

A project report on

“PHARMACY MANAGEMENT SYSTEM”

A project report submitted in partial fulfillment of requirement for Bachelor of Computer Application, fifth semester under Bangalore North University during the academic year 2022-2023.

BACHELOR OF COMPUTER APPLICATIONS IN 5th SEM

Submitted By:

Name:

Gagan S R

Surya K

Karthik Raja V

Register No:

R2013419

R2013456

R2013427

UNDER THE GUIDANCE OF

Mrs. Santhi P Theja

Department of Computer Science & Applications



SEA College of Science Commerce and Arts

Ekta Nagar, Near Ayyappanagar Circle Devasandra Main Road,
Virgonagar Post K. R Puram, Bangalore-560049

**SEA College of Science, Commerce and Arts
Bengaluru North University**



Certificate

This is to certify that the project work entitled “**Pharmacy Management System**”, is a bonafide work done by **Gagan S R (R2013419)**, **Surya K (R2013456)** and **Karthik Raja V (R2013427)** under the guidance of **Mrs. Santhi P Theja**, Department of Computer Science, & Applications. which is in partial fulfillment of the requirement for the award of Bachelor of Computer Applications from Bangalore North University, 2022-2023.

Internal Guide:

Head Of The Department:

Principal:

Examiners:

1).....

Date of examination:

2).....

Place: Bangalore

Acknowledgement

Above all we want to thank God Almighty for the blessings, health, wisdom, knowledge and strength he has given to us. It is only by his will that we are able to start and complete this project.

We wish to express our sincere gratitude to Late A. Krishnappa, the chairman of South East Asian Educational Trust for providing good infrastructures and adequate equipment's for the proper functioning of the institution. We also thank Dr. T N Muthe Gowda, the principal of SEA College of Science, Commerce and Arts for his tireless effort in managing the college making sure that everything runs in proper order.

We like to thank our HOD Mrs. Santhi P Theja , who is guide of our project for her constant support, advice and guidance which greatly contributed to the successful completion of this project. We also wish to express our gratitude the all the staffs in the Computer Science Department of their support, concern and encouragement not only to the project, but also in our studies.

We also want to thank our parents for their guidance, support, encouragement, advice and prayers. And also for providing all the materials we need, not only for our studies, but also for our well being if not for them, we will not be here today.

INDEX

S1 No	Title	Page No
1	Introduction	1
2	Abstract	2
3	Software Requirement Specification	3
4	Feasibility Study	4
5	Input and validation	5
6	System Analysis	6
7	Data Flow Diagrams	7
8	Overview of Technologies Used	9
9	Hierarchical Diagram	23
10	E-R Diagram	25
11	Testing	25
12	Screenshots	27
13	Coding	33
14	Conclusion	58
15	Bibliography	59

Introduction:

The "Pharmacy Management System" has been developed to override the problems prevailing in the practicing manual system. This software is supported to eliminate some cases reduce the hardships faced by this existing system. Moreover this system is designed for the particular need of the company to carry out operations in a smooth and effective manner.

The application is reduced as much as possible to avoid errors while entering the data. It also provides error message while entering invalid data. No formal knowledge is needed for the user to use this system. Thus by this all it proves it is user-friendly. Pharmacy Management System , as described above, can lead to error free, secure, reliable and fast management system. It can assist the user to concentrate on their other activities rather to concentrate on the record keeping. Thus it will help organization in better utilization of resources.

Every organization, whether big or small, has challenges to overcome and managing the informations of Medicines, Pharmacy, Company, Sells, Inventory. Every Pharmacy Management System has different Pharmacy needs, therefore we design exclusive employee management systems that are adapted to your managerial requirements. This is designed to assist in strategic planning, and will help you ensure that your organization is equipped with the right level of information and details for your future goals. Also, for those busy executive who are always on the go, our systems come with remote access features, which will allow you to manage your workforce anytime, at all times. These systems will ultimately allow you to better manage resources.

Abstract of the Project Pharmacy Management System:

The purpose of Pharmacy Management System is to automate the existing manual system by the help of computerized equipments and full-fledged computer software, fulfilling their requirements, so that their valuable data/information can be stored for a longer period with easy accessing and manipulation of the same. The required software and hardware are easily available and easy to work with.

Pharmacy Management System, as described above, can lead to error free, secure, reliable and fast management system. It can assist the user to concentrate on their other

activities rather to concentrate on the record keeping. Thus it will help organization in better utilization of resources.

The organization can maintain computerized records without redundant entries. That means that one need not be distracted by information that is not relevant, while being able to reach the information.

The aim is to automate its existing manual system by the help of computerized equipments and full-fledged computer software, fulfilling their requirements, so that their valuable data/information can be stored for a longer period with easy accessing and manipulation of the same. Basically the project describes how to manage for good performance and better services for the clients

Functionalities :

- Provides the searching facilities based on various factors. Such as Pharmacy, Stocks, Company, Inventory
- Pharmacy Management System also manage the Sells details online for company details, Inventory details, Pharmacy.
- It tracks all the information of Medicines, Sells, Company ect
- Manage the information of Medicines
- Shows the information and description of the Pharmacy, Stocks
- To increase efficiency of managing the Pharmacy, Medicines
- It deals with monitoring the information and transactions of Company.
- Manage the information of Pharmacy
- Editing, adding and updating of Records is improved which results in proper resource management of Pharmacy data.
- Manage the information of Company
- Integration of all records of Inventory.

Modules :

- Pharmacy Management Module: Used for managing the Pharmacy details.
- Sales Module : Used for managing the details of Sells
- Stocks Module : Used for managing the Stocks details
- Login Module: Used for managing the login details
- Users Module : Used for managing the users of the system
- Employees Module: Used for managing the employees
- Reports Module: Used to access the reports
- Profile Module : used to manage the profile
- Customers Module: Used to manage details of customer

Software Requirement Specification

The Software Requirements Specification is produced at the culmination of the analysis task. The function and performance allocated to software as part of system engineering are refined by establishing a complete information description, a detailed functional and behavioral description, an indication of performance requirements and design constraints, appropriate validation criteria, and other data pertinent to requirements.

➤ HARDWARE REQUIREMENT:

- PROCESSOR : Intel core i3 and later, AMD Ryzen 3 and later
- RAM : Minimum 4 GB
- HARD DISK : Minimum 40 GB
- OPERATING SYSTEM : Windows 7 and above
- OTHER : Monitor, Mouse, Keyboard etc.

➤ SOFTWARE REQUIREMENT:

- FRONT-END : visual Basic 6.0
- BACK-END : MS Access, MySQL , Oracle

Feasibility Study:

After doing the project Pharmacy Management System, study and analyzing all the existing or required functionalities of the system, the next task is to do the feasibility study for the project. All projects are feasible - given unlimited resources and infinite time. Feasibility study includes consideration of all the possible ways to provide a solution to the given problem. The proposed solution should satisfy all the user requirements and should be flexible enough so that future changes can be easily done based on the future upcoming requirements.

A. Economical Feasibility

This is a very important aspect to be considered while developing a project. We decided the technology based on minimum possible cost factor.

- All hardware and software cost has to be borne by the organization.
- Overall we have estimated that the benefits the organization is going to receive from the proposed system will surely overcome the initial costs and the later on running cost for system.

B. Technical Feasibility

This included the study of function, performance and constraints that may affect the ability to achieve an acceptable system. For this feasibility study, we studied complete functionality to be provided in the system, as described in the System Requirement Specification (SRS), and checked if everything was possible using different type of frontend and backend platformst.

C. Operational Feasibility

No doubt the proposed system is fully GUI based that is very user friendly and all inputs to be taken all self-explanatory even to a layman. Besides, a proper training has been conducted to let know the essence of the system to the users so that they feel comfortable with new system. As far our study is concerned the clients are comfortable and happy as the system has cut down their loads and doing.

Input Data and Validation of Project on Pharmacy Management System

- All the fields such as Pharmacy, Stocks, Inventory are validated and does not take invalid values
- Each form for Pharmacy, Medicines, Sells cannot accept blank value fields
- Avoiding errors in data
- Controlling amount of input
- Integration of all the modules/forms in the system.
- Preparation of the test cases.
- Preparation of the possible test data with all the validation checks.
- Actual testing done manually.
- Recording of all the reproduced errors.
- Modifications done for the errors found during testing.
- Prepared the test result scripts after rectification of the errors.
- Functionality of the entire module/forms.
- Validations for user input.
- Checking of the Coding standards to be maintained during coding.
- Testing the module with all the possible test data.
- Testing of the functionality involving all type of calculations etc.

System Analysis:

System analysis is a process of gathering and interpreting facts, diagnosing problems and the information about the Pharmacy Management System to recommend improvements on the system. It is a problem solving activity that requires intensive communication between the system users and system developers. System analysis or study is an important phase of any system development process. The system is studied to the minutest detail and analyzed. The system analyst plays the role of the interrogator and dwells deep into the working of the present system. The system is viewed as a whole and the input to the system are identified. The outputs from the organizations are traced to the various processes. System analysis is concerned with becoming aware of the problem, identifying the relevant and decisional variables, analyzing and synthesizing the various factors and determining an optimal or at least a satisfactory solution or program of action. A detailed study of the process must be made by various techniques like interviews, questionnaires etc. The data collected by these sources must be scrutinized to arrive to a conclusion. The conclusion is an understanding of how the system functions. This system is called the existing system. Now the existing system is subjected to close study and problem areas are identified. The designer now functions as a problem solver and tries to sort out the difficulties that the enterprise faces. The solutions are given as proposals. The proposal is then weighed with the existing system analytically and the best one is selected. The proposal is presented to the user for an endorsement by the user. The proposal is reviewed on user request and suitable changes are made. This is loop that ends as soon as the user is satisfied with proposal. Preliminary study is the process of gathering and interpreting facts, using the information for further studies on the system. Preliminary study is problem solving activity that requires intensive communication between the system users and system developers. It does various feasibility studies. In these studies a rough figure of the system activities can be obtained, from which the decision about the strategies to be followed for effective system study and analysis can be taken.

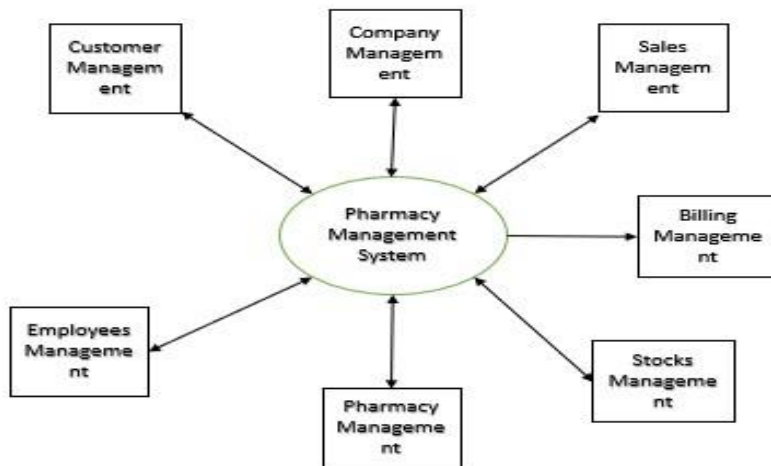
Data Flow Diagram:

A data flow (DFD) is a graphical system model that shows all of the main requirements for an information system in one datagram: inputs and outputs, processes, and data storage. A DFD describes what data flows rather than how it is processed. Everyone working on a development project can see all aspects of the system working together at once with DFD. That is one reason for its popularity. The DFD is also easy to read because it is graphical model. The DFD is mainly used during problem analysis. End Users, management, and all information systems workers typically can read and interpret the DFD with minimal training.

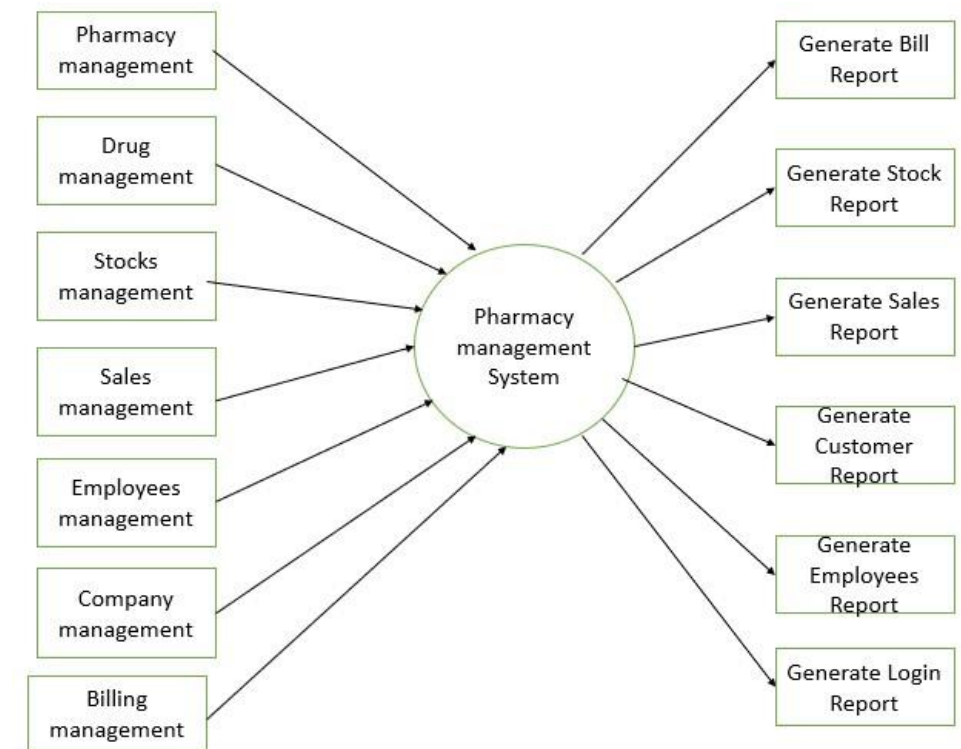
CONTEXT LEVEL DIAGRAM

DATA FLOW DIAGRAM A Graphical tool used to describe and analyses the movement of data through a system manual or automated including the process, stores of data and delays in the system. Data flow diagrams are the central tool and the basis from which other components are developed. The transformation of the data from input to output through process may be described logically and independently of the physical components associated with the system. They are termed logical DATA FLOW DIAGRAM. In contrast physical data flow diagrams show the actual implementation and movement of data between people, department and workstations□□□□

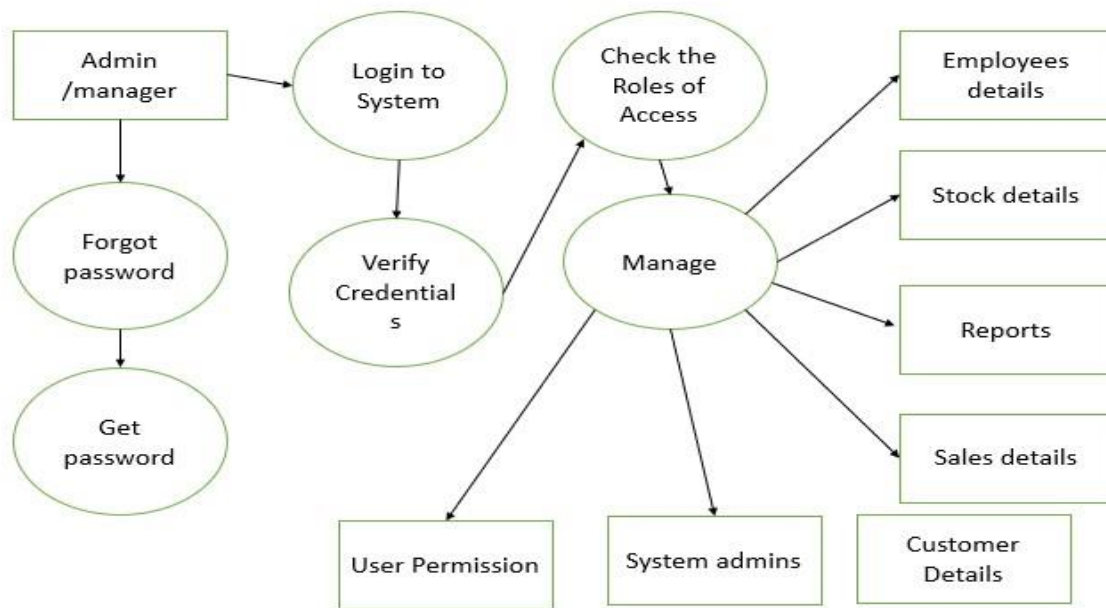
LEVEL 0 DFD(Data Flow Diagram)



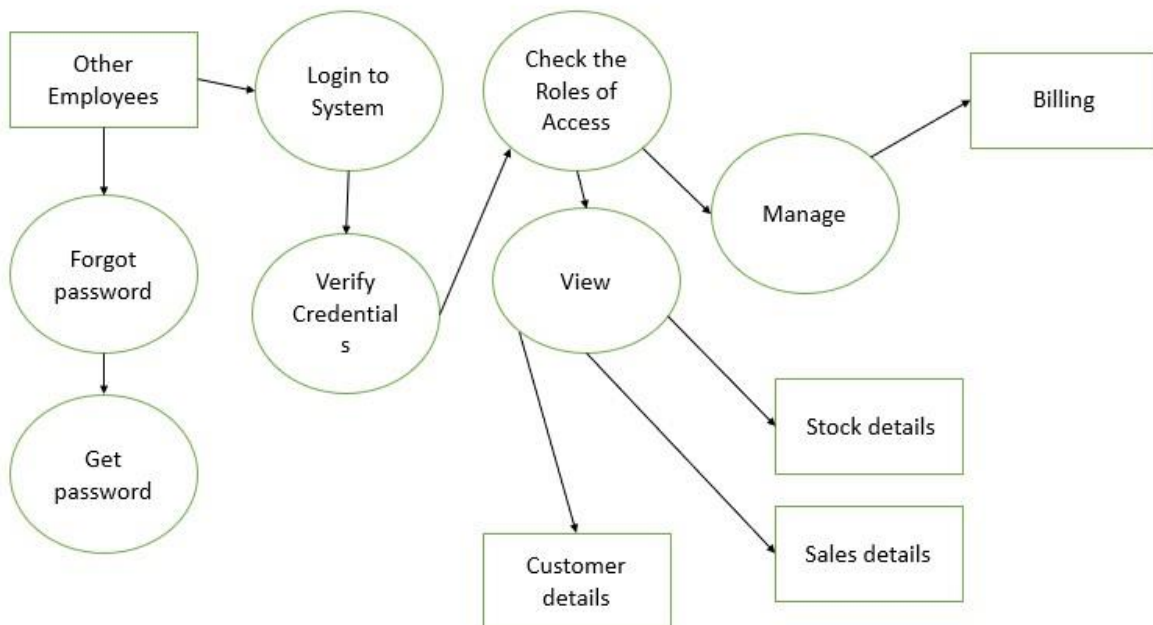
LEVEL 1 DFD(Data Flow Diagram)



LEVEL 2 DFD(Data Flow Diagram)



LEVEL 3 DFD(Data Flow Diagram)



Overview of Technologies Used

- Visual Basic 6.0
- Ms Access
- MySQL Oracle

ABOUT VB

Data access features allow u to create database and front-end applications for most popular database formats, including Microsoft SQL server and other enterprise-level databases. Active X technologies allow you to use the functionality provided by other applications, such as Microsoft word, Microsoft excel, and other windows applications can even automate applications and objects created using the professional or enterprise editions of visual basic. Internet capabilities make it easy to provide access to documents and applications across the Internet from within your application. Your finished application is a true exe file that uses run-time dynamic-link library (DLL) which you're freely distributes.

Three editions of VISUAL BASIC:

Visual Basic is available three editions, each graded to meet a specific set of developments requirements. The visual basic learning edition of standard edition allows programmers to easily create powerful applications for Microsoft windows 95 and window NT. It includes all intrinsic controls, plus grid, tab, and data-bound controls. The learning edition of visual basic incorporates a limited subset of the jet engine's data access capabilities in the data control and social data-bound controls. These features allow you to access existing data bases, but do not provide for creating new databases or data access objects. The professional edition provides computer professionals with a full-featured set of tools for developing solutions for other. It includes all the features of the learning edition plus additional Active X controls, including Internet controls, and Crystal Report Writer. It also includes all of the data control features, and full data access objects programming interface. The enterprise edition allows professionals to create robust distributed applications team setting. It includes all the features of the professional edition, plus the information manager, component manager, database management tools, the Microsoft visual source safe Tm project-oriented version control system and value. The enterprise edition adds a number of powerful client/server tools to visual Basic's data access capabilities. Including the Remote Data Control specifically designed to access remote ODBC client/server data bases Before you begin. (Check)

Hardware and System Requirements:

To run visual basic, you must have certain hardware and software installed on your computer. The system requirements include. Microsoft Windows NT 3.51 or later, or Microsoft Windows 95 80486 or higher microprocessor.

A hard disk with a minimum of 50 megabytes available space for a full installation.

A CD-ROM disc drive (optional)

16MB of RAM

A mouse or other suitable pointing device. The visual basic integrated development environment (IDE) consists of the following elements.

SDI or MDI Interface:

Visual Basic set to Multiple Document interface-MDI (discussed later) mode by default. To change to the single document mode-SDI mode follow the steps given below. We will be working in the SDI mode all of the IDE windows are free to be moved where on screen as long as visual basic is the current application, they will remain on top of any other applications.

To switch between SDI and MDI modes

1. Select options from the Tools menu. The options dialog box is displayed
2. Select the advanced tab
3. Check the SDI Development Environment check box to start in the SDI mode by default.

Note: This setting would take effect only the next time you start Visual Basic.

By now you should be comfortable identifying the many windows on the screen. Before we begin working with these controls let us understand Visual Basic's approach to programming.

Programming in Visual Basic is object base, here the focus is on the object. What is the object, the basic attributes of the (properties) how the user interacts with the object (events) and the actions in Visual Basic can be defined as combination of code and data that can be treated as one unit. An object can be an entire application or a piece of an application like a form or a control on a form e.g. a text box or a label a database or a chart are some other examples of objects.

Properties: Properties can be thought of as an object's attributes which define the appearance of the object like the size and color.

Events: Events are the responses which are triggered when the user interacts with the object at the time the program is executing e.g. a click, double click or drag.

Methods: Methods are actions that define the behavior of any object. E.g. move or save.

Therefore objects have properties, they respond to event and perform methods (action) you can use objects provided by Visual Basic to control other application's objects room within your visual basic applications, as well as create your own objects, and define additional properties and methods for them. Given below is a brief expression of all default visual basic objects/controls each of these objects/controls have their own set of properties, events, and methods with while you can control their appearance and behavior.

Active X control:

An Active X control is an extension to the Visual Basic toolbox. Active X control have the file name extension OCX. You can use the Active X controls provided with Visual Basic 5.0 or obtain additional controls from third-party developers. Visual Basic Active X controls are 32-bit controls. Some third-party developers offer Active X controls which an 16-bit controls, and these can not be used in Visual Basic version 5.0. You use Active X controls just as you would use any of the standard built-in controls, such as the Check Box control. When you add an Active X control to program, it becomes part of the development and runtime environment and provides new functionality for your applications.

Active X controls leverage your capabilities as a Visual Basic programmer by retaining some familiar properties, events, and methods. Such as the name property, which behave, as you would expect. Also the Active X controls feature methods and properties that greatly increase your flexibility and capability as a Visual Basic programmer.

Data Control:

The data control provides a relational interface to database files. Basically, a relational database is one that stores data of tables made up of columns and rows of data. In Visual Basic, columns are referred to as fields, and rows are referred to as records.

The Microsoft jet database engine that powers the data control views all databases as a set of relational tables, regardless of their physical file format. This means that when you use data from external databases(such as FoxPro, paradox, dBase, Microsoft Excel, Lotus 1-2-3 Text, or ODBC), you can use the same relational terms.

Tables:

A table is a logical grouping of related information arranged in rows and columns, similar to a spreadsheet table might contain a list of information about authors, such as their names date of birth, addresses, and pictures.

Fields:

Each column in a database table is called a field. Tables are defined by the fields they contain, with each field describing the data it is to hold. When creating a database, you assign a data type, maximum length, and other attributes to each field. Fields can contain characters, numbers, or even graphics. For example, the Authors table might have fields with the name and address as data type characters, the date of birth as data type date, and the author's photograph as type graphic.

Records:

Information about individual authors is kept in the rows of the table, called records. Generally, database table records are created such that no two rows are the same.

Indexes:

To make access to the data faster, most databases use indexes. Database table indexes are sorted lists that are faster to search than the tables. Each index entry points back to the database row it references. If the database (which does all of the searching) can look through an index first when looking for records (performing a query), its job is made easier and your data is returned faster.

Structured Query Language (SQL):

Other data is stored in the database, retrieving it is made easier by using an English-like language called structured Query Language, or SQL. SQL has evolved into the most widely accepted means to "converse" with a database. Basically, the user asks questions in the SQL language this question is called a query. The query usually contains the names of the tables to search, the names of the columns to return, and other information that sets the scope of the search. For example, an SQL query on Authors table might look like this: "Select Name , Picture from Authors where Date_of_Birth = # 2-7-1958#". This

SQL query would return the name and picture of all authors whose birthday is February 7, 1958. If any rows were returned. You could use bound controls to display the values. Because many of the external databases that Visual Basic recognizes are not relational in design, Visual Basic needs to convert external database structures to a relational model. Your code will not have to provide any specific logic to support this translation once the database has been opened—it is all done automatically.

Understanding Record set:

Visual Basic version 5.0 retrieves and displays database records using the recordset object provided by the Microsoft Jet database engine, version 3.5. A record set object represents the records in a base table or the records that result from running a query. The following tables list the three types of recordset objects available on the data control.

Record type description:

Table-type Recordset (dbOpen Table) A set of records that represents a single Database table that you can use to add, change, or delete records, **Dynaset-type Recordset (dbOpenDynaset)** A dynamic set of records that represents a database table or the results of a query containing fields from one or more tables. You can add, change, or delete records from a dynasettype Recordset and the changes will be reflected in the underlying table(s).

Snapshot-type Recordset (dbopenSnapshot) A static copy of a set of records that you can use to find data or generate reports. A snapshot-type Recordset can contain fields from one or more tables in a database but can't be updated. You can choose the type of Recordset object that you want the data control to create using the Recordset Type property. The default values vbDynaset Type.

Dynaset-and snapshot-type of Recordsets is stored in local memory. If you don't need your application to select fields from more than one table, and you are working with a non-ODBC source, the table-type. Recordset may be the most efficient in speed and memory use and in local TEMP disk space Recordset objects created in code (not available in the Visual Basic, learning Edition) can be assigned to the Recordset property of the data control using the set Statement. Set Data1.Recordset.MyRecordset. Similarly Recordset object created by one data control can be assigned to another data control at runtime.

The jet database engine provides a large number of database and recordset properties and methods. You can use these properties and methods directly with the data control by referring to data control's database and recordset properties.

Accessing Data:

Almost all applications require some form of data storage and manipulation, and Visual Basic provides a number of tools to meet these needs, including the data control and data-bound controls, data access objects, remote data objects, and remote data control.

This chapter focuses on the tools available in all editions of Visual Basic. The data control and data-bound controls. For stand-alone desktop applications, these tools are easy to use with little or no programming and allow you to create and access databases in a number of popular formats.

Data Access Options:

You can use the data controls to create applications that display, edit, and update information from many types of existing databases, including Microsoft Access, Brief, dBase, Microsoft Excel, Lotus 1-2-3, and standard ASCII text files as if they were true databases. In addition, the data control allows you to access and manipulate remote OpenDatabase connectivity (ODBC) databases such as Microsoft SQL Server and Oracle.

The data control implements data access by using the Microsoft Jet database engine, the same database engine that powers Microsoft Access. This technology gives you seamless access to many standard database formats and allows you to create data-aware applications without writing any code.

Data Access Controls:

You can use the data control to create simple database applications without writing any code at all. You can also use it together with Visual Basic code to create full-featured applications that give you a high degree of programming control over the behavior of your application's data. This topic will present more complex programming examples.

The data control can perform the following tasks without the use of code:

-----→ Connect to a local or remote database

-----→ Open a specified database table or define a set of records based on a Structured Query Language (SQL) query of the tables in that database.

-----→ Pass data fields to bound controls, where you can display or change the values.

-----→ Add new records or update a database based on any changes you make to data displayed in the bound controls.

-----→ Trap errors that occur as data is accessed.

-----→ Close the database

To create a database applications, you add the data control to your forms just as you would any other Visual Basic control. You can have as many data controls on your form as you need. As a rule, you will use one data control for each database table that you need to manipulate.

Creating a simple database application:

The following procedure gives you a brief overview of how to use the data control in a Visual Basic application. The example uses the Biblio.mdb sample database supplied with visual basic.

To use the data control in an application:

1. Select the data control in the Toolbox, and draw a data control on a form. After you draw the control on the form and size it, the caption appears as the default name of the control as Data1.
2. In the properties window, set the connect property to the type of database you want to use.
3. In the properties window set the database name property to the file or directory name of the database to which you want to connect. If your database is not available at design time, you will need to fill in the database name and record source properties at runtime.
4. Set the record source property to the name of the database table you want to access.

If the database is currently available, you can select a table from the dropdown list in the property window. If your database could not be found, the drop-down list will not appear in the record source settings box and an error message will appear.

5. Draw a text box on the form to display the database information. This control will be used to display and edit a selected field from the database. You can also use other data-bound controls including check boxes, picture boxes, image controls, labels, list boxes, combo boxes, and grid controls.
6. In the properties window, set the data source property for text1 to the name of the data control (data1). This binds the text box to the data control.
7. Add label and set its caption property to the name of the database field this text box will expose.
8. Set the data field property for text1 to the name of the field in the database table you want to view or modify.
9. Repeat steps 4,5,6,7 and 8 for each additional field you want to access. In the following example, the title,ISBN, and year published fields have been selected from the titles table.
10. Now run the application. You can use the four arrow buttons on the data control to move to the beginning of the data, to the end of the data, or from record to record through the data. You can modify the information in the database by changing the value displayed in any of the bound controls. When you click a button on the data control to move to a new record, Visual Basic automatically saves any changes you have made to the data.

Of course, you can add code to enhance your application further. The rest of this chapter shows you how to use the data control with bound controls to manipulate data, examine the structure of the database, and write event procedure to handle events that occur as data is accessed or updated.

Using the data bound controls:

Data bound controls are the data-aware controls through which you access information in a database. When a control is bound to the data control. Visual Basic applies field values from the current database record to that control. In turn, the control displays data to you and accepts your changes. If you change data in a bound control, those changes can be automatically written to the database as you move to another record.

Most bound controls are characterized by three data aware properties:

Data changes, Data field, and Data source

Property	Description
Data Changed	indicates whether a value displayed in abound
Control has changed.	
Data field	Specifies the name of a field in the recordset
Created by the data control.	
Data source	specifies the name of the data control to which
The control in bound.	

The steps in adding bound controls to your application are:

1. Draw the bound control the same form as the data control to which it will be bound.
2. Set the Data Source property to specify the data control to which it will be bound.
3. Set the Data Field property to a valid field in the data control are records.

If the database is available at design time a list of valid fields will be displayed in the Data Field Settings box in the property window. If the database is not available at design time, you will need to provide a valid field name at run time before data values will be posted to the control from the database.

You can have more than one bound control for a particular field but you do not need to provide a bound control for each field in the table. Neither the data control the bound controls need to be made visible. So you can incorporate data access capabilities into any form you design, manipulating the data control behind the scenes with Visual Basic code.

When you run your application the data control works together with the database to give you access to the current set of record, or records, with which you are working.

Using the arrow buttons on the data control, you can move from record to record and using the Bound Controls, you can view or edit the data displayed from which field. Whenever you click a button on the data control, Visual Basic automatically updates any changes you have made to the record set.

With Visual Basic code, it is also possible to use data-bound controls without binding them to control, using the DataBindings collection of the bound control.

Types of Bound Controls:

Visual Basic supports several built-in (intrinsic) controls that you can bind to the Data control, as well as several Data-Bound Active X (. OCX) controls (formally called custom or OLE controls). Many other data-aware controls are available from third parties and in the professional and Enterprise editions of Visual Basic.

Standard/Intrinsic Controls:

The standard Bound Controls that you can use with the Data Control include the Check Box, Image, Label, Picture Box, Text Box, List Box, Combo Box, OLE container Control.

ABOUT MSACCESS

MS Access for the Business Environment: MS Access as a Documentation Tool:

Database Diagramming

Monday Aug 2nd 2004 by William Pearson

Share:

Create a database diagram of an MSSQL Server 2000 database within an MS Access project. In this article, we create a database diagram to meet a documentation requirement, the support of a data dictionary.

About the Series ...

This article is a member of the series MS Access for the Business Environment. The primary focus of this series is an examination of business uses for the MS Access relational database management system. For more information on the series, as well as the hardware / software requirements to prepare for the tutorials we will undertake, please see the first article of our series, Create a Calculated Field with the Expression Builder.

Note: The majority of the procedures I demonstrate in the series will be undertaken within MS Access 2003, although the concepts that we explore in this article will apply to MS Access 2002, and beyond. Along with MS Access, additional application considerations apply for this tutorial, because it focuses upon activities that are performed, within an MS Access Project file, in conjunction with MSSQL Server 2000.

For those joining the series with this article, it is assumed that MSSQL Server 2000 is accessible to / installed on your PC, with the appropriate access rights to the MSSQL Server 2000 environment to parallel the steps of the article. Service Pack 3 / 3a is also assumed. If this is the first time MSSQL Server 2000 is being accessed from your machine, you may need to consult the MSSQL Server 2000 online documentation for installation and configuration instructions.

Overview

As virtually any developer of a database knows, documentation of the structure designed to contain data is a critical tool in many recurring scenarios, such as operation and general maintenance. Users of databases, particularly those who seek to extract data from them in an effective and efficient way, are often distracted and frustrated from completing their objectives because they cannot obtain even basic documentation, such as useful data dictionaries or current database diagrams. In my work with some of the largest companies in the world, who develop / purchase and operate databases that are often boggling to the imagination in size and complexity, I am no longer surprised to find that one of the most common issues in any business intelligence, OLAP, or other development effort I undertake, is incomplete documentation. In many cases, I find myself having to generate my own diagram, and, as crazy as it sounds, usually have to explain to several onlookers why I need to undertake this exercise instead of "just getting down to business."

In my opinion, one of the primary strengths of the MS Access project (.adp) is its portable, user friendly documentation capabilities, as I will overview in this and subsequent articles. This capability is useful whether one uses MS Access as an RDBMS or not.

The lion's share of my involvement with MS Access is comprised of migrating Access databases to larger RDMS', "upsizing" the core components to MSSQL Server and other platforms, where I often graft existing table designs into new, more robust schemas. In addition to database upsizing, I have recently begun "upsizing" MS Access reports to MSSQL Server Reporting Services (see my Database Journal article "Upsize" MS Access Reports to MS Reporting Services in this series) in various client or training scenarios. Whatever the objective of the engagement, I find myself quite often in those scenarios, to which I have already alluded, where no one can produce a data dictionary

(in some cases, it is "missing"; in others, the individual with whom I am discussing it is not even aware of what it is ...).

I therefore often need to generate a quick view of the database to facilitate development or maintenance tasks, building a query / dataset within Reporting Services or another enterprise reporting tool, performing a data-quality examination, and for other activities. In times like these, and in other scenarios where a quick database diagram can be efficient and useful, the MS Access project (.adp) can be a great documentation tool. In this article, we will explore using an MS-Access project for generating a database diagram. We will:

- Establish a hypothetical business need for a database diagram;

- Create an MS Access Project (.adp file) within which to perform the practice exercise;

- Establish connectivity with an MSSQL Server database;

- Create a database diagram, complete with table joins / relationship information;

- Explain navigation and methods that we encounter throughout our practice exercise;

- Explore distribution of the diagram through MS Word and other options.

Diagram Your MS Access Database

Objective and Business Scenario

In the following section, we will perform the steps required to diagram an MSSQL Server database using an MS Access project. Along the way, we will examine the various connections and settings that we need to establish to bring this about, as well as the navigation and general use of the diagramming interface that MS Access provides within a Project. Finally, we will discuss possible delivery options for the diagram once we have completed it.

For purposes of our practice procedure, we will assume that the information consumers in our business, a book publishing concern, have asked that we assist in planning a business intelligence engagement, which will include both relational and OLAP reporting. As part of the planning evolution, we have advised that the team assemble several references for the report authors, who are due to arrive on site in a couple of days. One of the primary references we have listed is a data dictionary for the organization's main OLTP database, pubs.

Our experience has been that a database diagram is a great start for a data dictionary, in that it displays all the objects in a way that helps us to easily see the tables, their

member columns, and the relationships between those tables. This is particularly useful to a report author (something, again, I am astounded to find is not offered as a standard assumption in many of the reporting engagements I have observed in action at various client sites). As we do not have MS Visio, or any other of the tools that are often used specifically for this purpose, available at present, we reject someone's suggestion that we "draw it out," and propose the use of MS Access, part of the existing Office 2003 applications we find on the PC's where we are working. We ask for access to the client database, simply to "read" it briefly to generate our diagram.

Considerations and Comments

For purposes of this exercise, we will be using MS Access in conjunction with MSSQL Server 2000, focusing upon the pubs database in MSSQL Server. Few practitioners with any exposure to MSSQL Server will be unfamiliar with the pubs database, which contains data modeled on a hypothetical book publishing company; Pubs is a very basic sample database that is available with all versions of MSSQL Server. If you do not see pubs listed in Enterprise Manager, or for some reason know it to have been removed from your PC, it is available from the original MSSQL Server installation disk and elsewhere.

We will be accessing pubs in MSSQL Server using an MS Access project (.adp file). An MS Access project is a data file that provides efficient, native-mode access to a Microsoft SQL Server database via OLE DB, used typically to develop both traditional and Web-based client / server applications, among other things. We are using it in meeting the hypothetical client business requirement because it provides easy access (to a local SQL Server database, a remote SQL Server database, or a local installation of SQL Server 2000 Desktop engine), within a fairly ubiquitous application (MS Access) that comes equipped with a reliable diagramming tool we can master quickly, and for which we have flexible output options.

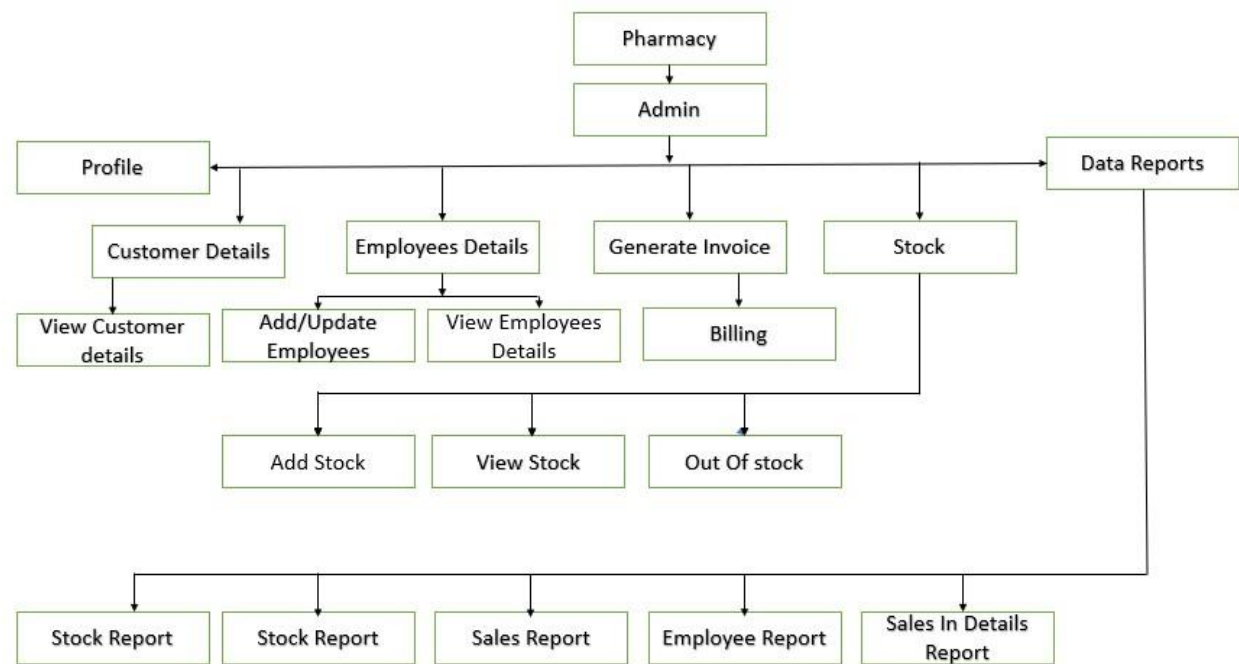
While our interaction with MSSQL Server will be quite minimal for purposes of this article, we will need the authority, access and privileges to access the pubs database, and to establish connectivity between it and MS Access.

Hands-On Procedure

An MS Access project (.adp), as we have stated, is a file that connects to an MSSQL Server database, providing an excellent platform from which to design client / server

applications. It works well in an environment where some or all development itself is taking place on a client, with the MSSQL Server database under consideration located on a server to which we can establish connectivity. An MS Access project differs from an MS Access database, because it contains database objects that are code- or HTML-based. We typically use the project objects to create an application; it does not store data or data definition based objects, such as tables, views, and so forth. In our scenario, the "real" database objects will exist in MSSQL Server, where we are really only reading them enough to generate a diagram of the database

HEIRARCHICAL DIAGRAM:



TESTING

SYSTEM TESTING:

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work Product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a

finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectation and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

_TYPES OF TESTING:

UNIT TESTING:

Unit testing involves the design of test cases that validate that The internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing f1 of individual software units of the application. It is done after the completion of an individual unit before integration.

This is a structural testing, that relies on knowledge of its construction and it invasive. Unit tests performs basic test at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specification and clearly defined inputs and expected results. Unit tests ensure that each unique path of a business process performs accurately to the documented specification and clearly defined inputs and expected results.

INTEGRATION TESTING:

Integration test are designed to test integrated software Components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specially aimed at exposing the problems that arise from the combination of components.

FUNCTIONAL TEST:

Functional test provide systematic demonstrations that functions tests are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing id centered on the following items:

Valid Input : Identified classes of valid input must be accepted.

Invalid Input : Identified classes of invalid inputs must be rejected.

Functions : Identified functions must be exercised.

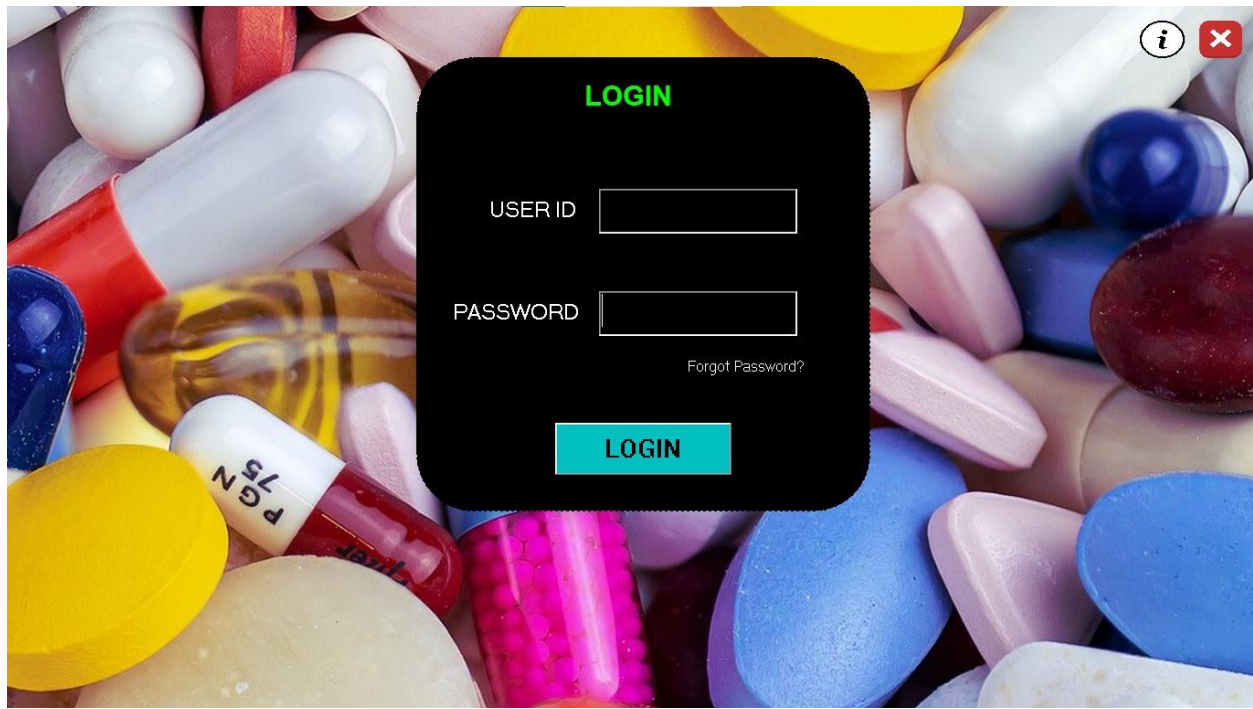
Output : Identified classes of application outputs must be exercised.

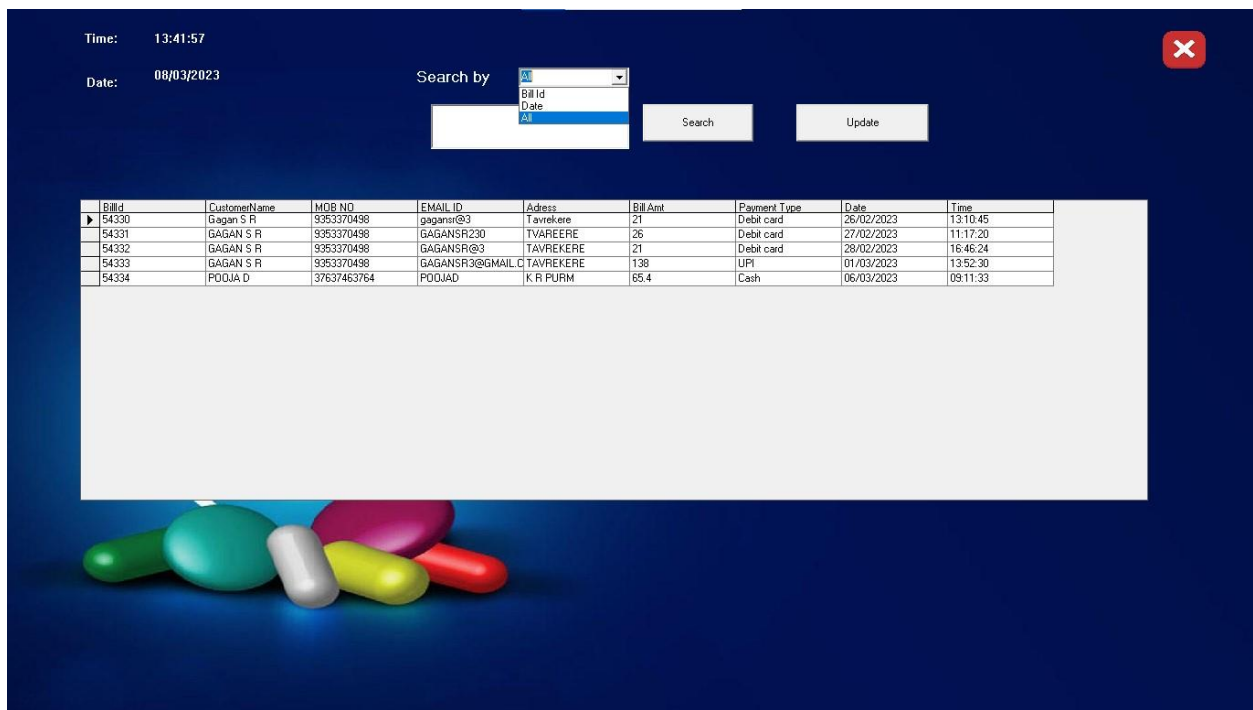
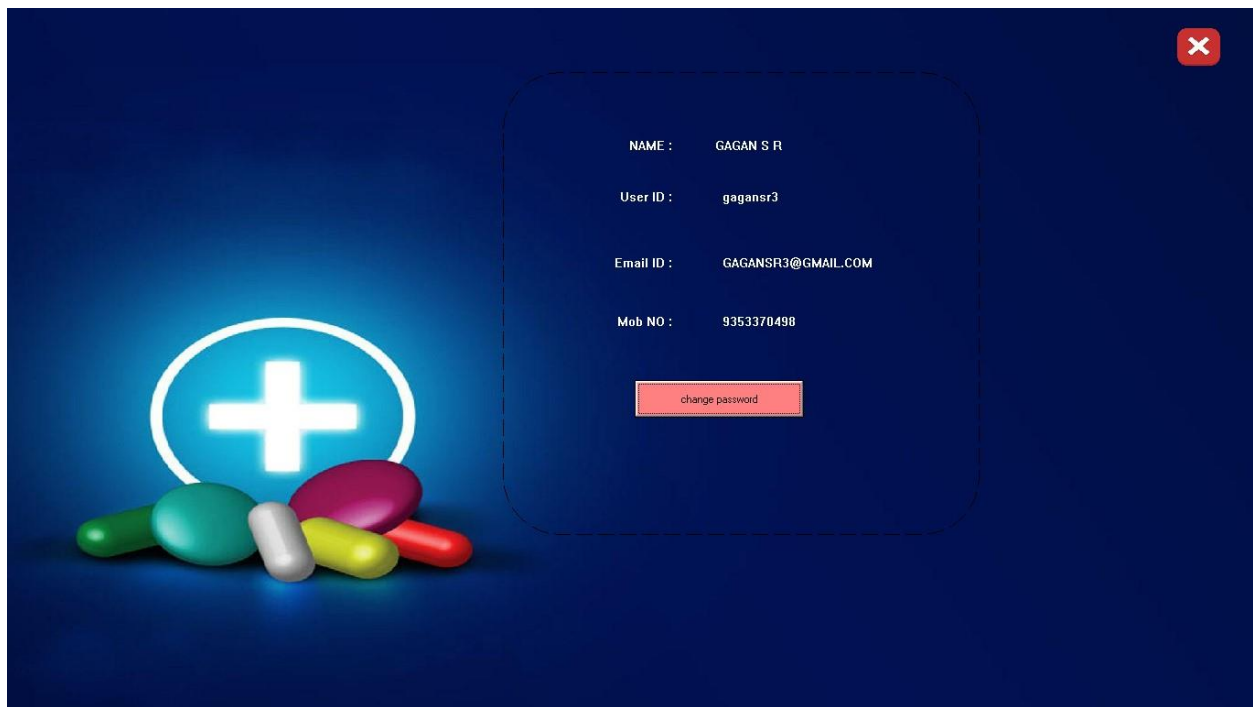
Systems Procedures: Interfacing system or procedures must be invoked.

organization and preparation of functional test is focused on Requirements, key functions, or special test cases. In addition Systematic coverage pertaining to identify Business process flows;

Data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current test is determined.

SCREEN SHOTS





Date: 00/03/2023
Time: 13:42:49

Search By

select
Name
All
Employee Id

Search

Add/Update Employees

✕

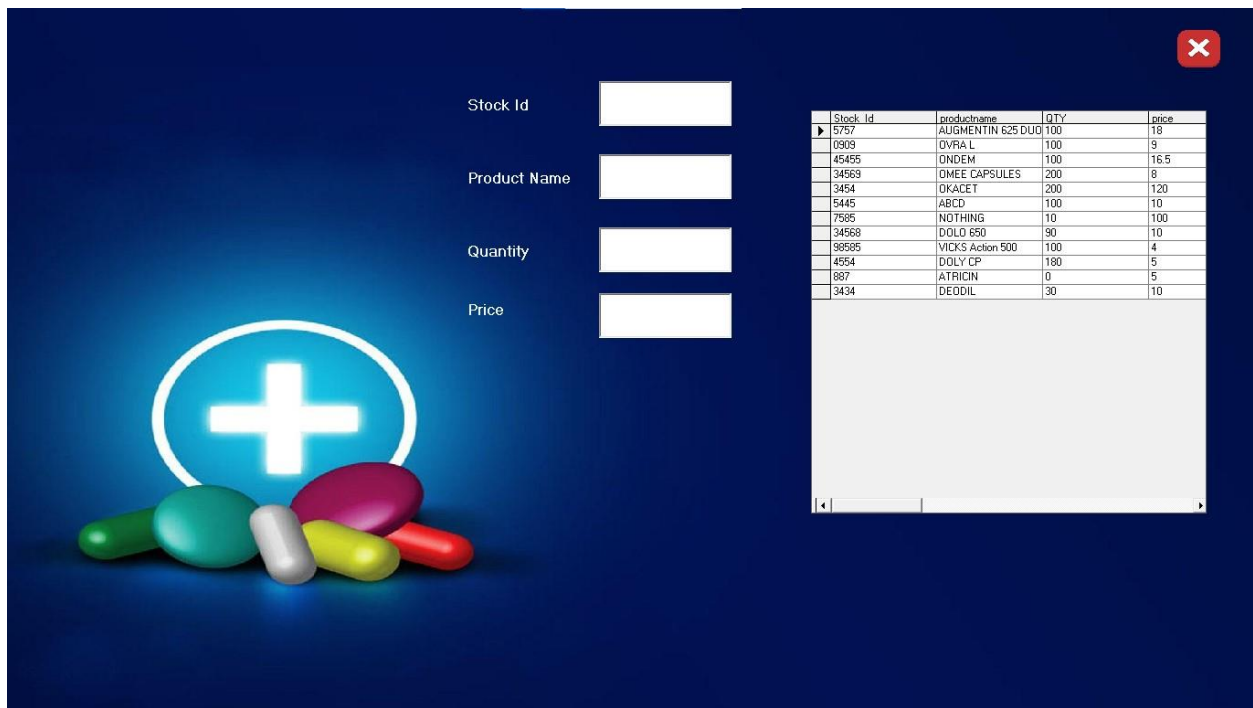
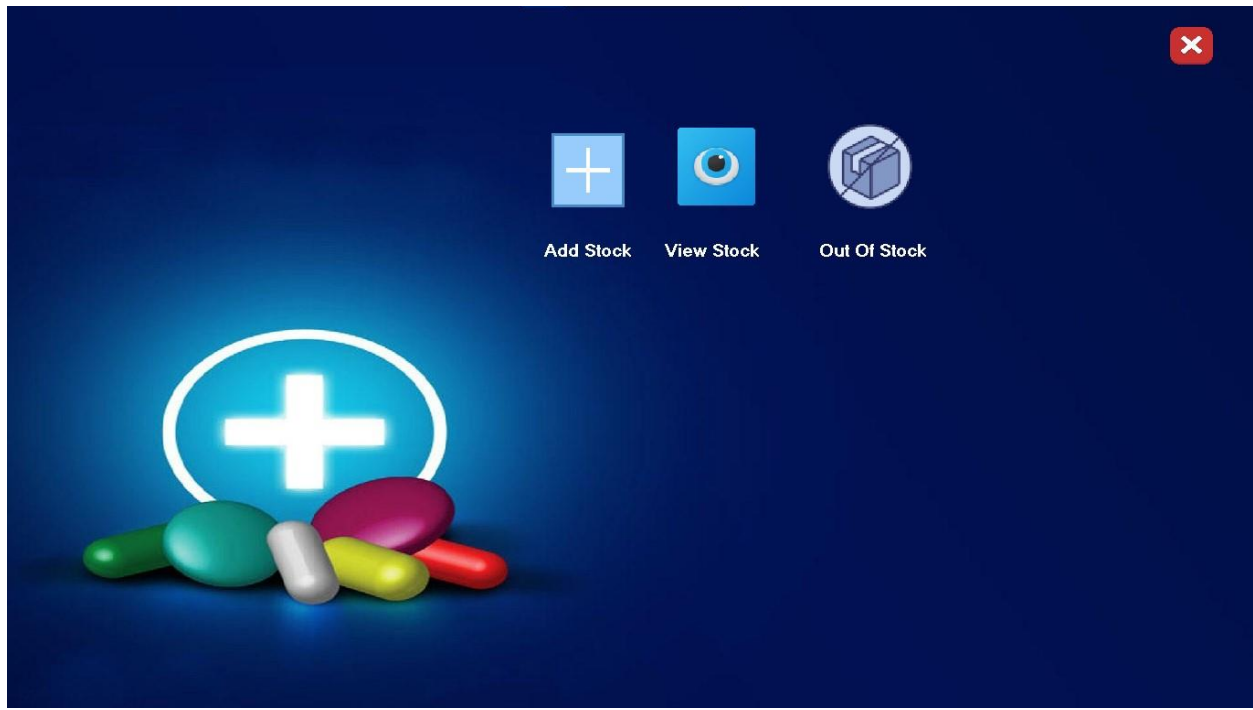
Employee Id	userid	DESIGNATION	password	Full Name	FATHERNAME	MOTHERNAME	ADDRESS	MOB NO	EMAIL	QUAL
3456	karthik19	cashier	123	Karthik Raja	venkatesh	sangeetha	Bangalore	990629857	karthikraja7081@gmail.com	BCA
3457	gaganr3	admin	123	Gagan S R	Ramakrishna	Kalpana V	Bangalore	9353370498	gaganr3@gmail.com	BCA
3458	suryak19	manager	suryak19	Surya K	Kuppaswamy	Vani	K R Puram	767676766	surya@gmail.com	BCA
3459	abc3	cashier	abc3	Abc	Abc	bc	Abc	877777	gaganr3@3	B tech
*										

Add / Update Employee


✕

Employee Id
User Id
Designation
password

Full Name
Father name
Mother Name
Adress
Mob no
Email
Qualification




Date 08/03/2023
Time 13:54:26
✕



Stock Id	productname	QTY	price
5757	AUGMENTIN 625 DUO	100	18
0909	OVRA L	100	9
45455	ONDEM	100	16.5
34563	OMEE CAPSULES	200	8
3454	OKACET	200	120
7585	NOTHING	10	100
34568	DOLD 500	90	10
36585	VICKS Action 500	100	4
4554	DOLY CP	180	5
887	ATRICIN	0	5
3434	DEODIL	30	10

Date 08/03/2023
Time 13:55:17
✕



Stock Id	productname	QTY	price
887	ATRICIN	0	5

Name
MOB
Email
Adress
Date
08/03/2023
Time
13:48:58

GSK PHARMACY

Bill No: 54335

SL no	Product Name	Rs(unit)	Qty	Amt

Total Amt: fetch ammt details

Product Name

select

Price

Quantity

Amt(incl. GST)

Bill amt

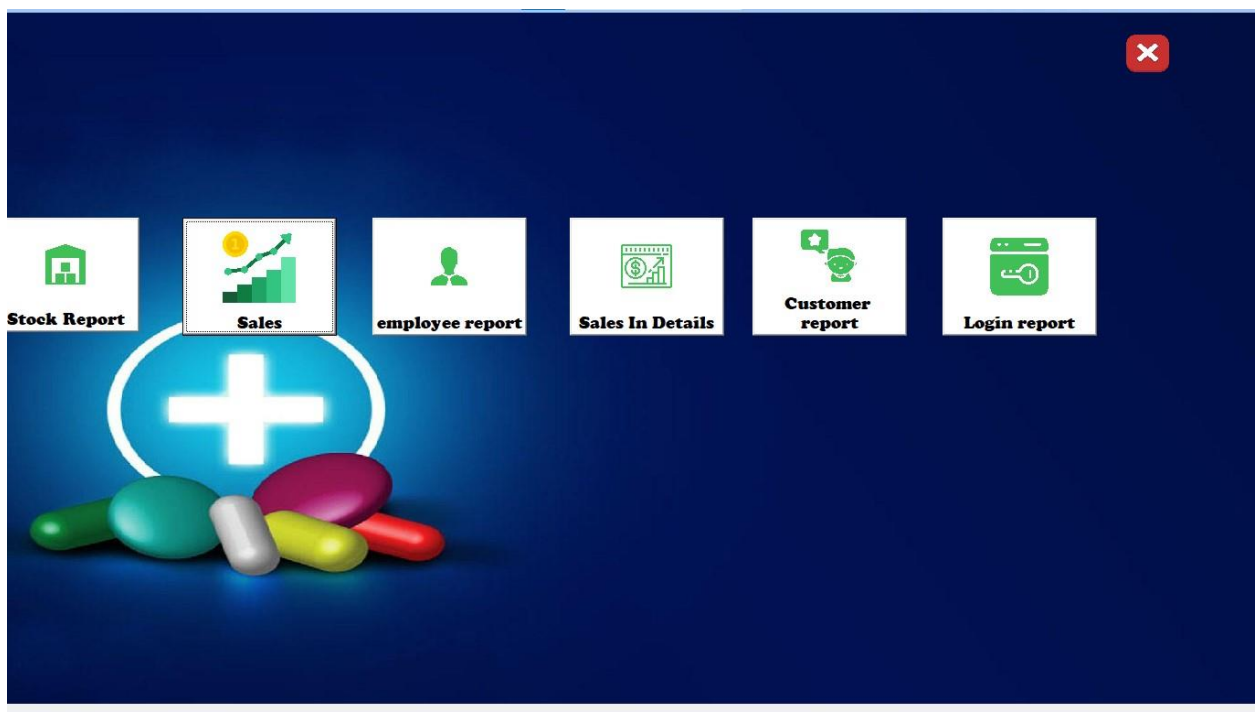
ADD

Payment Type

select

Payment

Generate new Bill



CODINGS

Splash Form:

```
Private Sub Timer1_Timer()  
If pb1.Value = 100 Then  
    login.Show  
    Timer1.Enabled = False  
    Unload Me  
Else  
    pb1.Value = pb1.Value + 1  
End If  
  
End Sub
```

Login form:

```
Dim flag As Integer  
Dim con As New ADODB.Connection  
Dim rs As New ADODB.Recordset  
Dim gs As New ADODB.Recordset  
Private Sub Command1_Click()  
    Set rs = New ADODB.Recordset  
    rs.CursorLocation = adUseClient  
    rs.Open "select * from details where userid='" + Text1.Text + "' and password='"  
    + Text2.Text + "'", con, adOpenDynamic, adLockPessimistic  
    If rs.EOF Then  
        MsgBox ("invalid credentials")  
        flag = 2
```

Else

```
MsgBox "Welcome " + rs.Fields(4) + " - " + StrConv(rs.Fields(2), vbUpperCase) + ""
```

```
flag = 1
```

```
user.Show
```

```
profile.lblname = StrConv(rs.Fields(4), vbUpperCase)
```

```
profile.lbluserid = rs.Fields(1)
```

```
profile.lblemail = StrConv(rs.Fields(9), vbUpperCase)
```

```
profile.lblmob = StrConv(rs.Fields(8), vbUpperCase)
```

```
user.Label1 = StrConv(rs.Fields(2), vbUpperCase)
```

```
If Not StrConv(rs.Fields(2), vbUpperCase) = "ADMIN" Or StrConv(rs.Fields(2), vbUpperCase) = "MANAGER" Then
```

```
user.Image4.Visible = False
```

```
user.Image4.Enabled = False
```

```
user.Label4.Visible = False
```

```
user.Label4.Enabled = False
```

```
user.Image3.Visible = False
```

```
user.Image3.Enabled = False
```

```
user.Label3.Visible = False
```

```
user.Label3.Enabled = False
```

```
End If
```

```
Timer1.Enabled = False
```

```
End If
```

```
If flag = 1 Then
```

```
a = "Successful"
```

```
Else
```

```
a = "Unsuccessful"
```

```
End If
```

```
s1 = "insert into logindata values('" + Text1.Text + "','" + Text2.Text + "','" + Label7.Caption + "','" + Label6.Caption + "','" + a + "')"
con.Execute s1
```

End Sub

Private Sub Form_Load()

Label7.Caption = Format(Date, "dd/mm/yyyy")

Timer1.Enabled = True

Set con = New ADODB.Connection

con.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data

Source=C:\Users\gagan_h9rbhos\OneDrive\Desktop\project\data.mdb"

'con.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:\Documents and Settings\Admin\My Documents\bill.mdb"

con.Open

Set rs = New ADODB.Recordset

rs.CursorLocation = adUseClient

rs.Open "select * from details", con, adOpenDynamic, adLockOptimistic

End Sub

Private Sub Image1_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)

Label4.Visible = False

End Sub

Private Sub Image2_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)

Label4.Visible = True

End Sub

Private Sub Image3_Click()

Unload Me

End Sub

```
Private Sub Label5_Click()  
Set rs = New ADODB.Recordset  
rs.CursorLocation = adUseClient  
rs.Open "select * from details where userid=" + InputBox("Enter Your USERID")  
+ "", con, adOpenDynamic, adLockPessimistic  
If rs.EOF Then  
MsgBox "invalid credentials", vbInformation  
Else  
MsgBox " Your password is " + rs.Fields(3)  
End If  
End Sub
```

```
Private Sub Timer1_Timer()  
Label6.Caption = Format(Time, "hh:mm:ss")  
End Sub
```

Master(user) form:

```
Private Sub Image2_Click()  
If Not Label1.Caption = "ADMIN" Or Label1.Caption = "MANAGER" Then  
stock.Image2.Visible = False  
stock.Image2.Enabled = False  
stock.Label6.Visible = False  
stock.Label6.Enabled = False  
End If  
stock.Show  
End Sub
```

```
Private Sub Image3_Click()  
customerdetails.Show
```

End Sub

Private Sub Image4_Click()

employees.Show

End Sub

Private Sub Image5_Click()

generateinvoice.Show

End Sub

Private Sub Image6_Click()

profile.Show

End Sub

Private Sub Image7_Click()

Unload Me

End Sub

Private Sub Image8_Click()

datareports.Show

End Sub

Profile form:

Dim con As New ADODB.Connection

Dim rs As New ADODB.Recordset

Private Sub Command1_Click()

Adodc1.RecordSource = "select * from details where userid='" +

Ibluserid.Caption + "'"

Adodc1.Refresh

If Adodc1.Recordset.EOF Then


```

MsgBox "wrong password please try again"
Elseif Adodc1.Recordset.Fields(3) = InputBox("enter the current password")
Then
MsgBox "password matched", vbInformation
Adodc1.Recordset.Fields(3) = InputBox("enter the new password")
Adodc1.Recordset.Update
MsgBox "password changed successfully, please re login", vbInformation
login.Show
Unload user
Unload Me
Else
MsgBox "wrong password please try again"
End If
End Sub

```

```

Private Sub Image3_Click()
Unload Me
End Sub

```

Customer details form:

```

Dim con As New ADODB.Connection
Dim rs As New ADODB.Recordset

```

```

Private Sub Calendar1_Click()
Text1.Text = Format(Calendar1.Value, "dd/mm/yyyy")
End Sub

```

```

Private Sub Command1_Click()
Set rs = New ADODB.Recordset
rs.CursorLocation = adUseClient

```

```
If Combo1.Text = "Bill Id" Then
rs.Open "select * from customerpurchase where BillId='" + Text1.Text + "'", con,
adOpenDynamic, adLockPessimistic
If rs.EOF Then
MsgBox " Not found", vbInformation
Else
Set DataGrid1.DataSource = rs
End If
ElseIf Combo1.Text = "Date" Then
rs.Open "select * from customerpurchase where Date='" + Text1.Text + "'", con,
adOpenDynamic, adLockPessimistic
If rs.EOF Then
MsgBox " Not found", vbInformation
Else
Set DataGrid1.DataSource = rs
End If
ElseIf Combo1.Text = "All" Then
rs.Open "select * from customerpurchase", con, adOpenDynamic,
adLockPessimistic
Set DataGrid1.DataSource = rs
Else
MsgBox " please select the option"
End If
End Sub
Private Sub Form_Load()
Label5.Caption = Format(Date, "dd/mm/yyyy")
If Not user.Label1.Caption = "ADMIN" Or user.Label1.Caption = "MANAGER"
Then
DataGrid1.AllowUpdate = False
Else
DataGrid1.AllowUpdate = True
```

End If

Set con = New ADODB.Connection

con.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=C:\Users\gagan_h9rbhos\OneDrive\Desktop\project\data.mdb"

con.Open

Set rs = New ADODB.Recordset

rs.CursorLocation = adUseClient

rs.Open "select * from customerpurchase", con, adOpenDynamic,
adLockPessimistic

Set DataGrid1.DataSource = rs

End Sub

Private Sub Image3_Click()

Unload Me

End Sub

Private Sub Text1_GotFocus()

If Combo1.Text = "Date" Then

Calendar1.Visible = True

End If

End Sub

Private Sub Text1_LostFocus()

Calendar1.Visible = False

End Sub

Private Sub Timer1_Timer()

Label3.Caption = Format(Time, "hh:mm:ss")

End Sub

Employee Details form:

```
Dim con As New ADODB.Connection
```

```
Dim rs As New ADODB.Recordset
```

```
Private Sub Command1_Click()
```

```
Set rs = New ADODB.Recordset
```

```
rs.CursorLocation = adUseClient
```

```
If Combo1.Text = "Name" Then
```

```
rs.Open "select * from details where Full_Name='" + Text1.Text + "'", con,  
adOpenDynamic, adLockPessimistic
```

```
    If rs.EOF Then
```

```
        MsgBox "Not found"
```

```
    Else
```

```
        Set DataGrid1.DataSource = rs
```

```
    End If
```

```
ElseIf Combo1.Text = "Employee Id" Then
```

```
rs.Open "select * from details where Employee_Id='" + Text1.Text + "'", con,  
adOpenDynamic, adLockPessimistic
```

```
    If rs.EOF Then
```

```
        MsgBox "Not found"
```

```
    Else
```

```
        Set DataGrid1.DataSource = rs
```

```
    End If
```

```
ElseIf Combo1.Text = "All" Then
```

```
rs.Open "select * from details", con, adOpenDynamic, adLockPessimistic  
Set DataGrid1.DataSource = rs
```

```
Else
```

```
MsgBox " please select the option"
```

```
End If
```

```
End Sub
```

```
Private Sub Command3_Click()  
addemployees.Show  
Unload Me  
End Sub
```

```
Private Sub Form_Load()  
Label4.Caption = Format(Date, "dd/mm/yyyy")  
Set con = New ADODB.Connection  
con.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data  
Source=C:\Users\gagan_h9rbhos\OneDrive\Desktop\project\data.mdb"  
con.Open  
Set rs = New ADODB.Recordset  
rs.CursorLocation = adUseClient  
rs.Open "select * from details", con, adOpenDynamic, adLockPessimistic  
Set DataGrid1.DataSource = rs  
End Sub
```

```
Private Sub Image3_Click()  
Unload Me  
End Sub
```

```
Private Sub Timer1_Timer()  
Label5.Caption = Format(Time, "hh:mm:ss")  
End Sub
```

Add / update Employees form:

```
Dim con As New ADODB.Connection  
Dim rs As New ADODB.Recordset
```

```
Private Sub Command1_Click()
```

```
If Text1.Text = "" Or Text2.Text = "" Or Text3.Text = "" Or Text4.Text = "" Or  
Text5.Text = "" Or Text6.Text = "" Or Text7.Text = "" Or Text8.Text = "" Or  
Text9.Text = "" Or Text10.Text = "" Or Text11.Text = "" Then
```

```
MsgBox " please fill all the details"
```

```
Else
```

```
Set rs = New ADODB.Recordset
```

```
rs.CursorLocation = adUseClient
```

```
rs.Open "select * from details", con, adOpenDynamic, adLockPessimistic
```

```
s1 = "insert into details values(" + Text1.Text + "," + Text2.Text + "," +  
Text3.Text + "," + Text4.Text + "," + Text5.Text + "," + Text6.Text + "," +  
Text7.Text + "," + Text8.Text + "," + Text9.Text + "," + Text10.Text + "," +  
Text11.Text + ")"
```

```
con.Execute s1
```

```
MsgBox "record has been added", vbCritical
```

```
Unload Me
```

```
employees.Show
```

```
End If
```

```
End Sub
```

```
Private Sub Command2_Click()
```

```
If Text1.Text = "" Or Text2.Text = "" Or Text3.Text = "" Or Text4.Text = "" Or  
Text5.Text = "" Or Text6.Text = "" Or Text7.Text = "" Or Text8.Text = "" Or  
Text9.Text = "" Or Text10.Text = "" Or Text11.Text = "" Then
```

```
MsgBox " Fill All the details"
```

```
Else
```

```
Adodc1.Recordset.Fields(1) = Text2.Text
```

```
Adodc1.Recordset.Fields(2) = Text3.Text
```

```
Adodc1.Recordset.Fields(3) = Text4.Text
```

```
Adodc1.Recordset.Update
```

```
Unload Me
```

```
employees.Show
```

MsgBox " data Updated Successfully"

End If

End Sub

Private Sub Form_Load()

Set con = New ADODB.Connection

con.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=C:\Users\gagan_h9rbhos\OneDrive\Desktop\project\data.mdb"

'con.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:\Documents and
Settings\Admin\My Documents\bill.mdb"

con.Open

End Sub

Private Sub Image3_Click()

employees.Show

Unload Me

End Sub

Private Sub Text1_GotFocus()

Command1.Enabled = False

Command1.Visible = False

Command2.Enabled = False

Command2.Visible = False

End Sub

Private Sub Text1_KeyPress(KeyAscii As Integer)

If (KeyAscii >= 65) Then

KeyAscii = 0

MsgBox "YOU CAN ONLY ENTER NUMBERS", vbExclamation, "Error"

End If

End Sub

```
Private Sub Text1_LostFocus()  
Adodc1.RecordSource = "select * from details where Employee_Id=" +  
Text1.Text + ""  
Adodc1.Refresh  
If Adodc1.Recordset.EOF Then  
Command1.Enabled = True  
Command1.Visible = True  
Text2.Text = ""  
Text3.Text = ""  
Text4.Text = ""  
Text5.Text = ""  
Text6.Text = ""  
Text7.Text = ""  
Text8.Text = ""  
Text9.Text = ""  
Text10.Text = ""  
Text11.Text = ""  
Else  
Command2.Enabled = True  
Command2.Visible = True  
Text2.Text = Adodc1.Recordset.Fields(1)  
Text3.Text = Adodc1.Recordset.Fields(2)  
Text4.Text = Adodc1.Recordset.Fields(3)  
Text5.Text = Adodc1.Recordset.Fields(4)  
Text6.Text = Adodc1.Recordset.Fields(5)  
Text7.Text = Adodc1.Recordset.Fields(6)  
Text8.Text = Adodc1.Recordset.Fields(7)  
Text9.Text = Adodc1.Recordset.Fields(8)  
Text10.Text = Adodc1.Recordset.Fields(9)  
Text11.Text = Adodc1.Recordset.Fields(10)
```


End If

End Sub

Private Sub Text9_KeyPress(KeyAscii As Integer)

If (KeyAscii >= 65) Then

KeyAscii = 0

MsgBox "YOU CAN ONLY ENTER NUMBERS", vbExclamation, "Error"

End If

End Sub

Generate Invoice form:

Dim num, bill, temp As Integer

Dim con As New ADODB.Connection

Dim rs As New ADODB.Recordset

Private Sub cmb1_LostFocus()

Set rs = New ADODB.Recordset

rs.Open " select * from stock1 where productname ='" & cmb1.Text & "'", con

If rs.EOF Then

MsgBox "please select the product"

Else

Text6.Text = rs.Fields(3)

End If

End Sub

Private Sub Command1_Click()

Adodc1.RecordSource = "select * from stock1 where productname='" +
cmb1.Text + "'"

Adodc1.Refresh

temp = Val(Adodc1.Recordset.Fields(2))

If Text1.Text = "" Or Text2.Text = "" Or Text3.Text = "" Or Text4.Text = "" Then

MsgBox " please Fill all the details"

```
Elseif Text6.Text = "" Then
    MsgBox "please select the product", vbInformation
Elseif Text7.Text = "" Then
    MsgBox "please input the qty", vbInformation
Elseif Val(Text7.Text) <= temp Then
    num = num + 1
    List1.AddItem cmb1.Text
    List2.AddItem Text6.Text
    List3.AddItem Text7.Text
    List4.AddItem Label25.Caption
    List5.AddItem num
    Text5.Text = num
    bill = bill + Val(Label25.Caption)
    Label16.Caption = bill
    Adodc1.Recordset.Fields(2) = temp - Val(Text7)
    Adodc1.Recordset.Update
    Set rs = New ADODB.Recordset
    rs.Open "select * from billingtemp ", con, adOpenDynamic, adLockPessimistic
    s1 = "insert into billingtemp values('" + Text5.Text + "','" + StrConv(Text1.Text,
    vbUpperCase) + "','" + cmb1.Text + "','" + Text6.Text + "','" + Text7.Text + "','" +
    Label25.Caption + "')"
    con.Execute s1
Else
    MsgBox "the selected product has " + Adodc1.Recordset.Fields(2) + " Quantity
    in the stock"
End If
End Sub

Private Sub Command2_Click()
    Set rs = New ADODB.Recordset
    rs.Open "customerpurchase", con, adOpenDynamic, adLockPessimistic
```

```
s1 = "insert into customerpurchase values('" + Label23.Caption + "','" +  
StrConv(Text1.Text, vbUpperCase) + "','" + Text2.Text + "','" +  
StrConv(Text3.Text, vbUpperCase) + "','" + StrConv(Text4.Text, vbUpperCase) +  
 "','" + Label16.Caption + "','" + Combo1.Text + "','" + Label19.Caption + "','" +  
Label18.Caption + "')"
con.Execute s1
MsgBox " Sold Successfully"
End Sub
```

```
Private Sub Command3_Click()
Text1.Text = ""
Text2.Text = ""
Text3.Text = ""
Text4.Text = ""
Text6.Text = ""
Text7.Text = ""
Label25.Caption = ""
Combo1.Text = ""
cmb1.Text = ""
num = 0
bill = 0
Label16.Caption = ""
Label23.Caption = Val(Label23.Caption) + 1
List1.Clear
List2.Clear
List3.Clear
List4.Clear
List5.Clear
End Sub

Private Sub Form_Load()
Set con = New ADODB.Connection
```

```
con.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=C:\Users\lgagan_h9rbhos\OneDrive\Desktop\project\data.mdb"
'con.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:\Documents and
Settings\Admin\My Documents\bill.mdb"
con.Open
Set rs = New ADODB.Recordset
rs.CursorLocation = adUseClient
rs.Open "select * from stock1", con, adOpenDynamic, adLockOptimistic
While Not rs.EOF
cmb1.AddItem rs.Fields(1)
rs.MoveNext
Wend
Label19.Caption = Format(Date, "dd/mm/yyyy")
Set rs = New ADODB.Recordset
rs.Open "select * from customerpurchase ", con, adOpenDynamic,
adLockPessimistic
rs.MoveLast
Label24.Caption = rs.Fields(0)
Label23.Caption = Val(Label24.Caption) + 1
End Sub

Private Sub Image3_Click()
Unload Me
End Sub

Private Sub Text2_KeyPress(KeyAscii As Integer)
If (KeyAscii >= 65) Then
KeyAscii = 0
MsgBox "YOU CAN ONLY ENTER NUMBERS", vbExclamation, "Error"
End If
End Sub
```

```
Private Sub Text6_KeyPress(KeyAscii As Integer)
If (KeyAscii >= 65) Then
KeyAscii = 0
MsgBox "YOU CAN ONLY ENTER NUMBERS", vbExclamation, "Error"
End If
End Sub
```

```
Private Sub Text7_Change()
Label25.Caption = Val(Text6.Text) + Val(Text7.Text) + (Val(Text6.Text) +
Val(Text7.Text) * (18 / 100))
End Sub
```

```
Private Sub Text7_KeyPress(KeyAscii As Integer)
If (KeyAscii >= 65) Then
KeyAscii = 0
MsgBox "YOU CAN ONLY ENTER NUMBERS", vbExclamation, "Error"
End If
End Sub
```

```
Private Sub Timer1_Timer()
Label18.Caption = Format(Time, "HH:MM:SS")
End Sub
```

Stock form:

```
Private Sub Image2_Click()
addstock.Show
End Sub
```

```
Private Sub Image3_Click()
```

```
viewstock.Show
```

```
End Sub
```

```
Private Sub Image4_Click()
```

```
outofstock.Show
```

```
End Sub
```

```
Private Sub Image7_Click()
```

```
Unload Me
```

```
End Sub
```

Add Stock Form:

```
Dim con As ADODB.Connection
```

```
Dim rs As ADODB.Recordset
```

```
Private Sub Command1_Click()
```

```
If Text1.Text = "" Or Text2.Text = "" Or Text3.Text = "" Or Text4.Text = "" Then
```

```
MsgBox " Fill All the details"
```

```
Else
```

```
Set rs = New ADODB.Recordset
```

```
rs.CursorLocation = adUseClient
```

```
rs.Open "select * from stock1", con, adOpenDynamic, adLockPessimistic
```

```
s1 = "insert into stock1 values(" + Text1.Text + "," + StrConv(Text2.Text,  
vbUpperCase) + "," + Text3.Text + "," + Text4.Text + ")"
```

```
con.Execute s1
```

```
Text1.Text = ""
```

```
Text2.Text = ""
```

```
Text3.Text = ""
```

```
Text4.Text = ""
```

Unload Me

addstock.Show

MsgBox "Stock Has been Added Scuccessfully", vbOKOnly

End If

End Sub

Private Sub Command2_Click()

Text1.Text = ""

Text2.Text = ""

Text3.Text = ""

Text4.Text = ""

End Sub

Private Sub Command3_Click()

If Text1.Text = "" Or Text2.Text = "" Or Text3.Text = "" Or Text4.Text = "" Then

MsgBox " Fill All the details"

Else

Adodc1.Recordset.Fields(1) = Text2.Text

Adodc1.Recordset.Fields(2) = Text3.Text

Adodc1.Recordset.Fields(3) = Text4.Text

Adodc1.Recordset.Update

Unload Me

addstock.Show

MsgBox "Data Updated Successfully"

End If

End Sub

Private Sub Form_Load()

Set con = New ADODB.Connection

con.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data

Source=C:\Users\gagan_h9rbhos\OneDrive\Desktop\project\data.mdb"

```
'con.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:\Documents and  
Settings\Admin\My Documents\bill.mdb"
```

```
con.Open
```

```
Set rs = New ADODB.Recordset
```

```
rs.CursorLocation = adUseClient
```

```
rs.Open "select * from stock1", con, adOpenDynamic, adLockPessimistic
```

```
Set DataGrid1.DataSource = rs
```

```
End Sub
```

```
Private Sub Image3_Click()
```

```
Unload Me
```

```
End Sub
```

```
Private Sub Text1_GotFocus()
```

```
Command1.Enabled = False
```

```
Command1.Visible = False
```

```
Command3.Enabled = False
```

```
Command3.Visible = False
```

```
End Sub
```

```
Private Sub Text1_KeyPress(KeyAscii As Integer)
```

```
If (KeyAscii >= 65) Then
```

```
KeyAscii = 0
```

```
MsgBox "YOU CAN ONLY ENTER NUMBERS", vbExclamation, "Error"
```

```
End If
```

```
End Sub
```

```
Private Sub Text1_LostFocus()
```



```
Adodc1.RecordSource = "select * from stock1 where Stock_Id='" + Text1.Text +  
""
```

```
Adodc1.Refresh
```

```
If Adodc1.Recordset.EOF Then
```

```
Command1.Enabled = True
```

```
Command1.Visible = True
```

```
Text2.Text = ""
```

```
Text3.Text = ""
```

```
Text4.Text = ""
```

```
Else
```

```
Command3.Enabled = True
```

```
Command3.Visible = True
```

```
Text2.Text = Adodc1.Recordset.Fields(1)
```

```
Text4.Text = Adodc1.Recordset.Fields(3)
```

```
Text3.Text = Adodc1.Recordset.Fields(2)
```

```
End If
```

```
End Sub
```

```
Private Sub Text3_KeyPress(KeyAscii As Integer)
```

```
If (KeyAscii >= 65) Then
```

```
KeyAscii = 0
```

```
MsgBox "YOU CAN ONLY ENTER NUMBERS", vbExclamation, "Error"
```

```
End If
```

```
End Sub
```

```
Private Sub Text4_KeyPress(KeyAscii As Integer)
```

```
If (KeyAscii >= 65) Then
```

```
KeyAscii = 0
```

```
MsgBox "YOU CAN ONLY ENTER NUMBERS", vbExclamation, "Error"
```

```
End If
```

End Sub

View Stock form :

Dim con As New ADODB.Connection

Dim rs As New ADODB.Recordset

Private Sub Form_Load()

Label2.Caption = Format(Date, "dd/mm/yyyy")

Set con = New ADODB.Connection

con.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data

Source=C:\Users\gagan_h9rbhos\OneDrive\Desktop\project\data.mdb"

'con.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:\Documents and Settings\Admin\My Documents\bill.mdb"

con.Open

Set rs = New ADODB.Recordset

rs.CursorLocation = adUseClient

rs.Open "select * from stock1", con, adOpenDynamic, adLockOptimistic

Set DataGrid1.DataSource = rs

End Sub

Private Sub Image7_Click()

Unload Me

End Sub

Private Sub Timer1_Timer()

Label4.Caption = Format(Time, "HH:MM:SS")

End Sub

Out Of Stock Form:

Dim con As New ADODB.Connection

Dim rs As New ADODB.Recordset

```
Private Sub Form_Load()  
Label2.Caption = Format(Date, "dd/mm/yyyy")  
Set con = New ADODB.Connection  
con.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data  
Source=C:\Users\gagan_h9rbhos\OneDrive\Desktop\project\data.mdb"  
'con.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:\Documents and  
Settings\Admin\My Documents\bill.mdb"  
con.Open  
Set rs = New ADODB.Recordset  
rs.CursorLocation = adUseClient  
Label5.Caption = 0  
rs.Open "select * from stock1 where QTY='" + Label5.Caption + "'", con,  
adOpenDynamic, adLockOptimistic  
Set DataGrid1.DataSource = rs  
End Sub
```

```
Private Sub Image7_Click()  
Unload Me  
End Sub
```

```
Private Sub Timer1_Timer()  
Label4.Caption = Format(Time, "HH:MM:SS")  
End Sub
```

Data Reports Form:

```
Private Sub Command1_Click()  
DataReport1.Show  
End Sub
```

```
Private Sub Command2_Click()
```

DataReport2.Show

End Sub

Private Sub Command3_Click()

DataReport3.Show

End Sub

Private Sub Command4_Click()

DataReport4.Show

End Sub

Private Sub Image3_Click()

Unload Me

End Sub

Conclusion



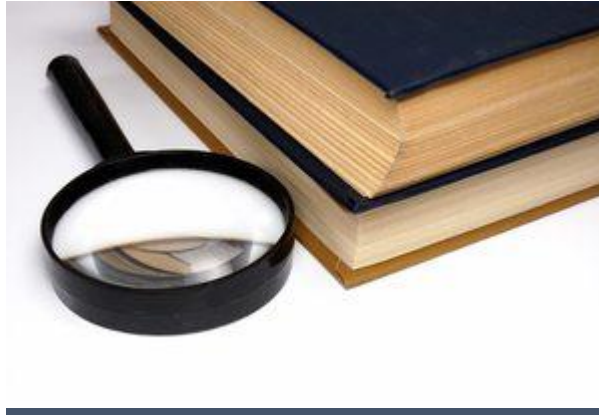
The implementation of the project using MSAccess as backend and Microsoft visual basic 6.0 as front end gave us the idea of how a database has to be organized and how it has to be maintained.

Since the implementation of a database application by us was related to **“PHARMACY MANAGEMENT SYSTEM”** We could know how real world constraints have to be dealt with and how, any problems that arose could be solved with the software.

We also learnt how to maintain a database by choosing this topic of interest by us and we have tried out level best in giving out a good application. One primary aspect we like to say here is we could not develop this application with all the features embedded in it due to time constraints and hence we tried to develop this application in the field of topics that are being left over.

And last but not the least we thank all our teaching and non- teaching staff, all over friends and our parents for giving us their co-operation and invaluable guidance at every phase of our project without which this application development could not be possible.

Bibilography



List of References: -

- ☐ Visual Basic Black Book by Stephen Holzner
- ☐ Programming In Visual Basic 6.0 by Julia Bradley & Anita Millspaugh.
- ☐ System Analysis and Design in Changing World by Satzinger.
- ☐ Software Engineering – A Practitioners Approach 7/e, by Roger S. Pressman.
- ☐ System Analysis and Design.
- ☐ Mastering VB 6.0

Web Reference

- ☐ www.wikipedia.org
- ☐ www.vbcode.com
- ☐ www.codeguru.com/vb
- ☐ www.vb.com
- ☐ www.oop.com

