

Improving Operational Efficiency with a Customized Management
Information System for Equipment Management

تحقيق كفاءة تشغيلية أعلى من خلال تصميم نظام معلومات إداري مُخصص لإدارة وصيانة
المعدات بشكل فعّال

By

RAED MOHAMMED AL-KUBATI

241001386

Supervision

DR. ASMA'A ABU ANZAH

Year

2024 / 1446

Copyright Notice

This research and all associated materials are the intellectual property of the University. Unauthorized use, reproduction, or distribution of any part of this content is strictly prohibited without the university's express consent.

Researcher

Abstract

In today's data-driven world, effective Management Information Systems (MIS) are essential for supporting informed decision-making and operational efficiency. This research develops a customized MIS tailored to the Saudi Diesel Equipment Company (SDEC) needs to improve equipment management processes. SDEC faces unique challenges related to delays in accessing vital equipment information, impacting customer satisfaction and operational productivity. Building on previous studies that highlight the limitations of traditional data management, this research integrates user-friendly interfaces and robust database technology to streamline data handling, enhance data accuracy, and deliver actionable insights through comprehensive analytics.

The designed MIS will optimize maintenance scheduling, improve equipment tracking, and support strategic, data-driven decision-making across the organization. This study contributes to scientific knowledge by examining the impact of automated data handling, analyzing the effectiveness of a customized graphical user interface (GUI), and providing insights applicable to other industries with similar operational needs. Ultimately, the study aims to enhance SDEC's operational efficiency, foster sustainable growth, and align the organization with evolving industry demands.

المستخلص

في عالم اليوم القائم على البيانات، تُعد نظم المعلومات الإدارية الفعّالة ضرورية لدعم اتخاذ القرارات المستنيرة وتعزيز الكفاءة التشغيلية. يهدف هذا البحث إلى تطوير نظام معلومات إداري مُخصص يلبي احتياجات الشركة السعودية لمعدات الديزل، لتحسين عمليات إدارة المعدات. تواجه الشركة تحديات فريدة تتعلق بالتأخير في الوصول إلى المعلومات الأساسية عن المعدات، مما يؤثر على رضا العملاء وإنتاجية العمليات. استناداً إلى الدراسات السابقة التي تسلط الضوء على محدودية أساليب إدارة البيانات التقليدية، يقوم هذا البحث بدمج واجهات سهلة الاستخدام مع تقنيات قواعد بيانات متقدمة لتبسيط معالجة البيانات، وتعزيز دقتها، وتوفير رؤية قابلة للتنفيذ من خلال تحليلات شاملة للمعلومات.

سيساهم النظام المصمم في تحسين جدولة الصيانة، وتعزيز تتبع المعدات، ودعم اتخاذ القرارات الاستراتيجية المستندة إلى البيانات على مستوى الشركة. كما يضيف هذا البحث إلى المعرفة العلمية من خلال دراسة تأثير الأتمتة في معالجة البيانات، وتحليل فعالية واجهة المستخدم الرسومية المُصممة خصيصاً، وتقديم رؤية قابلة للتطبيق في صناعات أخرى تواجه احتياجات تشغيلية مشابهة. يهدف البحث في النهاية إلى تعزيز الكفاءة التشغيلية لشركة المعدات السعودية، مما يدعم النمو المستدام ويجعل الشركة تتماشى مع المتطلبات المتطورة لإدارة المشاريع في منشأتها.

Acknowledgment

First and foremost, I extend my deepest gratitude and praise to Allah, the Almighty, for granting me the strength and opportunity to complete my master's degree and this final project report. This accomplishment would not have been possible without the support and contributions of many individuals, each of whom I wish to acknowledge here.

I am incredibly grateful to my wife, Pery, and friends, whose encouragement and unwavering support have been instrumental in this journey. My sincerest thanks also go to my project supervisor, Dr. Asma'a Abu Anzah, whose expert guidance, support, and encouragement have been invaluable throughout this project. Additionally, I would like to express my heartfelt appreciation to Dr. Eng. Khalid Hamad for his generous assistance with data collection and invaluable insights.

I am deeply indebted to the lecturers of the Faculty of Management, who have imparted knowledge that has broadened my understanding and opened my mind to new possibilities. Lastly, I wish to thank all my friends who have supported me and provided continuous support and motivation to complete my studies. May Allah bless you all abundantly.

Table of Contents

CHAPTER 1 INTRODUCTION	12
1.1 Overview	12
1.2 Problem Statement	16
1.3 Study Purposes	19
1.3.1 Study Importance	20
1.4 Project Scope	21
1.5 Report Organization	22
CHAPTER 2 LITERATURE REVIEW	24
2.1 Introduction	24
2.2 Information Management Concept	31
2.2.1 Information System Concept	32
2.2.2 Document and Record Management	32
2.2.3 Collaborative Tools	33
2.2.4 Information System in Inventory Management	33
2.3 Management of Equipment	34
2.3.1 Information System in Quality of Service	35
2.3.2 Planned Preventive Maintenance for Equipment	36
2.4 Database and Database System	38
2.4.1 Database Management System	39
2.4.2 Database System Hardware	40
2.4.3 Database System Software	41
2.5 Database Models	43
2.5.1 Conceptual Models	43
2.5.2 Implementation Models	44
2.6 Objective Oriented Programming	45
2.7 Summary	48

CHAPTER 3: METHODOLOGY	49
3.1 Introduction	49
3.2 System Study	49
3.3 Database Design	50
3.3.1 Tables Data Configuration	51
3.5 Development of Graphic User Interface	53
3.5.1 System Overview	53
3.6 Flow Chart of the Project	55
CHAPTER 4 RESULT AND DISCUSSION	57
4.1 Database Dictionary	57
4.1.2 Database Structure	61
4.2 Data Flow Mechanism	62
4.2.1 Data Flow (level 1)	62
4.3 User Interface Design	63
4.3.1 Login	63
4.3.2 Main Menu	Error! Bookmark not defined.
4.3.3 Profile	Error! Bookmark not defined.
4.3.4 Inventory	70
4.3.5 Work Order	Error! Bookmark not defined.
4.3.6 PPM Schedule	Error! Bookmark not defined.
4.3.7 PPM Notification	Error! Bookmark not defined.
4.3.8 Reports	80
4.3.9 Advance Search	Error! Bookmark not defined.
4.3.10 Reference Tables	Error! Bookmark not defined.
4.3.11 EMIS	Error! Bookmark not defined.
CHAPTER 5 CONCLUSION AND RECCOMENDATION	Error!
Bookmark not defined.	

5.1 Conclusion	Error! Bookmark not defined.
5.2 Recommendation for future work	90
Questionnaire	92
REFERENCES	Error! Bookmark not defined.
APPENDIX A	Error! Bookmark not defined.
APPENDIX B	Error! Bookmark not defined.

List of Tables

Table 4.1: Equipment Table and its Attributes and Description	57
Table 4.2: Site Table and its Attributes and Description	59
Table 4.3: Customer Table and its Attributes and Description	59
Table 4.4: Log in Table and its Attributes and Description	59
Table 4.5: Location Table and its Attributes and Description	60
Table 4.6: Work Order Table and its Attributes and Description	60

List of Figures

Figure No.	Description	Page No.
1.1: Information System Mechanism		13
3.1: Aspects of data collection		50
3.2: Table Data Configuration		52
3.3: System Overview		54
3.4: System Flowchart		56
4.1: Database Structure		61
4.2: Context Diagram		62
4.3: Data Flow Diagram (Level1)		63
4.4: Login Form		64
4.5: Error Message of Wrong Login		64
4.6: Login Process		65
4.7: Main Menu		67
4.8: Main Menu Hierarchy Diagram		69
4.9: Profile Dropdown Menu		70
4.10: The Equipment Form (The equipment details)		71
4.11: The Equipment Form (PPM schedule)		72
4.12: Customer Form		73
4.13: Inventory for Customer		73
4.14: Inventory for Equipment		74
4.15: Work Order Dropdown Menu		75
4.16: Work Order Form		76
4.17: Work Order Status Viewer		77

4.18: PPM Schedule Dropdown Menu	77
4.19: PPM Search for August Month	78
4.20: PPM Notification flowchart	79
4.21: PPM Notification	80
4.22: Reports Dropdown Menu	81
4.23: PPM Report	82
4.24: Work Order Report	83
4.25: Print Preview for WO Report	84
4.26: Advance Search Dropdown Menu	85
4.27: Searching by Type of Assist Form	85
4.28: Reference Tables Dropdown Menu	86
4.29: Reference Tables for Equipment	87
4.30: Reference Tables for Work Order	87
4.31: Reference Tables for Customer	88
4.32: About EMIS Form	88

CHAPTER 1

INTRODUCTION

1.1 Overview

In Today's Driven World Where Information Is Critical To Successful Management Practices – Data Holds The Key To Success Like Never Before In Helping Organizations Grow And Make Informed Decisions At Every Level Of Management Hierarchy Within A Company Or Institution Alike Enter The Management Information System (Mis) A Well-Designed System That Streamlines The Gathering And Processing Of Essential Information, For Enhanced Organizational Productivity And More Intelligent Decision Making Processes.

Past Research Has Shown The Difficulties Linked To Old School Data Management Methods. For Example, Chen And Colleagues in 2010, Pointed Out The Shortcomings Of Handling Data. It Is Suggested That Automation Be Used To Improve Precision And Processing Speed. Similarly, A 2014 Study By Elmasri And Navathe Showed That Database Systems Are More Effective Than File Systems When It Comes To Data Copy Stability And Retrieval. However, This Study Shows That The Emerging Consensus That Has Led To Stronger Trust In Data And Increased User Satisfaction With Its Operational Efficiency Demonstrates The Importance Of Integrated Systems. The Main Objective Of This Study Is To Create A Special Design For The Saudi Diesel Equipment Company (Sdec) That Includes The Management Information System (MIS)For This Company. The Technology Will Be Integrated Into The Database, And Its User-

Friendly Interface And The Improvement Of The Quality Of Services And Current Shortcomings Will Be Done By Focusing On The Management Of The Equipment System And Improving The Efficiency Of Its Operation.

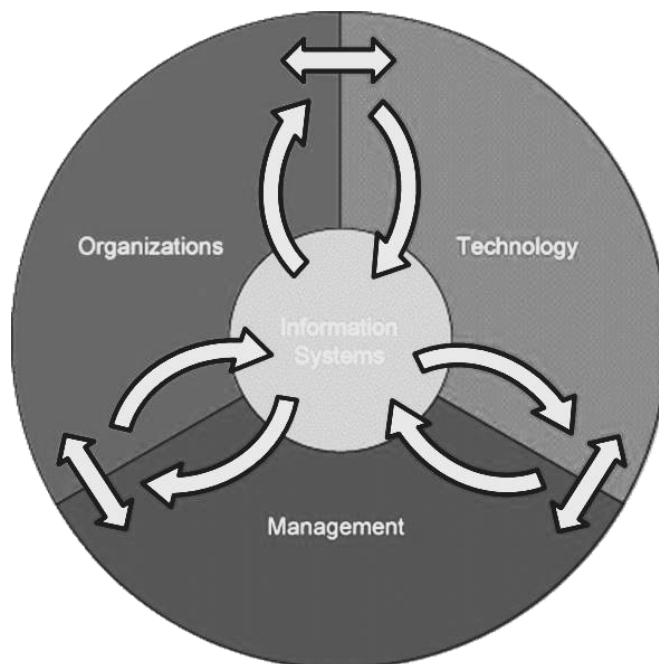


Figure 1.1: Information System Mechanism (Source: Best, 1988)

(Source: Adams, G.B. And Ingersoll, V.H, 1985)

Figure 1.1 Information Systems Are More Than Computers. Successfully Leveraging Information Systems Demands Understanding The Organization, Management, And the Information Technology Shaping These Systems. All Information Systems Are Organizationally And Managerially Solutions To The Challenges Posed By The Environment (Adams, 1985). Management Information Systems (Mis) Are Designed As Systems That Enable An Organization To Capture, Process, And Disseminate The Data Essential For Managing Organizations In The Best Way. Information Generated By Mis Is A Critical Necessary Ingredient For Sound Decision Making In Business And Management. (Davis, 1995).

Following A Defined Approach To Developing And Using Management Information Systems (MIS) Within A Company Is Vital For Both Bank Executives And Occ Inspectors (Ambler, 2006). Mis Should Adhere To Guidelines On Rules, Policies, And Practices That Are Uniformly Implemented Throughout The Organization In Creating, Maintaining, And Utilizing All Mis.

Management Frequently Applies Management Information Systems (Or MIS) Across Organizational Levels To Align With Its Long Term Strategic Goals And Objectives, As Atkinson Et Al. (1989) Mentioned. Additionally, The Accounting Systems That Play A Role In Maintaining Proper Control Over Financial Record-Keeping Processes Are Critical.

The Use Of Information Systems (IS) Has Opened Up Possibilities And Offered Remedies For Specific Challenges Encountered By Manufacturing Sectors. Businesses Have Utilized This To Optimize Their Processes, increase The Worth Of Their Products And Services, And Introduce Approaches To Conducting Business (According To Bancelhon Et Al., 1992).

Over The Years, The Manufacturing Sector Has Faced Challenges And Obstacles In Managing Many Equipment. There Has Been Increasing Demand From Customers For Service And Enhanced Satisfaction Levels. Furthermore, During This Period, The Industry Also Encountered Challenges Such As The Downturn In 2008 And The Impact Of The Natural Disaster In Japan In 2011 (Churcher, 2012). In North America, The Manufacturing Sector Workforce Exceeds 3 Million Individuals. This Underscores The Importance Of Integrating Management Information Systems In Manufacturing Operations. Such Integration Can Lead To Cost Savings In Production. Contribute To Enhancing Customer Satisfaction Across Dimensions (Blaha And Premeralni, 1997).

About Ten Years Ago, The Prevailing System In Most Small Businesses Was Based On The Use Of Paper, And The Managers Of These Companies Used Paper To Track All The Important Information Through It. All These Papers Were Placed In Storage In The File Cabinet In The Form Of Bundles And Folders, Each Of Which Had A Name Indicating What It Contained Inside. The Complete Reliance On Arranging This Data Inside The Files And Folders Was For The Intended Purpose Of Storing This Information. This Is The Ideal Way To Place Each File And Folder To Ensure The Coherence Between The Content And Not To Lose Information. For Example in A Hospital Environment, All Files And Folders Should Contain Details For Each Case, With A Folder Allocated For Each Patient Containing All The Necessary Information, Such As His Medical History And Previous Examination Records.

The Data Collected Was Typically Minimal, And The Organization's Manager Had Limited Reporting Requirements; Thus, The Manual Structure Effectively Functioned As A Data Repository (Chen Et Al., 2010) And Beyond As Companies Evolved And Reporting Standards Became Managing Data Manually In A File Format Became Increasingly Challenging. In Practice, Accessing And Utilizing Data In A Setting Has Become Time Consuming (Dittrich, 1986).

Unfortunately, Gathering Data From A Filing System Can Be Time Consuming And Burdensome. Despite Having A Designed System, Some Business Leaders Still Struggle With Meeting Government Mandated Reporting Deadlines That Require Weeks Of Dedicated Work Each Quarter. This Underscores The Need To Create Reporting Processes,

As Outlined By Elmasri And Navathe In 2014, A Computer-Based System Was Designed To Monitor And Generate Reports.

The Inherent Challenges Of File Systems Make Using A Database System Compared To The File Systems Multiple Isolated Files Approach. The Database Organizes Related Data Into A Single Storage Unit, Unlike The File System, Which Contains Disjointed Files. As A Result, The Database Transforms The Way Data Is Stored, Retrieved, And Managed By End Users. The Database Management System (Dbms) Provides Advantages Compared To File System Management As It Helps To Eliminate Most Of The Data Inconsistency Issues And Dependencies Associated With The File System Organization (Adapted From Dennis's Work In 2015).

It Is Important To Know That The Database Management System Is One Of The Parts That Play An Important Role In Database Systems. It Is Similar To The Smooth And Efficient Functioning Of The Heart Inside The Human Body That Drives It To Work. However, Just As Our Bodies Need Other Parts Besides The Heart To Function And Maintain Health, The Database System Also Requires Multiple Database Management Systems That Work Together In Harmony (As Discussed By Warner And Fossen In 1995).

1.2 Problem Statement

Companies Face The Growing Challenge Of Managing Their Equipment And Resources In Today's Changing Business World. This Challenge Is Quite Pronounced At Businesses Like Saudi Diesel Equipment Company (Sdec), Where The Variety Of Equipment And Services Has Been Rising, While Product Life Cycles Are Getting Shorter, And There Is A Move Towards Catering To Markets. The Current Systems Mainly Concentrate On

Assessment, Leaving Deficiencies In Tasks Like Inventory Control And Intricate Equipment Management.

A Significant Problem Highlighted In Sdec Is The (Issue Of Delays In Obtaining Information) Which Makes Up Around 75% Of Customer Grievances, According To A Study From 2014. Oftentimes, The Team Faces Many Challenges In Accessing Details Such As Warranty Periods And Planned Maintenance Schedules. This Leads To Customer Dissatisfaction Due To Low Efficiency. This Highlights The Need For A Management Information System That Combines Inventory Control And Equipment Monitoring Within The Structure.

Previous Research Findings

These Are The Most Prominent Of These Studies And Challenges Associated With The Practice Of Traditional Data Management, For Example:

- Chen Et Al. (2010) Confirmed That Manual Handling Of Data Is Ineffective And That Delivering This Data Through Automation Helps To Enhance Accuracy And Theft In Processing This Data.**
- Elmasri And Navathe (2014) Confirmed That Comparing Systems Within Databases With Systems Within Traditional Files Improves Data Consistency And Retrieval Capabilities, Explaining How They Work.**
- Mueller (2019) Confirmed The Importance Of Acid Properties In Maintaining Data Integrity For Transaction Management In Database Systems.**

These Studies Collectively Point Toward A Growing Consensus On The Necessity For Automated integrated Systems That Enhance Data Reliability And Operational Efficiency.

Contribution To Scientific Knowledge

This Research Project Seeks To Enhance Our Understanding By Creating A Management Information System (Mis) Customized To Meet The Unique Requirements Of Handling Equipment At Sdec (Sustainable Development And Environment Company). By Integrating Database Technology With User Interfaces, This Study Will Provide Insights Into The Following;

- Exploring Data Handling Techniques: The Research Investigates Automated Methods For Managing And Accessing Data, Focusing On Improving The Shortcomings Identified In The Studies.**
- Exploring The Effects Of A Crafted Graphical User Interface (Gui) On Enhancing User Satisfaction And Operational Productivity.**
- Practical Uses: The Results Will Provide Advice On Implementing Systems In Other Companies Dealing With Similar Issues.**

Contribution To The Community And Examples.

The Study Focuses Mainly On Saudi Arabian Manufacturing And Service-Oriented Organizations Prioritizing Equipment Management As Crucial To Their Operations.

- Enhanced Operational Efficiency Results In Decreased Information Access Delays And Enhanced Decision-Making Processes For Businesses.**
- Organizations Can Improve Customer Satisfaction By Optimizing Inventory And Equipment Management Processes To Meet Customer Demands And Enhance Service Efficiency.**

- **Aims For Studies: This Research Sets The Groundwork For Studies On Advancing Management Information Systems Across Diverse Industries To Enhance Comprehension Of Efficient Information Handling Systems.**

1.3 Study Purposes

Main Objective

To Establish A Management Information System (Mis), We Are Working To Improve Equipment Management And Operational Efficiency At Saudi Diesel Equipment Company (Sdec). This Includes Integrating Inventory Control And Monitoring Features To Streamline Operations Effectively.

Sub-Objectives

- **Improve Data Management: To Enhance The Accuracy And Efficiency Of Handling Data Related To Equipment Maintenance, Automated Data Management Methods Must Be Implemented.**
- **Designing Interfaces For Users: Create A User Interface (Gui) Using C And C++ Programming Languages To Enhance Smooth Communication With The Mis System.**
- **Setting Up Databases: Build A Database Using The Entity Relationship Model To Store And Manage Equipment-Related Data And Retrieve It Efficiently.**

- **Enhance The System With Features That Allow Tracking And Managing Inventory In Time To Maximize The Use Of Equipment Resources.**
- **Improving Customer Satisfaction: Evaluate How The New User Interface (Ui) Affects User Happiness And Productivity By Gathering User Feedback And Analyzing Performance Measures.**
- **Features For Reporting And Analysis: Integrate Reporting Features Into The Management Information System (Mis) To Offer Information On Equipment Effectiveness, Maintenance Timetables, And Stock Quantities.**
- **Maintenance Management System: Implementing A Scheduled Maintenance (Spm) System Can Help Guarantee That Equipment Is Regularly Maintained To Minimize Downtime And Improve The Quality Of Service Provided.**
- **Create Training Resources. Assist Employees In Utilizing The Management Information System (Mis) To Ensure A Seamless Transition And Successful Integration Of The New System.**

1.3.1 Study Importance

Scientific Importance

- **Contribution To Knowledge**

The Aim Of This Study Is To Discover The Best Methods For Dealing With Data, Know How To Manage It In An Innovative Way And To Contribute To The Scientific Literature By Enhancing And Understanding Information Systems And Their Management.

- **Foundation For Future Research**

This Research Sets A Framework By Integrating Database Technology With User Interfaces, Which Facilitates The Path To More Studies In Management Information Systems Designed Specifically For Multiple Industries.

- **Improvement Of Theoretical Models**

Providing Insight Into Effective Data Organization And Retrieval Methods Enhances The Results Of Theoretical Information Management Models.

Practical Importance

- **Enhanced Operational Efficiency**

This Study Addresses The Real Challenges In Equipment Management And Provides Practical Solutions That Can Reduce The Problem of Delays In Accessing Information And Improve Decision-Making Processes.

- **Increased Customer Satisfaction**

The Aim of This Research Is To Directly Improve Service Quality And Customer Satisfaction Levels In Manufacturing And Service-Oriented Organizations By Improving Inventory And Equipment Management Processes.

- **Guidance For Implementation**

The Study Will Provide Actionable Recommendations for Organizations Facing Similar Challenges, Which May Facilitate The Implementation Of An Effective Management Information System In Their Operations.

1.4 Project Scope

The Objective Of This Project Is To Create And Implement A Management Information System To Handle Different Types Of Equipment. This Tool Enables The Development And Utilization Of A Database Via An Interface For Swift And Direct Access To Essential Information By Manufacturers And Organizations.

The Project Consists of Two Parts. The Database and The User Interface, With The Database Being Developed To Set Up Tables For Storing All The Equipment-Related Information And Data. At The Same Time, The User Interface Will Be Developed To Create A User Menu To Efficiently Perform Database Tasks And Monitor Data. The Data Collection Process Will Focus On A Company In Saudi Arabia With Limited Resources And Capabilities, including Some Confidential Information Related To The Company's Customers.

1.5 Report Organization

This Report Consists of Five Sections. Let Us Delve Into The Content Of Each Chapter.

- Chapter 1 Introduces The Basic Principles Of The Project, Briefly Touches On The Idea Of The Equipment Management System, And Provides Information About The Reliability Of The System To Better Understand Its Application And Importance. The Project Objectives And Expected Features Define The Scope Of The System Design.**
- Chapter 2 Of The Thesis Explores The Background Literature And Historical Development Of The System That Led To Its Development And Integration Over Time. It Also Examines Studies And Research Initiatives That Aim To Clarify The Design Process And The Basic Principles Of Its Operation.**

- **Chapter 3 Of The Project Report Focuses On Explaining The Methodology And Providing A Diagram Of The Database Approach. In Addition, It Discusses The Design Of The User Interface To Facilitate User Interaction With The System And Explains How To Link The Database To The Graphical User Interface (Gui).**
- **Chapter 4 Should Present The Results Of The Database And The Results Of The Graphical User Interface (Gui) including A Hierarchical Diagram Of The System List With Its Functions, The Initial Screen Layout, And Each Model Interface Within The Gui.**
- **Chapter 5 Concludes The Project By Summarizing All Findings And Observations, Engaging In A Discussion Of The Project Itself, And Offering Suggestions For Strengthening And Expanding This Endeavor.**

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

Management Information System (MIS) Is Worked on As A Way To Organize And Share Information And As A Department Within The Structure Of The Organization. Using A Database System, This System Enhances The Collection, Association, And Visualization Of Information Across The Organization. Sambo (1992) Describes It As A System Used At Every Level In All Areas To Provide Management Methods With Information Derived From Internal And External Channels. Lucy (1989), Such An Approach, Believes That Management Information Systems (Mis) Serve As A Platform For Discussing The Constraints That Link The Tasks Of Dealing With Data In 2010, Transforming Data Into Information After It Was Sourced, And Delivering It In An Arrangement That Suits Management At All Levels For Effective Performance.

Theoretical Framework of The Study

This Study Is Based on The Foundations Of Management Information Systems As Database Management Practices. The Theoretical Model Delves Into How These Systems Improve Efficiency Within Organizations While Ensuring Data Accuracy And Optimizing User Interface Design.

Management Information Systems (Mis):

Mis Is Designed To Provide The Necessary Data For Making Sound Decisions Through Different Stages Of Management. It Integrates Internal And External Data Sources To Develop The Productivity Of The Organization And Provide Distinguished Services.

Database Management Systems (Dbms):

Dbms Are Very Essential, As They Process Transactions With High Efficiency To Maintain The Accuracy Of Information. For This Reason, Its Importance Lies In Ensuring The Safety Of These Systems In Addition To Storing And Coherent Data.

User Interface Design:

The Main Goal Through Which A Graphical User Interface (Gui) Was Created Is To Increase User Satisfaction By Providing An Effective Way To Interact With The Database System. This Is Achieved By Utilizing Programming Languages Such As C And C++ To Create An Interface.

Independent Variable

One Of The Most Important Tasks In Creating A Database Is Designing The User Interface And Developing System Features, Such As Inventory Management And Equipment Handling When Developing A Management Information System (Mis).

Dependent Variable

Operational Effectiveness And User Satisfaction Are Factors In Assessing The Impact Of Systems On Tasks Such As Data Access And Inventory Management To Improve The User Experience Of The Company.

What Will The Study Be Based On?

The Research Will Be Based On The Scientific Method Approach And Will Combine Primary And Secondary Data Sources For Analysis. It Will Address Interviews And Surveys With Users Of Management Information Systems To Collect Initial Data, While The Literature In The Company's Records Will Be Searched To Obtain Secondary Data. Hypotheses Will Be Formulated To Be Consistent With The Objectives Of This Study And Research Inquiries. The Basic Hypothesis Of This Research Study Is As Follows;

The Design And Implementation Of The Management Information System Do Not Appear To Be Related To Efficiency And User Satisfaction.

This Central Idea Leads To The Creation Of Hypotheses To Investigate The Detailed Connections Among Different Elements Of The Management Information System And The Resulting Outcomes.

The Research Will Use Methods And Software To Examine The Gathered Data And Assess The Proposed Hypotheses. This Will Include Methodologies, Like Correlation Analysis And Regression Modeling, To Ascertain The Significance Of The Connections Between The Variables.

Summary Of Past Studies

Chen And Colleagues Discussed The Difficulties Associated With Data Processing Tasks In 2010. This Research Paper Mainly Emphasizes The Drawbacks Of Data Management Practices And How They Lead To Delays And Errors That Lead To Lower Levels Of Overall Accuracy. This Research Emphasizes The Importance Of Improving Efficiency And

Accuracy In Adopting Automated Data Management Systems. When It Comes To Initiatives Aimed At Developing Data Processing Capabilities introducing Automation Reduces Errors Made By Humans. It Also Speeds Up Processes To Keep Up With The Need To Deal With The Increasing Amounts Of Data In Companies.

Masry And Navathy (2014) Explore The Distinction Between Database Systems And Traditional File Systems; In This Research Study, The Effectiveness Of Database Systems Is Compared To File Systems, Showing That Database Systems Provide Consistency And Retrieval Functions For Data. This Study Highlights The Benefits Of Databases That Improve Organization, Reduce Redundancy, And Provide Direct Access To Data. This Finding Holds Significance For Initiatives Aimed At Developing Data Storage Solutions, Underscoring The Importance Of Database Systems In Enhancing Data Integrity And Accessibility.

Markus And Colleagues (2013) Investigated The Importance Of Coding In Database Management Systems (Dbms); This Study Examines The Impact Of Coding In Dbms, Focusing On Its Role In Improving The Effectiveness And Features Of Databases. The Investigations Revealed That Incorporating Coding Into Dbmss Can Lead To Improved Transaction Processing And Query Optimization While Enabling Ad Hoc Operations. For Teams Working On Database Projects, This Study Emphasizes The Importance Of Using Coding Techniques To Build More Dynamic And Versatile Databases.

“Mueller’s 2019 Study Examines The Complexities Of Transaction Management And The Importance Of Acid Properties.” In Mueller’s Paper, He Delves Into Transaction Processing Within Databases. He Emphasizes The Importance Of The Acid Principles—Atomicity, Consistency, Isolation, And Durability—In Maintaining Data Integrity Standards. The

Study Emphasizes The Need For Databases To Ensure That Transactions Are Processed Reliably And Consistently To Avoid Data Accidents Or Loss. “The Efforts Of Organizations Seeking To Maintain First-Class Data Reliability Standards And Ensure Compliance With Acid Guidelines Become Essential For Trustworthy Transaction Management.

“Advances In Database Systems As Discussed By Zilio And Colleagues In 2012.” This Research Delves Into The Development Of Database Technologies, Specifically Examining Advances In Efficient Data Storage And Retrieval. It Emphasizes Improvements Such As Improved Data Organization And The Shift To Cloud-Based Storage That Have Transformed How We Access And Scale Data Systems. The Findings Highlight Critical Technological Trends And Improvements To Database Integration To Ensure That Projects Meet Current Efficiency And Scalability Standards.

“The 2018 Study By Nyborg And Maksimchuk Focused On The Tools Needed To Create Applications That Interact With Databases.” Nyborg And Maksimchuk Explore How Integrated Development Environments (Ides) Impact Database Application Development By Significantly Improving User Interface Design And System Usability. This Study Shows That Using An Ide Facilitates The Development Process And Increases Efficiency. These Results Support The Combination Of Ides In Projects That Focus On User Interfaces And Smooth Development Processes To Benefit Both Developers And End Users From A Well-Designed System.

“Sia And Rossi (2011) Explored The Evolution Of Database Systems Over Time. They Predicted Trends In The Field.” This Research Delves Into The Evolution Of Database Systems Over Time. It Predicts Trends, Such As The Emergence Of Nosql Databases And Cloud-Based Solutions In The Future Landscape. The Study Emphasizes Advances In Data

Management. It Makes Predictions About Where The Industry Is Headed. For Those Seeking To Protect Their Database Systems For Tomorrow's Needs, This Research Provides Insights Into Technological Changes To Help Keep Designs Relevant In An Ever-Changing Landscape.

“Decleva’s 2015 Research Investigated How Maintenance Impacts Information Systems.”
In His Research, Decleva Explores How Maintenance Impacts The Reliability And Effectiveness Of Information Systems By Revealing That Inadequate Maintenance Leads To Decreased Service Quality And System Reliability Over Time. This Highlights The Importance Of Implementing A Maintenance Strategy For Any Project To Ensure That Continuous Updates And Repairs Are Made To Maintain The System's Performance And Functionality.

“Wiederhold’s 2017 Study Examines The Applications Of Management Information Systems (Mis) In Inventory Management.” This Research Investigates How Mis Can Effectively Improve Inventory Management Procedures According To Wiederholds’ Findings, Which Show That Mis Plays A Role In Avoiding Stockouts And Optimizing Inventory Levels To Efficiently Enhance Effectiveness. This Study Recommends The Use Of Mis For Projects That Focus On Inventory Systems To Streamline Inventory Control And Enhance Operational Productivity.

“Effectiveness Of Reminder Systems In Maintenance Tasks By Kettinger And Lee, 2016.”
In Their Research, Kettinger And Lee Explore The Impact Of Reminder Systems In Increasing The Effective Completion Rate Of Maintenance Tasks. Their Research Indicates That Automated Notifications Contribute To Increasing The Efficiency Of Completing Tasks And Improving The Reliability Of Systems In Tasks That Involve Maintenance Work;

Incorporating A Reminder System Can Ensure That Essential Updates And Repairs Are Carried Out Promptly To Prevent Any Downtime.

Previous And Current Studies

1. Automation And Efficiency

- **Past Studies, Such As That By Chen Et Al. In 2010, Have Found Problems With Data Management. It Is Suggested That Automated Processes Be Used To Reduce Errors And Improve Efficiency.**
- **Recent Research Focuses On Exploring Advanced Automation Technologies Such As Intelligence And Machine Learning To Manage Data Sets And Real-Time Data Processing Needs.**

2. Data Integrity And Reliability

- **In Research, Scholars Such As Elmasri And Navathe (2014) Have Focused On Ensuring Data Integrity Through The Development Of Database Systems And Emphasizing The Importance Of Acid Properties To Maintain Consistent Transactions.**
- **Exploring Research Involves Examining Distributed Databases And Blockchain Technology To Maintain Data Integrity In Locations While Prioritizing Privacy And Security Measures.**

3. Programming And System Design

- **(In A 2013 Study, Marcos Et Al.). Emphasized The Use Of Programming Techniques To Enhance Database Efficiency.**

- **Research Is Currently Focusing On Delving Into Methods Such As Serverless Computing And Microservices To Improve Scalability And Adaptability.**

4. Technological Advancements

- **In Research, Experts Such As Zilio And Colleagues Predicted Trends Like Cloud Based Services And Relational Databases In 2012.**
- **Let Us Expand On These Studies By Delving Into Data Analysis Techniques, Exploring The Potentials Of Nosql Systems, And Leveraging Machine Learning For Optimization Purposes.**

5. Maintenance And Usability

- **In A Study From 2015, Deklev Emphasized The Importance Of Maintenance To Ensure The Reliability Of Systems.**
- **New Research Findings Suggest Employing Intelligence In Maintenance And User Experience Design To Develop More User Friendly Systems.**

2.2 Information Management Concept

The Concept Of Information Management Involves The Ability Of An Organization To Generate And Maintain Information So That Individuals Can Access It When Needed Through Appropriate Communication Channels At A Reasonable Cost. This Information Is Crucial To Decision-Making Processes (Langimo 1980). According To Best (1988) information Management Involves Efficiently Handling The Flow Of Information, Both Within And Outside The Organization, To Enhance Performance Levels By Effectively

Coordinating Production Processes And Storage Mechanisms. However, This Definition May Be Restrictive Because It Fails To Include Aspects Of Information Management, Such As Equal Access, Ownership Rights Of Content, And Ensuring Its Dissemination Among Employees And Resources Involved In The Process. As Robey Stated In 1979, The Primary Focus Is On Handling Information Within The Organization Through The Use Of Advanced Information Technologies.

2.2.1 Information System Concept

The Fast Evolution Of Computer Technology Has Sparked A Growing Interest Among People Seeking Computer Assistance To Solve Their Challenges And The Intricate Issues They Encounter. These Challenges Were Previously Thought To Be Within Human Nature And Decision-Making Processes, Particularly Within Organizations, Just A Few Years Ago.

The Significance Of Information Systems Is On The Rise Within Forward Thinking Organizations (Flemings And Von Halle in 1990). It Is Essential To Establish Procedures For Implementing Management Systems And Using Databases Efficiently To Ensure Cost-Effective Access In Organizations. Management Information And Information Systems Play A Role In Assisting The Decision-Making Process Within An Organization, According To Schewe (1979). For Instance, Mis Is Viewed As An Asset. Information Systems Are Mechanisms Designed To Take Data As Material And Transform It Through One Or More Processes To Produce Information (Davis Et Al., 1995).

2.2.2 Document And Record Management

Document And Record Management Plays A Role In Any Organization's Information System Setup. It Impacts Sectors Such As Accounting, Manufacturing, Financial Services,

Marketing, And Human Resources (As Mentioned By Dittrich In 1986). Once Data Is Available, An Information System Can Be Likened To A Library. It Is Structured In Line With The Organization's Needs And Guidelines, As Outlined By Gremillion And Pyburn In 1988. This Ensures That The Data Within The Organization Is Easily Accessible To Those Who Require It Promptly. Location And Retrieval Information Can Be Managed Smoothly And Logically With The System's Planning And Design In Place. Creating A Guide On Organizing And Indexing The Information Is Crucial In The System Design Phase. It Is Essential For Any Information System To Access And Retrieve Data And Maintain The Archived Information As Up To Date As Possible (Shah Et Al. 1989).

2.2.3 Collaborative Tools

Collaborative Tools Come In The Form Of Both Hardware And Software And Act As A Means To Share Information Externally Among Users To Facilitate The Exchange Of Data And Resources Effectively While Also Enabling Real Time Communication (Mirvis Et Al. 1991).

2.2.4 Information System In Inventory Management

Inventory Control Systems Are Essential In The Manufacturing Industry As They Are Mainly Employed To Keep Track Of The Quantity Of Each Product At Stock Levels Across Departments Like Sales Logistics And Production Engineering Within Manufacturing Operations Involved In Manufacturing Resource Planning (Mrpii).

Introduced To Integrate These Divisions Using A Management Information System, Which Has A Broader Scope Than Conventional Material Requirements Planning (Mrpi), As Noted

By Teoreyet AL.(1986). Mrpii Should Not Be Viewed As A System Due To Its Reach; It Should Instead Be Considered A Fundamental Part Of Corporate Culture (Alshawhi 2001).

Many Companies Often Use An Inventory Management System To Prevent Stock Levels And Shortages. The Tools For Organizing Inventory Data Were Developed To Meet The Organization's Requirements. They Previously Relied On Methods Like Copies And Spreadsheets For Inventory Counting, Which Was Time Consuming And Prone To Inaccuracies (Weiderhold, 1987). Typically Linked To And Akin To The Distribution System Is The Idea That Distributors Who Can Operate With Money Invested In Inventory Enjoy An Advantage Over Their Competitors (Flemming Et AL., 1990).

Keeping Track Of Inventory Is Crucial In The Manufacturing Industry As It Helps In Creating Work Orders And Bills Of Materials That Simplify The Production Process By Assisting Manufacturers In Arranging The Components And Equipment Needed For Tasks (Based On Research By Gremillion And Pyburn in 1988). To Handle Manufacturing Tasks Effectively, Manufacturers Can Generate Work Orders And Bills Of Materials With A Specified Series Of Steps That Need To Be Executed To Build The End Product. In The Realm Of Managing Inventory Systems, The Responsibility Lies In Overseeing Work Orders And Automated Processes, As Proposed By Wiederhold And Elmasri Back In 1979. For Manufacturers in Particular, The Practice Of Inventory Management Proves Advantageous By Enabling Them To Monitor Assets And Streamline The Handling Of Inventory Related Activities That Are Common Among Businesses In Different Sectors.

2.3 Management Of Equipment

Managing Equipment Is Crucially Important, Especially When It Comes To Manufacturing Machinery, Due To The Investments Involved And Their Impact On The

Potential Risk To Human Lives, The High Sensitivity Of Such Equipment, The Costly Maintenance Requirements, And The Relatively Short Lifespan, As Noted By Hawryszkiwycz In 1991.

Effectively Handling And Using Equipment Is Crucial For Maintaining An Organization's Efficiency And Keeping Its Equipment In Good Condition Since It Plays A Role In Companies And Organizations. However inadequate Understanding Of Equipment Choices, Skills Acquired, Maintenance And Repair Costs Management, Proper Support Systems, And Practical Leadership Abilities Lead To The Wastage Of Limited Resources In Certain Countries. This Can Ultimately Result In A Decline In The Quality Of Healthcare Services Provided.

2.3.1 Information System In Quality Of Service

According To Dekleva (1992), Companies Have Long Depended On Information Systems To Enhance The Effectiveness And Efficiency Of The Services They Provide To Clients. However, If These Systems Are Not Properly Maintained Within The Company, It May Result In A Reduction In The Quality Of Services The Company Offers. It Is Observed That Addressing Error Corrections And System Failures Can Frequently Interfere With The Execution Of System Upgrades And Additions When Examining The Relationship Between The Quality Of Information Systems And Their Maintenance Procedures. Information Systems (Is) Management Proponents Stress The Need To See Is Strategy As A Viewpoint On How Businesses Function And Are Run (Chen Et Al., 2010).

If A Company's Customers Lack Confidence In The Information Systems Being Utilized For The Services They Receive, They Might Consider Switching To Another Organization (Kim And Lochovsky, 1989). According To The Literature in The Information Systems (Is)

Field, Customers Gauge An Organization's System Quality By Assessing The Level Of Service Offered And The Minimum Standard Of Service They Are Willing To Accept. For Instance, A Customer Might Decide Not To Avail Of The Services Offered By A Company Because The Technology Being Utilized Is Outdated Or Incapable Of Meeting The Customer's Requirements.

Some Writers Have Explored The Technical Challenges Associated With Information System Production In The Manufacturing Sector (For Example, Manchanda Et Al., 2009). However, Limited Investigations Exist On Information System Quality And Upkeep In Organizational Settings Where Management Plays A Role In Preserving The Functionality Of Their Systems. It Can Be Extremely Difficult To Balance The Need For Managers In An Organization To Protect The Integrity Of The Information System With The Goal Of Providing High Quality Services To Customers (Agus 2004).

2.3.2 Planned Preventive Maintenance For Equipment

Scheduled Maintenance, Or Ppm Includes A Variety Of Tasks That Must Be Performed On Equipment To Ensure It Operates At Its Best Over Time. These Tasks Include Ensuring That Equipment Parts Are Properly Tested, Cleaned, Lubricated, And Calibrated For Wear, Tear, Or Potential Failure. The Key To Planned Maintenance (Ppm) Is User Involvement And Dedication To The Process From The Beginning. Users Must Take Responsibility For Engage In Maintenance Activities. Actively Take Part In Carrying Them Out Regularly.

Users Are Advised To Carry Out Maintenance Tasks And Collaborate With The Technical Engineer On A Weekly Basis For Any Necessary Follow Up Work. For Technical Fixes Under

The Engineer's Purview, It Is Advisable To Schedule Them Every Six Months As Per Howes's (2001) Recommendation.

In Order To Create A Planned Maintenance System, It's Crucial To Maintain A Well-Maintained Record System in Place That Is Organized Effectively And Efficiently. The Maintenance Manual Provided By The Equipment Manufacturer Can Be Incorporated Into A Program For Implementation Of The System. If Computer Accessibility Is Not Feasible, An Alternative Option Would Be To Establish A Filing System. The Administrative Structure For Planned Maintenance Necessitates The Inclusion Of The Following Elements;

- **Equipment Inventory**

All Medical Equipment Must Be Documented And Logged Into The System Under The Supervision Of The Hospital's Service Workshop, With Details Such As Its Whereabouts And Records Of Repairs And Maintenance Conducted By The Manufacturer Or Suppliers.

- **Definition Of Maintenance Task**

Maintaining Every Piece Of Equipment In A Dependable State Requires Clearly Outlining The Necessary Tasks And Efforts Involved, Which Can Be Determined By Referring To The Manufacturer's Documentation And Details About The Product.

- **Establishing Intervals Of Maintenance**

The Frequency At Which Tasks Should Be Carried Out Depends On The Nature Of The Task At Hand – A Utilized Item Would Require More Frequent Cleaning And Inspection

Compared To One That Is Used Less Frequently; However, It Is Essential To Establish Minimum Standards To Adhere To In Any Case; While The Manufacturer's Manual Recommendations Can Offer Guidance On The Frequency Of Maintenance Procedures; It Is Ultimately The Actual Usage That Should Dictate The Required Maintenance Routine.

- **Reminder System**

Maintenance Of Tools And Machinery Requires Attention And Care. After The Equipment Has Been Sorted, The Maintenance Program Must Be Carefully Implemented. It May Be Important To Implement A Reminder System As It Is Useful To Ensure That The Team Members Responsible For This Are Informed Of The Scheduled Maintenance Tasks.

Whether Using A Physical Card Index Or A Digital Computer Program, It Is Essential To Document The Scheduled Date For Each Equipment Maintenance Session. The Manager Should Prepare By Identifying The Tasks That Need To Be Completed And Creating A Schedule Each Week (As Suggested By Kittinger And Lee In 1997).

- **Special Test Equipment**

Those In Charge Of Managing And Maintaining Equipment Should Have Access To Testing Tools To Ensure That Medical Equipment Operates Correctly And Meets Electrical Safety Standards.

- **Technical Library**

Make Sure You Have Access To A Library That Includes Installation Guides And Spare Parts Manuals For Each Piece Of Equipment, Electronic And Component Data Books, And Relevant Technical References Stored Together.

2.4 Database And Database System

In Terms, A Database Is Like A Storage Space For Information. This Suggests That Creating Databases Is Essentially An Extension Of A Practice That Dates Back To The Stages Of Writing. It Can Be Linked To The Outcome Of Any Form Of Record Keeping Activity That Predates The Computer Era. However, This Explanation May Seem Broad For Contexts, And Thus, It Is Further Defined As Our Understanding Evolves (According To Siau And Rossi, 2011).

A Database System Must Be Established To Run A Business Effectively. We Refer To A Business As An Endeavor Encompassing Endeavors Such As An Airline Company, An Institution, Or A Manufacturing Firm, To Even Something As Diverse As Collecting Stamps Or Keeping Track Of Contacts For Sending Out New Year Greetings (Stephens 2010).

Throughout This Project, We Use An Example Of A School Database System. The School Manages Information About Its Students And Teachers including Courses Offered And Grades Earned By Students. We Also Keep A Record Of Advisors' Duties Toward Students And Aspects Of School Operations That We Will Delve Into. These Data Elements Form The Data. In Other Words, The Essential Information Needed For The School To Function Correctly. Data Used In Operations Is Collected From Student Application Forms, Enrollment Records, Grade Sheets, And Schedules. This Information Is Then Leveraged To Produce Output Data, Like Transcripts, Enrollment Documents, Administrative Reports, Etc.

Note That No Computer Is Essential For Utilizing Such A Database; A School In The 1930s Would Have Kept The Same Database In Paper Form. Then Again, The Presence Of

Computers To Store And Control The Data Changes Client Desires: We Hope To Store More Data And Use It More Refinedly (Zilio Et Al., 2004).

2.4.1 Database Management System

A Database Management System (Dbms) Is A Collection Of Data Storage Entities (Hardware, Software, And Users) That Helps An Organization Effectively Manage Its Daily Data Operations. For Example, If Our School Has 10,000 Students Each Year, And Each Student Creates About Ten Records A Year Over The Course Of Ten Years, Then That School Has A Million Grade Records. It Is Quite A Challenge To Extract Records Meeting Criteria From This Particular Dataset; Following Today's Standards, This Collection Of Records Seems Somewhat Limited In Scope! With The Trend Of "Grade Inflation" In Mind, A Key Question We Could Tackle Is Analyzing The Progression Of Grades In Introductory Programming Courses For 10 Years. It Is Evident That Efficient Data Retrieval Plays A Role In Database Systems (Teorey, 1990).

Many Database Management Systems Handle Users Trying To Access Shared Data Items, Sometimes The Same Data Item. For Instance, Let Us Consider Setting Up An Automated Student Registration System Where Students Can Register Using Terminals Or Computers (Sabharwal 2001). The Database Is Assumed To Include Information On Course Capacities And Current Seat Availability. Few Students Are Interested In Signing Up For The Cs210 Course In The Spring Semester Of 2003; The Course Has Limited Capacity. Cannot Accommodate All Enrollment Requests. If There's One Seat Remaining In The Class And It's The Seat Up For Grabs, The Database Needs To Handle Conflicting Requests And Permit A Single Registration (Referencing Fahrner Et Al., 1995).

2.4.2 Database System Hardware

Database Management Systems Are Typically Installed In General-Purpose Computers, As Highlighted By (Rumbaugh Et Al. 1991), Due To The Influence Of Hardware Attributes In Shaping The Development Of Dbms Platforms.

It Is Helpful To Categorize Computer Memory Into Two Types. Interior Memory And External Memory. For Our Purposes Today! While Some Internal Memory, Like Rom, Is Permanent And Cannot Be Altered By Programs' Stored Data, We Are Focusing Solely On Flexible Memory That Can Be Modified By Software Applications In This Discussion. This Type Of Memory Is Commonly Referred To As Ram. However, It Is Worth Noting That It Is Volatile, Which Means Any Power Supply Interruption Can Lead To Data Loss (Dittrich Et Al., 2012).

Regarding Storage Options, Disks And Tapes Of Information Storage Are Considered Basic Types Of Technology Used In Computing Systems For Saving Data Over Long Periods Without Power Loss Or Degradation In Content Quality. According To The Implementation, A Magnetic Tapes-Based Environment Will Only Access Data Because It Cannot Provide Filtration On Time Fully, So It Is Normally Used For Data Backup. When Analyzing The Memory Requirements Of A Database Management System (Dbms), We Must Consider The Following;

- The Data Of A Database Management System Should Be Persistent – The Data Must Be Available Even After The Tasks Are Done By The User Of The System.**

Furthermore, The Data Must Remain Intact Even If The System Encounters Disruptions.

- A Database Management System Needs To Access Data Efficiently.**
- Storing Such A Volume Of Data Should Be Done Using A Cost Storage Medium.**

The Requirements Are Currently Met In Today's Era Through Magnetic Disks.

2.4.3 Database System Software

Users Interact With Database Systems Using Query Languages To Retrieve And Manipulate Data Effectively In A Dbms Environment Characterized By Its Data Structures And Retrieval Methods. Data Recovery Is The Process Of Retrieving Information Stored In A Database That Meets Criteria Set By The User In A Search Query. Data Updates Involve Modifying Data By Making Changes Or Additions And Deleting It. Programming In Database Management Systems (Dbms) Languages Is Done Differently Than Programming In High Level Programming Languages Like C Or Pascal. Projects Written In C, Pascal, Or Pl / 1 Directly Carry Out An Algorithm To Solve A Problem (Elmasri And Navathe, 2014). When A Question Is Phrased In A Database Query Language, It Simply Outlines The Problem Without Delving Into The Specifics Of Writing The Code To Address It. This Task Is Typically Left To A Group Within The Team. "Database Management System Programming Was Discussed In A Study By Marcos And Others In 2003."

This Programming Approach Is Known As The Procedural Coding Style In Computer Programming. One Of The Functions Of Database Management Systems (Dbmss) Is Transaction Management, Which Involves Handling A Sequence Of Database Operations That Need To Be Executed At All—A Concept Referred To As Atomicity. The Best Way To Illustrate This Concept Is To Transfer Money Between Accounts A And B In A Bank Database. The Money Transfers Should Not Affect The Amount Of Money In The Bank Accounts. This Is An Important Condition For Consistency In The Database, As Mueller Pointed Out In 1999.

Typically, At Any Moment In Time, Transactions Happen Simultaneously In The Database System. The Transaction Management Component Ensures That The Progress Of Any Transaction Does Not Influence The Progress Of One Transaction. This Is Known As The Isolation Property Of Transactions. Ultimately, The Effect Of A Transaction On The State Of The Database Must Be Permanent, Meaning That It Must Persist In The Database Even After The Transaction Is Completed. This Is Referred To As The Transaction Durability Property. In Short, The Basic Properties Of Transactions Mentioned Above Are Commonly Referred To As The Acid Properties, Which Stand For Atomicity, Consistency, Isolation, And Durability. Typically, Database Management System Software Includes Tools For Application Development In Addition To Query Languages. These Tools Serve The Purpose Of Aiding In User Interface Development And Encompass Frameworks And Nonprocedural Programming Languages That Blend Database Querying With Various User Interfaces, Etc. (Naiburg And Maksimchuck, 2001).

2.5 Database Models

In Databases, A Database Model Is A System Of Organization That Portrays How Data Is Structured And Connected Within The Database Itself. There Are Two Categories Of Database Models. Conceptual Models And Implementation Models (Navathe And Gadgil, 1982).

2.5.1 Conceptual Models

The Entity Relation (E-R) Based Data Model Stands As A Lofty Conceptual Framework Showcasing Entity Attributes And Connections Information. All Discussed In Chapter 3 Previously. An Entity Relation (E-R) Based Illustration Depicts The Database Structure Overall By Portraying A Scenario Through Objects And Their Interconnections (As Noted By Sasa In 1992). In The World Of Data Modeling Tools Like The One Mentioned Here (Lightstone Et Al., 2010), Their Primary Purpose Is To Showcase The Data Needs And Connections Among Entities (Referred To As Tables In A Database). In Management Information Systems (Mis), Tables Play A Role In Organizing And Streamlining The System's Data Flow By Breaking It Down Into Manageable Units Of Information. Each Entity Consists Of Attributes (Fields), Where Each Attribute Is A Piece Of Data Associated With A Database Item. A Specific Type Of Attribute Known As A Key Serves As An Identifier For An Individual Entity Within The Database. For Example, The Customerid Acts Here Like A Key For The Equipment Entity In The Database System Along With Attributes Like Customername And Customeraddress, Etc., Which Are Essential For Describing The Connections Within Various Database Entities And Enabling Efficient Cross-Table Queries According To Marshall And Mckay (2000). In Databases, We Typically Encounter Three Types Of Relationships Among Entities:

- In One, To One Relationships Each Item In The Table Corresponds To One Item In The Second Table.**
- In One Too Many Relationships, Every Entry In The Table Can Match With Entries In The Second Table, While Each Entry In The Second Table Matches With Just One Entry In The First Table.**

- **Too Many Relationships In Databases Occur When Each Entry In The Table Links To Multiple Entries In The Second Table And Vice Versa (Referenced From Loomis Et Al., 1987).**

2.5.2 Implementation Models

The Focus Is Put On How The Information Is Stored In The Database Or How The Data Structures Are Used To Depict What Is Shown To Users. The Implementation Models Include The Database Model, The Network Database Model, The Relational Database Model And The Object Oriented Database Model (Schneider, 2013).

Relationships And Data Are Shown Through Tables Without Pointers Or Links; Instead, They Are Connected Based On The Values They Contain (P.K., F.K.). Each Table In The System Comprises Rows And Columns That Intersect With Each Other (As Mentioned By Hernandez In 2013). These Tables. Also Known As Relations. Are Linked To One Another Through A Shared Characteristic Data Point. Even Though The Tables Operate Independently From Each Other, Retrieving Data Across Tables Is Straightforward. As A Result Of The Model Design Choices— Its Emphasis On Structured Data And Relationships—Provides A Fundamental Framework For Managing Data Efficiently By Reducing The Common Redundancies Often Present In File Systems (Mcfarlane Et Al., 1998).

2.6 Objective Oriented Programming

Object Oriented Design (Oo), Along With Object Oriented Programming (Also Known As Oop), Is Gaining Popularity In The Tech Landscape. Understanding The Principles Of Oop And Various Programming Aspects In Object-Oriented Languages Is Crucial. One Can

Create Object-Oriented Programs Using A Variety Of High Level Languages Like C Or Pascal. On The Other Hand, Modern Programming Languages Such As Ada, C++, And Fortran 90 (F90) Come With Features That Facilitate Widespread Adoption And Effectiveness Of Object Oriented Programming (Mirvis Et Al. 1991). Before F90 Came Into Existence, C++ Had Already Been. Is Now Considered One Of The Most Commonly Used Programming Languages In The World.

F90 Was Designed To Incorporate Most Of The Features Of C++ Despite Being An Oop Language Itself In Nature; Of Adapting To New Standards, Many Authors Refer To The Outdated F77 Standard And Claim That Fortran Is Incompatible With Oop Practices. This Misconception Will Be Addressed Here (Satzinger, 2011).

Cutting Edge Object Oriented Languages Offer Software Engineers Three Capabilities That Improve And Reorganize The Setup Of Projects. Abstraction inheritance, And Polymorphism (Or Generic Functionality). Other Relevant Aspects Include Objects, Classes, And Data Storage Mechanisms. An Object Combines Data Types Into A Collection That Defines A New Variable Type Or Structure. A Class Brings Together The Entity Types And Associated Data That Represent Its State With Methods (Functions And Subroutines) That Access And/Or Manipulate This Data. An Example Of A Class Is Created By Combining Data And Functions In An Object-Oriented

Programming Languages Like C++ And F90 Are A Departure From Languages Where Data And Functions Are Separate Entities. The Object Oriented Approach Mirrors Real World Entities With Both Data And Functionality Encapsulated Within An Entity Using A "Module" Block In F90 Or A "Class' Block In C++. This Model Also Includes A Method Where Certain Information And Supporting Routines Can Be Hidden From The User To

Some Extent Or Entirely. Typically, The Accessibility Of The Declarations And Routines Within A Class Is Determined By "Private" Qualifiers. Concealing Information Enables One To Safeguard Data In One Section Of A System From Being Accessed, Particularly Altered In Sections Of The Application. In C++, Data And Capabilities Default To "Private " Unless Specified As "Public." On The Contrary, The F90 Language Opts For The Default Protection Mode Decision. In An F90 "Module," It Is The "Contains" Statement That Links The Data Variables And Functions Preceding It To The Functions And Subroutines That Follow (Au Et AL., 2002).

Progressive Systems Work By Using Existing Classes Of Data And Functionality To Organize Classes Without The Need To Recode The Customized Functionality From Classes For Later Ones To Use – A Concept Known As Inheritance In Programming Terms. For Example, If We Define An Employee Class In A Program And Then Create A Manager Class In The Codebase, The Manager Class Would Automatically Inherit Most Of The Data. Functionality Present In The Employee Class. We Would Only Need To Add The Information And Skills Necessary For A Manager In That Case. We May Also Need A Method To Redefine Employee Class Functions That Differ From Those Of A Manager Class. By Using The Concept Of A Class Hierarchy, Less Programming Effort Is Needed To Create The Program. In The Fortran 90 Programming Language Classes, Code Is Incorporated Into Classes Using The "Utilization" Statement Followed By The Name Of The "Module" Statement That Defines The Class (Marshall Et AL., 2000). An Example Of An Application That Utilizes Object Oriented Programming Is Visual Basic – Also Known As Vb, Which Aims To Simplify Software Development Processes And Enhance Efficiency While Retaining The Capability To Produce Sophisticated Programs With Intelligent Features. In Practice, Visual Basic

Language, For Instance, Is Made To Be Easily Understood By Humans – The Source Code Itself Is Clear Without Needing Notes For Explanation Purposes. The Visual Basic Program Also Features Elements Such As Code Snippets And Intellisense That Auto Generate Code For Components Inserted By The Developer. Another Feature Known As "Autocorrect" Can Help Debug The Code During Program Execution.

Programming Languages, Like C++, Power Builder, And Power Soft, Play A Role In The Evolving Landscape Of Desktop Computing To Meet Development Goals Effectively And Efficiently. Vb Is Continually Undergoing Enhancements At Microsoft To Stay Relevant In The Changing Technology Landscape, With Its Advantages And Features:

- It Is A Comprehensive System For Developing Windows Applications That Allow Us To Utilize Built Functions And Subprograms For A Variety Of Tasks And Create Personalized Libraries And Objects That Can Be Integrated Into The Application As Needed Or Shared With Others.**
- It Is A Product That Countless Developers And Designers Of Varying Skill Levels Favor.**
- Most Other Products From Companies Usually Offer Support For It.**
- It Advocates For The Concept Of Object Oriented Design.**
- It Offers Flexibility And Reliability While Being User Friendly Due To Its User Interface (Gui).**

2.7 Summary

In Today's World, Data And Information Are Crucial For Every Organization In Many Ways. Advances In Information Processing and Data Management Are Progressing Quickly, Making It Challenging To Keep Up With Them. As Mentioned Earlier in The Literature Review, information and Its Management Are Vital in Today's Industry.

CHAPTER 3

METHODOLOGY

3.1 Introduction

There are two components to this project. The first one is about the database itself Designing how to organize information in the database and relate data elements using a database model. The second is developing and designing a graphical user interface. This interface will get data from the database and create a user menu you can use to do the database-related task and also track the data properly.

3.2 System Study

investigation was carried out at the Saudi Diesel Equipment Company (SDEC) to identify the root cause of an issue and develop strategies for managing it. The primary focus was on enhancing customer service by gathering data; however, determining what constitutes data posed a challenge. Following examination and analysis within the company premises, it was determined that the essential data to be collected pertained to equipment and its associated information.

The collected data in SDEC is based on four important aspects, as shown in Figure 3.1. Firstly, equipment data includes equipment name, equipment type, manufacturer, and warranty period. Secondly, customer data includes Customer ID, customer name, customer address, and customer phone. Thirdly, the location includes the location code, location site, location name, and location shortcut. Lastly is the service data, which includes the planned

preventive Maintenance for each piece of equipment and the life cycle for each piece of equipment.

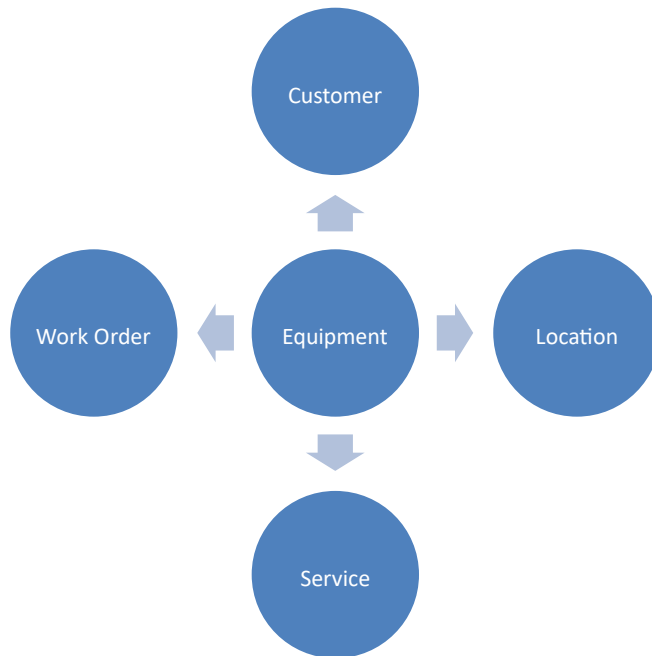


Figure 3.1: Aspects of data collection

3.3 Database Design

After you've pinpointed the data and collected all the information regarding it, the subsequent move involves determining the method to incorporate this data into the database. Database systems function as a mechanism for monitoring data. They are designed to ensure accuracy and integrity in the information they manage. Databases can take forms akin to spreadsheets or word-processing applications. Their primary function is not for graphic design or presentation purposes. They are not meant to replace word processing or spreadsheet software. At the same time, spreadsheet and word processing software tools enable users to create tables to store data. These databases lack the complexity and relational capabilities that are typical of databases.

By making use of Open Database Connectivity (ODBC), we can facilitate the transfer of information, between applications and databases effectively. This particular endeavor is centered around utilizing MS Access to build databases because of its programming support and a wide array of database management tools. The initial step involves designing tables and populating them with information to kickstart the process of creating the database. For this project, we worked with six tables, as mentioned earlier. Equipment Table (Equipment) Customer Table (Customert) Location Table (Locationt) Work Order Table (WorkOrdert) and Login Table (Logint).

After choosing the tables to set up and deciding on the data to include in each one of them, remember to label the fields in every table with names. For instance, essential information, like the equipment's name and the manufacturers model should be incorporated into the Equipment table guarantee that the data in each spreadsheet is well arranged and structured effectively.

3.3.1 Tables Data Configuration

The database must be configured with the data type for each table to ensure that the right data is read and written to avoid rejection when different data types are placed in it. Figure 3.2 illustrates that every data entry into the EquipmentT Table is assigned the data type; for example, setting the price field as "Currency" ensures that only currency values are accepted, preventing any text or date entries. Thus, it aids in data organization and minimizes errors and duplication in the future.

EquipmentT		CustomerT
	Field Name	Data Type
?	Asset#	AutoNumber
	EquipmentName	Short Text
	Manufacturer	Short Text
	Model	Short Text
	Serial#	Short Text
	EquipmentType	Short Text
	SupplierID	Short Text
	Price	Currency
	Site#	Short Text
	MD#	Short Text
	LocationID	Short Text
	StartUp	Date/Time
	UnderWarranty	Yes/No
	EndOfWarranty	Date/Time

Figure 3.2: Table Data Configuration

Access also includes the following types of data. A short description of how each data type is used is provided below :

- **Text:** This kind of data typically consists of letters or a mix of numbers and alphabetic characters.
- **Note to self:** This data format suits text exceeding 255 characters.
- **Number:** The number data type is commonly employed to input values used in calculations.
- **Data/Time:** primarily utilized for marking dates and times.
- **Currency:** refers to data used to represent values and numerical amounts.
- **Autonumber:** a data type that helps you automatically increase your record.
- **Yes/No:** Is this type of data used when there are no values?

3.5 Development of Graphic User Interface

To effectively create and design a graphical user interface (GUI) it is important to select the software and programming language that best suits the project requirements, at hand. For this project, the implementation of the GUI system in C and C++ was chosen due to their compatibility with object binding and embedding within the database environment. Furthermore, having a coding platform is essential to ensure a coding experience throughout the development process. For this purpose, Visual Basic was selected as the GUI design and development program.

3.5.1 System Overview

In order to create a software system that's easy for users to navigate and manage equipment information effectively (EMIS), it is essential to adhere to specific procedures and methods. Firstly, it is important to gather the information and details from the company in order to develop the system effectively. Next, the system database should be established using specialized software based on the collected information. Lastly, a user-friendly interface needs to be designed using application software like Microsoft Visual Studio.

The project has two components: the database and the user interface are involved in this venture. Using a database to create tables that store information and data about equipment. The information is sourced from the Saudi Diesel Equipment Company (SDEC) in Dammam, Saudi Arabia. Based on the collected data, six tables have been set up in Microsoft Access 2010. Specified attributes have been identified, and a primary key has been selected for each table to ensure entity integrity. Many businesses rely on an asset number (Asset #) identifier for their equipment items. As a result, this asset number is designated as the key in the equipment table. It serves as the foreign key for all other tables.

Next came the process of standardizing each table by structuring and optimizing the data in the database according to guidelines to remove duplicate and irrelevant information before linking the tables.

Once the database of the system was created with Microsoft Access a user friendly menu was developed to execute database tasks and present information effectively. OLE DB (Object Linking and Embedding Database) was utilized to access the database via Microsoft Visual Studio. To show requested data outcomes. The available database functions were determined for user interaction with the system. Each task is carried out using a form, with the necessary tools like text boxes and labels included. The reports were then created based on the user's needs. Every component is coded individually. Then, it is put together to create the equipment management information system (EMIS). The system's features were tested using sample data and queries. In Figure 3. 2, You can see an overview of the system and its logo.



Figure 3.3: System Overview

3.6 Flow Chart of the Project

Flowcharts are diagrams that visually depict a sequence of steps in a process. Symbols represent each step and provide a description of it. These symbols are connected with arrows to show the flow of the process. The flowchart illustrates the progression of data in a way that's easier to understand than reading it in plain text.

The flowchart illustrates the step-by-step progression of the project criteria concerning the "Development of GUI, for Effective Equipment Management Information System." Once the topic is addressed, tasks are sequentially carried out to achieve the goal of collecting data and designing the database in order to develop the system effectively.

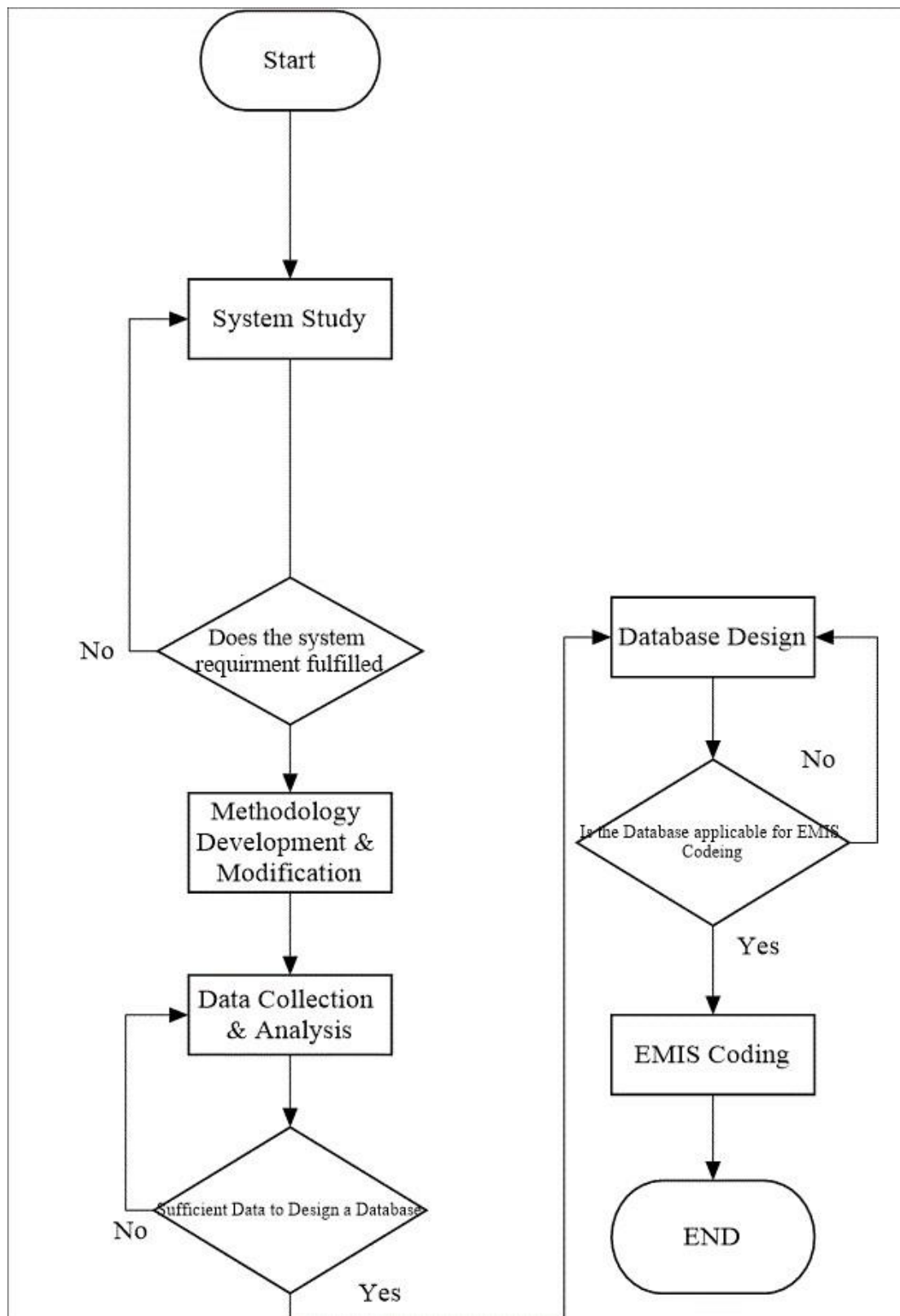


Figure 3.4: System Flowchart

CHAPTER 4

RESULT AND DISCUSSION

4.1 Database Dictionary

To organize the data effectively the initial step involves defining the database using a database tool to assist in this process efficiently. The database management system (DBMS), in turn utilizes this data dictionary to locate data components structures and connections to alert users about any challenges that may arise when coding intricate relationships, for each program. Therefore, the data dictionary employed for every table within the database is outlined in Table 1 through Table 7 below;

Table 4.1: Equipment Table and its Attributes and Description

Table Name: EquipmentT			
Field Name	Data Type	Size	Constraints
Asset#	Autonumber	Long Integer	primary key, unique, not null
EquipmentName	Text	255 char	
Manufacturer	Text	255 char	
Serial#	Text	255 char	
EquipmentType	Text	255 char	
CustomerID	Text	255 char	foreign key for CustomerT
Price	Currency	Long Integer	> = 0
Site#	Text	255 char	foreign key for SiteT
LocationID	Text	255 char	foreign key for LocationT

StartUp	Date/Time	General Date	
UnderWarranty	Yes/No	255 char	True/False
EndOfWarranty	Date/Time	General Date	
Retired	Date/Time	General Date	
PPM'sYearly	Text	255 char	
PPMschedule	Text	255 char	
PPM'sMonth	Text	255 char	
1stPPM	Date/Time	Short Date	
1st PPM'Status	Text	255 char	
2ndPPM	Date/Time	Short Date	
2nd PPM'Status	Text	255 char	
3rdPPM	Date/Time	Short Date	
3rd PPM'Status	Text	255 char	
4thPPM	Date/Time	Short Date	
4th PPM'Status	Text	255 char	
5thPPM	Date/Time	Short Date	
5th PPM'Status	Text	255 char	
6thPPM	Date/Time	Short Date	
6th PPM'Status	Text	255 char	
7thPPM	Date/Time	Short Date	
7th PPM'Status	Text	255 char	
8thPPM	Date/Time	Short Date	
8th PPM'Status	Text	255 char	
9thPPM	Date/Time	Short Date	
9th PPM'Status	Text	255 char	
10thPPM	Date/Time	Short Date	
10th PPM'Status	Text	255 char	
PO#	Text	255 char	
AcceptanceForm	OLE Object	depend	

EquipmentPic	OLE Object	depend	
Remark	Memo	255 char	

Table 4.2: Site Table and its Attributes and Description

Table Name: SiteT			
Field Name	Data Type	Size	Constraints
Site#	Text	255 char	primary key, unique, not null
SiteName	Text	255 char	

Table 4.3: Customer Table and its Attributes and Description

Table Name: SupplierT			
Field Name	Data Type	Size	Constraints
CustomerID	Text	255 char	primary key, unique, not null input mask: LL99
CustomerName	Text	255 char	
Address	Text	255 char	
City	Text	255 char	
CustomerPhone	Text	255 char	
CustomerE-mail	Hyperlink	255 char	

Table 4.4: Log in Table and its Attributes and Description

Table Name: LogInT			
Field Name	Data Type	Size	Constraints
ID	Autonumber	Long Integer	primary key, unique, not null
Username	Text	255 char	
Password	Text	255 char	

Password confirmation	Text	255 char	Password confirmation must be like Password
------------------------------	------	----------	---

Table 4.5: Location Table and its Attributes and Description

Table Name: LocationT			
Field Name	Data Type	Size	Constraints
LocationID	Text	255 char	primary key, unique, not null
LocationName	Text	255 char	

Table 4.6: Work Order Table and its Attributes and Description

Table Name: WorkOrderT			
Field Name	Data Type	Size	Constraints
WO#	Autonumber	Long Integer	primary key, unique, not null
CallDate	Date/Time	Short Date	
CallTime	Date/Time	Short Date	
Site#	Text	255 char	foreign key for SiteT
CallerName	Text	255 char	
Callerphone	Text	255 char	
CallerE-mail	Hyperlink	255 char	
Asset#	Number	Long Integer	foreign key for EquipmentT
Urgency	Yes/No	255 char	True/False
Reason	Text	255 char	
RequestDetail	Memo	255 char	
TechnicalTeam	Text	255 char	
SparePart	Yes/No	255 char	True/False
WO'Status	Text	255 char	
Corrective Action	Text	255 char	
Comment	Memo	255 char	

4.1.2 Database Structure

The database for this project is structured using Microsoft Access software, as it plays a role in the project's operation. It encompasses operations like storing and fetching data and serves as the repository for all vital information. Since it forms the core of the system, it must be meticulously designed to ensure a functional and productive system.

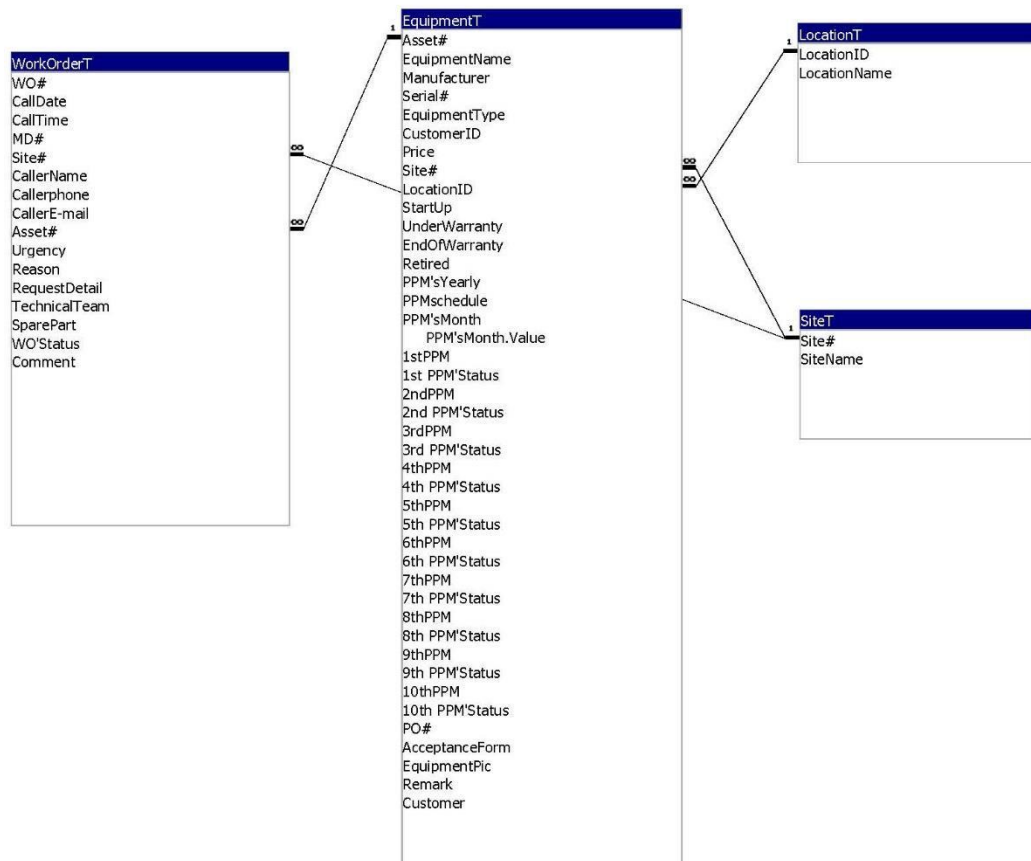


Figure 4.1: Database Structure

Linkage is implemented through the concept of integrity in databases, where each table is linked to another table using a field known as a foreign key, effectively establishing connections between them. One to many relationships are employed in the database model. This structure can be visualized in Figure 4.1.

4.2 Data Flow Mechanism

A data flow diagram (DFD) is a tool used to illustrate how data moves between entities or tables within a system as it progresses and accumulates information over time. The visual representation of the data flow in the system helps in understanding its operations and implementation details as it undergoes specification. Designers typically sketch context-level DFDs to provide an overview of the relationships between the system and its entities. The high level Data Flow Diagram (DFD) is expanded to serve as proof or indication of a concept of the system being duplicated.

In a data flow diagram, the context diagram is considered the level known as level 0. It is the most basic representation of the system, with just one process node and a general overview of how things work. In Figure 4.2. Equipment Management Information System (EMIS) the context diagram illustrates the connection, between users and the database.

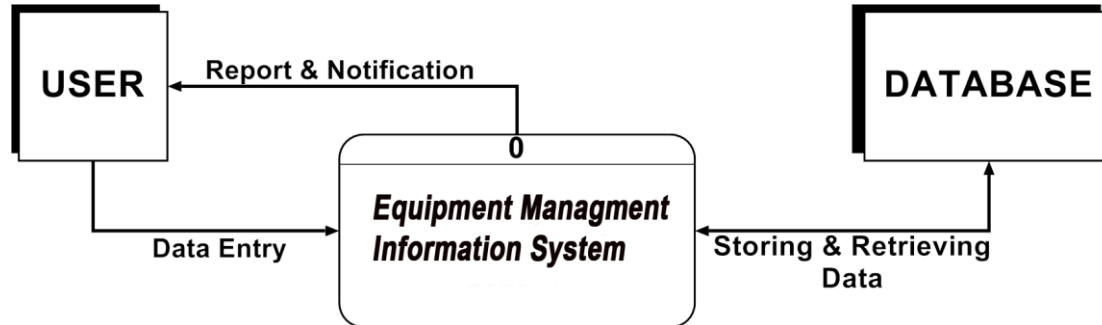


Figure 4.2: Context Diagram

4.2.1 Data Flow (level 1)

Figure 4.3 Illustrates the data flow, within the database system where users can access reports and input data into any of the databases available, to them.

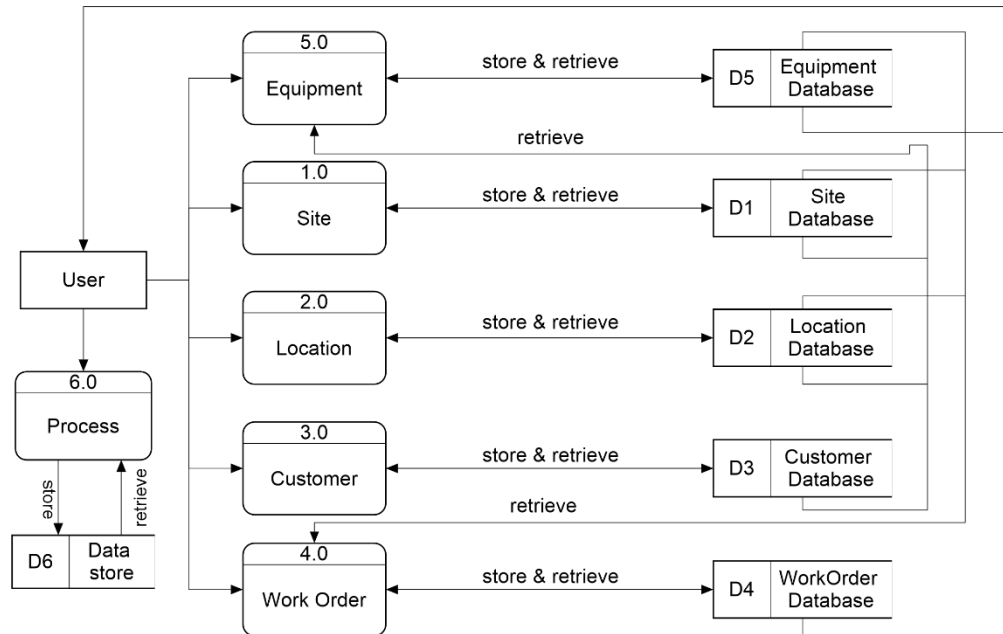


Figure 4.3: Data Flow Diagram (Level1)

4.3 User Interface Design

The equipment management information system (EMIS) includes interfaces for users to engage with the system, which also produce outputs to update users about outcomes and any error messages that may have arisen. Here is a rundown of the interfaces within the system;

4.3.1 Login

The login form is essential for entering the page of the equipment management information system (EMIS) along with its features, as indicated in Figure 4.4.



Figure 4.4: Login Form

It is employed as a security measure to prevent entry into the system requiring users to input the username and password to proceed further which is then cross checked with the database for validation before granting access to the user to log in successfully onto the system. The username and password must be present in the database. It can be configured using the "Change Username or Password" form that will be displayed. If the username and password you enter don't match up correctly, the system will show an error message, as depicted in Figure 4.5.



Figure 4.5: Error Message of Wrong Login

The system will automatically stop the program if the user tries to log in, then three times to enhance system security measures. The login procedure illustrated in figure 4.6 Depicts whether the user is allowed access, to the equipment management information system (EMIS).

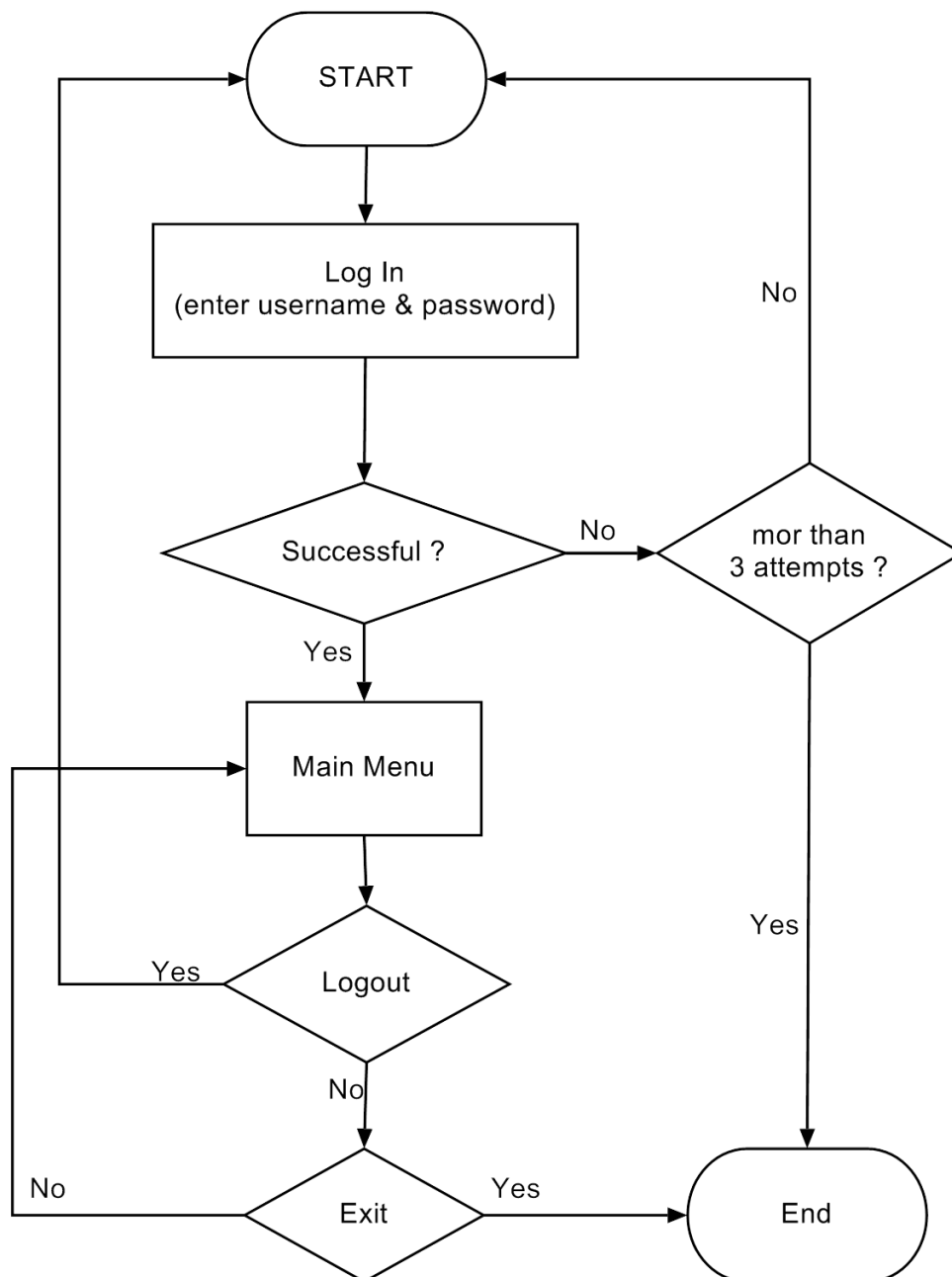


Figure 4.6: Login Process

When the system user starts the software application and reaches the login page, they must provide their credentials to confirm their identity as a user. The credentials include a username and password stored in the database. Access to the system will be allowed once the entered password matches. Afterward, the user can use the equipment management information system (EMIS) through the main menu interface. If the username and password do not match initially when logging in to the system user account, three attempts are allowed to enter the credentials without any issue. However, if the user fails to log in after these three attempts, the system will automatically log out. Log out itself. To log out of the system or exit, select the "Log Out" option. Use the "Log Out" tab in the main menu. You can use each function separately, not at the same time.

4.3.2 Main Menu

The system's central hub is the menu where users can choose from options to carry out specific tasks (see Figure 4.7). There are drop-down menus on the page allowing users to select and access interfaces. In the “Profile” section, users can update their username and password, create user accounts, and log out of the system. The “Inventory” section of the system interface allows users to access the inventory database and enter data or search for existing information related to equipment and customer details, as well as site and location databases. The “Work Order” feature of the system interface allows users to create work orders or search for and update existing work orders. The “PPM Schedule” section of the system interface gives users access to a search and edit form to view and update details related to planned maintenance schedules and review notifications within the system. In the “Reports” section of the system interface, menu options include the ability for users to create

management reports such as planned maintenance reports and available work order reports. Users have the option to make use of the "Advanced Search" function to delve into search capabilities within the system database. Users also have the ability to refer to tables like customer tables for purposes, under "Reference Tables". Lastly, humans can select "About EMIS" to gain insight into the system and its originator.



Figure 4.7: Main Menu

The menu structure displayed in Figure 4.8 provides an overview of the menus and their respective functions within the equipment management information system (EMIS). The diagram comprises three tiers ;

1. The first level comprises eight sections; "Profile," "Inventory," "Work Order," "PPM schedule," "Reports," "Advanced Search," "Reference Tables," and "About HEMS." These are called the main menus of the system

One heading can be opened at a time. At this level, the items are always in view for the user.

2. The following stage is the submenu, the second level heading.

"Examples of a second level heading include 'Changing Username or Password.'"

This section includes tools and features like "Equipment," "WO Status," "PPM reports," and "Search by Month," which users can access after selecting the heading option.

3. Under second-level headings like "New" and "Search," Can you please provide me access to this document? Update: the third-level heading will be displayed with objects to the user upon selecting the second heading.

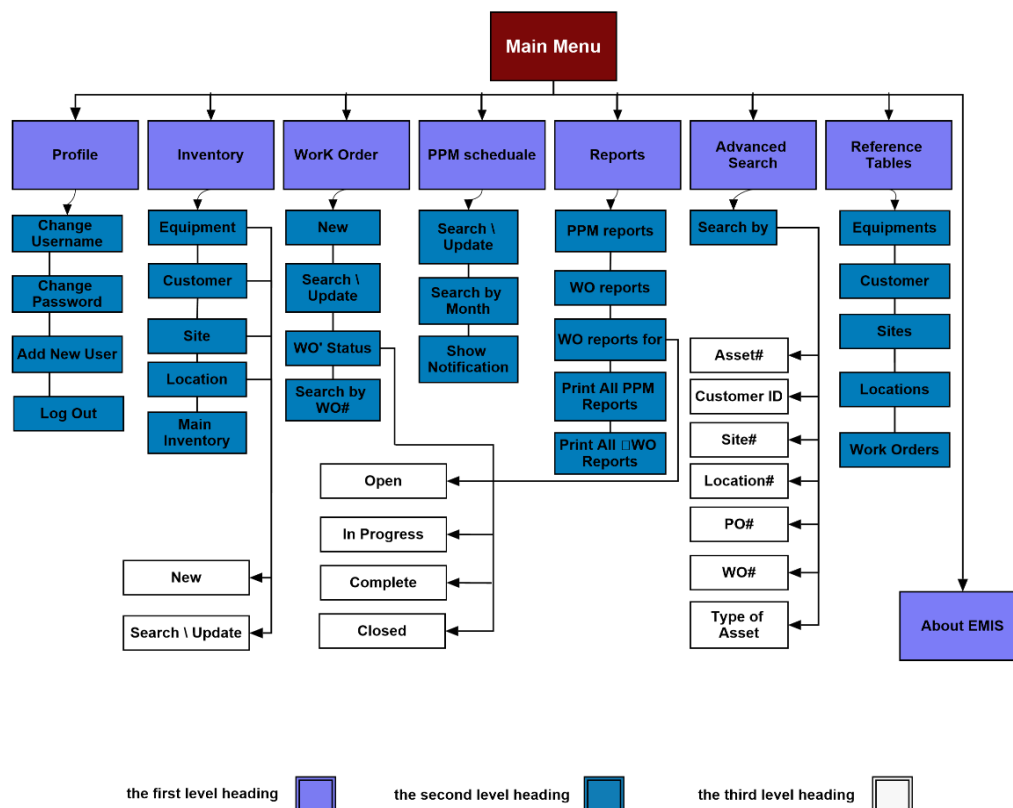


Figure 4.8: Main Menu Hierarchy

You can carry out actions like "Save Record." "Cancel" under the "headings. In the context of "Search Let me know if there is anything else I can help you with. Update" headings in the system interface is where you would use the "Edit" function, while for "Reports," the available action is limited to printing as, per the menu hierarchy diagram depicted in figure 4.8

4.3.3 Profile

The profile dropdown menu provides users with options to select and execute desired actions, such as changing usernames and passwords or adding users to the system before logging out. Users may utilize the "Change Username" or "Change Password" form to update their login information accordingly. In Figure 4.9 of the interface design document (IDD), users have the ability to add a user by inputting a username and a new password into the designated fields, followed by updating the "Login Table" in the system.

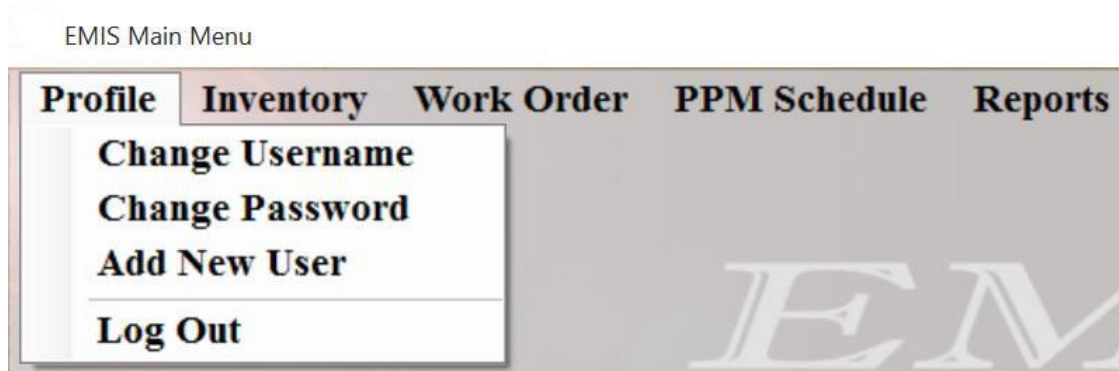


Figure 4.9: Profile Dropdown Menu

4.3.4 Inventory

In the systems menu, under "Inventory," users will find interfaces for accessing the primary database inventory and equipment information related to customers and locations within the system interface. Users can interact with the database by adding records or

searching and updating existing ones. Within the equipment form displayed in Figures 4.10 and 4.11, users can input equipment details covered by information and planned preventative maintenance schedules for each piece. In this form, there are combo boxes linked to databases other than the equipment database including "customer ID," "Site#," and "Location ID." System users are highly advised to complete these databases before filling out the equipment database to ensure the accuracy and effectiveness of the data in the system database.

In the equipment window displayed in Figure 4.10, there is a range of information that can serve as a reference for the company, such as the purchase date and warranty expiration date of the equipment. This information assists in customer interactions and issue resolution during the warranty period.

The screenshot shows a software window titled "Equipments" with a standard Windows interface (minimize, maximize, close buttons). Inside the window, there is a logo on the top left consisting of a green circular arrow and the text "EMIS". To the right of the logo, the text "PPM" is displayed. Below the logo and "PPM" text, there are two tabs: "Equipment Details" (which is selected) and "PPM Schedule". The "Equipment Details" tab contains a form with the following fields:

Asset ID:	16	Customer:	[Dropdown]
Name:	90 KVA Generator	Price:	90000
Manufacturer:	SDEC	PO #:	1
Serial #:	SDH-90	Site #:	29
Type:	Power Equipment	Location:	JED
Warranty:	☒ Yes, Under Warranty	Start Up:	[Dropdown]
Remark:	[Text Area]	Warranty Ends:	[Dropdown]
		Retired:	[Dropdown]

At the bottom right of the form, there are two buttons: "Cancel" and "Update Record".

Figure 4.10: The Equipment Form (The equipment details)

In Figure 4.11, there are windows in the equipment inventory, such as the PPM schedule window, which provides customers with information on the PPM schedule for each equipment and the status of each PPM period. Additionally, it will offer the team detailed information about the type of maintenance required—whether it is external on site or internal within the company's workshop—and the customer's preferred month for maintenance (PPM). This helps both parties coordinate schedules efficiently and expedite the maintenance process.

PPM Period	Date	Status	Completed
1st PPM	28/02/2023	Completed	☒
2nd PPM	13/08/2023	Not Yet	☒
3rd PPM	11/02/2024	Not Yet	☒
4th PPM	15/08/2024	Not Yet	☒
5th PPM	13/02/2025	Not Yet	☒
6th PPM		Not Yet	☐
7th PPM		Not Yet	☐
8th PPM		Not Yet	☐
9th PPM		Not Yet	☐
10th PPM		Not Yet	☐

Figure 4.11: The Equipment Form (PPM schedule)

The form displayed in Figure 4.12 enables users to input customer details, such as information regarding the customers of various equipment types. The "customer ID" textbox has an input format restriction (LL99). This means that the first two characters must be letters, and the following two characters must be numbers.

ManageCustomer

Customer Manager

Customer ID:

Customer Name:

Customer Address:

City:

Phone Number:

Email:

Close Save / Update

Figure 4.12: Customer Form

The primary inventory option provides access to inventory database options, such as the customer inventory database depicted in Figure 4.13, which allows users to view customer information and their respective referral IDs, streamlining the process of adding equipment to the database or identifying which customers are associated with a particular referral ID.

MainInventory

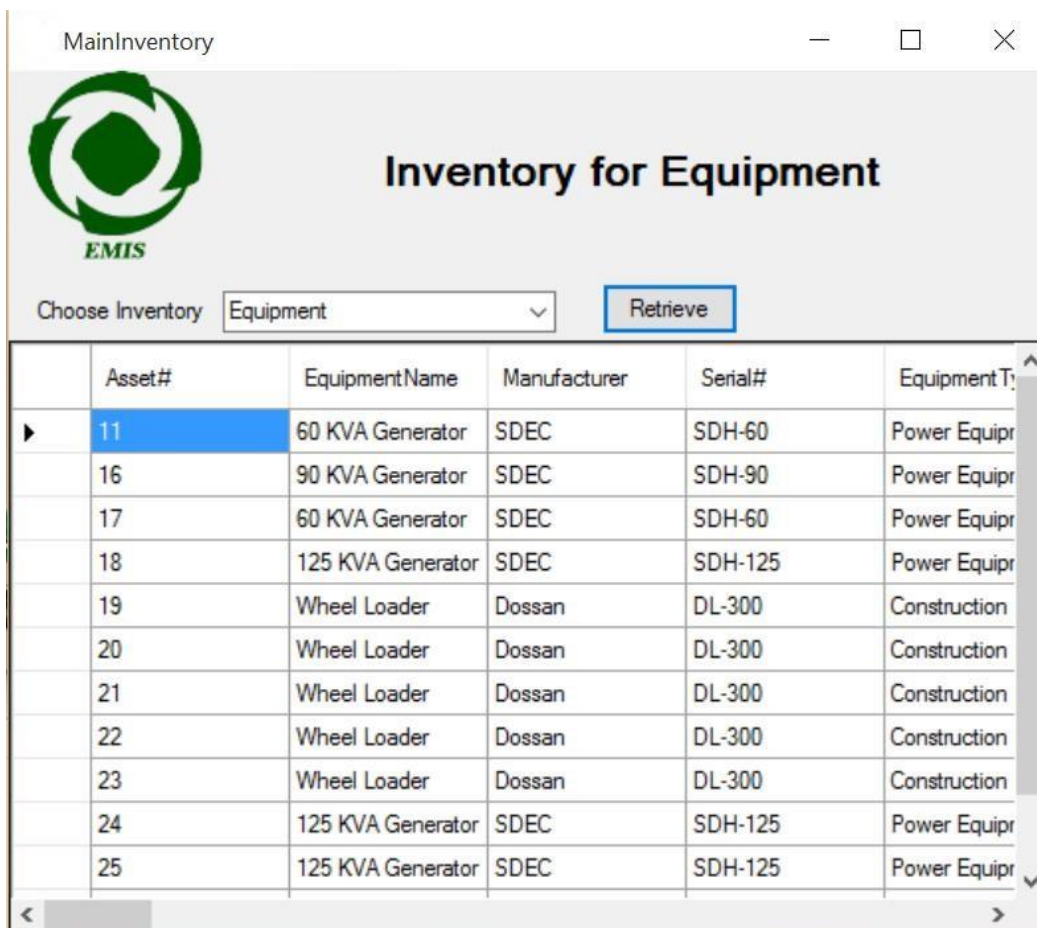
Inventory for Customer

Choose Inventory: Customer Retrieve

	CustomerID	CustomerName	Address	City	CustomerPhone
▶	ST	SDGC STOCK		North Providence	
	SC	SME CRUSHER ...			
	SA	SAUDI ARABIAN...			
	SD	SDEC			
	AA	A.A. TURKI COR...			
	AI	ALLCHEM INTE...			
	HT	HONGTAK			
	EG	EVERDIGM			
	AB	AL BAKREIN EST.			
	AC	AL BECITEEN			
	AM	AL MIRA CENTE...			
	BA	BRITISH AEROS...		Riyadh	

Figure 4.13: Inventory for Customer

Additionally, it can access inventory databases such as the equipment inventory database (as illustrated in Figure 4.14), the location inventory database, and the site inventory database within the system interface connected to Microsoft Access for updates on current inventory status and facilitates linking additional data to various submenus in the system.



	Asset#	EquipmentName	Manufacturer	Serial#	Equipment Type
▶	11	60 KVA Generator	SDEC	SDH-60	Power Equipment
	16	90 KVA Generator	SDEC	SDH-90	Power Equipment
	17	60 KVA Generator	SDEC	SDH-60	Power Equipment
	18	125 KVA Generator	SDEC	SDH-125	Power Equipment
	19	Wheel Loader	Dossan	DL-300	Construction
	20	Wheel Loader	Dossan	DL-300	Construction
	21	Wheel Loader	Dossan	DL-300	Construction
	22	Wheel Loader	Dossan	DL-300	Construction
	23	Wheel Loader	Dossan	DL-300	Construction
	24	125 KVA Generator	SDEC	SDH-125	Power Equipment
	25	125 KVA Generator	SDEC	SDH-125	Power Equipment

Figure 4.14: Inventory for Equipment

4.3.5 Work Order

The "Work Order" dropdown menu allows users to either create a work order or search for and update existing ones within the system interface. Effective work order management is crucial from contact to archival stages as it guarantees traceability and enables precise analysis.

Furthermore, the user has the capability to search for and modify orders based on their status or work order number (WO #), as depicted in Figure 4.15.

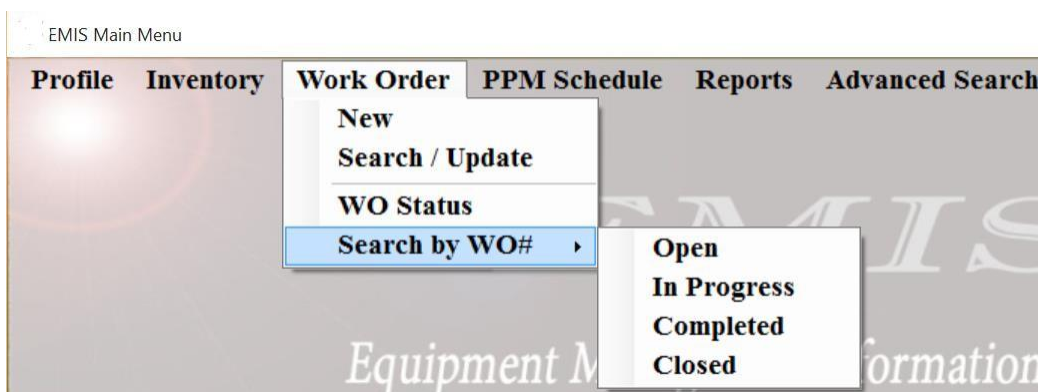
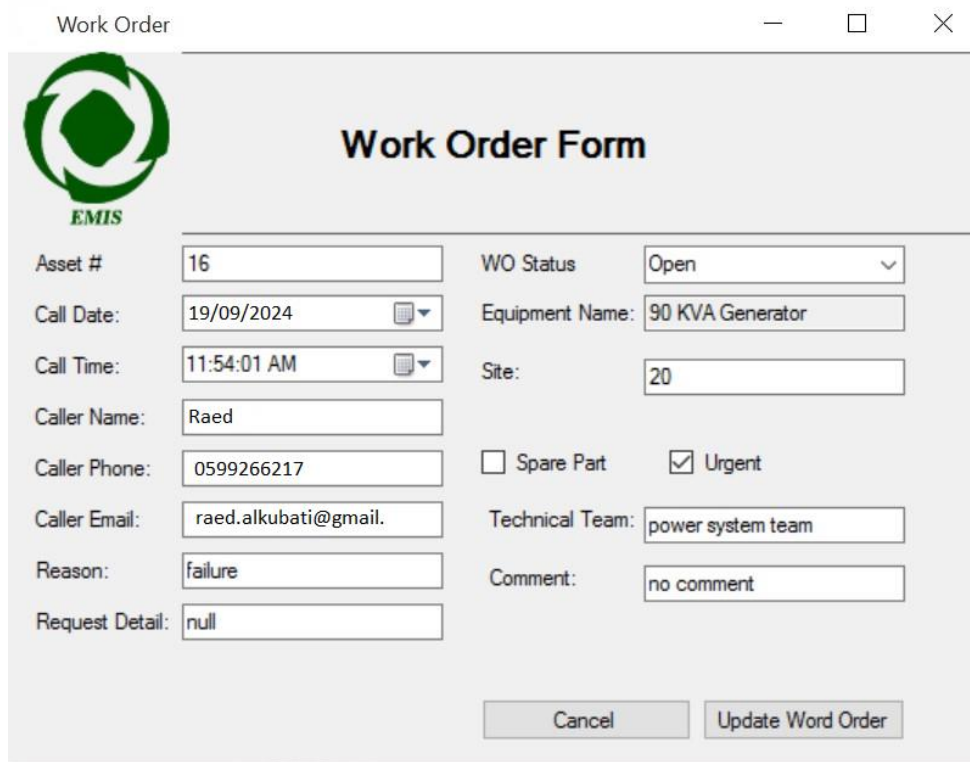


Figure 4.15: Work Order Dropdown Menu

The form displayed in Figure 4.16 allows users to create and complete a work order for equipment stored in the system's database. Various dropdown menus in this form are linked to databases such as "Asset#," "Site#," and "Location#." Henceforth, system users are advised to populate these databases before entering information into the work order database to guarantee the accuracy and effectiveness of the data within the system database. Additionally, when looking up work orders and their status, such as checking for work orders that still need to be completed and closed, you can simply click "search by WO#" and select "open" to view all the work orders as illustrated in figure 4.17 of the system interface design

document. These features have been implemented to streamline communication between the team and customers, allowing them to access equipment information before heading to the site. It will also help management stay informed about the status of the work order until it is completed.



Work Order

 **Work Order Form**

Asset #	16	WO Status	Open
Call Date:	19/09/2024	Equipment Name:	90 KVA Generator
Call Time:	11:54:01 AM	Site:	20
Caller Name:	Raed	☐ Spare Part	☒ Urgent
Caller Phone:	0599266217	Technical Team:	power system team
Caller Email:	raed.alkubati@gmail.	Comment:	no comment
Reason:	failure		
Request Detail:	null		

Figure 4.16: Work Order Form

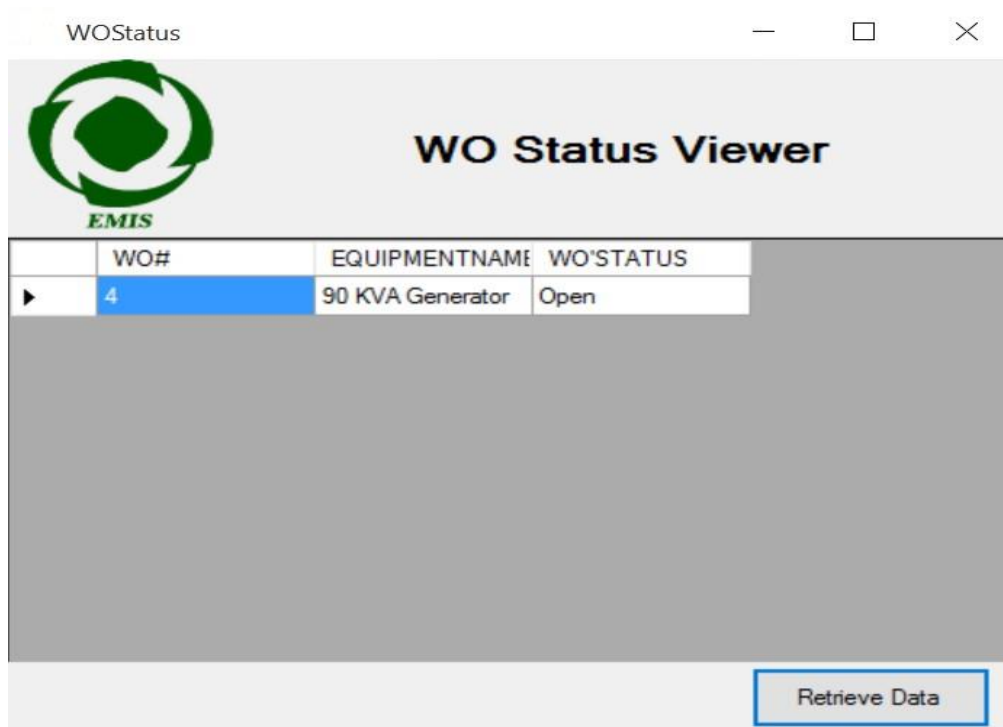


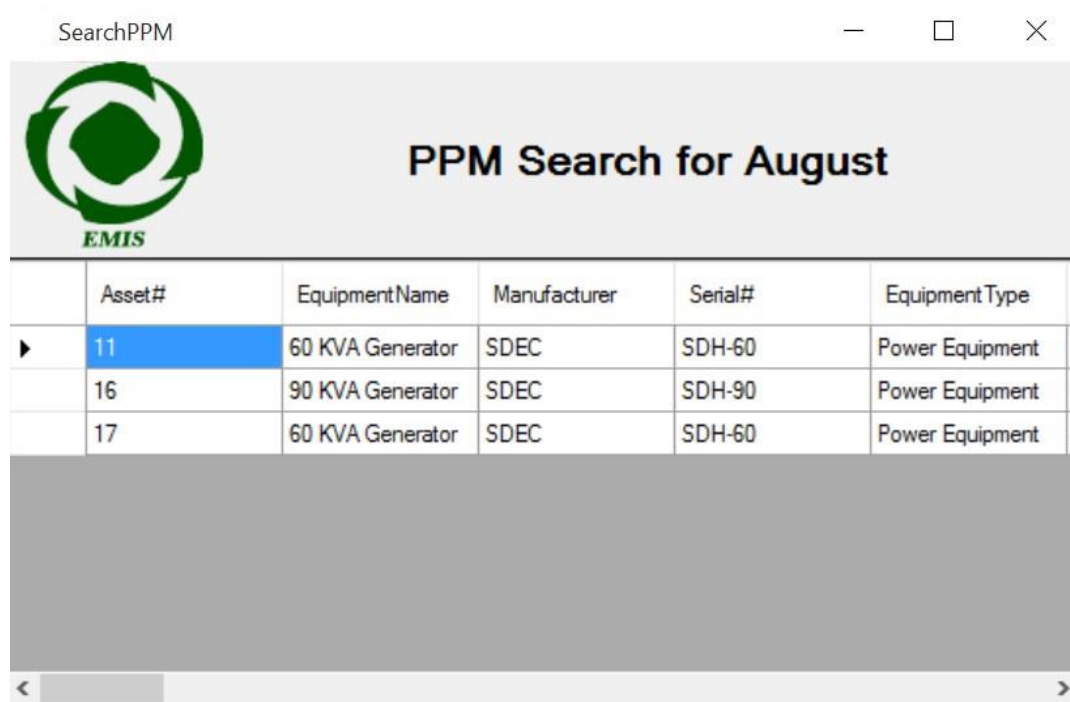
Figure 4.17: Work Order Status Viewer

4.3.6 PPM Schedule

In the menu, under "PPM schedule," users can access a search. Update the form to view and modify details for the equipment's PPM schedules stored in the database. As illustrated in Figures 4.18 and 4.19, respectively in, the system interface allows users to search for PPM schedules for equipment in the database. Additionally, users can. Review notifications related to planned maintenance by selecting the "Show Notification" option, as depicted in Figure 4.20 displayed on the screen.



Figure 4.18: PPM Schedule Drop down Menu



	Asset#	EquipmentName	Manufacturer	Serial#	EquipmentType
▶	11	60 KVA Generator	SDEC	SDH-60	Power Equipment
	16	90 KVA Generator	SDEC	SDH-90	Power Equipment
	17	60 KVA Generator	SDEC	SDH-60	Power Equipment

Figure 4.19: PPM Search for August Month

4.3.7 PPM Notification

The PPM notification feature is set up to inform system users of asset maintenance schedules displaying the notifications, on the bottom right of the menu screen, for easy access and visibility as depicted in Figure 4.20 showcasing the detailed notification flowchart outlining the entire process.

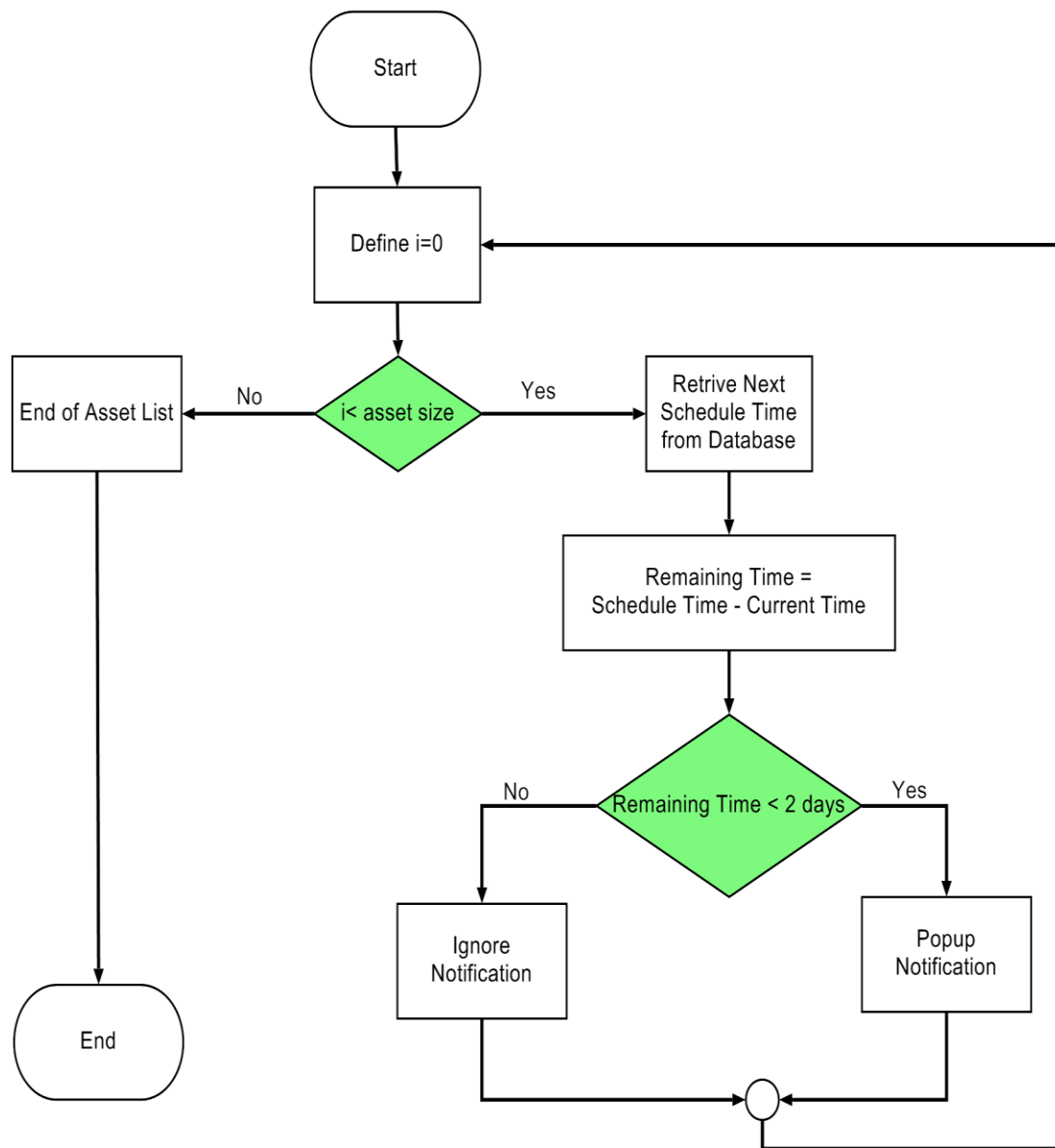


Figure 4.20: PPM Notification flowchart

When displaying the notification message to users or stakeholders, the system first verifies all assets listed in the database to see if their next scheduled maintenance time coincides with the time. Assets that do not have a warranty are excluded from receiving the notification alert. To determine the remaining time before scheduled maintenance, the system calculates it using the formula provided below :

$$\Delta_t = T_schedule - T_current$$

If the time difference is, under 48 hours, and the value Δt , the notification will be displayed (see Figure 4. 21). Otherwise the notification, for that asset will be delayed. A notification message could contain asset details based on scheduled times.

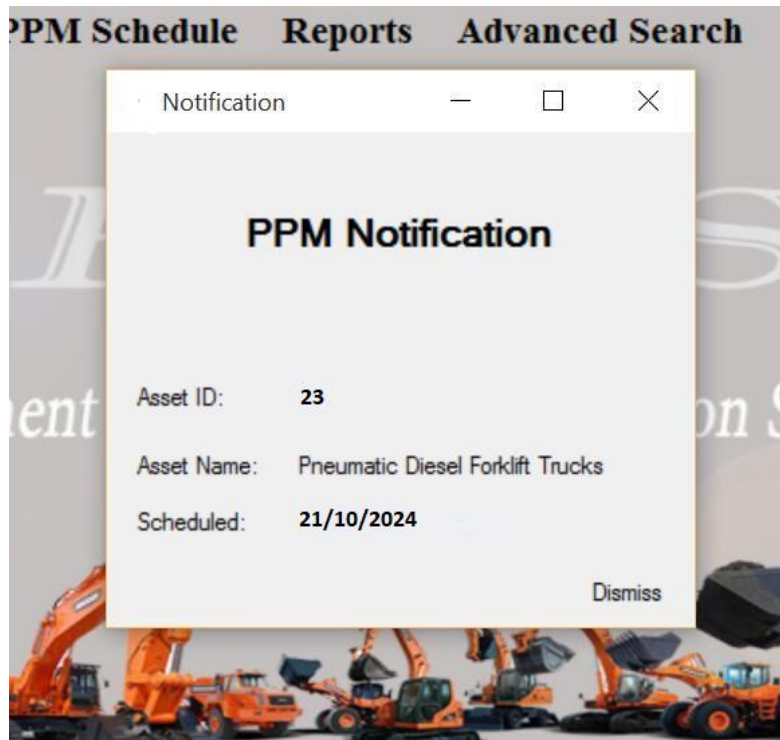


Figure 4.21: PPM Notification

4.3.8 Reports

Generating a report is vital within the system as it conveys information in a presentable manner to assist top management in staying informed and aligned with their goals. Within the "Reports" menu displayed in Figure 4.21, users are provided with options to create and review management reports such as the PPM report (figure 4.22), the WO report (figure 4.23), and print all reports, within the system (figure 4.24). The user also has the option to

search for reports based on the status of the work order, for analysis purposes, like checking In Progress work orders.

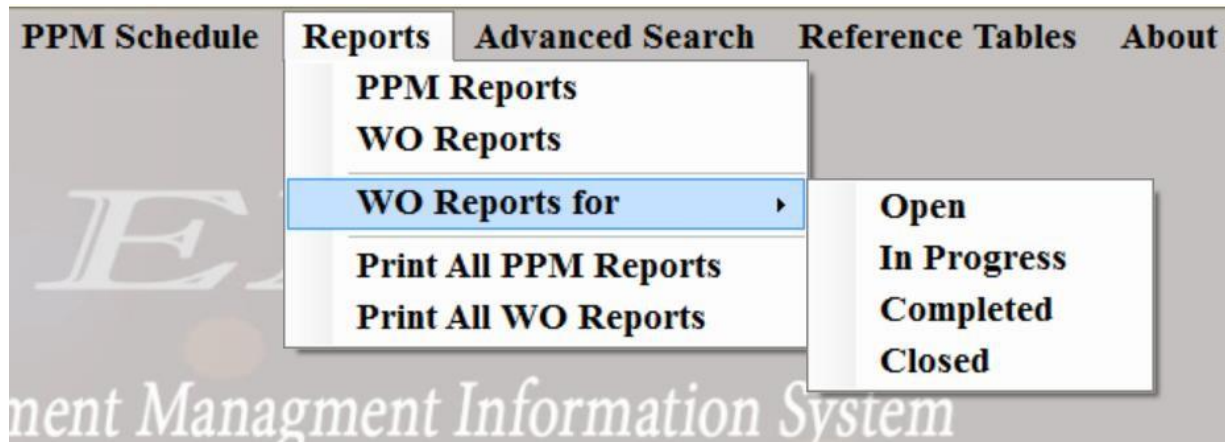


Figure 4.22: Reports Drop down Menu



Planned Preventive Maintenance (PPM) Report

Asset #:	11	Equipment Name:	60 KVA Generator
Supplier ID:	EG	Supplier Name:	SDEC
Under Warranty:	True	End of Warranty:	01/01/2016
PPM's Yearly:	2	PPM Schedule:	Internal
PPM's Month:	August;February		

PPM's Status

1st PPM:	26/02/2013	Completed
2nd PPM:	26/08/2013	Completed
3rd PPM:	26/02/2014	Completed
4th PPM:	26/08/2014	Completed
5th PPM:	26/02/2015	Completed

PPM's Status

6th PPM:	26/08/2015	Completed
7th PPM:	00/00/0000	Not Yet
8th PPM:	00/00/0000	Not Yet
9th PPM:	00/00/0000	Not Yet
10th PPM:	00/00/0000	Not Yet

[Print](#)

Figure 4.23: PPM Report

WOReport



Work Order Report

WO #:

Asset #:

16

WO' Status:

Open

Equipment Name:

90 KVA Generator

Call Date:

19/09/2024 11:54:01 AM

MD #:

No Data

Call Time:

19/09/2024 11:54:01 AM

Site #:

20

Caller Name:

Raed

Caller Phone:

0599266217

Caller E-mail:

Raed.alkubati@gmail.com

Reason:

failure

Request Detail:

null

Technical Team:

power system team

☒ Urgent

☐ Spare Part

Comment:

no comment

Back

Next

Figure 4.24: Work Order Report

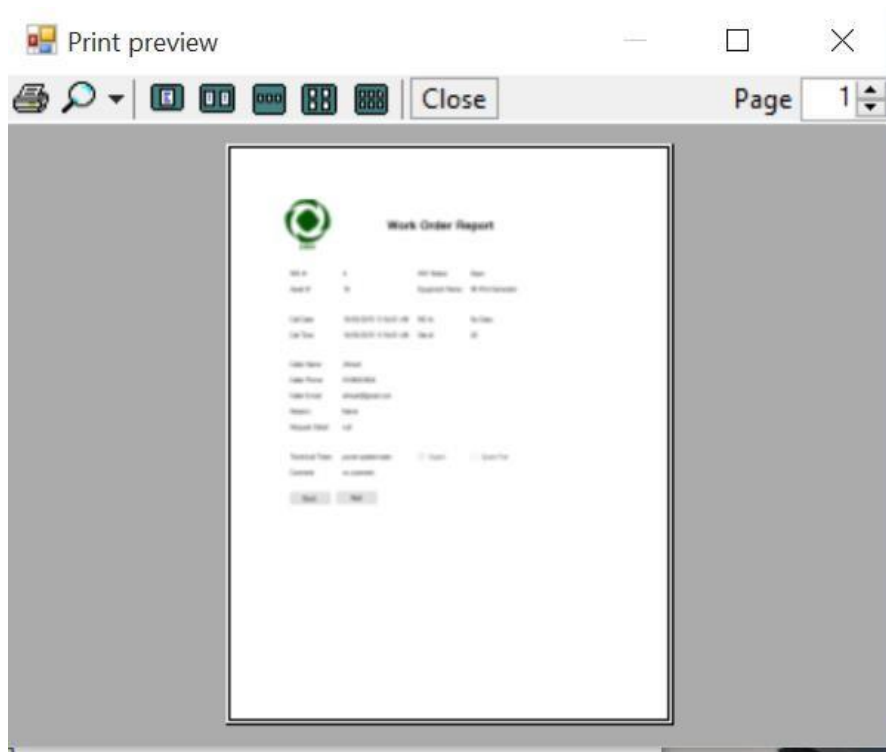


Figure 4.25: Print Preview for WO Report

4.3.9 Advance Search

There are methods to locate an equipment record in the system's database indicated in Figure 4.26, located within the "Advanced Search" section of the menu.

I'm looking for a record using the "Asset#," "Customer ID," and "Site#" identifiers.

The identification number of the location or equipment record such, as "Location ID," "PO#" "WO#" or "Type of Asset." Furthermore utilizing the search feature allows users to quickly locate information such, as power equipment exclusively registered in the system. By selecting the "Type of Assets" search option and inputting "Power Equipment " users can easily access all equipment categorized under power equipment in Figure 4.27

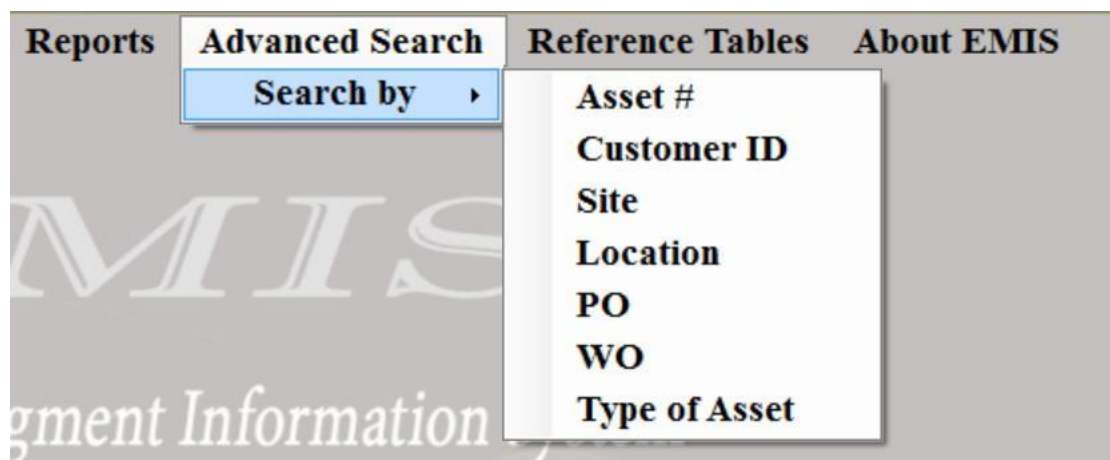


Figure 4.26: Advanced Search Dropdown Menu

The screenshot shows a window titled 'AdvancedSearcher' with a search result table. The window header includes the EMIS logo and the text 'Searching Asset with Asset Type power equipment'. The table has the following columns: Asset#, EquipmentName, Manufacturer, Serial#, Equipment Type, and CustomerID. The first row is highlighted in blue.

Asset#	EquipmentName	Manufacturer	Serial#	Equipment Type	CustomerID
11	60 KVA Generator	SDEC	SDH-60	Power Equipment	EG
16	90 KVA Generator	SDEC	SDH-90	Power Equipment	AA
17	60 KVA Generator	SDEC	SDH-60	Power Equipment	AA
18	125 KVA Generator	SDEC	SDH-125	Power Equipment	AB
24	125 KVA Generator	SDEC	SDH-125	Power Equipment	AB
25	125 KVA Generator	SDEC	SDH-125	Power Equipment	AB

Figure 4.27: Searching by Type of Assist Form

4.3.10 Reference Tables

The system users can refer to the tables in the database under the "Reference Tables" section located in the menu, as depicted in Figure 4.28.

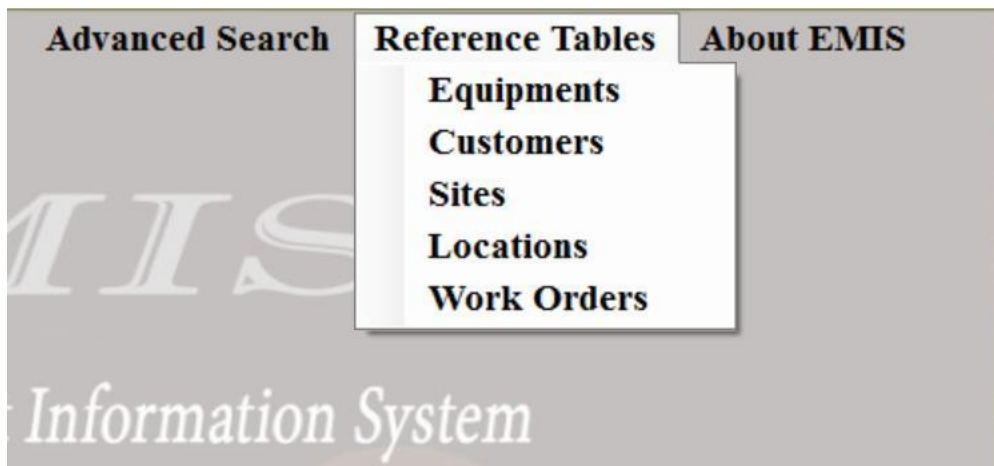


Figure 4.28: Reference Tables Dropdown Menu

This tool assists users in examining and verifying the information stored in database tables, as depicted in Figures 4.29 and 4.30, which serve as reference tables for equipment and work orders. The primary purpose of a reference table is to offer guidance to the user; for example, when a user needs to check a customer's ID to finalize an equipment record, this interface facilitates that process, as illustrated in Figure 4.31.

ReferenceTables



Reference Table for Equipment

	Asset#	EquipmentName	Manufacturer	Serial#	EquipmentType	CustomerID
▶	11	60 KVA Generator	SDEC	SDH-60	Power Equipment	EG
	16	90 KVA Generator	SDEC	SDH-90	Power Equipment	AA
	17	60 KVA Generator	SDEC	SDH-60	Power Equipment	AA
	18	125 KVA Generator	SDEC	SDH-125	Power Equipment	AB
	19	Wheel Loader	Dossan	DL-300	Construction Equi...	DZ
	20	Wheel Loader	Dossan	DL-300	Construction Equi...	AS
	21	Wheel Loader	Dossan	DL-300	Construction Equi...	AS
	22	Wheel Loader	Dossan	DL-300	Construction Equi...	AS
	23	Wheel Loader	Dossan	DL-300	Construction Equi...	AS
	24	125 KVA Generator	SDEC	SDH-125	Power Equipment	AB
	25	125 KVA Generator	SDEC	SDH-125	Power Equipment	AB
	26	Crawler Excavator	Dossan	DX225LCA	Construction Equi...	OC
	27	Pneumatic Diesel...	Dossan	D15S	Lifting Equipment	EG

Figure 4.29: Reference Tables for Equipment

ReferenceTables



Reference Table for Work Orders

	WO#	CallDate	CallTime	Site#	CallerName
▶	4	19/09/2015 11:5...	19/09/2015 11:5...	20	Ahmed

Figure 4.30: Reference Tables for Work Order

ReferenceTables



Reference Table for Suppliers

	CustomerID	CustomerName	Address	City	CustomerPhone	Cus
▶	ST	SDGC STOCK		North Providence		
	SC	SME CRUSHER ...				
	SA	SAUDI ARABIAN...				
	SD	SDEC				
	AA	A.A. TURKI COR...				
	AI	ALLCHEM INTE...				
	HT	HONGTAK				
	EG	EVERDIGM				
	AB	AL BAKREIN EST.				
	AC	AL BECITEEN				
	AM	AL MIRA CENTE...				
	BA	BRITISH AEROS...		Riyadh		
	GL	GREEN LAND L...				
	OC	AL OSAIS CONT...				
	DS	DOSSAN				
	AS	AFRICA MUSLIM...				
	3722	AL MABANI GEN...				
	5622	ARABIAN CONS...				

Figure 4.31: Reference Tables for Customer

4.3.11 EMIS

If users want to learn more about the system and the designer's information, they can choose "About EMIS" from the menu. The image in Figure 4.30 Displays the form for "About EMIS"

About

Equipment Management Information System



Designed Developed By Raed Alkubati

Email: Raed.alkubati@gmail.com

Figure 4.32: About EMIS Form

CHAPTER 5

CONCLUSION AND RECCOMENDATION

5.1 Conclusion

The equipment management information system (EMIS), briefly mentioned here, is created utilizing the relational database idea, with Microsoft Access as the backend and Microsoft Visual Studio as the frontend software toolset in use. One of EMIS's advantages over applications lies in its ability to handle a variety of system functions like storing equipment details and generating reports, as well as providing PPM notifications that prompt users about events. This feature differentiates it from systems that merely store data without processing it into actionable reports or sending user reminders. Furthermore, users can make changes to the database tables using interfaces provided by the system, which also features search and delete options for locating various records, like equipment and customer data in (EMIS).

A series of steps and procedures must be followed to develop this system effectively and meet project deadlines on time. Firstly, retrieving data and project requirements from the Saudi Diesel Equipment Company is essential. Then, the database was constructed using Microsoft Access with normalization rules and a data dictionary. Then, the graphical user interface for each module was created using Microsoft Visual Studio. To assess the performance and effectiveness of the system, we conducted tests using data samples entered into the database. The results were positive.

Users of the EMIS system can monitor equipment and its planned preventive maintenance (PPM) schedules to ensure that each piece of equipment functions correctly and undergoes regular efficiency and accuracy checks as needed for operation and performance optimization with a few simple keystrokes in the system interface. By leveraging the generated reports and providing data insights, upper management can efficiently save time when making decisions and promptly take necessary actions at any given moment.

The Equipment Management Information System (EMIS) is incredibly valuable and accessible to anyone, regardless of age or level of computer expertise. This system has been designed with a user interface that allows users to easily navigate through menus and carry out tasks while managing data in the database efficiently. Whether you're new to computers or an expert in the field, EMIS caters to your needs by providing an approach to data storage that eliminates paperwork and saves time.

The project goals have been met by collecting the data and creating the database for the system; following that step is the creation and proper installation of the graphical user interface, which includes a range of features and functions for users to interact with and manage data within the database.

5.2 Recommendation for future work

Additional enhancements and features could be incorporated into the Equipment Management Information System (EMIS) to boost its performance and provide users with options and functionalities. The existing security measures need improvement as they are less effective than desired. Different access level rights could be introduced to distinguish

between user authorization levels, allowing certain users to make modifications to the database while restricting others to only adding equipment or suppliers. An alert system that prompts and motivates users to update their passwords regularly can be introduced to prevent password use.

To improve the system functionality and better meet user requirements, we could introduce features, like statistical analysis tools, to provide users with options and analytical functions for their needs. These tools would be helpful in generating a summary table that outlines the scheduled maintenance work orders (PPM) as well as their progress on a monthly basis and over specific time frames. This enables users to differentiate between tasks completed within the time frame and those delayed. Furthermore, enhanced data validation will lead to a potent system as users will not need to be concerned about inadvertently inputting incorrect characters or unfamiliar data into the system. This will contribute to an improved system.

Questionnaire:

1. Hypothesis Structure and Focus Areas

- **Hypothesis 1 examines whether the MIS enhances service quality, particularly speed, customer satisfaction, error reduction, and communication.**
- **Hypothesis 2 investigates how the MIS impacts decision-making efficiency by providing accurate data, reducing guesswork, and enabling quick access to necessary information.**
- **Hypothesis 3 focuses on the role of MIS in managing and maintaining equipment, assessing aspects like maintenance tracking, cost reduction, and operational efficiency.**
- **Hypothesis 4 looks at the user interface's design, checking if it is intuitive, supportive, and organized for a seamless experience.**
- **Hypothesis 5 assesses how MIS improves inter-departmental coordination, communication, and workflow organization across the organization.**

2. Survey Design and Distribution

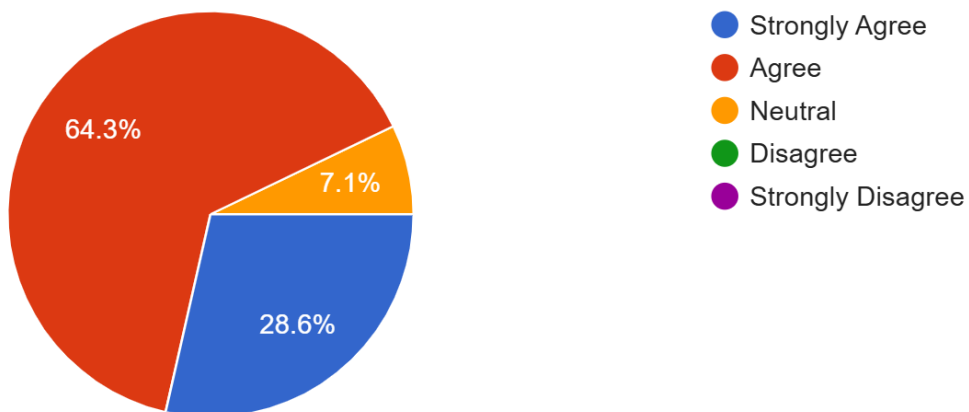
- **The survey includes 25 questions spread across five hypotheses, with five questions per hypothesis to capture a comprehensive view.**

- 56 participants responded, which is moderate sample size for preliminary insights but might benefit from a larger sample for more generalized conclusions.

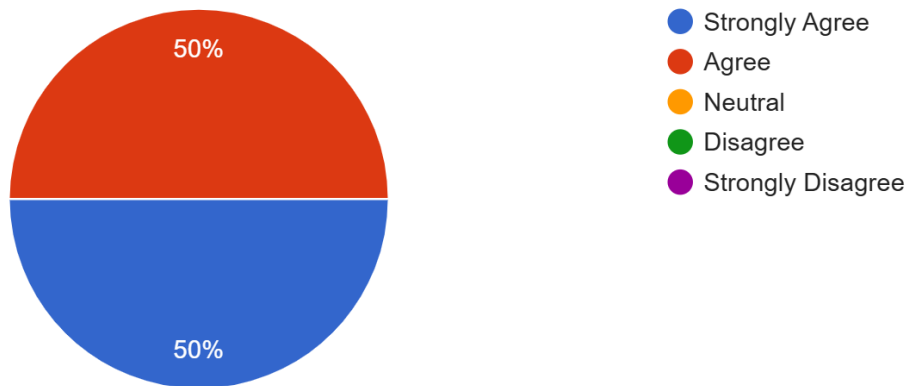
The results were as follows:

Hypothesis 1: The management information system contributes to improving the quality of service

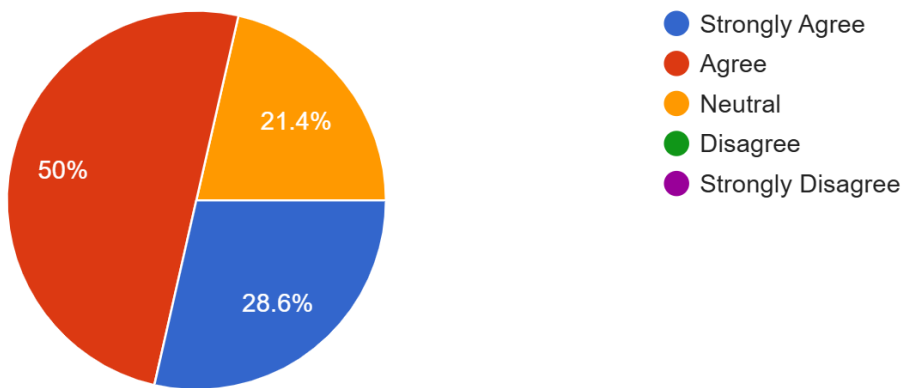
1/ I believe that the management information system helps in providing faster services



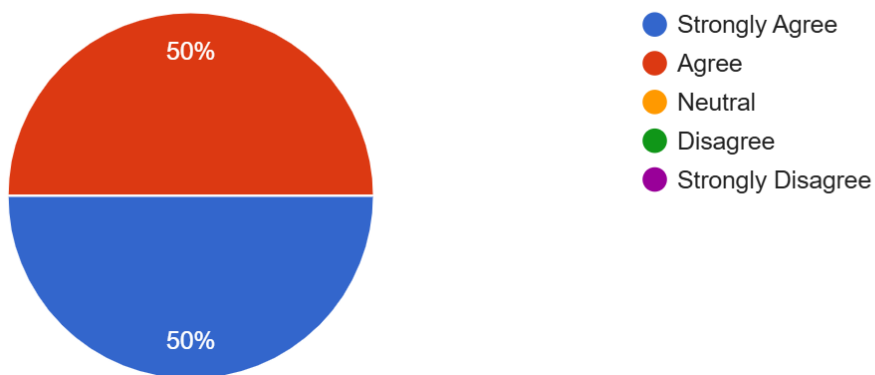
2 / I think the system contributes to customer satisfaction.



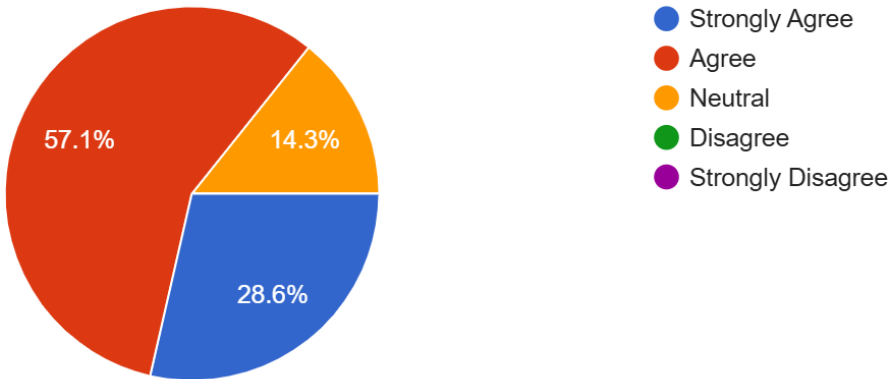
3 / I believe the system reduces errors in service delivery



4 / I think the system simplifies tracking customer requests and inquiries

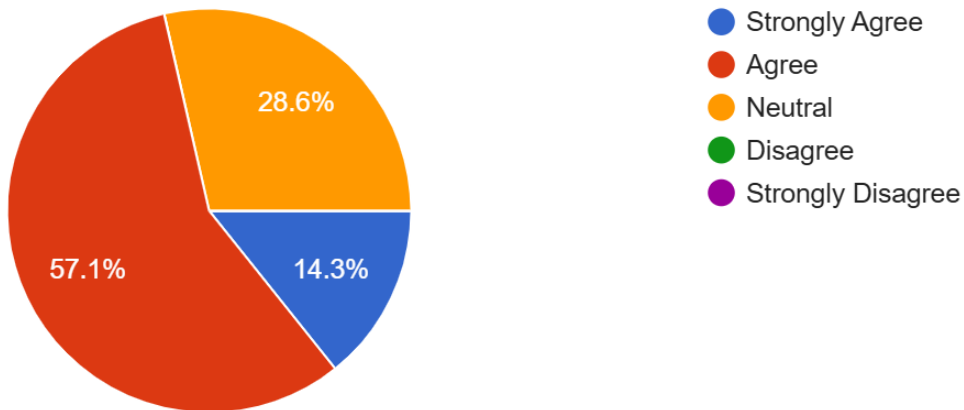


5 / I believe the system enhances internal communication to improve service quality

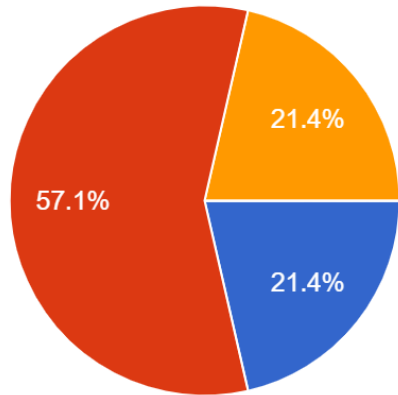


Hypothesis 2: The MIS contributes to improving decision-making efficiency

1 / I believe the system provides accurate data that aids in decision-making

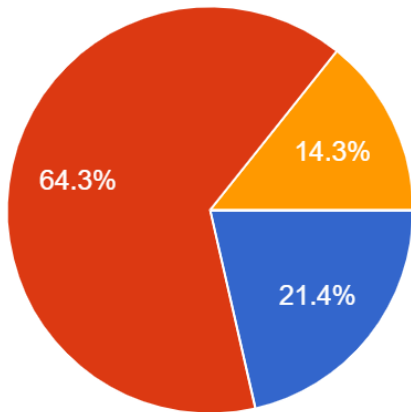


2 / I think the system helps speed up decision-making processes.



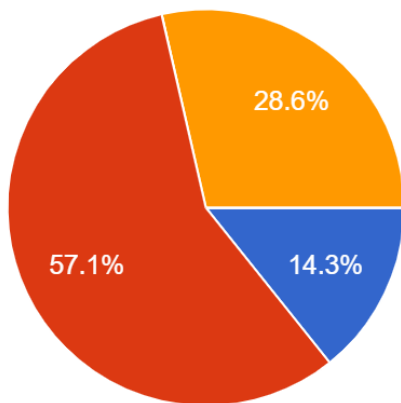
- Strongly Agree
- Agree
- Neutral
- Disagree
- Strongly Disagree

3 / I believe the information available in the system is comprehensive and supports better decisions



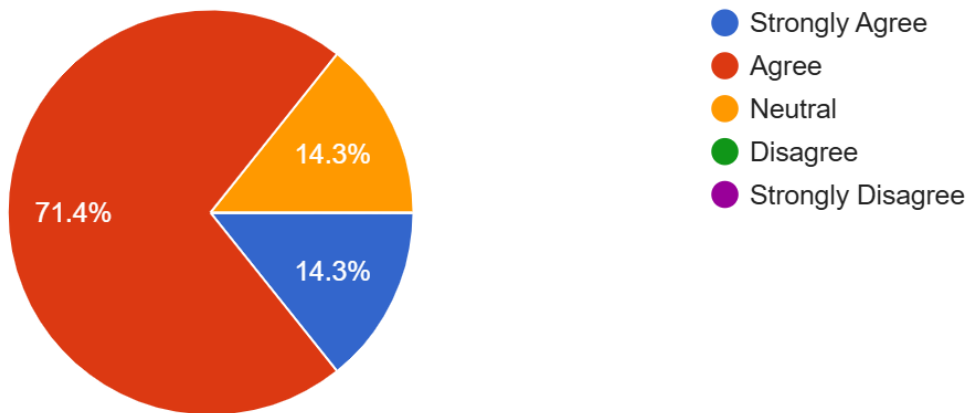
- Strongly Agree
- Agree
- Neutral
- Disagree
- Strongly Disagree

4 / I think the system reduces reliance on guesswork in decision-making



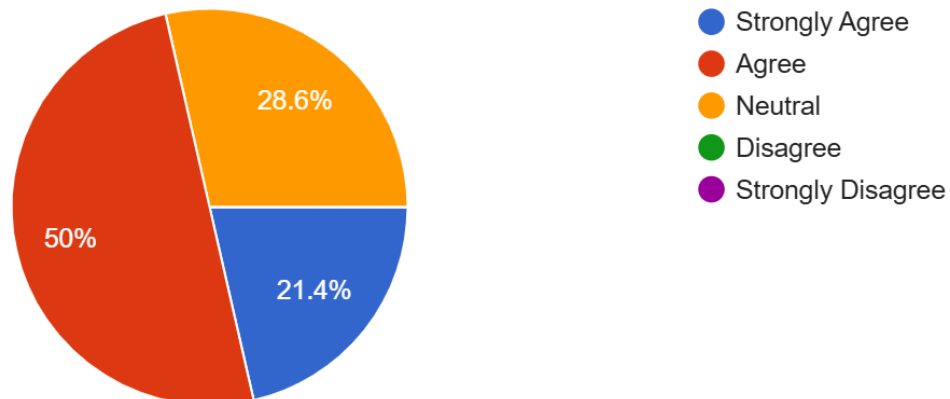
- Strongly Agree
- Agree
- Neutral
- Disagree
- Strongly Disagree

5 / I believe the system provides instant access to data needed for decision-making

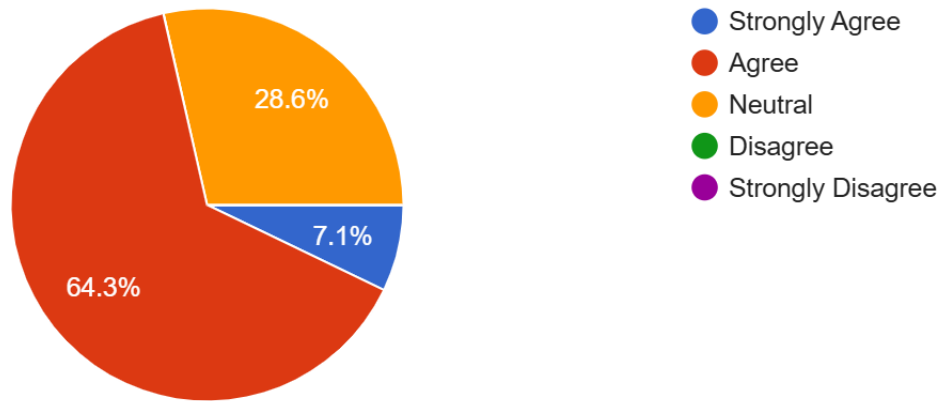


Hypothesis 3: The MIS contributes to improving equipment management and maintenance

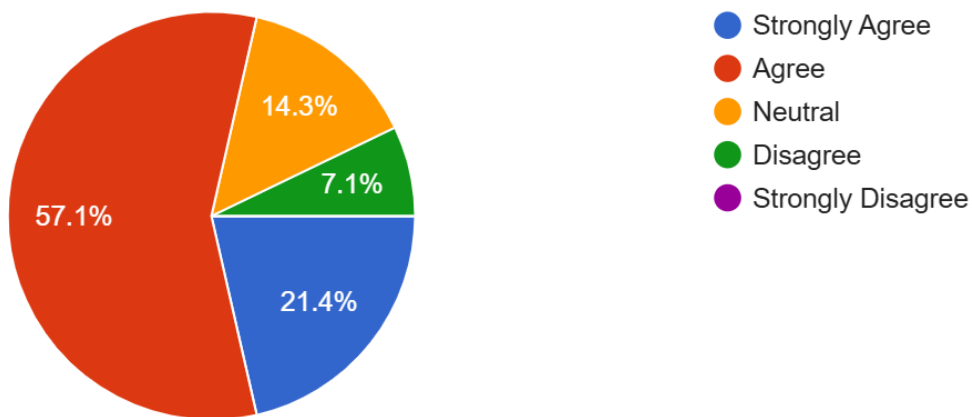
1 / I believe the system simplifies tracking equipment maintenance



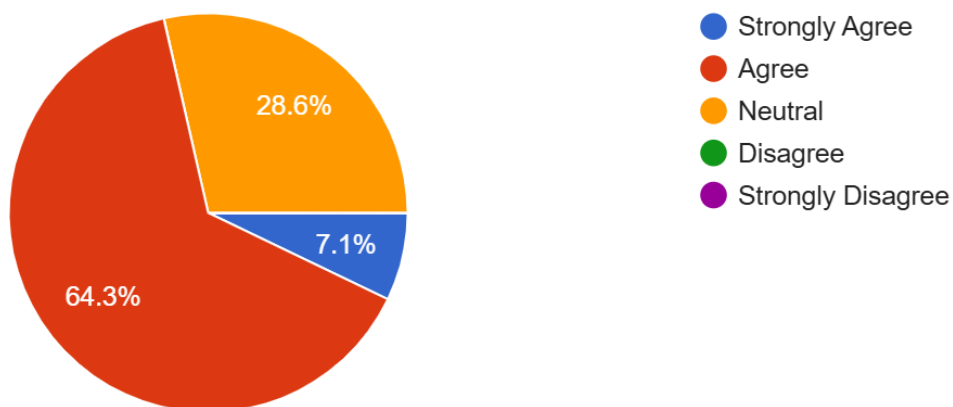
2 / I think the system enables better planning for preventive maintenance.



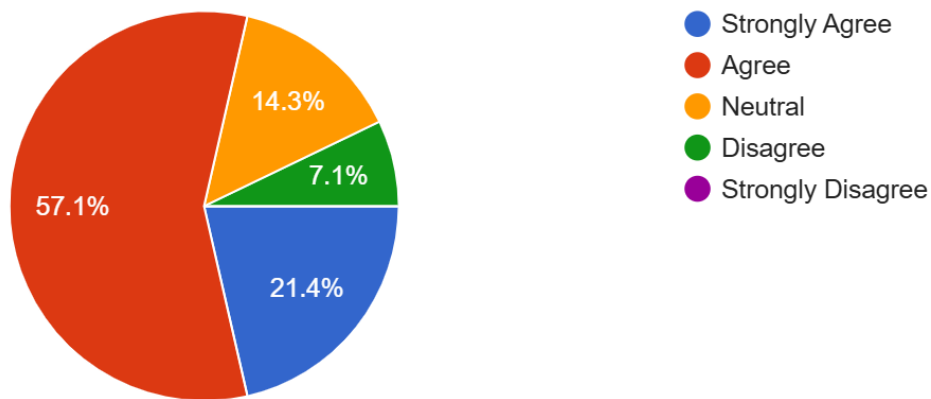
3 / I believe the system reduces equipment downtime, leading to higher operational efficiency



4 / I think the system provides accurate and timely information on maintenance schedules

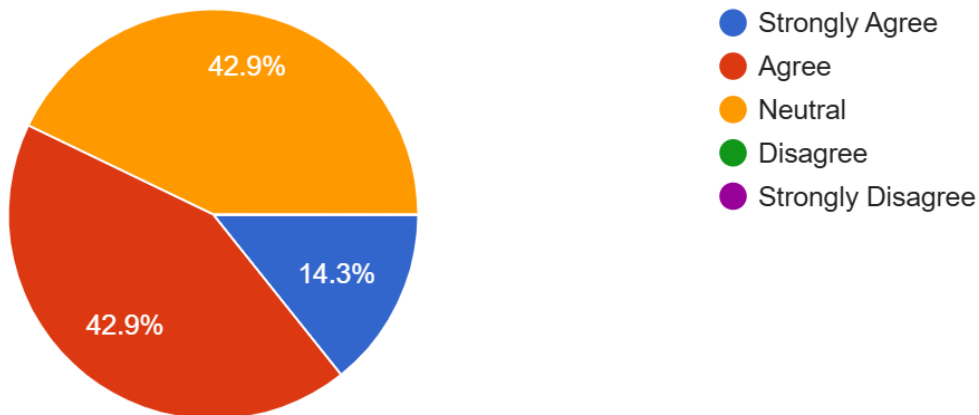


5 / I believe the system helps to lower maintenance costs by optimizing the scheduling and tracking of repairs

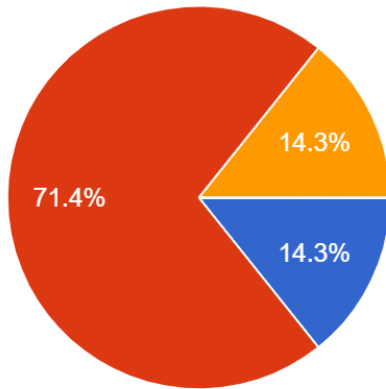


Hypothesis 4: The MIS interface is user-friendly and provides a comfortable user experience

1 / I believe the system interface is intuitive and easy to navigate

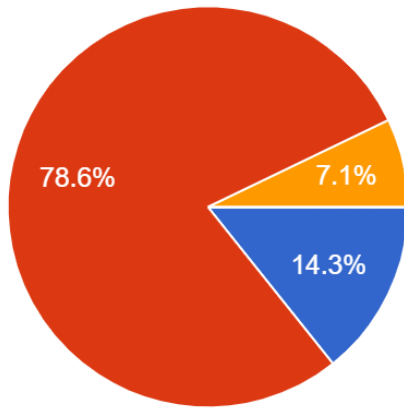


2 / I think the design of the system allows users to complete tasks quickly and efficiently



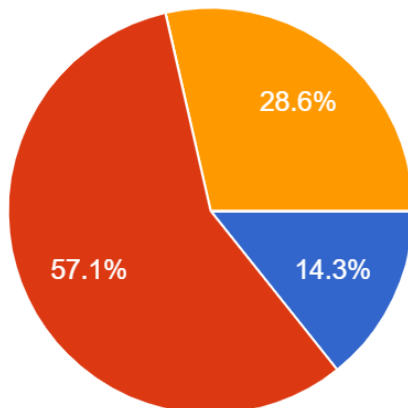
- Strongly Agree
- Agree
- Neutral
- Disagree
- Strongly Disagree

3 / I believe the system offers a pleasant and supportive experience for daily operations



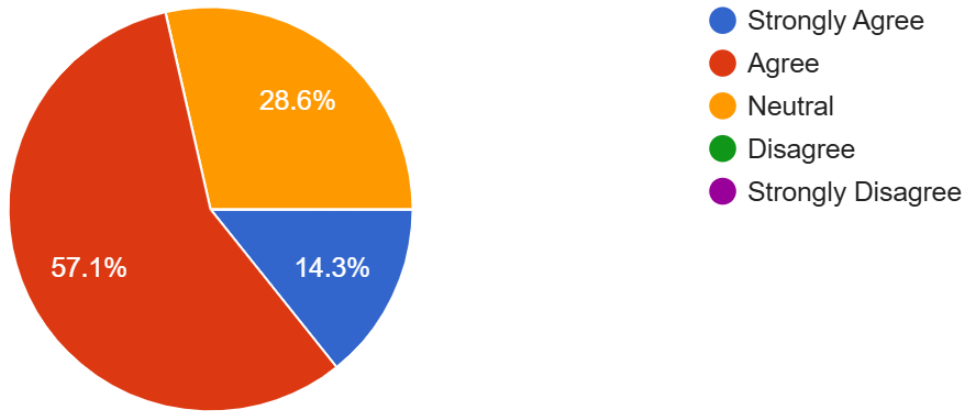
- Strongly Agree
- Agree
- Neutral
- Disagree
- Strongly Disagree

4 / I think the system interface is well-organized, making information easily accessible



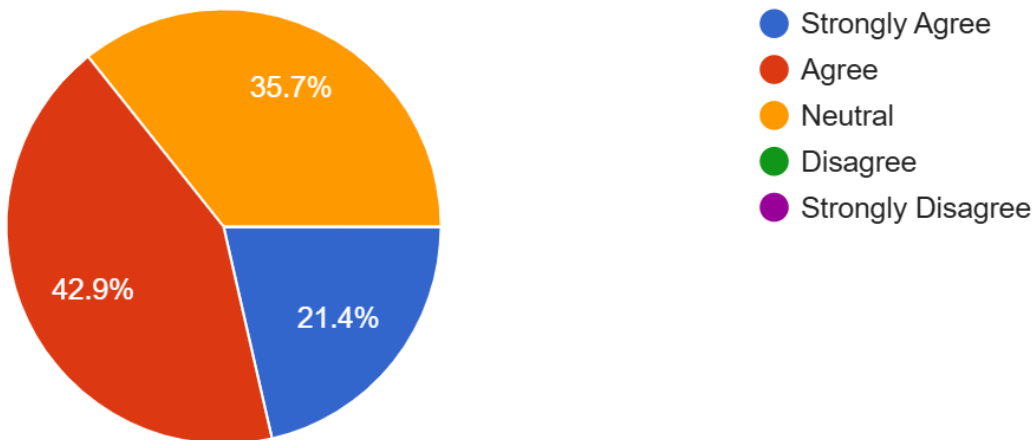
- Strongly Agree
- Agree
- Neutral
- Disagree
- Strongly Disagree

5 / I believe the system facilitates smooth transitions between different screens and tasks without delays

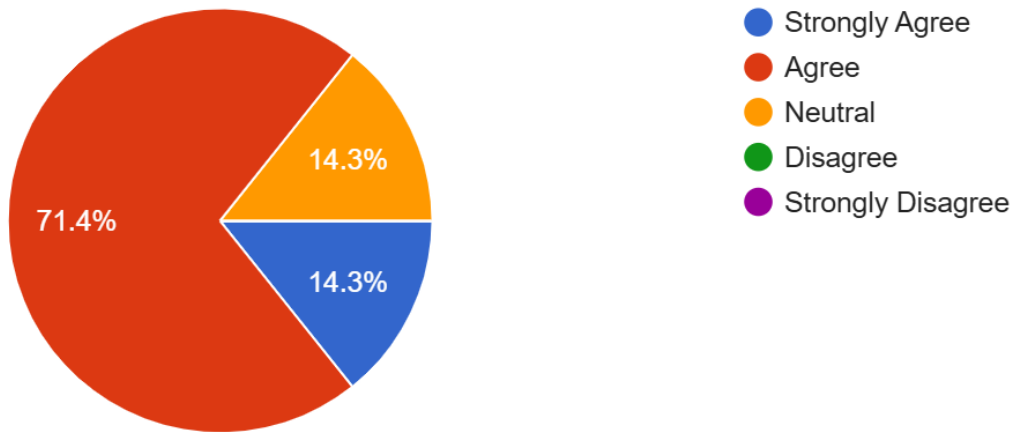


Hypothesis 5: The system helps improve coordination between different departments in the organization

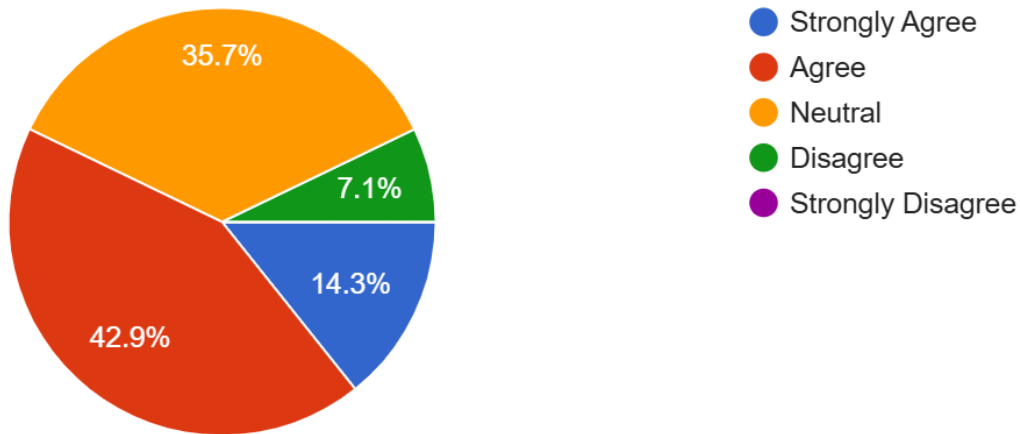
1 / I believe the system enhances collaboration and information sharing between departments



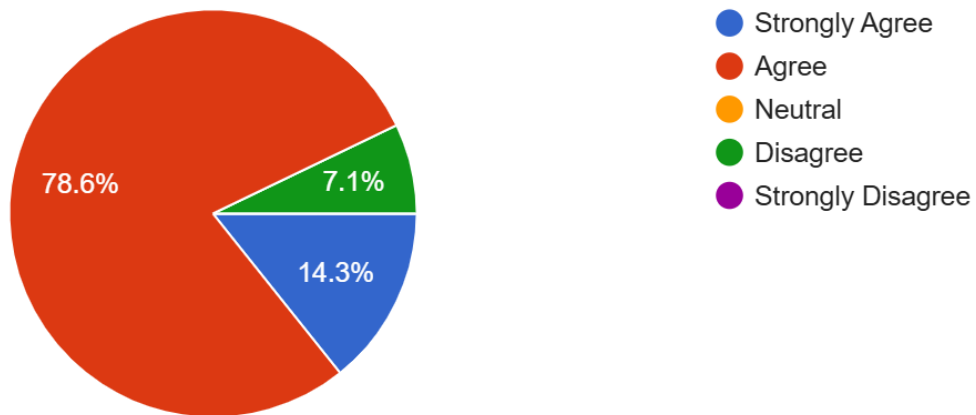
2 / I think the system allows departments to communicate smoothly, supporting overall workflow



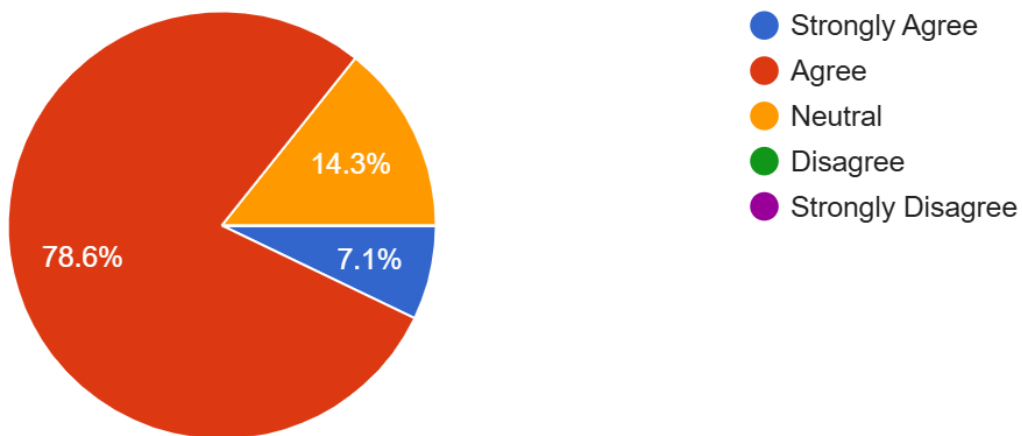
3 / I believe the system reduces duplicated efforts by providing shared access to critical information



4 / I think the system organizes workflows effectively, ensuring tasks are coordinated across departments



5 / I believe the system allows each department to access necessary information from other departments, leading to more efficient operations



3. Interpretation of Results

Assuming the results are available or scored (such as a Likert scale for agreeability or effectiveness), the findings could reveal the following insights:

- **Service Quality:** Responses to Hypothesis 1 can indicate whether the MIS is perceived to positively impact speed and accuracy, which can be linked to overall customer satisfaction and internal communication improvements.
- **Decision-Making Efficiency:** Hypothesis 2 results can help assess how MIS influences the quality of decision-making, particularly in supporting data-driven choices and reducing dependency on less reliable methods.
- **Equipment Management:** Hypothesis 3 insights can identify if MIS contributes to effective maintenance management, a key factor for reducing downtime and optimizing costs.
- **User Experience:** Hypothesis 4 results provide direct feedback on the interface's usability, showing if it is an asset for users or if design areas need improvement.
- **Coordination and Workflow:** Responses to Hypothesis 5 could clarify how well the MIS supports cross-departmental workflows, helping to eliminate redundancies and ensure smooth task transitions.

4. Potential Insights and Recommendations

- **High Positive Scores** across questions could indicate a generally effective MIS that aligns well with institutional goals, suggesting the system may continue with minor adjustments.
- **Mixed Responses** might show areas for improvement; for example, if participants note challenges in the user interface, it might benefit from a redesign to enhance intuitiveness.
- **Low Scores** in specific areas like decision-making or inter-departmental coordination would suggest targeted adjustments to enhance those functions, possibly with training or additional MIS features.

REFERENCES

Books

1. Ambler, S. W., & Sadalage, P. J. (2006). *Refactoring databases: Evolutionary database design*. Pearson Education.
2. Blaha, M., & Premerlani, W. (1997). *Object-oriented modeling and design for database applications*. Prentice Hall.
3. Dittrich, K. R., Dayal, U., & Buchmann, A. P. (2012). *On object-oriented database systems*. Springer Science & Business Media.
4. Elmasri, R., & Navathe, S. B. (2014). *Fundamentals of database systems* (6th ed.). Pearson.
5. Hernandez, M. J. (2013). *Database design for mere mortals: A hands-on guide to relational database design*. Pearson Education.
6. Lightstone, S. S., Teorey, T. J., & Nadeau, T. (2010). *Physical database design: The database professional's guide to exploiting indexes, views, storage, and more*. Morgan Kaufmann.
7. Naiburg, E. J., & Maksimchuck, R. A. (2001). *UML for database design*. Addison-Wesley.
8. Satzinger, J., Jackson, R., & Burd, S. D. (2011). *Systems analysis and design in a changing world*. Cengage Learning.
9. Schneider, D. I. (2013). *An introduction to programming using Visual Basic 2012*. Prentice Hall.
10. Stephens, R. (2010). *Visual Basic graphics programming*. Wiley.

Journal Articles

1. Alshawi, A. (2001). Logistics in the Internet age: Towards a holistic information and processes picture. *Logistics Information Management*, 14(4), 235–241.
2. Atkinson, M. P., Bancilhon, F., DeWitt, D. J., Dittrich, K. R., Maier, D., & Zdonik, S. B. (1989). The object-oriented database system manifesto. *Dood*, 89, 40–57.
3. Best, D. P. (1988). The future of information management. *International Journal of Information Management*, 8(1), 13–24.
4. Chen, D. Q., Mocker, M., Preston, D. S., & Teubner, A. (2010). Information systems strategy: Reconceptualization, measurement, and implications. *MIS Quarterly*, 34(2), 233–A8.
5. Siau, K., & Rossi, M. (2011). Evaluation techniques for systems analysis and design modeling methods—A review and comparative analysis. *Information Systems Journal*, 21(3), 249–268.

Conference Papers

1. Marshall, P., & McKay, J. (2000). Rethinking information systems planning in strategic business networks. In *Proceedings of the Americas Conference on Information Systems* (p. 855).
2. Zilio, D. C., Rao, J., Lightstone, S., Lohman, G., Storm, A., Garcia-Arellano, C., & Fadden, S. (2004). DB2 design advisor: Integrated automatic physical database design. In *Proceedings of the Thirtieth International Conference on Very Large Data Bases* (Vol. 30, pp. 1087–1097).

APPENDIX A

The codes for each form:

Login

Public Class Form1

Dim counter As Integer = 0

Private Sub timeTimer_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles timeTimer.Tick labelTime.Text = Date.Now.ToString
End Sub

Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

If (textUsername.Text.Length = 0) Then

MsgBox("Please enter your username", MsgBoxStyle.OkOnly, "Username Required")

Return End If

If (textUsername.Text.Length = 0) Then

MsgBox("Please enter your username", MsgBoxStyle.OkOnly, "Username Required")

Return End If

If (loginAccount(textUsername.Text, textPassword.Text)) Then

textUsername.Text = "" textPassword.Text = ""

textUsername.Focus() counter = 0 Me.Hide()

Form_MainMenu.Show() Else counter = counter + 1

```
MsgBox("Either your username or your password is incorrect. Please try again",  
MsgBoxStyle.OkOnly, "Invalid User Credentials")    textPassword.Text = ""    End If
```

```
If (counter = 3) Then  
    MsgBox("Access Denied", MsgBoxStyle.OkOnly, "Exit System")  
    Me.Close()    End If
```

```
End Sub
```

```
Private Function loginAccount(ByVal username As String, ByVal password As  
String) As Boolean
```

```
    Dim result As Boolean = False
```

```
    Dim connection As New  
OleDb.OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" &  
Application.StartupPath & "\FYP.accdb;Persist Security Info=False;")
```

```
connection.Open()
```

```
    Dim command As New OleDb.OleDbCommand("SELECT ID FROM users WHERE  
USERNAME = " & username & " AND PASSWORD = " & password & """)
```

```
command.Connection = connection
```

```
    Dim reader As OleDb.OleDbDataReader = command.ExecuteReader()
```

```
If (reader.Read() = True) Then
```

```
    Form_MainMenu.setUserInfo(username, password)
```

```
connection.Close()    result = True    End If
```

```
Return result    End Function
```

```
Private Sub Form1_Load(ByVal
```

sender As System.Object, ByVal e

As

System.EventArgs) Handles MyBase.Load

End Sub

End Class

Main Menu

Public Class Form_MainMenu

Dim connection As New

OleDb.OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" &
Application.StartupPath & "\FYP.accdb;Persist Security Info=False;")

Dim username As String Dim

password As String

Private Sub WOStatusToolStripMenuItem_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles WOStatusToolStripMenuItem.Click

WOStatus.Show()

End Sub

Private Sub ChangeUsernameToolStripMenuItem_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ChangeUsernameToolStripMenuItem.Click

Dim currentUsername As String = username

Dim newUsername As String = InputBox("Please enter your new username", "Changing
Username", currentUsername)

If (newUsername.Length > 0) Then

If (newUsername.Equals(currentUsername) = False) Then

If (executeSQL("UPDATE users SET USERNAME = " & newUsername &

```

    "" WHERE USERNAME = "" & currentUsername & """) > 0) Then

setUserInfo(newUsername, password)

    MsgBox("Your username has been changed successfully", MsgBoxStyle.OkOnly,
"Username Changed")

    Else

        MsgBox("An error has occurred. Please try again later.", MsgBoxStyle.OkOnly,
"Account Update Failed")

    End If

    Else

        MsgBox("Please enter proper username", MsgBoxStyle.OkOnly,
"Invalid Input")

    End If

    End If

End Sub

```

```

Private Sub Form_MainMenu_FormClosed(ByVal sender As System.Object, ByVal e
As System.Windows.Forms.FormClosedEventArgs) Handles MyBase.FormClosed

```

```

    Form1.Show()

```

```

End Sub

```

```

Private Function executeSQL(ByVal SQL As String) As Integer

```

```

    Console.WriteLine(SQL)    Dim result As

```

```

Boolean = False    connection.Open()

```

```

    Dim command As New OleDb.OleDbCommand(SQL)

```

```

command.Connection = connection

```

```

    Dim affected As Integer

```

```

    Try    affected = command.ExecuteNonQuery()

```

```

Catch ex As Exception    affected = -1

```

```
MsgBox(ex.Message, MsgBoxStyle.OkOnly, "Operation Failed")
```

```
End Try
```

```
connection.Close() Return
```

```
affected End Function
```

```
Public Sub setUserInfo(ByVal username As String, ByVal password As String)
```

```
Me.username = username
```

```
Me.password = password
```

```
End Sub
```

```
Private Sub ChangePasswordToolStripMenuItem_Click(ByVal sender As  
System.Object, ByVal e As System.EventArgs) Handles  
ChangePasswordToolStripMenuItem.Click
```

```
Dim currentPassword As String = password
```

```
Dim newPassword As String = InputBox("Please enter your new password", "Changing  
Password", currentPassword)
```

```
If (newPassword.Length > 0) Then
```

```
If (newPassword.Equals(currentPassword) = False) Then
```

```
If (executeSQL("UPDATE users SET [PASSWORD] = '" & newPassword  
& "' WHERE USERNAME = '" & username & "'") > 0) Then
```

```
setUserInfo(username, newPassword)
```

```
MsgBox("Your password has been changed successfully", MsgBoxStyle.OkOnly,  
"Password Changed")
```

```
Else
```

```
MsgBox("An error has occurred. Please try again later.", MsgBoxStyle.OkOnly,  
"Account Update Failed")
```

```
End If
```

```
Else
```

```
MsgBox("Please enter proper password", MsgBoxStyle.OkOnly,
```

"Invalid Input")

End If End If

End Sub

Private Sub AddNewUserToolStripMenuItem_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
AddNewUserToolStripMenuItem.Click

Dim newUser_username As String = InputBox("Please enter new username", "Adding New
User")

If (newUser_username.Length > 0) Then

Dim newUser_password As String = InputBox("Please enter password", "Adding New
User")

If (newUser_password.Length > 5) Then

If (executeSQL("insert into users (USERNAME, [PASSWORD])
VALUES('" & newUser_username & "', '" & newUser_password & "')") > 0) Then

MsgBox("New user has been added successfully",
MsgBoxStyle.OkOnly, "Successful")

Else

MsgBox("An error has occurred. Please try again later.", MsgBoxStyle.OkOnly,
"Addition Failed")

End If

Else

MsgBox("Minimum password size is 5 characters. Operation
Cancelled", MsgBoxStyle.OkOnly, "Operation Cancelled")

End If

Else

MsgBox("Invalid username received. Operation Cancelled",
MsgBoxStyle.OkOnly, "Operation Cancelled")

End If

End Sub

```
Private Sub MainInventoryToolStripMenuItem_Click(ByVal sender As  
System.Object, ByVal e As System.EventArgs) Handles  
MainInventoryToolStripMenuItem.Click MainInventory.Show()
```

```
End Sub
```

```
Private Sub NewToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As  
System.EventArgs) Handles NewToolStripMenuItem.Click
```

```
WorkOrder.setMode("new") WorkOrder.Show()
```

```
End Sub
```

```
Private Sub SearchUpdateToolStripMenuItem_Click(ByVal sender As  
System.Object, ByVal e As System.EventArgs) Handles  
SearchUpdateToolStripMenuItem.Click
```

```
Dim recordID As String = InputBox("Please enter WO#", "Searching Work  
Order", "")
```

```
If (recordID.Length > 0) Then
```

```
WorkOrder.setMode("update")
```

```
WorkOrder.showRecord(recordID)
```

```
WorkOrder.Show()
```

```
Else
```

```
MsgBox("Please enter proper ID", MsgBoxStyle.OkOnly, "Invalid  
Input")
```

```
End If
```

```
End Sub
```

```
Private Sub OpenToolStripMenuItem_Click(ByVal sender As System.Object,  
ByVal e As System.EventArgs) Handles OpenToolStripMenuItem.Click
```

```
WOSearcher.filterWO("Open") WOSearcher.Show()
```

```
End Sub
```

```
Private Sub InProgressToolStripMenuItem_Click(ByVal sender As  
System.Object, ByVal e As System.EventArgs) Handles  
InProgressToolStripMenuItem.Click
```

```
    WOSearcher.filterWO("In Progress")
```

```
    WOSearcher.Show()
```

```
End Sub
```

```
Private Sub CompletedToolStripMenuItem_Click(ByVal sender As System.Object,  
ByVal e As System.EventArgs) Handles CompletedToolStripMenuItem.Click
```

```
    WOSearcher.filterWO("Completed")
```

```
    WOSearcher.Show()
```

```
End Sub
```

```
Private Sub ClosedToolStripMenuItem_Click(ByVal sender As System.Object,  
ByVal e As System.EventArgs) Handles ClosedToolStripMenuItem.Click
```

```
    WOSearcher.filterWO("Closed")    WOSearcher.Show()
```

```
End Sub
```

```
Private Sub EquipmentToolStripMenuItem_Click(ByVal sender As System.Object,  
ByVal e As System.EventArgs) Handles EquipmentToolStripMenuItem.Click  
Inventory.filterInventory("Equipment")
```

```
    Inventory.Show()
```

```
End Sub
```

```
Private Sub SupplierToolStripMenuItem_Click(ByVal sender As System.Object,  
ByVal e As System.EventArgs) Handles SupplierToolStripMenuItem.Click
```

```
    Inventory.filterInventory("Customer")
```

```
    Inventory.Show()
```

```
End Sub
```

```
Private Sub SiteToolStripMenuItem_Click(ByVal sender As System.Object,  
ByVal e As System.EventArgs) Handles SiteToolStripMenuItem.Click
```

```
    Inventory.filterInventory("Site")
```

```
    Inventory.Show()
```

```
End Sub
```

```
Private Sub LocationToolStripMenuItem_Click(ByVal sender As System.Object,  
ByVal e As System.EventArgs) Handles LocationToolStripMenuItem.Click
```

```
    Inventory.filterInventory("Location")
```

```
    Inventory.Show()
```

```
End Sub
```

```
Private Sub EquipmentsToolStripMenuItem_Click(ByVal sender As  
System.Object, ByVal e As System.EventArgs) Handles EquipmentsToolStripMenuItem.Click
```

```
    ReferenceTables.filterInventory("Equipments")
```

```
    ReferenceTables.Show()
```

```
End Sub
```

```
Private Sub SuppliersToolStripMenuItem_Click(ByVal sender As System.Object,  
ByVal e As System.EventArgs) Handles SuppliersToolStripMenuItem.Click
```

```
    ReferenceTables.filterInventory("Customers")
```

```
    ReferenceTables.Show()
```

```
End Sub
```

```
Private Sub SitesToolStripMenuItem_Click(ByVal sender As System.Object,  
ByVal e As System.EventArgs) Handles SitesToolStripMenuItem.Click
```

```
    ReferenceTables.filterInventory("Sites")
```

```
ReferenceTables.Show()
```

```
End Sub
```

```
Private Sub LocationsToolStripMenuItem_Click(ByVal sender As System.Object,  
ByVal e As System.EventArgs) Handles LocationsToolStripMenuItem.Click
```

```
ReferenceTables.filterInventory("Locations")
```

```
ReferenceTables.Show()
```

```
End Sub
```

```
Private Sub WorkOrdersToolStripMenuItem_Click(ByVal sender As  
System.Object, ByVal e As System.EventArgs) Handles WorkOrdersToolStripMenuItem.Click
```

```
ReferenceTables.filterInventory("Work Orders")
```

```
ReferenceTables.Show()
```

```
End Sub
```

```
Private Sub AssetToolStripMenuItem_Click(ByVal sender As System.Object,  
ByVal e As System.EventArgs) Handles AssetToolStripMenuItem.Click
```

```
Dim id As String = InputBox("Please enter Asset#", "Searching  
Equipments", "")
```

```
If (id.Length > 0) Then
```

```
AdvancedSearcher.searchAssetByID(id)
```

```
AdvancedSearcher.Show()
```

```
Else
```

```
MsgBox("Please enter proper ID", MsgBoxStyle.OkOnly, "Invalid  
Input")
```

```
End If
```

```
End Sub
```

```
Private Sub SuplierIDToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles SuplierIDToolStripMenuItem.Click
```

```
    Dim keyword As String = InputBox("Please enter Customer ID", "Searching Equipments", "")
```

```
    If (keyword.Length > 0) Then
```

```
        AdvancedSearcher.searchAssetBySupplierID(keyword)
```

```
        AdvancedSearcher.Show()
```

```
    Else
```

```
        MsgBox("Please enter proper Customer ID", MsgBoxStyle.OkOnly, "Invalid Input")
```

```
    End If
```

```
End Sub
```

```
Private Sub SiteToolStripMenuItem1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles SiteToolStripMenuItem1.Click
```

```
    Dim keyword As String = InputBox("Please enter Site ID", "Searching Equipments", "")
```

```
    If (keyword.Length > 0) Then
```

```
        AdvancedSearcher.searchAssetBySiteID(keyword)
```

```
        AdvancedSearcher.Show()
```

```
    Else
```

```
        MsgBox("Please enter site ID", MsgBoxStyle.OkOnly, "Invalid Input")
```

```
    End If
```

```
End Sub
```

```
Private Sub MDToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MDToolStripMenuItem.Click
```

```
    Dim keyword As String = InputBox("Please enter MD ID", "Searching Equipments", "")
```

```

If (keyword.Length > 0) Then
    AdvancedSearcher.searchAssetByMDID(keyword)
    AdvancedSearcher.Show()
Else
    MsgBox("Please enter proper MD ID", MsgBoxStyle.OkOnly, "Invalid
Input")
End If
End Sub

```

```

Private Sub LocationToolStripMenuItem1_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles LocationToolStripMenuItem1.Click
    Dim keyword As String = InputBox("Please enter Location", "Searching
Equipments", "")
    If (keyword.Length > 0) Then
        AdvancedSearcher.searchAssetByLocation(keyword)
        AdvancedSearcher.Show()
    Else
        MsgBox("Please enter proper Location", MsgBoxStyle.OkOnly, "Invalid
Input")
    End If
End Sub

```

```

Private Sub POToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles POToolStripMenuItem.Click
    Dim keyword As String = InputBox("Please enter PO", "Searching
Equipments", "")
    If (keyword.Length > 0) Then
        AdvancedSearcher.searchAssetByPO(keyword)
        AdvancedSearcher.Show()
    Else

```

```

        MsgBox("Please enter proper PO", MsgBoxStyle.OkOnly, "Invalid
Input")

    End If

End Sub

```

```

Private Sub WOToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles WOToolStripMenuItem.Click

```

```

    Dim keyword As String = InputBox("Please enter WO", "Searching
Equipments", "")

```

```

    If (keyword.Length > 0) Then

```

```

        AdvancedSearcher.searchAssetByWO(keyword)

```

```

        AdvancedSearcher.Show()

```

```

    Else

```

```

        MsgBox("Please enter proper WO", MsgBoxStyle.OkOnly, "Invalid
Input")

```

```

    End If

```

```

End Sub

```

```

Private Sub TypeOfAssetToolStripMenuItem_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
TypeOfAssetToolStripMenuItem.Click

```

```

    Dim keyword As String = InputBox("Please enter Asset Type", "Searching
Equipments", "")

```

```

    If (keyword.Length > 0) Then

```

```

        AdvancedSearcher.searchAssetByAssetType(keyword)

```

```

        AdvancedSearcher.Show()

```

```

    Else

```

```

        MsgBox("Please enter proper Asset Type", MsgBoxStyle.OkOnly,
"Invalid Input")

```

```

    End If

```

End Sub

```
Private Sub SearchByMonthToolStripMenuItem_Click(ByVal sender As  
System.Object, ByVal e As System.EventArgs) Handles  
SearchByMonthToolStripMenuItem.Click
```

```
    Dim keyword As String = InputBox("Please enter month in number (ex: 1 for January)",  
"Searching PPM", "")    If (keyword.Length > 0) Then
```

```
        SearchPPM.searchPPMByMonth(keyword)
```

```
        SearchPPM.Show()
```

```
    Else
```

```
        MsgBox("Please enter proper Asset Type", MsgBoxStyle.OkOnly,  
"Invalid Input")
```

```
    End If
```

```
End Sub
```

```
Private Sub SearchUpdateToolStripMenuItem1_Click(ByVal sender As  
System.Object, ByVal e As System.EventArgs) Handles  
SearchUpdateToolStripMenuItem1.Click
```

```
    Dim recordID As String = InputBox("Please enter Asset#", "Searching PPM", "")
```

```
    If (recordID.Length > 0) Then        PPM.searchPPM(recordID)
```

```
        PPM.Show()
```

```
    Else
```

```
        MsgBox("Please enter proper ID", MsgBoxStyle.OkOnly, "Invalid  
Input")
```

```
    End If
```

```
End Sub
```

```
Private Sub LogOutToolStripMenuItem_Click(ByVal sender As System.Object,  
ByVal e As System.EventArgs) Handles LogOutToolStripMenuItem.Click
```

```
    Me.Close()
```

End Sub

```
Private Sub ShowNotificationToolStripMenuItem_Click(ByVal sender As  
System.Object, ByVal e As System.EventArgs) Handles  
ShowNotificationToolStripMenuItem.Click
```

```
showNotification() End Sub
```

```
Private Function checkDate(ByRef dateObj As Object) As Nullable(Of Date)
```

```
    If (IsDBNull(dateObj)) Then
```

```
        Return New Nullable(Of Date)()
```

```
    Else
```

```
        Return dateObj
```

```
    End If End Function
```

```
Private Function findClosestDate(ByVal date1 As Nullable(Of Date), ByVal date2 As  
Nullable(Of Date), ByVal date3 As Nullable(Of Date), ByVal date4 As Nullable(Of Date), ByVal  
date5 As Nullable(Of Date), ByVal date6 As Nullable(Of  
Date), ByVal date7 As Nullable(Of Date), ByVal date8 As Nullable(Of Date),  
ByVal date9 As Nullable(Of Date), ByVal date10 As Nullable(Of Date)) As Date
```

```
    Dim finalDate As Date = Date.Now
```

```
    Dim nearest As Long = Long.MaxValue
```

```
    If (date1.HasValue) Then
```

```
        If (Math.Abs(DateDiff(DateInterval.Second, date1.Value, Date.Now))  
< nearest) Then nearest = DateDiff(DateInterval.Second, date1.Value, Date.Now)
```

```
    finalDate = date1.Value
```

```
    End If End If
```

```
    If (date2.HasValue) Then
```

```
        If (Math.Abs(DateDiff(DateInterval.Second, date2.Value, Date.Now))
```

```

< nearest) Then          nearest = DateDiff(DateInterval.Second, date2.Value, Date.Now)

finalDate = date2.Value

    End If      End If


    If (date3.HasValue) Then

        If (Math.Abs(DateDiff(DateInterval.Second, date3.Value, Date.Now))
< nearest) Then          nearest = DateDiff(DateInterval.Second, date3.Value, Date.Now)

finalDate = date3.Value

        End If      End If


    If (date4.HasValue) Then

        If (Math.Abs(DateDiff(DateInterval.Second, date4.Value, Date.Now))
< nearest) Then          nearest = DateDiff(DateInterval.Second, date4.Value, Date.Now)

finalDate = date4.Value

        End If      End If


    If (date5.HasValue) Then

        If (Math.Abs(DateDiff(DateInterval.Second, date5.Value, Date.Now))
< nearest) Then          nearest = DateDiff(DateInterval.Second, date5.Value, Date.Now)

finalDate = date5.Value

        End If      End If


    If (date6.HasValue) Then

        If (Math.Abs(DateDiff(DateInterval.Second, date6.Value, Date.Now))
< nearest) Then          nearest = DateDiff(DateInterval.Second, date6.Value, Date.Now)

finalDate = date6.Value

        End If      End If

```

```

    If (date7.HasValue) Then
        If (Math.Abs(DateDiff(DateInterval.Second, date7.Value, Date.Now))
< nearest) Then          nearest = DateDiff(DateInterval.Second, date7.Value, Date.Now)

finalDate = date7.Value

        End If      End If

    If (date8.HasValue) Then
        If (Math.Abs(DateDiff(DateInterval.Second, date8.Value, Date.Now))
< nearest) Then          nearest = DateDiff(DateInterval.Second, date8.Value, Date.Now)

finalDate = date8.Value

        End If      End If

    If (date9.HasValue) Then
        If (Math.Abs(DateDiff(DateInterval.Second, date9.Value, Date.Now)) < nearest) Then
            nearest = DateDiff(DateInterval.Second, date9.Value, Date.Now)          finalDate =
date9.Value

        End If      End If

    If (date10.HasValue) Then
        If (Math.Abs(DateDiff(DateInterval.Second, date10.Value, Date.Now))
< nearest) Then          nearest = DateDiff(DateInterval.Second, date10.Value, Date.Now)

finalDate = date10.Value

        End If      End If

    Return finalDate End Function

```

```
Private Enum DateCompare
```

```
    LessThanEqualTo    GreaterThanEqualTo
```

```
End Enum
```

```
Private Function GetNearestDate(ByVal source As IEnumerable(Of DateTime), _
```

```
    ByVal target As DateTime, _
```

```
    ByVal dt As DateCompare) As Nullable(Of DateTime)
```

```
Dim result As Nullable(Of DateTime) = Nothing
```

```
Dim lowestDifference As TimeSpan = TimeSpan.MaxValue
```

```
Dim difference As TimeSpan
```

```
For Each _date As DateTime In source
```

```
    If dt = DateCompare.LessThanEqualTo And _date > target Then
```

```
        Continue For
```

```
    ElseIf dt = DateCompare.GreaterThanEqualTo And _date < target Then
```

```
Continue For        End If
```

```
        If target > _date Then        difference =
```

```
target - _date        Else        difference = _date -  
target
```

```
        End If
```

```
        If difference < lowestDifference Then
```

```
lowestDifference = difference        result = _date
```

```
End If
```

```
Next
```

```
Return result End Function
```

```
Private Sub PPMReportToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles PPMReportToolStripMenuItem.Click
```

```
Dim recordID As String = InputBox("Please enter Asset#", "Searching PPM Report", "")
```

```
If (recordID.Length > 0) Then
```

```
PPMReport.Show()
```

```
PPMReport.loadReport(recordID)
```

```
Else
```

```
MsgBox("Please enter proper ID", MsgBoxStyle.OkOnly, "Invalid Input")
```

```
End If
```

```
End Sub
```

```
Private Sub PrintAllPPMReportsToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles PrintAllPPMReportsToolStripMenuItem.Click
```

```
PPMReport.printAll()
```

```
End Sub
```

```
Private Sub WOReportsToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles WOReportsToolStripMenuItem.Click
```

```
Dim recordID As String = InputBox("Please enter WO#", "Searching WO Report", "")
```

```
If (recordID.Length > 0) Then
```

```
WOReport.Show()
```

```
WOReport.loadReport(recordID)
```

```
Else
```

```
MsgBox("Please enter proper ID", MsgBoxStyle.OkOnly, "Invalid
```

Input")

End If

End Sub

Private Sub PriToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles PriToolStripMenuItem.Click

WOReport.printAll()

End Sub

Private Sub OpenToolStripMenuItem1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles OpenToolStripMenuItem1.Click

WOReport.loadReportsByStatus("Open")

End Sub

Private Sub InProgressToolStripMenuItem1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles InProgressToolStripMenuItem1.Click

WOReport.loadReportsByStatus("In Progress")

End Sub

Private Sub CompletedToolStripMenuItem1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles CompletedToolStripMenuItem1.Click

WOReport.loadReportsByStatus("Completed")

End Sub

Private Sub ClosedToolStripMenuItem1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles ClosedToolStripMenuItem1.Click

WOReport.loadReportsByStatus("Closed")

End Sub

Private Sub showNotification()

```
Dim command As New OleDb.OleDbCommand("SELECT EquipmentT.[Asset#],  
EquipmentT.EquipmentName, EquipmentT.[1stPPM], EquipmentT.[2ndPPM],  
EquipmentT.[3rdPPM], EquipmentT.[4thPPM], EquipmentT.[5thPPM],  
EquipmentT.[6thPPM], EquipmentT.[7thPPM], EquipmentT.[8thPPM],  
EquipmentT.[9thPPM], EquipmentT.[10thPPM] FROM EquipmentT where (  
(EquipmentT.[1stPPM] > now() AND EquipmentT.[1stPPM] < now() + 2) OR  
(EquipmentT.[2ndPPM] > now() AND EquipmentT.[2ndPPM] < now() + 2) OR  
(EquipmentT.[3rdPPM] > now() AND EquipmentT.[3rdPPM] < now() + 2) OR  
(EquipmentT.[4thPPM] > now() AND EquipmentT.[4thPPM] < now() + 2) OR  
(EquipmentT.[5thPPM] > now() AND EquipmentT.[5thPPM] < now() + 2) OR  
(EquipmentT.[6thPPM] > now() AND EquipmentT.[6thPPM] < now() + 2) OR  
(EquipmentT.[7thPPM] > now() AND EquipmentT.[7thPPM] < now() + 2) OR  
(EquipmentT.[8thPPM] > now() AND EquipmentT.[8thPPM] < now() + 2) OR  
(EquipmentT.[9thPPM] > now() AND EquipmentT.[9thPPM] < now() + 2) OR  
(EquipmentT.[10thPPM] > now() AND EquipmentT.[10thPPM] < now() + 2))")
```

```
Console.WriteLine(command.CommandText)      connection.Open()
```

```
command.Connection = connection
```

```
Dim dataReader As OleDb.OleDbDataReader = command.ExecuteReader
```

```
While (dataReader.Read())
```

```
    Console.WriteLine(dataReader.Item(2))
```

```
Dim d1 As Nullable(Of Date) = checkDate(dataReader.Item(2))
```

```
Dim d2 As Nullable(Of Date) = checkDate(dataReader.Item(3))
```

```
Dim d3 As Nullable(Of Date) = checkDate(dataReader.Item(4))
```

```
Dim d4 As Nullable(Of Date) = checkDate(dataReader.Item(5))
```

```
Dim d5 As Nullable(Of Date) = checkDate(dataReader.Item(6))
```

```
Dim d6 As Nullable(Of Date) = checkDate(dataReader.Item(7))
```

```
Dim d7 As Nullable(Of Date) = checkDate(dataReader.Item(8))
```

```
Dim d8 As Nullable(Of Date) = checkDate(dataReader.Item(9))
```

```

        Dim d9 As Nullable(Of Date) = checkDate(dataReader.Item(10))
        Dim d10 As
Nullable(Of Date) = checkDate(dataReader.Item(11))

```

```

        Notification.showNotification(dataReader.Item(0), dataReader.Item(1),
findClosestDate(d1, d2, d3, d4, d5, d6, d7, d8, d9, d10))
    End While
connection.Close()
End Sub

```

```

Private Sub Form_MainMenu_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
    Timer1.Start()
    Timer1.Interval = 3000

End Sub

```

```

Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Timer1.Tick
    Timer1.Enabled = False
    showNotification()

End Sub

```

```

Private Sub NewToolStripMenuItem1_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles NewToolStripMenuItem1.Click
    ManageLocation.addNew()

End Sub

```

```

Private Sub SearchUpdateToolStripMenuItem2_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
SearchUpdateToolStripMenuItem2.Click
    Dim recordID As String = InputBox("Please enter Location ID",
"Searching for Location", "")

    If (recordID.Length > 0) Then
        ManageLocation.searchUpdate(recordID)
    End If
End Sub

```

Else

MsgBox("Please enter proper ID", MsgBoxStyle.OkOnly, "Invalid
Input")

End If

End Sub

Private Sub NewToolStripMenuItem2_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles NewToolStripMenuItem2.Click

ManageMD.addNew()

End Sub

Private Sub SearchUpdateToolStripMenuItem3_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
SearchUpdateToolStripMenuItem3.Click

Private Sub NewToolStripMenuItem3_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles NewToolStripMenuItem3.Click

ManageSite.addNew()

End Sub

Private Sub SearchUpdateToolStripMenuItem4_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
SearchUpdateToolStripMenuItem4.Click

Dim recordID As String = InputBox("Please enter Site ID", "Searching for Site", "")
If (recordID.Length > 0) Then

ManageSite.searchUpdate(recordID)

Else

MsgBox("Please enter proper ID", MsgBoxStyle.OkOnly, "Invalid
Input")

End If

End Sub

```
Private Sub NewToolStripMenuItem4_Click(ByVal sender As System.Object,  
ByVal e As System.EventArgs) Handles NewToolStripMenuItem4.Click
```

```
    ManageSupplier.addNew()
```

End Sub

```
Private Sub SearchUpdateToolStripMenuItem5_Click(ByVal sender As  
System.Object, ByVal e As System.EventArgs) Handles  
SearchUpdateToolStripMenuItem5.Click
```

```
    Dim recordID As String = InputBox("Please enter Customer ID",  
"Searching for Customer", "")
```

```
    If (recordID.Length > 0) Then
```

```
        ManageSupplier.searchUpdate(recordID)
```

```
    Else
```

```
        MsgBox("Please enter proper ID", MsgBoxStyle.OkOnly, "Invalid  
Input")
```

```
    End If
```

End Sub

```
Private Sub AboutMenuItem_Click(ByVal sender As System.Object, ByVal e As  
System.EventArgs) Handles AboutMenuItem.Click
```

```
    About.ShowDialog()
```

End Sub

```
Private Sub SearchUpdateToolStripMenuItem6_Click(ByVal sender As  
System.Object, ByVal e As System.EventArgs) Handles  
SearchUpdateToolStripMenuItem6.Click
```

```
    Dim recordID As String = InputBox("Please enter Equipment ID",  
"Searching for Equipment", "")
```

```
    If (recordID.Length > 0) Then
```

```
PPM.searchPPM(recordID)
```

```
PPM.Show()
```

```
Else
```

```
MsgBox("Please enter proper ID", MsgBoxStyle.OkOnly, "Invalid  
Input")
```

```
End If
```

```
End Sub
```

```
Private Sub NewToolStripMenuItem5_Click(ByVal sender As System.Object,  
ByVal e As System.EventArgs) Handles NewToolStripMenuItem5.Click
```

```
PPM.setMode("new")
```

```
PPM.ShowDialog()
```

```
End Sub
```

```
End Class
```

Inventory

```
Public Class MainInventory
```

```
Dim connection As New  
OleDb.OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" &  
Application.StartupPath & "\FYP.accdb;Persist Security Info=False;")
```

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As  
System.EventArgs) Handles Button1.Click filterInventory(ComboBox1.Text)
```

```
End Sub
```

```
Public Sub filterInventory(ByVal inventoryType As String)
```

```
Label1.Text = "Inventory for " & inventoryType
```

```
Dim sql As String = "" Dim recordSet
```

```
As String = ""
```

If (inventoryType.Equals("Equipment")) Then

```
sql = "SELECT EquipmentT.[Asset#], EquipmentT.EquipmentName,  
EquipmentT.Manufacturer, EquipmentT.Model, EquipmentT.[Serial#],  
EquipmentT.EquipmentType, EquipmentT.CustomerID, EquipmentT.Price,  
EquipmentT.[Site#], EquipmentT.[MD#], EquipmentT.LocationID,  
EquipmentT.StartUp, EquipmentT.UnderWarranty, EquipmentT.EndOfWarranty,  
EquipmentT.Retired, EquipmentT.[PPM's Yearly], EquipmentT.PPMschedule,  
EquipmentT.[PPM's Month], EquipmentT.[1st PPM], EquipmentT.[1st PPM'Status],  
EquipmentT.[2nd PPM], EquipmentT.[2nd PPM'Status], EquipmentT.[3rd PPM],  
EquipmentT.[3rd PPM'Status], EquipmentT.[4th PPM], EquipmentT.[4th PPM'Status],  
EquipmentT.[5th PPM], EquipmentT.[5th PPM'Status], EquipmentT.[6th PPM],  
EquipmentT.[6th PPM'Status], EquipmentT.[7th PPM], EquipmentT.[7th PPM'Status],  
EquipmentT.[8th PPM], EquipmentT.[8th PPM'Status], EquipmentT.[9th PPM],  
EquipmentT.[9th PPM'Status], EquipmentT.[10th PPM], EquipmentT.[10th  
PPM'Status], EquipmentT.[PO#], EquipmentT.AcceptanceForm,  
EquipmentT.EquipmentPic, EquipmentT.Remark, EquipmentT.supplier FROM  
EquipmentT"      recordSet =
```

"EQUIPMENTT"

ElseIf (inventoryType.Equals("Customer")) Then

```
sql = "SELECT CustomerT.CustomerID, CustomerT.CustomerName,  
CustomerT.Address, CustomerT.City, CustomerT.CustomerPhone,  
CustomerT.[CustomerE-mail] FROM CustomerT"
```

recordSet = "CustomerT"

ElseIf (inventoryType.Equals("Site")) Then sql = "SELECT

```
SiteT.[Site#], SiteT.SiteName FROM SiteT"      recordSet = "SiteT"
```

ElseIf (inventoryType.Equals("Location")) Then

```
sql = "SELECT LocationT.LocationID, LocationT.LocationName FROM  
LocationT"      recordSet = "LOCATIONT"      End If
```

```

    Dim dataadapter As New OleDb.OleDbDataAdapter(sql, connection)    Dim
ds As New DataSet()    connection.Open()    dataadapter.Fill(ds, recordSet)
connection.Close()    DataGridView1.DataSource = ds
    DataGridView1.DataMember = recordSet

End Sub

```

```

Private Sub MainInventory_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

```

```

End Sub
End Class

```

Location

```

Public Class ManageLocation

```

```

    Dim connection As New
OleDb.OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" &
Application.StartupPath & "\FYP.accdb;Persist Security Info=False;")
    Dim ID As String = ""

```

```

Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click

```

```

    Me.Close()

End Sub

```

```

Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button2.Click

```

```

    If (checkInputs()) Then

```

```

    If (ID.Length = 0) Then

        If (executeSQL("INSERT INTO LOCATIONT(LOCATIONID, LOCATIONNAME)
VALUES('" & TextBox1.Text & "', '" & TextBox2.Text & "')") > 0) Then

            MsgBox("New location has been added", MsgBoxStyle.OkOnly,
"Location Added")

            Me.Close()

        Else

            MsgBox("Unexpected error occured", MsgBoxStyle.OkOnly,
"Location Add Failed")

        End If

    Else

        If (executeSQL("UPDATE LOCATIONT SET LOCATIONID = '" &
TextBox1.Text & "', LOCATIONNAME = '" & TextBox2.Text & "' WHERE LOCATIONID =
'" & ID & "'") > 0) Then

            MsgBox("Location has been updated", MsgBoxStyle.OkOnly,
"Location Updated")

            Me.Close()

        Else

            MsgBox("Unexpected error occured", MsgBoxStyle.OkOnly,
"Location Update Failed")

        End If

    End If

End If

End Sub

Private Function checkInputs() As Boolean

    If (TextBox1.Text.Length = 0) Then

        MsgBox("Please enter location ID", MsgBoxStyle.OkOnly, "Missing
Input")

        Return False
    
```

End If

If (TextBox2.Text.Length = 0) Then

MsgBox("Please enter location Name", MsgBoxStyle.OkOnly, "Missing
Input")

Return False End If

Return True End Function

Private Function executeSQL(ByVal SQL As String) As Integer

Console.WriteLine(SQL) Dim result As

Boolean = False connection.Open()

Dim command As New OleDb.OleDbCommand(SQL)

command.Connection = connection

Dim affected As Integer

Try affected = command.ExecuteNonQuery()

Catch ex As Exception affected = -1

MsgBox(ex.Message, MsgBoxStyle.OkOnly, "Operation Failed")

End Try

connection.Close() Return

affected End Function

Public Sub addNew()

TextBox1.Text = ""

TextBox2.Text = ""

ID = ""

```

    Me.Show()

End Sub

Public Sub searchUpdate(ByVal recordID As String)

    Dim command As New OleDb.OleDbCommand("SELECT LocationT.[LocationID],
LocationT.[LocationName] from LocationT where LocationT.[LocationID] = " & recordID & """)

    Console.WriteLine(command.CommandText)

connection.Open()      command.Connection = connection

    Dim dataReader As OleDb.OleDbDataReader = command.ExecuteReader

    ID = ""

    If (dataReader.Read()) Then

        ID = recordID

        TextBox1.Text = dataReader.Item(0)      TextBox2.Text =

dataReader.Item(1)

        Me.Show()

    Else

        MsgBox("No record has been found", MsgBoxStyle.OkOnly, "Record not found")

    End If      connection.Close()  End Sub

Private Sub ManageLocation_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

    End Sub

End Class

```

Site

Public Class ManageSite

Dim connection As New

OleDb.OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" &
Application.StartupPath & "\FYP.accdb;Persist Security Info=False;")

Dim ID As String = ""

Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click

Me.Close()

End Sub

Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button2.Click

If (checkInputs()) Then

If (ID.Length = 0) Then

If (executeSQL("INSERT INTO SiteT([Site#], SiteName) VALUES(" &
& TextBox1.Text & ", " & TextBox2.Text & ")") > 0) Then

MsgBox("New Site has been added", MsgBoxStyle.OkOnly, "Site
Added")

Me.Close()

Else

MsgBox("Unexpected error occured", MsgBoxStyle.OkOnly,
"Site Add Failed")

End If

Else

If (executeSQL("UPDATE SiteT SET [Site#] = " & TextBox1.Text & ", SiteName = " &
& TextBox2.Text & " WHERE [Site#] = " & ID & "")) > 0) Then

MsgBox("Site has been updated", MsgBoxStyle.OkOnly, "Site
Updated")

Me.Close()

```

Else
    MsgBox("Unexpected error occured", MsgBoxStyle.OkOnly,
"Site Update Failed")
End If
End If
End If
End Sub

Private Function checkInputs() As Boolean
    If (TextBox1.Text.Length = 0) Then
        MsgBox("Please enter Site ID", MsgBoxStyle.OkOnly, "Missing Input")
        Return False    End If

    If (TextBox2.Text.Length = 0) Then
        MsgBox("Please enter Site Name", MsgBoxStyle.OkOnly, "Missing
Input")
        Return False    End If

    Return True    End Function

Private Function executeSQL(ByVal SQL As String) As Integer
    Console.WriteLine(SQL)    Dim result As
Boolean = False    connection.Open()

    Dim command As New OleDb.OleDbCommand(SQL)
command.Connection = connection

    Dim affected As Integer

```

```

        Try            affected = command.ExecuteNonQuery()

Catch ex As Exception    affected = -1

        MsgBox(ex.Message, MsgBoxStyle.OkOnly, "Operation Failed")

End Try

connection.Close()    Return

affected    End Function

Public Sub addNew()

    TextBox1.Text = ""

    TextBox2.Text = ""

    ID = ""

    Me.Show()

End Sub

Public Sub searchUpdate(ByVal recordID As String)

    Dim command As New OleDb.OleDbCommand("SELECT SiteT.[Site#],
SiteT.[SiteName] from SiteT where SiteT.[Site#] = " & recordID & """)

    Console.WriteLine(command.CommandText)    connection.Open()

    command.Connection = connection

    Dim dataReader As OleDb.OleDbDataReader = command.ExecuteReader

    ID = ""

    If (dataReader.Read()) Then

        ID = recordID

```

```

        TextBox1.Text = dataReader.Item(0)        TextBox2.Text =
dataReader.Item(1)

        Me.Show()
    Else
        MsgBox("No record has been found", MsgBoxStyle.OkOnly, "Record not found")
    End If

    connection.Close()    End Sub

    Private Sub ManageSite_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

        End Sub

    End Class

Customer

    Public Class ManageSupplier

        Dim connection As New
OleDb.OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" &
Application.StartupPath & "\FYP.accdb;Persist Security Info=False;")

        Dim ID As String = ""

        Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click

            Me.Close()

        End Sub

        Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button2.Click

            If (checkInputs()) Then

                If (ID.Length = 0) Then

                    If (executeSQL("INSERT INTO CustomerT([CustomerID],
CustomerName, Address, City, CustomerPhone, [CustomerE-mail]) VALUES(" &

```

```
TextBox1.Text & ", " & TextBox2.Text & ", " & TextBox3.Text & ", " & TextBox4.Text & ",  
" & TextBox5.Text & ", " & TextBox6.Text & ")") > 0)
```

```
Then
```

```
    MsgBox("New Customer has been added", MsgBoxStyle.OkOnly,  
"Customer Added")
```

```
    Me.Close()
```

```
Else
```

```
    MsgBox("Unexpected error occured", MsgBoxStyle.OkOnly,  
"Customer Add Failed")
```

```
End If
```

```
Else
```

```
    If (executeSQL("UPDATE CustomerT SET [CustomerID] = " &  
TextBox1.Text & ", CustomerName = " & TextBox2.Text & ", Address = " &  
TextBox3.Text & ", City = " & TextBox4.Text & ", CustomerPhone = " & TextBox5.Text & ",  
[CustomerE-mail] = " & TextBox6.Text & " WHERE  
[CustomerID] = " & ID & """) > 0) Then
```

```
        MsgBox("Customer has been updated", MsgBoxStyle.OkOnly,  
"Customer Updated")
```

```
        Me.Close()
```

```
Else
```

```
    MsgBox("Unexpected error occured", MsgBoxStyle.OkOnly,  
"Customer Update Failed")
```

```
End If
```

```
End If
```

```
End If
```

```
End Sub
```

```
Private Function checkInputs() As Boolean
```

```
    If (TextBox1.Text.Length = 0) Then
```

```
        MsgBox("Please enter Customer ID", MsgBoxStyle.OkOnly, "Missing  
Input")
```

```
        Return False    End If
```

```

If (TextBox2.Text.Length = 0) Then
    MsgBox("Please enter Customer Name", MsgBoxStyle.OkOnly, "Missing
Input")
    Return False    End If

If (TextBox6.Text.Length = 0) Then
    MsgBox("Please enter Phone Number", MsgBoxStyle.OkOnly, "Missing
Input")
    Return False    End If

If (TextBox5.Text.Length = 0) Then
    MsgBox("Please enter Email Address", MsgBoxStyle.OkOnly, "Missing Input")
    Return False

End If

Return True    End Function

Private Function executeSQL(ByVal SQL As String) As Integer
    Console.WriteLine(SQL)    Dim result As
Boolean = False    connection.Open()

    Dim command As New OleDb.OleDbCommand(SQL)
    command.Connection = connection

    Dim affected As Integer
    Try    affected = command.ExecuteNonQuery()
Catch ex As Exception    affected = -1

    MsgBox(ex.Message, MsgBoxStyle.OkOnly, "Operation Failed")

```

End Try

connection.Close() Return

affected End Function

Public Sub addNew()

 TextBox1.Text = ""

 TextBox2.Text = ""

 ID = ""

 Me.Show()

End Sub

Public Sub searchUpdate(ByVal recordID As String)

 Dim command As New OleDb.OleDbCommand("SELECT CustomerT.[CustomerID],
CustomerT.[CustomerName], CustomerT.Address, CustomerT.City,
CustomerT.CustomerPhone, CustomerT.[CustomerE-mail] from CustomerT where
CustomerT.[CustomerID] = " & recordID & """)

 Console.WriteLine(command.CommandText)

 connection.Open() command.Connection = connection

 Dim dataReader As OleDb.OleDbDataReader = command.ExecuteReader

 ID = ""

 If (dataReader.Read()) Then

 ID = recordID

 TextBox1.Text = dataReader.Item(0)

 TextBox2.Text = dataReader.Item(1)

 TextBox3.Text = dataReader.Item(2)

```

        TextBox4.Text = dataReader.Item(3)

        TextBox5.Text = dataReader.Item(4)        TextBox6.Text =
dataReader.Item(5)

        Me.Show()

    Else

        MsgBox("No record has been found", MsgBoxStyle.OkOnly, "Record not found")

    End If        connection.Close()    End Sub

Private Sub ManageSupplier_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

End Sub

End Class

```

Equipment

```
Public Class PPM
```

```

    Dim connection As New
OleDb.OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" &
Application.StartupPath & "\FYP.accdb;Persist Security Info=False;")

    Dim assetID As String

```

```

Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click    If

(checkInputs() = False) Then

        Return    End If

```

```

If (Button1.Text.StartsWith("Save")) Then

```

```

    Dim SQL As String = "INSERT INTO EquipmentT ( EquipmentName,

```

```

Manufacturer, Model, [Serial#], EquipmentType, CustomerID, Price, [Site#],
[MD#], LocationID, StartUp, UnderWarranty, EndOfWarranty, Retired,
[PPM'sYearly], PPMschedule, [1stPPM], [1st PPM'Status], [2ndPPM], [2nd
PPM'Status], [3rdPPM], [3rd PPM'Status], [4thPPM], [4th PPM'Status], [5thPPM],
[5th PPM'Status], [6thPPM], [6th PPM'Status], [7thPPM], [7th PPM'Status],
[8thPPM], [8th PPM'Status], [9thPPM], [9th PPM'Status], [10thPPM], [10th
PPM'Status], [PO#], AcceptanceForm, EquipmentPic, Remark, supplier ) VALUES (" &
textName.Text & ", " & TextManufacturer.Text & ", " & textModel.Text &
", " & textSerial.Text & ", " & textType.Text & ", " & comboSupplier.Text
& ", " & textPrice.Text & ", " & textSite.Text & ", " & TextMD.Text & ",
" & textLocation.Text & ", " & dtStartup.Text & ", " &
checkWarranty.Checked & ", " & dtWarrantyEnds.Text & ", " & dtRetired.Text &
", " & textYearly.Text & ", " & textScheduleType.Text & ", " &
If(dtPPM.Enabled, dtPPM.Text, "NULL") & ", " & If(dtPPM.Enabled,
comboStatus1.Text, "NULL") & ", " & If(dtPPM2.Enabled, dtPPM2.Text, "NULL") &
", " & If(dtPPM2.Enabled, comboStatus2.Text, "NULL") & ", " &
If(dtPPM3.Enabled, dtPPM3.Text, "NULL") & ", " & If(dtPPM3.Enabled, comboStatus3.Text,
"NULL") & ", " & If(dtPPM4.Enabled, dtPPM4.Text, "NULL") &
", " & If(dtPPM4.Enabled, comboStatus4.Text, "NULL") & ", " &
If(dtPPM5.Enabled, dtPPM5.Text, "NULL") & ", " & If(dtPPM5.Enabled, comboStatus5.Text,
"NULL") & ", " & If(dtPPM6.Enabled, dtPPM6.Text, "NULL") &
", " & If(dtPPM6.Enabled, comboStatus6.Text, "NULL") & ", " &
If(dtPPM7.Enabled, dtPPM7.Text, "NULL") & ", " & If(dtPPM7.Enabled, comboStatus7.Text,
"NULL") & ", " & If(dtPPM8.Enabled, dtPPM8.Text, "NULL") &
", " & If(dtPPM8.Enabled, comboStatus8.Text, "NULL") & ", " &
If(dtPPM9.Enabled, dtPPM9.Text, "NULL") & ", " & If(dtPPM9.Enabled, comboStatus9.Text,
"NULL") & ", " & If(dtPPM10.Enabled, dtPPM10.Text, "NULL") & ", " &
If(dtPPM10.Enabled, comboStatus10.Text, "NULL") & ", " & textPO.Text & ", NULL, NULL,
" & textRemark.Text & ", " & ")

```

```

SQL = SQL.Replace("NULL", "NULL")

```

```

Console.WriteLine(SQL)

```

```

If (executeSQL(SQL) > 0) Then

```

```

    MsgBox("New equipment has been added successfully", MsgBoxStyle.OkOnly,
"Equipment Added")

```

```

    Me.Close()

```

```

Else

```

```

    MsgBox("Unexpected error has occurred. Please try again later",
MsgBoxStyle.OkOnly, "Add Failed")

```

```

End If

```

Else

'update

If (assetID = -1) Then

MsgBox("There is no PPM found. Update Failed",
MsgBoxStyle.OkOnly, "Update Failed")

Else

Dim SQL As String = "UPDATE EquipmentT SET
EquipmentT.EquipmentName = " & textName.Text & ", EquipmentT.Manufacturer =
" & TextManufacturer.Text & ", EquipmentT.Model = " & textModel.Text & ",
EquipmentT.[Serial#] = " & textSerial.Text & ", EquipmentT.EquipmentType = "
& textType.Text & ", EquipmentT.CustomerID = " & comboSupplier.Text & ",
EquipmentT.Price = " & textPrice.Text & ", EquipmentT.[Site#] = " & textSite.Text & ",
EquipmentT.[MD#] = " & TextMD.Text & ",
EquipmentT.LocationID = " & textLocation.Text & ", EquipmentT.StartUp = " & dtStartup.Text
& ", EquipmentT.UnderWarranty = " & checkWarranty.Checked & ",
EquipmentT.EndOfWarranty = " & dtWarrantyEnds.Text & ", EquipmentT.Retired = " &
dtRetired.Text & ", EquipmentT.[PPM's Yearly] = " & textYearly.Text & ",
EquipmentT.PPMschedule = " & textScheduleType.Text & ", EquipmentT.[PO#] = "
& textPO.Text & ", EquipmentT.Remark = " & textRemark.Text & ",
EquipmentT.supplier = ""

If (dtPPM.Enabled) Then

SQL = SQL + ", EquipmentT.[1stPPM] = " & dtPPM.Text & ",
EquipmentT.[1st PPM'Status] = " & comboStatus1.Text & ""

Else

SQL = SQL + ", EquipmentT.[1stPPM] = NULL, EquipmentT.[1st
PPM'Status] = NULL"

End If

If (dtPPM2.Enabled) Then

SQL = SQL + ", EquipmentT.[2ndPPM] = " & dtPPM2.Text & ",
EquipmentT.[2nd PPM'Status] = " & comboStatus2.Text & ""

Else

SQL = SQL + ", EquipmentT.[2ndPPM] = NULL, EquipmentT.[2nd

PPM'Status] = NULL"

End If

If (dtPPM3.Enabled) Then

SQL = SQL + ", EquipmentT.[3rdPPM] = '" + dtPPM3.Text + "',
EquipmentT.[3rd PPM'Status] = '" + comboStatus3.Text + "'"

Else

SQL = SQL + ", EquipmentT.[3rdPPM] = NULL, EquipmentT.[3rd
PPM'Status] = NULL"

End If

If (dtPPM4.Enabled) Then

SQL = SQL + ", EquipmentT.[4thPPM] = '" + dtPPM4.Text + "',
EquipmentT.[4th PPM'Status] = '" + comboStatus4.Text + "'"

Else

SQL = SQL + ", EquipmentT.[4thPPM] = NULL, EquipmentT.[4th
PPM'Status] = NULL"

End If

If (dtPPM5.Enabled) Then

SQL = SQL + ", EquipmentT.[5thPPM] = '" + dtPPM5.Text + "',
EquipmentT.[5th PPM'Status] = '" + comboStatus5.Text + "'"

Else

SQL = SQL + ", EquipmentT.[5thPPM] = NULL, EquipmentT.[5th
PPM'Status] = NULL"

End If

If (dtPPM6.Enabled) Then

SQL = SQL + ", EquipmentT.[6thPPM] = '" + dtPPM6.Text + "',

EquipmentT.[6th PPM'Status] = "" + comboStatus6.Text + ""

Else

SQL = SQL + ", EquipmentT.[6thPPM] = NULL, EquipmentT.[6th PPM'Status] = NULL"

End If If

(dtPPM7.Enabled) Then

SQL = SQL + ", EquipmentT.[7thPPM] = "" + dtPPM7.Text + "",
EquipmentT.[7th PPM'Status] = "" + comboStatus7.Text + ""

Else

SQL = SQL + ", EquipmentT.[7thPPM] = NULL, EquipmentT.[7th PPM'Status] = NULL"

End If

If (dtPPM8.Enabled) Then

SQL = SQL + ", EquipmentT.[8thPPM] = "" + dtPPM8.Text + "",
EquipmentT.[8th PPM'Status] = "" + comboStatus8.Text + ""

Else

SQL = SQL + ", EquipmentT.[8thPPM] = NULL, EquipmentT.[8th PPM'Status] = NULL"

End If

If (dtPPM9.Enabled) Then

SQL = SQL + ", EquipmentT.[9thPPM] = "" + dtPPM9.Text + "",
EquipmentT.[9th PPM'Status] = "" + comboStatus9.Text + ""

Else

SQL = SQL + ", EquipmentT.[9thPPM] = NULL, EquipmentT.[9th PPM'Status] = NULL"

End If

```

        If (dtPPM10.Enabled) Then

            SQL = SQL + ", EquipmentT.[10thPPM] = '" + dtPPM10.Text +
            "', EquipmentT.[10th PPM'Status] = '" + comboStatus10.Text + "'"

        Else

            SQL = SQL + ", EquipmentT.[10thPPM] = NULL,
            EquipmentT.[10th PPM'Status] = NULL"

        End If

        SQL = SQL + " WHERE EquipmentT.[Asset#] = " & assetID

        Console.WriteLine(SQL)

        If (executeSQL(SQL) > 0) Then

            MsgBox("Record has been updated successfully",
            MsgBoxStyle.OkOnly, "Record Updated")

            Me.Close()

        Else

            MsgBox("Unexpected error has occurred. Please try again later",
            MsgBoxStyle.OkOnly, "Update Failed")

        End If

    End If

End If

End Sub

Private Function checkInputs() As Boolean

    If (textName.Text.Length = 0) Then

        MsgBox("Please enter equipment name", MsgBoxStyle.OkOnly, "Missing
        Input")
    
```

Return False End If

If (TextManufacturer.Text.Length = 0) Then

 MsgBox("Please enter manufacturer name", MsgBoxStyle.OkOnly, "Missing Input")

Return False End If

If (textModel.Text.Length = 0) Then

 MsgBox("Please enter equipment model", MsgBoxStyle.OkOnly, "Missing Input")

Return False End If

If (textSerial.Text.Length = 0) Then

 MsgBox("Please enter equipment serial no", MsgBoxStyle.OkOnly, "Missing Input")

Return False

End If

If (textType.Text.Length = 0) Then

 MsgBox("Please enter equipment type", MsgBoxStyle.OkOnly, "Missing Input")

Return False End If

If (comboSupplier.Text.Length = 0) Then

 MsgBox("Please select Customer ID", MsgBoxStyle.OkOnly, "Missing Input")

Return False End If

If (textPrice.Text.Length = 0) Then

```
MsgBox("Please enter equipment price", MsgBoxStyle.OkOnly, "Missing  
Input")
```

```
Return False      End If
```

```
If (textSite.Text.Length = 0) Then
```

```
MsgBox("Please enter equipment site", MsgBoxStyle.OkOnly, "Missing  
Input")
```

```
Return False      End If
```

```
If (TextMD.Text.Length = 0) Then
```

```
MsgBox("Please enter equipment MD", MsgBoxStyle.OkOnly, "Missing  
Input")
```

```
Return False      End If
```

```
If (textLocation.Text.Length = 0) Then
```

```
MsgBox("Please enter equipment location", MsgBoxStyle.OkOnly, "Missing Input")  
Return False      End If
```

```
If (dtStartup.Text.Length = 0) Then
```

```
MsgBox("Please enter warranty start date", MsgBoxStyle.OkOnly,  
"Missing Input")
```

```
Return False      End If
```

```
If (dtWarrantyEnds.Text.Length = 0) Then
```

```
MsgBox("Please enter warranty end date", MsgBoxStyle.OkOnly,  
"Missing Input")
```

```
Return False      End If
```

```
If (dtRetired.Text.Length = 0) Then
```

```
MsgBox("Please enter equipment retired date", MsgBoxStyle.OkOnly,  
"Missing Input")
```

```
Return False      End If
```

```
If (textYearly.Text.Length = 0) Then
```

```
MsgBox("Please enter yearly PPM", MsgBoxStyle.OkOnly, "Missing  
Input")
```

```
Return False      End If
```

```
If (textScheduleType.Text.Length = 0) Then
```

```
MsgBox("Please enter schedule type", MsgBoxStyle.OkOnly, "Missing  
Input")
```

```
Return False      End If
```

```
If (textMonth.Text.Length = 0) Then
```

```
MsgBox("Please enter PPM Month", MsgBoxStyle.OkOnly, "Missing  
Input")
```

```
Return False      End If
```

```
If (textPO.Text.Length = 0) Then
```

```
MsgBox("Please enter equipment PO", MsgBoxStyle.OkOnly, "Missing  
Input")
```

```
Return False      End If
```

```
Return True      End Function
```

```
Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)  
Handles Button2.Click
```

```
Me.Close()
```

End Sub

Private Sub getSuppliers()

Dim command As New OleDb.OleDbCommand("SELECT SupplierT.[CustomerID]
from CustomerT") connection.Open() command.Connection = connection

Dim dataReader As OleDb.OleDbDataReader = command.ExecuteReader

comboCustomer.Items.Clear() While (dataReader.Read())

comboSupplier.Items.Add(dataReader.Item(0)) End While

connection.Close() End Sub

Public Sub searchPPM(ByVal ID As String) assetID = -1

Dim command As New OleDb.OleDbCommand("SELECT EquipmentT.EquipmentName,
EquipmentT.Manufacturer, EquipmentT.Model, EquipmentT.[Serial#],
EquipmentT.EquipmentType, EquipmentT.CustomerID, EquipmentT.Price,
EquipmentT.[Site#], EquipmentT.[MD#], EquipmentT.LocationID,
EquipmentT.StartUp, EquipmentT.UnderWarranty, EquipmentT.EndOfWarranty,
EquipmentT.Retired, EquipmentT.[PPM's Yearly], EquipmentT.PPMschedule,
EquipmentT.[PPM's Month], EquipmentT.[1st PPM], EquipmentT.[1st PPM'Status],
EquipmentT.[2nd PPM], EquipmentT.[2nd PPM'Status], EquipmentT.[3rd PPM],
EquipmentT.[3rd PPM'Status], EquipmentT.[4th PPM], EquipmentT.[4th PPM'Status],
EquipmentT.[5th PPM], EquipmentT.[5th PPM'Status], EquipmentT.[6th PPM],
EquipmentT.[6th PPM'Status], EquipmentT.[7th PPM], EquipmentT.[7th PPM'Status],
EquipmentT.[8th PPM], EquipmentT.[8th PPM'Status], EquipmentT.[9th PPM],
EquipmentT.[9th PPM'Status], EquipmentT.[10th PPM], EquipmentT.[10th
PPM'Status], EquipmentT.[PO#], EquipmentT.Remark, EquipmentT.supplier FROM
EquipmentT WHERE EquipmentT.[Asset#] = " & ID)

connection.Open() command.Connection = connection

Dim dataReader As OleDb.OleDbDataReader = command.ExecuteReader

dtPPM.Enabled = False

dtPPM2.Enabled = False dtPPM3.Enabled

```

= False      dtPPM4.Enabled = False

dtPPM5.Enabled = False      dtPPM6.Enabled

= False      dtPPM7.Enabled = False

dtPPM8.Enabled = False      dtPPM9.Enabled

= False      dtPPM10.Enabled = False

comboStatus1.Enabled = False

comboStatus2.Enabled = False

comboStatus3.Enabled = False

comboStatus4.Enabled = False

comboStatus5.Enabled = False

comboStatus6.Enabled = False

comboStatus7.Enabled = False

comboStatus8.Enabled = False

comboStatus9.Enabled = False

comboStatus10.Enabled = False      If

(dataReader.Read()) Then      assetID = ID

'EquipmentT.[Asset#],      textAssetID.Text

= ID

'EquipmentT.EquipmentName,

      If (Not IsDBNull(dataReader.Item(0))) Then

          textName.Text = dataReader.Item(0)

      End If

      If (Not IsDBNull(dataReader.Item(1))) Then

```

'EquipmentT.Manufacturer,

TextManufacturer.Text = dataReader.Item(1)

End If

If (Not IsDBNull(dataReader.Item(2))) Then

'EquipmentT.Model, textModel.Text =

dataReader.Item(2)

End If

If (Not IsDBNull(dataReader.Item(3))) Then

'EquipmentT.[Serial#], textSerial.Text =

dataReader.Item(3)

End If

If (Not IsDBNull(dataReader.Item(4))) Then

'EquipmentT.EquipmentType,

textType.Text = dataReader.Item(4)

End If

If (Not IsDBNull(dataReader.Item(5))) Then

'EquipmentT.CustomerID, comboSupplier.Text =

dataReader.Item(5)

End If

'EquipmentT.Price,

```
        If (Not IsDBNull(dataReader.Item(6))) Then            textPrice.Text =  
dataReader.Item(6)  
        End If
```

```
        If (Not IsDBNull(dataReader.Item(7))) Then  
            'EquipmentT.[Site#],            textSite.Text =  
dataReader.Item(7).ToString  
        End If
```

```
        If (Not IsDBNull(dataReader.Item(8))) Then  
            'EquipmentT.[MD#],  
            TextMD.Text = dataReader.Item(8)  
        End If
```

```
        If (Not IsDBNull(dataReader.Item(9))) Then  
            'EquipmentT.LocationID,            textLocation.Text =  
dataReader.Item(9)  
        End If
```

```
        If (Not IsDBNull(dataReader.Item(10))) Then  
            'EquipmentT.Startup,            dtStartup.Text =  
dataReader.Item(10)  
        End If
```

```

        If (Not IsDBNull(dataReader.Item(11))) Then
'EquipmentT.UnderWarranty,          checkWarranty.Checked =
dataReader.Item(11)

        End If

        If (Not IsDBNull(dataReader.Item(12))) Then
'EquipmentT.EndOfWarranty,          dtWarrantyEnds.Text =
dataReader.Item(12)

        End If

'EquipmentT.Retired,
        If (Not IsDBNull(dataReader.Item(13))) Then          dtRetired.Text =
dataReader.Item(13)

        End If

        If (Not IsDBNull(dataReader.Item(14))) Then
'EquipmentT.[PPM's Yearly],          textYearly.Text =
dataReader.Item(14)

        End If

        If (Not IsDBNull(dataReader.Item(15))) Then
'EquipmentT.PPMschedule,          textScheduleType.Text =
dataReader.Item(15)

        End If

```

```

        If (Not IsDBNull(dataReader.Item(16))) Then
'EquipmentT.[PPM'sMonth],          textMonth.Text =
dataReader.Item(16)

        End If

        If (Not IsDBNull(dataReader.Item(17))) Then          'EquipmentT.[1stPPM],
dtPPM.Text = dataReader.Item(17)          CheckBox1.Checked = True
dtPPM.Enabled = True          End If

        If (Not IsDBNull(dataReader.Item(18))) Then
'EquipmentT.[1st PPM'Status],          comboStatus1.Text =
dataReader.Item(18)          CheckBox1.Checked = True
comboStatus1.Enabled = True

        End If

        If (Not IsDBNull(dataReader.Item(19))) Then
          'EquipmentT.[2ndPPM],          dtPPM2.Text =
dataReader.Item(19)          CheckBox2.Checked = True
dtPPM2.Enabled = True          End If

        If (Not IsDBNull(dataReader.Item(20))) Then
'EquipmentT.[2nd PPM'Status],          comboStatus2.Text =
dataReader.Item(20)          comboStatus2.Enabled = True

          CheckBox2.Checked = True

        End If

```

```

If (Not IsDBNull(dataReader.Item(21))) Then
    'EquipmentT.[3rdPPM],          dtPPM3.Text =
dataReader.Item(21)          dtPPM3.Enabled = True
    CheckBox3.Checked = True
End If

If (Not IsDBNull(dataReader.Item(22))) Then
    'EquipmentT.[3rd PPM'Status],    comboStatus3.Text =
dataReader.Item(22)          comboStatus3.Enabled = True
    CheckBox3.Checked = True
End If

If (Not IsDBNull(dataReader.Item(23))) Then
    'EquipmentT.[4thPPM],          dtPPM4.Text =
dataReader.Item(23)          dtPPM4.Enabled = True
    CheckBox4.Checked = True
End If

If (Not IsDBNull(dataReader.Item(24))) Then
    'EquipmentT.[4th PPM'Status],    comboStatus4.Text =
dataReader.Item(24)          comboStatus4.Enabled = True
    CheckBox4.Checked = True
End If

If (Not IsDBNull(dataReader.Item(25))) Then

```

```

        'EquipmentT.[5thPPM],          dtPPM5.Text =
dataReader.Item(25)          dtPPM5.Enabled = True

        CheckBox5.Checked = True
End If

If (Not IsDBNull(dataReader.Item(26))) Then
    'EquipmentT.[5th PPM'Status],      comboStatus5.Text = dataReader.Item(26)
comboStatus5.Enabled = True

    CheckBox5.Checked = True
End If

If (Not IsDBNull(dataReader.Item(27))) Then
    'EquipmentT.[6thPPM],          dtPPM6.Text =
dataReader.Item(27)          dtPPM6.Enabled = True

    CheckBox6.Checked = True
End If

If (Not IsDBNull(dataReader.Item(28))) Then
    'EquipmentT.[6th PPM'Status],      comboStatus6.Text =
dataReader.Item(28)          comboStatus6.Enabled = True

    CheckBox6.Checked = True
End If

If (Not IsDBNull(dataReader.Item(29))) Then
    'EquipmentT.[7thPPM],          dtPPM7.Text =
dataReader.Item(29)          dtPPM7.Enabled = True

```

CheckBox7.Checked = True

End If

If (Not IsDBNull(dataReader.Item(30))) Then

'EquipmentT.[7th PPM>Status], comboStatus7.Text =

dataReader.Item(30) comboStatus7.Enabled = True

CheckBox7.Checked = True

End If

If (Not IsDBNull(dataReader.Item(31))) Then

'EquipmentT.[8thPPM], dtPPM8.Text =

dataReader.Item(31) dtPPM8.Enabled = True

CheckBox8.Checked = True

End If

If (Not IsDBNull(dataReader.Item(32))) Then

'EquipmentT.[8th PPM>Status], comboStatus8.Text =

dataReader.Item(32) comboStatus8.Enabled = True

CheckBox8.Checked = True

End If

If (Not IsDBNull(dataReader.Item(33))) Then

'EquipmentT.[9thPPM], dtPPM9.Text =

dataReader.Item(33) dtPPM9.Enabled = True

CheckBox9.Checked = True

End If

```
If (Not IsDBNull(dataReader.Item(34))) Then
'EquipmentT.[9th PPM>Status],          comboStatus9.Text =
dataReader.Item(34)          comboStatus9.Enabled = True
    CheckBox9.Checked = True
End If
```

```
If (Not IsDBNull(dataReader.Item(35))) Then
'EquipmentT.[10thPPM],          dtPPM10.Text =
dataReader.Item(35)          dtPPM10.Enabled = True
    CheckBox10.Checked = True
End If
```

```
If (Not IsDBNull(dataReader.Item(36))) Then
'EquipmentT.[10th PPM>Status],          comboStatus10.Text =
dataReader.Item(36)          comboStatus10.Enabled = True
    CheckBox10.Checked = True
End If
```

```
If (Not IsDBNull(dataReader.Item(37))) Then
'EquipmentT.[PO#],          textPO.Text =
dataReader.Item(37)
End If
```

```
If (Not IsDBNull(dataReader.Item(38))) Then
```

```

        'EquipmentT.Remark,          textRemark.Text =
dataReader.Item(38)

    End If

    If (Not IsDBNull(dataReader.Item(39))) Then
        'EquipmentT.supplier()      comboSupplier.Text =
dataReader.Item(39)

    End If    End If

connection.Close() End Sub

Public Sub setMode(ByVal mode As String)    getSuppliers()

    If (mode.Equals("new")) Then

        Button1.Text = "Save Record"      textAssetID.Text = "Auto
Generated"

        dtPPM.Enabled = False

dtPPM2.Enabled = False      dtPPM3.Enabled =
False      dtPPM4.Enabled = False

dtPPM5.Enabled = False      dtPPM6.Enabled =
False      dtPPM7.Enabled = False

dtPPM8.Enabled = False      dtPPM9.Enabled =
False      dtPPM10.Enabled = False

comboStatus1.Enabled = False

comboStatus2.Enabled = False

comboStatus3.Enabled = False

comboStatus4.Enabled = False

```

comboStatus5.Enabled = False

comboStatus6.Enabled = False

comboStatus7.Enabled = False

comboStatus8.Enabled = False

comboStatus9.Enabled = False

comboStatus10.Enabled = False

Else

Button1.Text = "Update Record"

End If

End Sub

Private Function executeSQL(ByVal SQL As String) As Integer Dim

result As Boolean = False connection.Open()

Dim command As New OleDb.OleDbCommand(SQL)

command.Connection = connection

Dim affected As Integer

Try affected = command.ExecuteNonQuery()

Catch ex As Exception affected = -1

MsgBox(ex.Message, MsgBoxStyle.OkOnly, "Operation Failed")

End Try

connection.Close() Return

affected End Function

Private Sub CheckBox1_CheckedChanged(ByVal sender As System.Object, ByVal e

As System.EventArgs) Handles CheckBox1.CheckedChanged

dtPPM.Enabled = CheckBox1.Checked comboStatus1.Enabled =

CheckBox1.Checked

End Sub

Private Sub CheckBox2_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles CheckBox2.CheckedChanged

dtPPM2.Enabled = CheckBox2.Checked comboStatus2.Enabled =

CheckBox2.Checked

End Sub

Private Sub CheckBox3_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles CheckBox3.CheckedChanged

dtPPM3.Enabled = CheckBox3.Checked comboStatus3.Enabled =

CheckBox3.Checked

End Sub

Private Sub CheckBox4_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles CheckBox4.CheckedChanged

dtPPM4.Enabled = CheckBox4.Checked comboStatus4.Enabled =

CheckBox4.Checked

End Sub

Private Sub CheckBox5_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles CheckBox5.CheckedChanged

dtPPM5.Enabled = CheckBox5.Checked comboStatus5.Enabled =

CheckBox5.Checked

End Sub

```
Private Sub CheckBox10_CheckedChanged(ByVal sender As System.Object, ByVal e As  
System.EventArgs) Handles CheckBox10.CheckedChanged dtPPM6.Enabled =  
CheckBox10.Checked comboStatus6.Enabled = CheckBox10.Checked
```

End Sub

```
Private Sub CheckBox9_CheckedChanged(ByVal sender As System.Object, ByVal e  
As System.EventArgs) Handles CheckBox9.CheckedChanged  
dtPPM7.Enabled = CheckBox9.Checked comboStatus7.Enabled =  
CheckBox9.Checked
```

End Sub

```
Private Sub CheckBox8_CheckedChanged(ByVal sender As System.Object, ByVal e  
As System.EventArgs) Handles CheckBox8.CheckedChanged  
dtPPM8.Enabled = CheckBox8.Checked comboStatus8.Enabled =  
CheckBox8.Checked
```

End Sub

```
Private Sub CheckBox7_CheckedChanged(ByVal sender As System.Object, ByVal e  
As System.EventArgs) Handles CheckBox7.CheckedChanged  
dtPPM9.Enabled = CheckBox7.Checked comboStatus9.Enabled =  
CheckBox7.Checked
```

End Sub

```
Private Sub CheckBox6_CheckedChanged(ByVal sender As System.Object, ByVal e
```

```

As System.EventArgs) Handles CheckBox6.CheckedChanged

dtPPM10.Enabled = CheckBox6.Checked      comboStatus10.Enabled =
CheckBox6.Checked

End Sub

```

```

Private Sub PPM_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

End Sub

End Class

```

PPM notification

```

Public Class Notification

Private Sub Label4_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Label4.Click

Me.Close()

End Sub

Public Sub showNotification(ByVal id As String, ByVal name As String, ByVal PPMDate As
String)      labelID.Text = id      labelName.Text = name      labelDate.Text = PPMDate
Me.ShowDialog()

End Sub

```

```

Private Sub Notification_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

```

End Sub

End Class

PPM report

Public Class PPMReport

Dim connection As New

OleDb.OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" &
Application.StartupPath & "\FYP.accdb;Persist Security Info=False;")

Public Sub loadReport(ByVal ID As String)

Dim command As New OleDb.OleDbCommand("SELECT EquipmentT.EquipmentName,
EquipmentT.Manufacturer, EquipmentT.CustomerID, EquipmentT.UnderWarranty,
EquipmentT.EndOfWarranty, EquipmentT.[PPM's Yearly], EquipmentT.PPMschedule,
EquipmentT.[PPM's Month], EquipmentT.[1st PPM], EquipmentT.[1st PPM'Status],
EquipmentT.[2nd PPM], EquipmentT.[2nd PPM'Status], EquipmentT.[3rd PPM],
EquipmentT.[3rd PPM'Status], EquipmentT.[4th PPM], EquipmentT.[4th PPM'Status],
EquipmentT.[5th PPM], EquipmentT.[5th PPM'Status], EquipmentT.[6th PPM],
EquipmentT.[6th PPM'Status], EquipmentT.[7th PPM], EquipmentT.[7th PPM'Status],
EquipmentT.[8th PPM], EquipmentT.[8th PPM'Status], EquipmentT.[9th PPM],
EquipmentT.[9th PPM'Status], EquipmentT.[10th PPM], EquipmentT.[10th PPM'Status]
FROM EquipmentT WHERE EquipmentT.[Asset#] = " & ID)

connection.Open() command.Connection = connection

Dim dataReader As OleDb.OleDbDataReader = command.ExecuteReader

If (dataReader.Read()) Then

'EquipmentT.[Asset#],

labelAssetID.Text = ID

'EquipmentT.EquipmentName,

```

        If (Not IsDBNull(dataReader.Item(0))) Then            labelEquipName.Text =
dataReader.Item(0)

        End If

        'EquipmentT.Manufacturer,
            If (Not IsDBNull(dataReader.Item(1))) Then
                labelSupName.Text = dataReader.Item(1)
            End If

        'EquipmentT.CustomerID,
            If (Not IsDBNull(dataReader.Item(2))) Then        labelSupID.Text =
dataReader.Item(2)

            End If

        'EquipmentT.UnderWarranty,
            If (Not IsDBNull(dataReader.Item(3))) Then        labelUnderWarranty.Text =
dataReader.Item(3)

            End If

        'EquipmentT.EndOfWarranty,
            If (Not IsDBNull(dataReader.Item(4))) Then        labelEndWarranty.Text =
dataReader.Item(4)

            End If

        'EquipmentT.[PPM's Yearly],

```

```

        If (Not IsDBNull(dataReader.Item(5))) Then            labelYearly.Text =
dataReader.Item(5)

        End If

        'EquipmentT.PPMschedule,
        If (Not IsDBNull(dataReader.Item(6))) Then            labelSchedule.Text =
dataReader.Item(6)

        End If

        'EquipmentT.[PPM'sMonth],
        If (Not IsDBNull(dataReader.Item(7))) Then            labelMonth.Text =
dataReader.Item(7).ToString

        End If

        If (Not IsDBNull(dataReader.Item(8))) Then
            'EquipmentT.[1stPPM],            label1stPPM.Text =
dataReader.Item(8)

        End If

        If (Not IsDBNull(dataReader.Item(9))) Then            'EquipmentT.[1st
PPM'Status],            labelStatus1.Text = dataReader.Item(9)

        End If

        If (Not IsDBNull(dataReader.Item(10))) Then
            'EquipmentT.[2ndPPM],            label2ndPPM.Text =
dataReader.Item(10)

```

End If

If (Not IsDBNull(dataReader.Item(11))) Then

'EquipmentT.[2nd PPM'Status], labelStatus2.Text =
dataReader.Item(11)

End If

If (Not IsDBNull(dataReader.Item(12))) Then

'EquipmentT.[3rdPPM], label3rdPPM.Text =
dataReader.Item(12)

End If

If (Not IsDBNull(dataReader.Item(13))) Then

'EquipmentT.[3rd PPM'Status], labelStatus3.Text =
dataReader.Item(13)

End If

If (Not IsDBNull(dataReader.Item(14))) Then

'EquipmentT.[4thPPM], label4thPPM.Text =
dataReader.Item(14)

End If

If (Not IsDBNull(dataReader.Item(15))) Then

'EquipmentT.[4th PPM'Status], labelStatus4.Text =
dataReader.Item(15)

End If

If (Not IsDBNull(dataReader.Item(16))) Then

'EquipmentT.[5thPPM], label5thPPM.Text =
dataReader.Item(16)

End If

If (Not IsDBNull(dataReader.Item(17))) Then

'EquipmentT.[5th PPM>Status], labelStatus5.Text =
dataReader.Item(17)

End If

If (Not IsDBNull(dataReader.Item(18))) Then

'EquipmentT.[6thPPM], label6thPPM.Text =
dataReader.Item(18)

End If

If (Not IsDBNull(dataReader.Item(19))) Then

'EquipmentT.[6th PPM>Status], labelStatus6.Text =
dataReader.Item(19)

End If

If (Not IsDBNull(dataReader.Item(20))) Then

'EquipmentT.[7thPPM], label7thPPM.Text =
dataReader.Item(20) End If

```

        If (Not IsDBNull(dataReader.Item(21))) Then
'EquipmentT.[7th PPM>Status],          labelStatus7.Text =
dataReader.Item(21)

        End If

        If (Not IsDBNull(dataReader.Item(22))) Then
'EquipmentT.[8thPPM],          label8thPPM.Text =
dataReader.Item(22)

        End If

        If (Not IsDBNull(dataReader.Item(23))) Then
'EquipmentT.[8th PPM>Status],          labelStatus8.Text =
dataReader.Item(23)

        End If

        If (Not IsDBNull(dataReader.Item(24))) Then
'EquipmentT.[9thPPM],          label9thPPM.Text =
dataReader.Item(24)

        End If

        If (Not IsDBNull(dataReader.Item(25))) Then
'EquipmentT.[9th PPM>Status],          labelStatus9.Text =
dataReader.Item(25)

        End If

```

```

        If (Not IsDBNull(dataReader.Item(26))) Then
            'EquipmentT.[10thPPM],          label10thPPM.Text =
dataReader.Item(26)

        End If

        If (Not IsDBNull(dataReader.Item(27))) Then
            'EquipmentT.[10th PPM>Status],    labelStatus10.Text =
dataReader.Item(27)

        End If

        Else

            MsgBox("Sorry, Asset ID could not be found", MsgBoxStyle.OkOnly,
"Invalid ID")

            Me.Close()      End If

connection.Close()  End Sub

Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click

    Button1.Visible = False

    PrintForm1.PrintAction = Printing.PrintAction.PrintToPreview

    PrintForm1.Print()      Button1.Visible = True

End Sub

Public Sub printAll()

    Dim command As New OleDb.OleDbCommand("SELECT EquipmentT.[Asset#] FROM
EquipmentT")      connection.Open()      command.Connection = connection

```

```

Dim dataReader As OleDb.OleDbDataReader = command.ExecuteReader

Dim listID As List(Of String) = New List(Of String)()

While (dataReader.Read())

listID.Add(dataReader.Item(0))      End While

connection.Close()

```

```

Button1.Visible = False      Me.Visible = True

```

```

For i As Integer = 0 To listID.Count      loadReport(listID.Item(i))

```

```

PrintForm1.PrintAction = Printing.PrintAction.PrintToPrinter

```

```

PrintForm1.Print()

```

```

Next

```

```

Button1.Visible = True

```

```

Me.Close()

```

```

End Sub

```

```

Private Sub PPMReport_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

```

```

End Sub

```

Reference Tables

```

Public Class ReferenceTables

```

```

Dim connection As New

```

```

OleDb.OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" &
Application.StartupPath & "\FYP.accdb;Persist Security Info=False;")

```

```

Public Sub filterInventory(ByVal inventoryType As String)      Label1.Text =
"Reference Table for " & inventoryType

    Dim sql As String = ""      Dim recordSet

As String = ""

    If (inventoryType.Equals("Equipments")) Then

        sql = "SELECT EquipmentT.[Asset#], EquipmentT.EquipmentName,
EquipmentT.Manufacturer, EquipmentT.Model, EquipmentT.[Serial#],
EquipmentT.EquipmentType, EquipmentT.CustomerID, EquipmentT.Price,
EquipmentT.[Site#], EquipmentT.[MD#], EquipmentT.LocationID,
EquipmentT.StartUp, EquipmentT.UnderWarranty, EquipmentT.EndOfWarranty,
EquipmentT.Retired, EquipmentT.[PPM's Yearly], EquipmentT.PPMschedule,
EquipmentT.[PPM's Month], EquipmentT.[1st PPM], EquipmentT.[1st PPM'Status],
EquipmentT.[2nd PPM], EquipmentT.[2nd PPM'Status], EquipmentT.[3rd PPM],
EquipmentT.[3rd PPM'Status], EquipmentT.[4th PPM], EquipmentT.[4th PPM'Status],
EquipmentT.[5th PPM], EquipmentT.[5th PPM'Status], EquipmentT.[6th PPM],
EquipmentT.[6th PPM'Status], EquipmentT.[7th PPM], EquipmentT.[7th PPM'Status],
EquipmentT.[8th PPM], EquipmentT.[8th PPM'Status], EquipmentT.[9th PPM],
EquipmentT.[9th PPM'Status], EquipmentT.[10th PPM], EquipmentT.[10th
PPM'Status], EquipmentT.[PO#], EquipmentT.AcceptanceForm,
EquipmentT.EquipmentPic, EquipmentT.Remark, EquipmentT.supplier FROM
EquipmentT"      recordSet =

"EQUIPMENTT"

        ElseIf (inventoryType.Equals("Customers")) Then

            sql = "SELECT CustomerT.CustomerID, CustomerT.CustomerName,
CustomerT.Address, CustomerT.City, CustomerT.CustomerPhone,
CustomerT.[CustomerE-mail] FROM CustomerT"

recordSet = "CustomerT"

            ElseIf (inventoryType.Equals("Sites")) Then      sql = "SELECT

SiteT.[Site#], SiteT.SiteName FROM SiteT"      recordSet = "SiteT"

            ElseIf (inventoryType.Equals("Locations")) Then

                sql = "SELECT LocationT.LocationID, LocationT.LocationName FROM

LocationT"      recordSet = "LOCATIONT"

```

ElseIf (inventoryType.Equals("Work Orders")) Then

```
sql = "SELECT WorkOrderT.[WO#], WorkOrderT.CallDate,  
WorkOrderT.CallTime, WorkOrderT.[MD#], WorkOrderT.[Site#],  
WorkOrderT.CallerName, WorkOrderT.Callerphone, WorkOrderT.[CallerE-mail],  
WorkOrderT.[Asset#], WorkOrderT.Urgency, WorkOrderT.Reason,  
WorkOrderT.RequestDetail, WorkOrderT.TechnicalTeam, WorkOrderT.SparePart,  
WorkOrderT.[WO'Status], WorkOrderT.Comment FROM WorkOrderT"
```

recordSet = "WorkOrderT" End If

Dim dataadapter As New OleDb.OleDbDataAdapter(sql, connection) Dim

ds As New DataSet() connection.Open() dataadapter.Fill(ds, recordSet)

connection.Close()

DataGridView1.DataSource = ds

DataGridView1.DataMember = recordSet

End Sub

Private Sub ReferenceTables_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

End Sub

End Class

PPM search

Public Class SearchPPM

Dim connection As New

OleDb.OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" &
Application.StartupPath & "\FYP.accdb;Persist Security Info=False;")

```

Public Sub searchPPMByMonth(ByVal month As Integer)    Label1.Text =
"PPM Search for " & getMonth(month)

```

```

    Dim sql As String = "SELECT * FROM EQUIPMENTT WHERE ( MONTH([1stPPM]) = "
& month & " OR MONTH([2NDPPM]) = " & month & " OR MONTH([3RDPPM]) = " & month
& " OR MONTH([4THPPM]) = " & month & " OR MONTH([5THPPM]) = " & month & " OR
MONTH([6THPPM]) = " & month & " OR MONTH([7THPPM]) = " & month & " OR
MONTH([8THPPM]) = " & month & " OR MONTH([9THPPM]) = " & month & " OR
MONTH([10THPPM]) = " & month & ")"

```

```

    Dim dataadapter As New OleDb.OleDbDataAdapter(sql, connection)    Dim
ds As New DataSet()    connection.Open()    dataadapter.Fill(ds,
"EQUIPMENTT")    connection.Close()    DataGridView1.DataSource = ds

DataGridView1.DataMember = "EQUIPMENTT"

End Sub

```

```

Private Function getMonth(ByVal month As Integer) As String

```

```

    If (month = 1) Then

        Return "January"
    ElseIf (month = 2) Then

        Return "February"
    ElseIf (month = 3) Then

        Return "March"
    ElseIf (month = 4) Then

        Return "April"
    ElseIf (month = 5) Then

        Return "May"
    ElseIf (month = 6) Then

        Return "June"
    ElseIf (month = 7) Then

```

```

        Return "July"
    ElseIf (month = 8) Then
        Return "August"
    ElseIf (month = 9) Then
        Return "September"
    ElseIf (month = 10) Then
        Return "October"
    ElseIf (month = 11) Then
        Return
"November"
    End If

    Return "December" End

Function

```

```

Private Sub SearchPPM_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

```

```

End Sub

```

```

End Class

```

WO report

```

Public Class WOREport

```

```

    Dim connection As New
OleDb.OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" &
Application.StartupPath & "\FYP.accdb;Persist Security Info=False;")

```

```

    Dim reportList As List(Of String) = New List(Of String)()

```

```

    Dim reportListIndex As Integer = -1

```

Public Sub loadReport(ByVal ID As String)

checkSpare.Enabled = False checkUrgent.Enabled = False

Dim command As New OleDb.OleDbCommand("SELECT WorkOrderT.CallDate,
WorkOrderT.CallTime, WorkOrderT.[MD#], WorkOrderT.[Site#],
WorkOrderT.CallerName, WorkOrderT.Callerphone, WorkOrderT.[CallerE-mail],
WorkOrderT.[Asset#], WorkOrderT.Urgency, WorkOrderT.Reason,
WorkOrderT.RequestDetail, WorkOrderT.TechnicalTeam, WorkOrderT.SparePart,
WorkOrderT.[WO'Status], WorkOrderT.Comment, EquipmentT.EquipmentName FROM
WorkOrderT, EquipmentT WHERE EquipmentT.[Asset#] = WorkOrderT.[Asset#] AND
WorkOrderT.[WO#] = " & ID) connection.Open()

command.Connection = connection

Dim dataReader As OleDb.OleDbDataReader = command.ExecuteReader

If (dataReader.Read()) Then

'WorkOrderT.[WO#], labelWOID.Text

= ID

'WorkOrderT.CallDate,

If (Not IsDBNull(dataReader.Item(0))) Then labelCallDate.Text =

dataReader.Item(0)

End If

'WorkOrderT.CallTime,

If (Not IsDBNull(dataReader.Item(1))) Then labelCallTime.Text =

dataReader.Item(1)

End If

'WorkOrderT.[MD#],

<pre> If (Not IsDBNull(dataReader.Item(2))) Then dataReader.Item(2) End If 'WorkOrderT.[Site#], If (Not IsDBNull(dataReader.Item(3))) Then dataReader.Item(3) End If 'WorkOrderT.CallerName, If (Not IsDBNull(dataReader.Item(4))) Then dataReader.Item(4) End If 'WorkOrderT.Callerphone, If (Not IsDBNull(dataReader.Item(5))) Then dataReader.Item(5) End If 'WorkOrderT.[CallerE-mail], If (Not IsDBNull(dataReader.Item(6))) Then dataReader.Item(6) End If 'WorkOrderT.[Asset#], </pre>	<pre> labelMDID.Text = labelSiteID.Text = labelCallName.Text = labelCallPhone.Text = labelCallEmail.Text = </pre>
--	---

```
        If (Not IsDBNull(dataReader.Item(7))) Then            labelAssetID.Text =  
dataReader.Item(7).ToString  
        End If
```

```
        If (Not IsDBNull(dataReader.Item(8))) Then  
            'WorkOrderT.Urgency,  
            checkUrgent.Checked = dataReader.Item(8)  
        End If
```

```
        If (Not IsDBNull(dataReader.Item(9))) Then  
            'WorkOrderT.Reason,            labelReason.Text =  
dataReader.Item(9)  
        End If
```

```
        If (Not IsDBNull(dataReader.Item(10))) Then  
            'WorkOrderT.RequestDetail,            labelDetail.Text =  
dataReader.Item(10)  
        End If
```

```
        If (Not IsDBNull(dataReader.Item(11))) Then  
            'WorkOrderT.TechnicalTeam,            labelTeam.Text =  
dataReader.Item(11)  
        End If
```

```
        If (Not IsDBNull(dataReader.Item(12))) Then
```

```

        'WorkOrderT.SparePart,          checkSpare.Checked =
dataReader.Item(12)

    End If

    If (Not IsDBNull(dataReader.Item(13))) Then
        'WorkOrderT.[WO'Status],          labelStatus.Text =
dataReader.Item(13)

    End If

    If (Not IsDBNull(dataReader.Item(14))) Then
        'WorkOrderT.Comment
        LabelComment.Text = dataReader.Item(14)
    End If

    If (Not IsDBNull(dataReader.Item(15))) Then
        'WorkOrderT.Comment          labelEquipName.Text =
dataReader.Item(15)

    End If

Else

    MsgBox("Sorry, Work Order ID could not be found",
MsgBoxStyle.OkOnly, "Invalid ID")

    Me.Close()    End If

connection.Close() End Sub

```

```

Public Sub loadReportsByStatus(ByVal status As String)

```

```

    Dim command As New OleDb.OleDbCommand("SELECT WorkOrderT.[WO#] FROM

```

```
WorkOrderT WHERE WorkOrderT.[WO'Status] = ' ' & status & ' '))
```

```
connection.Open()      command.Connection = connection
```

```
Dim dataReader As OleDb.OleDbDataReader = command.ExecuteReader
```

```
reportList.Clear()      While (dataReader.Read())
```

```
reportList.Add(dataReader.Item(0))
```

```
End While
```

```
connection.Close()
```

```
If (reportList.Count = 0) Then      reportListIndex = -1
```

```
MsgBox("Sorry, there is no WO found with status " & status)
```

```
Me.Close()
```

```
Else
```

```
Button2.Visible = True
```

```
Button3.Visible = True      reportListIndex = 0
```

```
loadReport(reportList.Item(reportListIndex))
```

```
Me.Show()
```

```
End If
```

```
End Sub
```

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)  
Handles Button1.Click
```

```
Button1.Visible = False
```

```
PrintForm1.PrintAction = Printing.PrintAction.PrintToPreview
```

```
PrintForm1.Print()      Button1.Visible = True
```

```
End Sub
```

```

Public Sub printAll()

    Dim command As New OleDb.OleDbCommand("SELECT WorkOrderT.[WO#]
FROM WorkOrderT")    connection.Open()    command.Connection = connection

    Dim dataReader As OleDb.OleDbDataReader = command.ExecuteReader

    Dim listID As List(Of String) = New List(Of String)()

    While (dataReader.Read())

listID.Add(dataReader.Item(0))    End While

connection.Close()

    Button1.Visible = False    Me.Visible = True

    For i As Integer = 0 To listID.Count

        loadReport(listID.Item(i))

        PrintForm1.PrintAction = Printing.PrintAction.PrintToPrinter

        PrintForm1.Print()

    Next

    Button1.Visible = True

    Me.Close()

End Sub

Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click    If (reportListIndex > 0) Then
reportListIndex = reportListIndex - 1
loadReport(reportList.Item(reportListIndex))

Else

    MsgBox("Beginning of Work Order List", MsgBoxStyle.OkOnly,

```

"Beginning of List")

End If

End Sub

Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button3.Click

If (reportListIndex < reportList.Count - 1) Then reportListIndex =
reportListIndex + 1 loadReport(reportList.Item(reportListIndex))

Else

MsgBox("End of Work Order List", MsgBoxStyle.OkOnly, "End of List")

End If

End Sub

Private Sub WOReport_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load

End Sub

End Class

WO form

Public Class WorkOrder

Dim connection As New
OleDb.OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" &
Application.StartupPath & "\FYP.accdb;Persist Security Info=False;")

Dim WOID As String

Public Sub showStatus(ByVal status As String)

Dim SQLCommand As OleDb.OleDbCommand = New OleDb.OleDbCommand("SELECT
WorkOrderT.[WO#], WorkOrderT.CallDate, WorkOrderT.CallTime, WorkOrderT.[MD#],
WorkOrderT.[Site#], WorkOrderT.CallerName, WorkOrderT.Callerphone,
WorkOrderT.[CallerE-mail], WorkOrderT.[Asset#], WorkOrderT.Urgency,

```
WorkOrderT.Reason, WorkOrderT.RequestDetail, WorkOrderT.TechnicalTeam,
WorkOrderT.SparePart, WorkOrderT.[WO'Status], WorkOrderT.Comment FROM WorkOrderT
WHERE WorkOrderT.[WO'Status] = " & status & """)    connection.Open()
```

```
Dim dataReader As OleDb.OleDbDataReader    dataReader =
```

```
SQLCommand.ExecuteReader
```

```
While (dataReader.Read())
```

```
End While
```

```
End Sub
```

```
Public Sub showRecord(ByVal recordID As String)
```

```
WOID = recordID
```

```
Dim command As New OleDb.OleDbCommand("SELECT CallDate, CallTime,
[MD#], [Site#], CallerName, Callerphone, [CallerE-mail], [Asset#], Urgency,
Reason, RequestDetail, TechnicalTeam, SparePart, [WO'Status], Comment FROM
WORKORDERT WHERE [WO#] = " & recordID)
```

```
connection.Open()    command.Connection =
```

```
connection
```

```
Dim dataReader As OleDb.OleDbDataReader = command.ExecuteReader
```

```
If (dataReader.Read()) Then    dt_Date.Text = dataReader.Item(0)
```

```
dt_Time.Text = dataReader.Item(1)    textMD.Text = dataReader.Item(2)
```

```
textSite.Text = dataReader.Item(3)    textCallerName.Text = dataReader.Item(4)
```

```
textCallerPhone.Text = dataReader.Item(5)    textCallerEmail.Text =
```

```
dataReader.Item(6)    textAssetID.Text = dataReader.Item(7)
```

```
checkUrgent.Checked = dataReader.Item(8).Equals(True)    textReason.Text =
```

```

dataReader.Item(9)          textRequestDetail.Text = dataReader.Item(10)

textTechnicalTeam.Text = dataReader.Item(11)          checkSparePart.Checked =
dataReader.Item(12).Equals(True)          comboStatus.Text = dataReader.Item(13)

textComment.Text = dataReader.Item(14)          End If          connection.Close()

findAsset(textAssetID.Text)  End Sub

```

```

Private Function executeSQL(ByVal SQL As String) As Integer          Dim
result As Boolean = False          connection.Open()

Dim command As New OleDb.OleDbCommand(SQL)

command.Connection = connection

Dim affected As Integer

Try          affected = command.ExecuteNonQuery()

Catch ex As Exception          affected = -1

MsgBox(ex.Message, MsgBoxStyle.OkOnly, "Operation Failed")

End Try

connection.Close()          Return

affected  End Function

```

```

Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click

If (textAssetID.Text.Length = 0) Then

MsgBox("Please enter Asset ID", MsgBoxStyle.OkOnly, "Missing Input")

ElseIf (dt_Date.Text.Length = 0) Then

MsgBox("Please enter Caller Date", MsgBoxStyle.OkOnly, "Missing Input")

ElseIf (dt_Time.Text.Length = 0) Then

MsgBox("Please enter Caller Time", MsgBoxStyle.OkOnly, "Missing Input")

```

```

ElseIf (textCallerName.Text.Length = 0) Then
    MsgBox("Please enter Caller Name", MsgBoxStyle.OkOnly, "Missing Input")
ElseIf (textCallerPhone.Text.Length = 0) Then
    MsgBox("Please enter Caller Phone", MsgBoxStyle.OkOnly, "Missing Input")
ElseIf (textCallerEmail.Text.Length = 0) Then
    MsgBox("Please enter Caller Email", MsgBoxStyle.OkOnly, "Missing Input")
ElseIf (textReason.Text.Length = 0) Then
    MsgBox("Please enter Reason", MsgBoxStyle.OkOnly, "Missing Input") ElseIf
(textRequestDetail.Text.Length = 0) Then
    MsgBox("Please enter Request Detail", MsgBoxStyle.OkOnly, "Missing Input")
ElseIf (textEquipmentName.Text.Length = 0) Then
    MsgBox("Equipment Asset# is invalid", MsgBoxStyle.OkOnly, "Missing Input")
ElseIf (textSite.Text.Length = 0) Then
    MsgBox("Please enter Site ID", MsgBoxStyle.OkOnly, "Missing Input")
ElseIf (textTechnicalTeam.Text.Length = 0) Then
    MsgBox("Please enter Technical Team", MsgBoxStyle.OkOnly, "Missing Input")
ElseIf (textComment.Text.Length = 0) Then
    MsgBox("Please enter Comment", MsgBoxStyle.OkOnly, "Missing Input")
Else
    If (Button1.Text.StartsWith("Save")) Then
        If (executeSQL("INSERT INTO WorkOrderT ( CallDate, CallTime,
[MD#], [Site#], CallerName, Callerphone, [CallerE-mail], [Asset#], Urgency,
Reason, RequestDetail, TechnicalTeam, SparePart, [WO'Status], Comment ) VALUES ('" &
dt_Date.Value & "', '" & dt_Time.Value & "', '" & textMD.Text & "', '" & textSite.Text & "', '" &
textCallerName.Text & "', '" & textCallerPhone.Text &
'", '" & textCallerEmail.Text & "', '" & textAssetID.Text & "', '" & checkUrgent.Checked & "', '" &
textReason.Text & "', '" & textRequestDetail.Text & "', '" & textTechnicalTeam.Text & "', '" &
checkSparePart.Checked & "', '" & comboStatus.Text & "', '" & textComment.Text & '"') > 0)
Then
            MsgBox("New Work Order has been added successfully", MsgBoxStyle.OkOnly,
"Work Order Added")

```

Me.Close()

End If

Else

If (executeSQL("UPDATE WorkOrderT SET CallDate = " & dt_Date.Text & ",
CallTime = " & dt_Time.Text & ", [MD#] = " & textMD.Text & ", [Site#] = " & textSite.Text
& ", CallerName = " & textCallerName.Text
& ", Callerphone = " & textCallerPhone.Text & ", [CallerE-mail] = " & textCallerEmail.Text &
", [Asset#] = " & textAssetID.Text & ", Urgency = " & checkUrgent.Checked & ", Reason = "
& textReason.Text & ", RequestDetail = "
& textRequestDetail.Text & ", TechnicalTeam = " & textTechnicalTeam.Text & ", SparePart = "
& checkSparePart.Checked & ", [WO'Status] = " &
comboStatus.Text & ", Comment = " & textComment.Text & " WHERE [WO#] = " &
WOID) > 0) Then

MsgBox("Work Order has been updated successfully", MsgBoxStyle.OkOnly, "Work
Order Added")

Me.Close()

End If

End If

End If

End Sub

Public Sub setMode(ByVal mode As String) If

(mode.Equals("new")) Then textAssetID.Text =

"" textCallerEmail.Text = ""

textCallerName.Text = "" textCallerPhone.Text =

"" textComment.Text = ""

textEquipmentName.Text = "" textMD.Text = ""

textReason.Text = "" textRequestDetail.Text = ""

textSite.Text = "" textTechnicalTeam.Text = ""

comboStatus.SelectedIndex = 0 dt_Date.Text =

```

Date.Now.ToString      dt_Time.Text =

Date.Now.ToString      Button1.Text = "Save Word
Order"

    ElseIf (mode.Equals("update")) Then
        Button1.Text = "Update Word Order"
    End If

End Sub

Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button2.Click

    Me.Close()

End Sub

Private Sub textAssetID_Leave(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles textAssetID.Leave      findAsset(textAssetID.Text)

End Sub

Private Sub findAsset(ByVal id As String)

    Dim command As New OleDb.OleDbCommand("SELECT EQUIPMENTNAME FROM
EQUIPMENTT WHERE [ASSET#] = " & id)

connection.Open()      command.Connection =
connection

    Dim dataReader As OleDb.OleDbDataReader = command.ExecuteReader

    If (dataReader.Read()) Then      textEquipmentName.Text =

dataReader.Item(0)      Else      textEquipmentName.Text = ""

```

End If connection.Close()

End Sub

Private Sub WorkOrder_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load

End Sub

End Class

WO searcher

Public Class WOSearcher

Dim connection As New
OleDb.OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" &
Application.StartupPath & "\FYP.accdb;Persist Security Info=False;")

Public Sub filterWO(ByVal status As String)

Label1.Text = "Searching Word Orders as " & status

Dim sql As String = "SELECT WorkOrderT.[WO#], WorkOrderT.CallDate,
WorkOrderT.CallTime, WorkOrderT.[MD#], WorkOrderT.[Site#],
WorkOrderT.CallerName, WorkOrderT.Callerphone, WorkOrderT.[CallerE-mail],
WorkOrderT.[Asset#], EQUIPMENTT.EQUIPMENTNAME, WorkOrderT.Urgency,
WorkOrderT.Reason, WorkOrderT.RequestDetail, WorkOrderT.TechnicalTeam,
WorkOrderT.SparePart, WorkOrderT.[WO'Status], WorkOrderT.Comment FROM
WORKORDERT, EQUIPMENTT WHERE WORKORDERT.[ASSET#] =
EQUIPMENTT.[ASSET#] AND
WorkOrderT.[WO'Status] = " & status & ""

Dim dataadapter As New OleDb.OleDbDataAdapter(sql, connection) Dim
ds As New DataSet() connection.Open() dataadapter.Fill(ds,
"WORKORDERT") connection.Close() DataGridView1.DataSource = ds
DataGridView1.DataMember = "WORKORDERT"

End Sub

Private Sub WOSearcher_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load

End Sub

End Class

WO status

Public Class WOStatus

Dim connection As New
OleDb.OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" &
Application.StartupPath & "\FYP.accdb;Persist Security Info=False;")

Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

Dim sql As String = "SELECT [WO#], EQUIPMENTT.EQUIPMENTNAME,
[WO'STATUS] FROM WORKORDERT, EQUIPMENTT WHERE WORKORDERT.[ASSET#]
=
EQUIPMENTT.[ASSET#]"

Dim dataadapter As New OleDb.OleDbDataAdapter(sql, connection) Dim
ds As New DataSet() connection.Open() dataadapter.Fill(ds,
"WORKORDERT") connection.Close()

DataGridView1.DataSource = ds

DataGridView1.DataMember = "WORKORDERT"

End Sub

Private Sub WOStatus_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load

End Sub

End Class

Advanced Searcher

Public Class AdvancedSearcher

```

Dim connection As New
OleDb.OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" &
Application.StartupPath & "\FYP.accdb;Persist Security Info=False;")

```

```

Public Sub searchAssetByID(ByVal ID As String)

```

```

    Label1.Text = "Searching Asset with ID " & ID

```

```

    Dim sql As String = "SELECT * FROM EQUIPMENTT WHERE [ASSET#] = " & ID

```

```

    Dim dataadapter As New OleDb.OleDbDataAdapter(sql, connection)    Dim

```

```

ds As New DataSet()    connection.Open()    dataadapter.Fill(ds,

```

```

"EQUIPMENTT")    connection.Close()    DataGridView1.DataSource = ds

```

```

    DataGridView1.DataMember = "EQUIPMENTT"

```

```

End Sub

```

```

Public Sub searchAssetBySupplierID(ByVal ID As String)

```

```

    Label1.Text = "Searching Asset with Customer ID " & ID

```

```

    Dim sql As String = "SELECT * FROM EQUIPMENTT WHERE CUSTOMERID = '" &
ID & "'"

```

```

    Dim dataadapter As New OleDb.OleDbDataAdapter(sql, connection)    Dim

```

```

ds As New DataSet()    connection.Open()    dataadapter.Fill(ds,

```

```

"EQUIPMENTT")

```

```

    connection.Close()

```

```

    DataGridView1.DataSource = ds

```

```

    DataGridView1.DataMember = "EQUIPMENTT"

```

```

End Sub

```

```
Public Sub searchAssetBySiteID(ByVal ID As String)
```

```
Label1.Text = "Searching Asset with Site ID " & ID
```

```
Dim sql As String = "SELECT * FROM EQUIPMENTT WHERE [SITE#] = " & ID & """"
```

```
Dim dataadapter As New OleDb.OleDbDataAdapter(sql, connection) Dim  
ds As New DataSet() connection.Open() dataadapter.Fill(ds,  
"EQUIPMENTT") connection.Close() DataGridView1.DataSource = ds  
DataGridView1.DataMember = "EQUIPMENTT"
```

```
End Sub
```

```
Public Sub searchAssetByMDID(ByVal ID As String)
```

```
Label1.Text = "Searching Asset with MD ID " & ID
```

```
Dim sql As String = "SELECT * FROM EQUIPMENTT WHERE [MD#] = " & ID & """"
```

```
Dim dataadapter As New OleDb.OleDbDataAdapter(sql, connection) Dim  
ds As New DataSet() connection.Open() dataadapter.Fill(ds,  
"EQUIPMENTT") connection.Close() DataGridView1.DataSource = ds  
DataGridView1.DataMember = "EQUIPMENTT"
```

```
End Sub
```

```
Public Sub searchAssetByLocation(ByVal ID As String)
```

```
Label1.Text = "Searching Asset with Location ID " & ID
```

```
Dim sql As String = "SELECT * FROM EQUIPMENTT WHERE LOCATIONID = " &  
ID & """"
```

```

    Dim dataadapter As New OleDb.OleDbDataAdapter(sql, connection)    Dim
ds As New DataSet()    connection.Open()    dataadapter.Fill(ds,
"EQUIPMENTT")    connection.Close()    DataGridView1.DataSource = ds
    DataGridView1.DataMember = "EQUIPMENTT"
End Sub

```

```

Public Sub searchAssetByPO(ByVal ID As String)

```

```

    Label1.Text = "Searching Asset with PO ID " & ID

```

```

    Dim sql As String = "SELECT * FROM EQUIPMENTT WHERE [PO#] = '" & ID &
""""

```

```

    Dim dataadapter As New OleDb.OleDbDataAdapter(sql, connection)    Dim
ds As New DataSet()    connection.Open()    dataadapter.Fill(ds,
"EQUIPMENTT")    connection.Close()    DataGridView1.DataSource = ds
    DataGridView1.DataMember = "EQUIPMENTT"
End Sub

```

```

Public Sub searchAssetByWO(ByVal ID As String)

```

```

    Label1.Text = "Searching Asset with WO ID " & ID

```

```

    Dim sql As String = "SELECT * FROM WORKORDERT WHERE [WO#] = " & ID

```

```

    Dim dataadapter As New OleDb.OleDbDataAdapter(sql, connection)

```

```

    Dim ds As New DataSet()    connection.Open()

```

```

dataadapter.Fill(ds, "EQUIPMENTT")

```

```

connection.Close()    DataGridView1.DataSource = ds

```

```

    DataGridView1.DataMember = "EQUIPMENTT"

```

End Sub

Public Sub searchAssetByAssetType(ByVal ID As String)

Label1.Text = "Searching Asset with Asset Type " & ID

Dim sql As String = "SELECT * FROM EQUIPMENTT WHERE EQUIPMENTTYPE = " & ID & ""

Dim dataadapter As New OleDb.OleDbDataAdapter(sql, connection) Dim

ds As New DataSet() connection.Open() dataadapter.Fill(ds,

"EQUIPMENTT") connection.Close() DataGridView1.DataSource = ds

DataGridView1.DataMember = "EQUIPMENTT"

End Sub

Private Sub AdvancedSearcher_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load

End Sub

End Class

APPENDIX B

Gantt Chart for Project Phase I

WORK DONE	Week													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Read about Project														
Project Objective														
Introduction														
Review Previous Papers														
Review about System														
Analays of Procedure and Method														
Preparing for Presentation														

Gantt Chart for Project Phase II

WORK DONE	Week													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Data Collection														
Designing the database														
Finalization of Database														
Coding the software														
Verification of result														
Documentation														
Preparation of presentation														