuestionTab;

JLabel  fromLabel = new JLabel ("From :"),
        toLabel = new JLabel ("To:"),
        yearLabel = new JLabel ("Year: ");
//displays the menu

JButton  indLog = new JButton("Individual Log"),
        allLog = new JButton("All Log"),
        viewContent = new JButton("View Content"),
        quit = new JButton("Exit");

JTextField fromText = new JTextField(),
        toText = new JTextField(),
        yearText = new JTextField();

JTextArea resultText = new JTextArea(1,40);

final String choices[] = { "Jan 1","Jan 16","Feb 1","Feb 16",
                           "Mar 1","Mar 16", "Apr 1","Apr 16",
                           "May 1","May 16","Jun 1","Jun 16",
                           "Jul 1","Jul 16","Aug 1","Aug 16",
                           "Sep 1","Sep 16","Oct 1","Oct 16",
                           "Nov 1","Nov 16","Dec 1","Dec 16",};

JComboBox fromChoice = new JComboBox(choices);

JScrollPane scr1Questions;

int rows;
Connection myConnection;
Statement myStatement;
ResultSet myResultSet;

public void handleException(Exception e1)
{
    resultText.setText("Error: " + e1.getMessage());
    e1.printStackTrace();
    if (e1 instanceof SQLException)
    {
        while ((e1 = ((SQLException)
e1).getNextException()) != null)
        {
            System.out.println(e1);
        }
    }
}

public void buildTable()
{
    myTable = new DefaultTableModel();
    myTable.addColumn(columnNames[0]);
    myTable.addColumn(columnNames[1]);
    myTable.addColumn(columnNames[2]);
    myTable.addColumn(columnNames[3]);
}

```

creates the structure for
the window

opens connection to
database

```
try
{
String sqlString = "SELECT * from DailyLog";
myStatement = myConnection.createStatement();
myResultSet = myStatement.executeQuery(sqlString);
int row = 0;
while(myResultSet.next())
{
    rowData[0]=myResultSet.getString(1);
    rowData[1]=myResultSet.getString(2);
    rowData[2]=myResultSet.getString(7);
    myTable.addRow(rowData);
    row++;
}

if(row==0) {
    System.out.println("Table is empty");
    boolean isempty = true;
    System.out.println(row);
}
catch (Exception e1)
{
}
}

public void launch()
{
    resultText.setForeground(new Color(200,201,175));
    resultText.setBackground(Color.darkGray);
    textPanel.add(resultText);
    inputPanel.setLayout(new GridLayout(3,2));
    inputPanel.add(fromLabel);
    inputPanel.add(fromChoice);

    inputPanel.add(toLabel);
    inputPanel.add(toText);
    inputPanel.add(yearLabel);
    inputPanel.add(yearText);

    toText.setEditable(false);
    openConnection();
    buildTable();
    QuestionTab = new JTable(myTable);

    frame.addWindowListener(new WindowAdapter()
    {
        public void windowClosing(WindowEvent e) {
            closeConnection();
            frame.dispose();
        }
    });

    fromChoice.addItemListener(new ItemListener()
    {
        public void itemStateChanged(ItemEvent check)
        {

```



```

        int fchoice=fromChoice.getSelectedIndex()+1;
        if (fchoice==1)
            toText.setText("Jan 15");
        else if (fchoice==2)
            toText.setText("Jan 31");
        else if (fchoice==3)
            toText.setText("Feb 15");
        else if (fchoice==4)
            toText.setText("Feb 29");
        else if (fchoice==5)
            toText.setText("Mar 15");
        else if (fchoice==6)
            toText.setText("Mar 31");
        else if (fchoice==7)
            toText.setText("Apr 15");
        else if (fchoice==8)
            toText.setText("Apr 30");
        else if (fchoice==9)
            toText.setText("May 15");
        else if (fchoice==10)
            toText.setText("MAy 31");
        else if (fchoice==11)
            toText.setText("Jun 15");
        else if (fchoice==12)
            toText.setText("Jun 30");
        else if (fchoice==13)
            toText.setText("Jul 15");
        else if (fchoice==14)
            toText.setText("Jul 31");
        else if (fchoice==15)
            toText.setText("Aug 15");
        else if (fchoice==16)
            toText.setText("Aug 31");
        else if (fchoice==17)
            toText.setText("Sep 15");
        else if (fchoice==18)
            toText.setText("Sep 30");
        else if (fchoice==19)
            toText.setText("Oct 15");
        else if (fchoice==20)
            toText.setText("Oct 31");
        else if (fchoice==21)
            toText.setText("Nov 15");
        else if (fchoice==22)
            toText.setText("Nov 30");
        else if (fchoice==23)
            toText.setText("Dec 15");
        else
            toText.setText("Dec 31");
    }
});

```

```

indLog.addActionListener(new ActionListener()

```

```

{
    public void actionPerformed(ActionEvent e){

```

```

        try

```

```

        {

```

```

            String

```

```

codeInput=JOptionPane.showInputDialog("Enter Code to Search");
            System.out.println(codeInput);

```

```

            for(int i=0;i<myTable.getRowCount();i++)

```



```

        {
            myTable.removeRow(i);
            i--;
        }
        int row = 0;
        /*String sqlString = "SELECT * from DailyLog";
        myStatement = myConnection.createStatement();
        myResultSet =
myStatement.executeQuery(sqlString);
        int row = 0;
        while(myResultSet.next())
        {
            rowData[0]=myResultSet.getString(1);
            rowData[1]=myResultSet.getString(2);
            rowData[2]=myResultSet.getString(7);
            myTable.removeRow(row);
            row++;
        }*/

```

```

String sqlString1 = "SELECT * from DailyLog where
CODE='"+codeInput+"'";
        myStatement = myConnection.createStatement();
        myResultSet =
myStatement.executeQuery(sqlString1);

```

```

        while(myResultSet.next())
        {
            rowData[0]=myResultSet.getString(1);
            rowData[1]=myResultSet.getString(2);
            rowData[2]=myResultSet.getString(7);
            myTable.addRow(rowData);
            row++;
        }

        if(row==0) {
            System.out.println("Table is empty");
            boolean isempty = true;
            System.out.println(row);
        }
        catch (Exception e1)
        {
        }
    }
}
});
allLog.addActionListener(new ActionListener()
{
    public void actionPerformed(ActionEvent e)
    {
        try

```

```

        {
            String sqlString = "SELECT * from
DailyLog";
            myStatement =
myConnection.createStatement();
            myResultSet =
myStatement.executeQuery(sqlString);
            int row = 0;
            while(myResultSet.next())
            {

```



```

rowData[0]=myResultSet.getString(1);
rowData[1]=myResultSet.getString(2);
rowData[2]=myResultSet.getString(7);

myTable.addRow(rowData);
row++;
}

if(row==0) {
    System.out.println("Table
is empty");

    boolean isempty = true;
    System.out.println(row);
}
catch (Exception e1)
{
}

});

viewContent.addActionListener(new ActionListener()
{
    public void actionPerformed(ActionEvent e)
    {
        if(scr1Questions.isVisible()) {
            scr1Questions.setVisible(false);
            frame.pack();
        }
        else{
            scr1Questions.setVisible(true);
            frame.pack();
        }
    }
});

quit.addActionListener(new ActionListener()
{
    public void actionPerformed(ActionEvent e)
    {
        closeConnection();
        frame.dispose();
    }
});

JPanel rowlbutn = new JPanel();
rowlbutn.add(indLog);
rowlbutn.add(allLog);
rowlbutn.add(viewContent);
rowlbutn.add(quit);
buttonPanel.setLayout(new BorderLayout());
buttonPanel.add(rowlbutn,BorderLayout.NORTH);
scr1Questions = new JScrollPane(QuestionTab);

scr1Questions.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_ALWAYS);

scr1Questions.setHorizontalScrollBarPolicy(JScrollPane.HORIZONTAL_SCROLLBAR_ALWAYS);
scr1Questions.setPreferredSize(new Dimension(50, 150));

```



```

scrQuestions.setMinimumSize(new Dimension(10, 10));
buttonPanel.add(scrQuestions, BorderLayout.SOUTH);

frame.getContentPane().add(textPanel, BorderLayout.NORTH);
frame.getContentPane().add(inputPanel, BorderLayout.CENTER);
frame.getContentPane().add(buttonPanel, BorderLayout.SOUTH);

int rowvalue = 1;
if(isempty==false)
    rowvalue =
1+Integer.parseInt(myTable.getValueAt(myTable.getRowCount()-1,0).toString());
frame.pack();
frame.setVisible(true);
}

```

```

public void openConnection( )
{
    String theSource;
    theSource = "jdbc:odbc:CSTAS";
    try
    {
        Class.forName ("sun.jdbc.odbc.JdbcOdbcDriver");
        myConnection =
DriverManager.getConnection(theSource);
        resultText.setText("Connected");
    }
    catch (Exception e)
    {
        handleException(e);
    }
}

```

```

public void closeConnection( )
{
    try
    {
        myConnection.close( );
    }
    catch (Exception e1)
    {
        handleException(e1);
    }
}

```

```

class DuplicateIDException extends Exception
{
    public void DuplicateIDException() {
    }
}

```

```

class RecordNotFoundException extends Exception
{
    public void RecordNotFoundException() {
    }
}

```

```

}

```


Login.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;
import java.net.*;
import java.sql.*;
import java.io.*;
```

```
public class Login extends JFrame
{
```

```
    JTextField txtUsername = new JTextField("");
    JPasswordField passTxt = new JPasswordField(10);
```

defining variables

```
    JLabel lblUsername = new JLabel("Username: "),
        lblPass = new JLabel("Password: ");
```

```
    JButton btnLogin = new JButton("Login"),
        btnCancel = new JButton("Cancel");
    JPanel fieldPanel = new JPanel(),
        inputPanel = new JPanel(),
        btnPanel = new JPanel();
```

```
    int rows;
    Connection myConnection;
    Statement myStatement;
    ResultSet myResultSet;
```

```
    public Login()
    {
```

```
        ///////
        setTitle("Login");
        setResizable(false);
        setLocation(400,250);
```

```
        ///////
```

```
        openConnection();
        fieldPanel.setLayout(new GridLayout(2,2));
        fieldPanel.add(lblUsername);
        fieldPanel.add(txtUsername);
        fieldPanel.add(lblPass);
        fieldPanel.add(passTxt);
        inputPanel.setLayout(new BorderLayout());
        inputPanel.add(fieldPanel, BorderLayout.PAGE_START);
        getContentPane().add(inputPanel, BorderLayout.NORTH);
        btnPanel.setLayout(new FlowLayout(FlowLayout.CENTER));
        btnPanel.add(btnLogin);
        btnPanel.add(btnCancel);
        getContentPane().add(btnPanel, BorderLayout.SOUTH);
        pack();
```

//creates structure for
//window

```
        setVisible(true);
        btnLogin.addActionListener(new ActionListener()
        {
```

```
            public void actionPerformed(ActionEvent e)
            {
                if(txtUsername.getText().equals(""))
```

//instructions

```
                JOptionPane.showMessageDialog(null, "Username is  
Required for Login", "Username Error", JOptionPane.INFORMATION_MESSAGE);
            }
        }
    }
```



```

else
{
    String password =
string.copyValueOf(passTxt.getPassword());
    try
    {
        String username = txtUsername.getText();
        String sqlString = "SELECT PASSWORD,ACCESS
from User where USERNAME='"+username+"'";
        Statement pl = myConnection.createStatement();
        pl.clearBatch();
        ResultSet myResultSet =
pl.executeQuery(sqlString);
        if(myResultSet.next())
        {
            if(password.equals(myResultSet.getString(1)))
            {
                String access =
myResultSet.getString(2);
                System.out.println(access);
                System.out.println("Log in
successful");
                JOptionPane.showMessageDialog(null,
"Log In Successful", "Login Successful", JOptionPane.INFORMATION_MESSAGE);
                new CSTAS(access);
                closeConnection();
                dispose();
            }
            else
            {
                System.out.println("Password
failed");
                JOptionPane.showMessageDialog(null, "Check your password", "Login
Failed", JOptionPane.INFORMATION_MESSAGE);
            }
        }
        else
        {
            JOptionPane.showMessageDialog(null,
"Username Not Found", "Login Failed", JOptionPane.INFORMATION_MESSAGE);
        }
    }
    catch (Exception e1)
    {
        System.out.println(e1.getMessage());
    }
}

});
addWindowListener(new WindowAdapter()
{
    public void windowClosing(WindowEvent e)
    {
        System.exit(0);
    }
});
btnCancel.addActionListener(new ActionListener()

```



```

    {
        public void actionPerformed(ActionEvent e)
        {
            System.exit(0);
        }
    }
};

}

public void openConnection()
{
    String theSource;
    theSource = "jdbc:odbc:CSTAS";
    try
    {
        Class.forName ("sun.jdbc.odbc.JdbcOdbcDriver");
        myConnection =
DriverManager.getConnection(theSource);
        System.out.println("Connected");
    }
    catch (Exception e)
    {
        System.out.println(e.getMessage());
    }
}

public void closeConnection()
{
    try
    {
        myConnection.close( );
    }
    catch (Exception e1)
    {
    }
}

}

public static void main(String[] args)
{
    new Login();
}

}

```

//opens a connection
//to the database

SensorReader.java

```

import java.io.*;
import josx.rcxcomm.*;
import josx.platform.rcx.*;

```

```

/**
 * This program is a sample program
 * that successfully read information from
 * the card
 */

```

```

public class SensorReader {
    public static void main(String args[]) {
        int sensorID, sensorValue;
        RCXPort port = null;
        try {

```



```

port = new RCXPort();
DataOutputStream out = new DataOutputStream(port.getOutputStream());
while (true) {
    sensorID = port.getInputStream().read();
    sensorValue = Sensor.readSensorValue(sensorID, 0);
    LCD.showNumber(sensorValue);
    out.writeShort(sensorValue);
}

```

ReadSensor.java

```

import java.io.*;
import josx.rcxcomm.*;

/**
 * This program is a sample program
 * that was tested to transmit
 * light values to the PC via the IR link
 */

public class ReadSensor {

    public static void main(String[] args) {

        try {

            RCXPort port = new RCXPort();

            InputStream is = port.getInputStream();
            OutputStream os = port.getOutputStream();
            DataInputStream dis = new DataInputStream(is);
            DataOutputStream dos = new DataOutputStream(os);

            System.out.println("Reading Light Sensor");
            int sendTime = (int)System.currentTimeMillis();
            for(int i=0;i<20;i++) {
                dos.writeByte(1);
                dos.flush();

                int n = dis.readShort();

                System.out.println("Received " + n);
            }
            System.out.println("Time = " + ((int)System.currentTimeMillis() -
sendTime));
        }
        catch (Exception e) {
            System.out.println("Exception " + e.getMessage());
        }
    }
}

```