**SIMPLE ONLINE BANKING SYSTEM**

**INTRODUCTION**

During the past several decades’ personnel function has been transformed from a relatively obscure record keeping staff to central and top level management function. There are many factors that have influenced this transformation like technological advances, professionalism, and general recognition of human beings as most important resources.

A computer based management system is designed to handle all the primary information required to calculate monthly statements of customer account which include monthly statement of any month. Separate database is maintained to handle all the details required for the correct statement calculation and generation.

This project intends to introduce more user friendliness in the various activities such as record updation, maintenance, and searching. The searching of record has been made quite simple as all the details of the customer can be obtained by simply keying in the identification or account number of that customer. Similarly, record maintenance and updation can also be accomplished by using the account number with all the details being automatically generated. These details are also being promptly automatically updated in the master file thus keeping the record absolutely up-to-date.

The entire information has maintained in the database or Files and whoever wants to retrieve can’t retrieve, only authorization user can retrieve the necessary information which can be easily be accessible from the file.

**OBJECTIVE OF THE PROJECT**

A computer based management system is designed to handle all the primary information required to create customer account holder name, Account number, Make Deposit, Widthdraw, Check Balance . Separate database is maintained to handle all the details required for the correct store of information and query request.

This project intends to introduce more user friendliness in the various activities such as record updation, maintenance, and searching. The searching of record has been made quite simple as all the details of the customer can be obtained by simply keying in the identification or account number of that customer. Similarly, record maintenance and updation can also be accomplished by using the account number with all the details being automatically generated. These details are also being promptly automatically updated in the master file thus keeping the record absolutely up-to-date.

The main objective of our project is providing the different typed of customers facility, the main objective of this system is to find out the actual customer service. Etc.

         IT SHOULD FULFILL ALMOST ALL THE PROCESS REQUIREMENTS OF ANY BANK.

* It should increase the productivity of bank by utilizing the working  hours more and more, with minimum manpower.

This project includes the entire upgraded feature required for the computerization banking system. This system is very easy to use, so that any user can use without getting pre-knowledge about this. Its very much user friendly and meet almost all daily working process requirements. This system is completely GUI based and can be use by mouse and as well as keyboard. This system is melded in such a way that has got all features to upgrade without making much change in existing components.

**FEASIBILITY ANALYSIS:-**

             Depending on the results of the initial investigation, the survey is expanded to a more detailed feasibility study. A feasibility study is a test of a system proposal. According to its workability, impact on the organization, ability to meet user’s needs and effective use of the resources its main task done during the feasibility study are:-

1.      Evaluation of existing system and procedures. Our group went to various Banking Professionals to gather information about the software system. They are using and evaluating those system and the procedures invoked in it during the period of feasibility study.

2.      Analysis of alternative candidate systems after studying the various systems we derived various alternatives through which we develop our project and evaluated the alternative. The most appropriate is selected.

**FEASIBILITY STUDY**

The only tangible benefit provided by the proposed system is that the paper work is reduced to the minimum and hence the reduction in cost incurred on Stationary and its storage. The system provides many benefits that can’t be measured in terms of Money for e.g. user’s friendliness, more user response being more efficient.

  TECHNICAL FEASIBILITY:-

 The proposed system is technically feasible as it can be developed easily with the help of available technology. The proposed system requires MS – VISUAL Studio 2005 using VB.Net as a Interface for Programming & back-end as MS-SQL Server 2000 for storing/maintaining database. The database can be easily interconnected using MS-SQL Server 2000.

  OPERATIONAL FEASIBILITY:-

Automation makes our life easy. The proposed system is highly user friendly and is much easily able to interact with the system. Therefore the users will readily accept the system as data entry and making queries can be easily done.

**SYSYTEM REQUIREMENTS**

Hardware specifications

Hardware is a set of physical components, which performs the functions of applying appropriate, predefined instructions. In other words, one can say that electronic and mechanical parts of computer constitute hardware.

This package is designed on a powerful programming language Visual Basic. It is a powerful Graphical User Interface. The backend is ACCESS, which is used to maintain database. It can run on almost all the popular microcomputers. The following are the minimum hardware specifications to run this package: -

Personal Computer: -

   It minimum contains P-III

  Processor with 128 MB RAM

Software Requirements**:**

The software is a set of procedures of coded information or a program which when fed into the computer hardware, enables the computer to perform the various tasks. Software is like a current inside the wire, which cannot be seen but its effect can be felt.

1. Operating System:- Windows NT / 2000 / XP/Windows 8-11 Version

2. Application Software:- Application software uses front end Java script and database access such MySQL

**SOFTWARE DEVELOPMENT LIFE CYCLE**

A system development life cycle is a logical process by which system analysts, software engineers, programmers, and end users build information systems and computer applications to solve business problems and needs.

The major phases involved in the MIS development process are referred to as system development life cycle. Each phase of the development process must have well defined objectives ,and at the end of each phase ,progress towards meeting the objectives must be evaluated.

The development process should not continue until the objectives of all prior phases have been met.

System development life cycle is a phased approach to analysis and design to ensure that systems are best developed.

The system development life cycle can be divided into seven phases as shown in fig

**INTRODUCTION TO FRONT END TOOL**

A Java web application is a collection of dynamic resources (such as Servlets, JavaServer Pages, Java classes and jars) and static resources (HTML pages and pictures). A Java web application can be deployed as a WAR (Web ARchive) file.

A WAR file is a zip file which contains the complete content of the corresponding web application.

**NEED FOR JAVA PROGRAMMING:-**

  Technology is rapidly evolving across the globe. A new software language or framework is developed in every two years to support disruptive business innovation. Popular Java frameworks jQuery,[AngularJS](https://www.kelltontech.com/angularjs-development), and Backbone.js has completely changed the landscape of [applications development in Java](https://www.kelltontech.com/java-development-services); NoSQL databases and JavaScript has revolutionized data storage and structuring methods.

Companies need not only adopt emerging technologies, they can also improve the existing technologies in use. For instance, Java 8 includes several features that have existed for long in the form of other languages.

Java is considered to be a "safe" software platform unless the JVM is hampered by a third-party application developer.  Following factors are the major reasons why the banking industry considers Java to be safe.

**ADVANTAGES of using javascript PROGRAMMING:-**

* Java is Simple
* Any language can be considered as simple if it is easy to learn and understand. The syntax of Java is straightforward, easy to write, learn, maintain, and understand, the code is easily debugable.
* Moreover, Java is less complex than the languages like C and C++, because many of the complex features of these languages are being removed from Java such as explicit pointers concept, storage classes, operator overloading, and many more.
* Java is an Object-Oriented Programming language
* Java is an object-oriented language that helps us to enhance the flexibility and reusability of the code. Using the OOPs concept, we can easily reuse the object in other programs.
* It also helps us to increase security by binding the data and functions into a single unit and not letting it be accessed by the outside world. It also helps to organize the bigger modules into smaller ones so they are easy to understand.
* Java is a secure language
* Java reduces security threats and risks by avoiding the use of explicit pointers. A pointer stores the memory address of another value that can cause unauthorized access to memory.
* This issue is resolved by removing the concept of pointers. Also, there is a Security manager in Java for each application that allows us to define the access rules for classes.
* Java is cheap and economical to maintain
* Java programs are cheap to develop and maintain as these programs are dependent on a specific hardware infrastructure to run. We can easily execute them on any machine that reduces the extra cost to maintain.
* Java is platform-independent
* Java offers a very effective boon to its users by providing the feature of platform independence that is Write Once Run Anywhere(WORA) feature.
* The compiled code, i.e the byte code of java is platform-independent and can run on any machine irrespective of the operating system. We can run this code on any machine that supports the Java Virtual Machine(JVM) as shown in the figure below

**INTRODUCTION TO BACK END TOOL**

**Introduction to SQL: -**

SQL is a standard computer language for accessing and manipulating databases.

         SQL stands for **S**tructured **Q**uery **L**anguage.

         SQL allows you to access a database.

         SQL is an ANSI standard computer language.

         SQL can execute queries against a database.

         SQL can retrieve data from a database.

         SQL can insert new records in a database.

         SQL can delete records from a database.

         SQL can update records in a database.

         SQL is easy to learn.

  SQL is an ANSI (American National Standards Institute) standard computer language for accessing and manipulating database systems. SQL statements are used to retrieve and update data in a database. SQL works with database programs like MS Access, DB2, Informix, MS SQL Server, Oracle, Sybase, etc.

  Unfortunately, there are many different versions of the SQL language, but to be in compliance with the ANSI standard; they must support the same major keywords in a similar manner (such as SELECT, UPDATE, DELETE, INSERT, WHERE, and others).

  **SQL DATABASE TABLES: -**

A database most often contains one or more tables. Each table is identified by a name (e.g. "Customers" or "Orders"). Tables contain records (rows) with data.

**Below is an example of a table called "Persons": -**

|  |  |  |  |
| --- | --- | --- | --- |
| **Last Name** | **First Name** | **Address** | **City** |
| Hansen | Ola | Timoteivn 10 | Sandnes |
| Svendson | Tove | Borgvn 23 | Sandnes |
| Pettersen | Kari | Storgt 20 | Stavanger |

  The table above contains three records (one for each person) and four columns (Last Name, First Name, Address, and City).

  **SQL QUERIES: -**

With SQL, we can query a database and have a result set returned.

  **A query like this: -**

|  |
| --- |
| SELECT Last Name FROM Persons |

  **Gives a result set like this: -**

|  |
| --- |
| **Last Name** |
| Hansen |
| Svendson |
| Petersen |

**SQL Data Manipulation Language (DML)**

  SQL (Structured Query Language) is syntax for executing queries. But the SQL language also includes syntax to update, insert, and delete records.

  These query and update commands together form the Data Manipulation Language (DML) part of SQL: -

  **SELECT** - extracts data from a database table

  **UPDATE** - updates data in a database table

  **DELETE** - deletes data from a database table

  **INSERT INTO** - inserts new data into a database table

**SQL DATA DEFINITION LANGUAGE (DDL)**

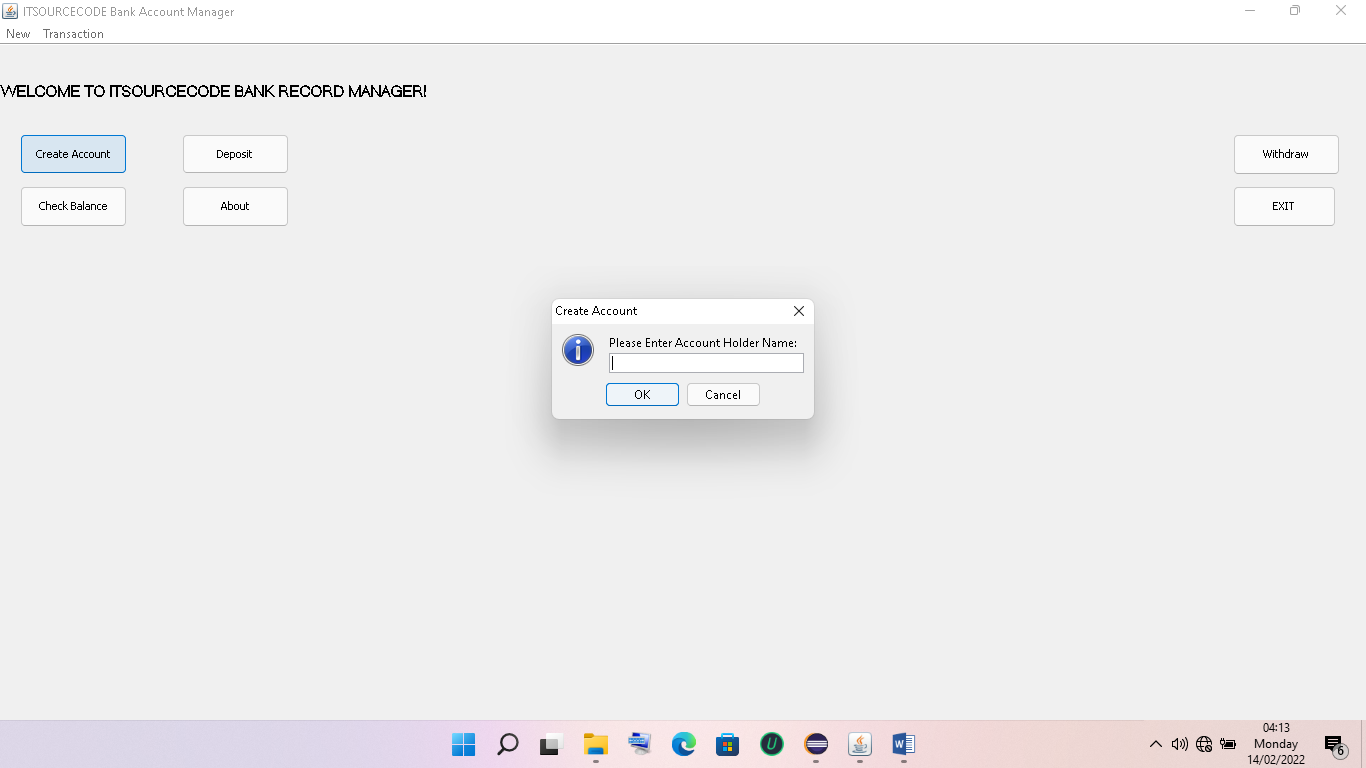
  The Data Definition Language (DDL) part of SQL permits database tables to be created or deleted. We can also define indexes (keys), specify links between tables, and impose constraints between database tables.

  **The most important DDL statements in SQL are: -**

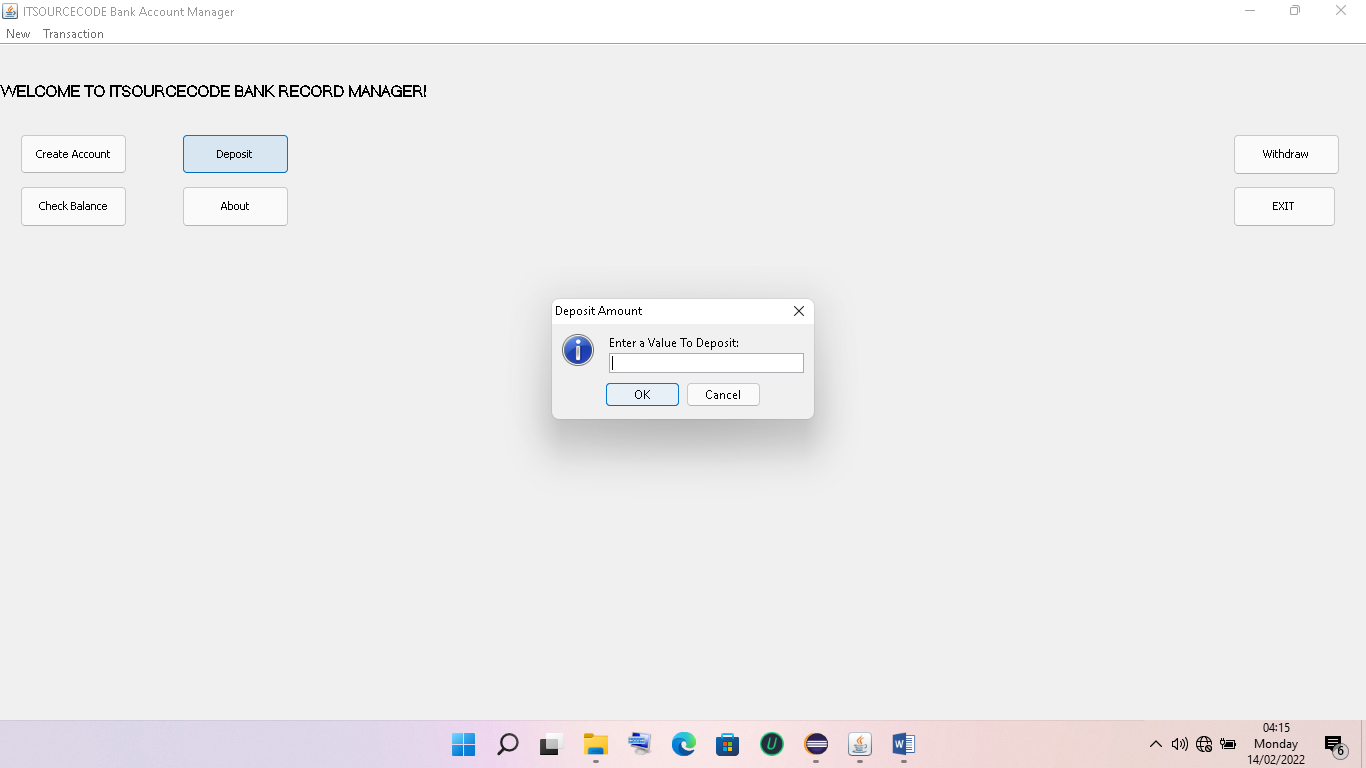
* **CREATE TABLE** - creates a new database table
* **ALTER TABLE**- alters (changes) a database table
* **DROP TABLE** - deletes a database table
* **CREATE INDEX** - creates an index (search key)
* **DROP INDEX** - deletes an index MS SQL SERVER 2000

**Modules**

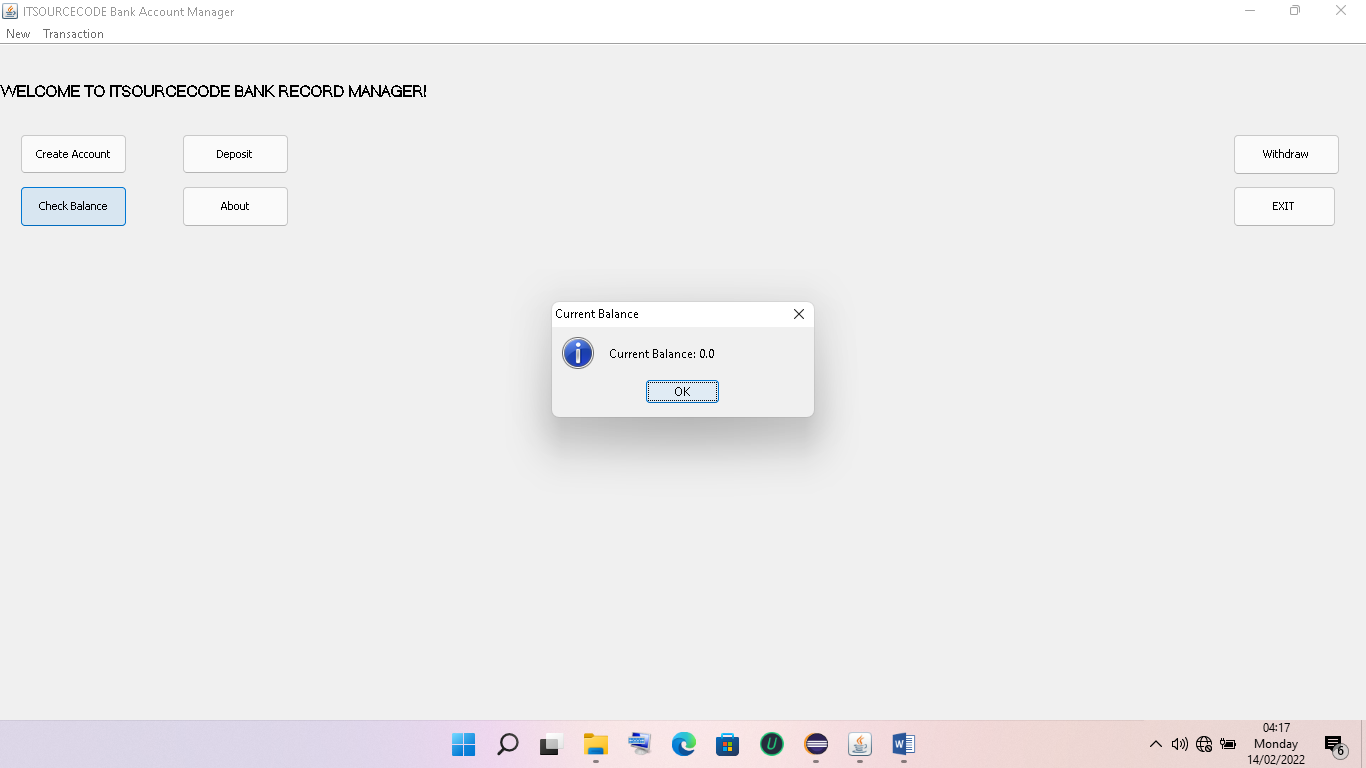
**1.Create Account Module**

****

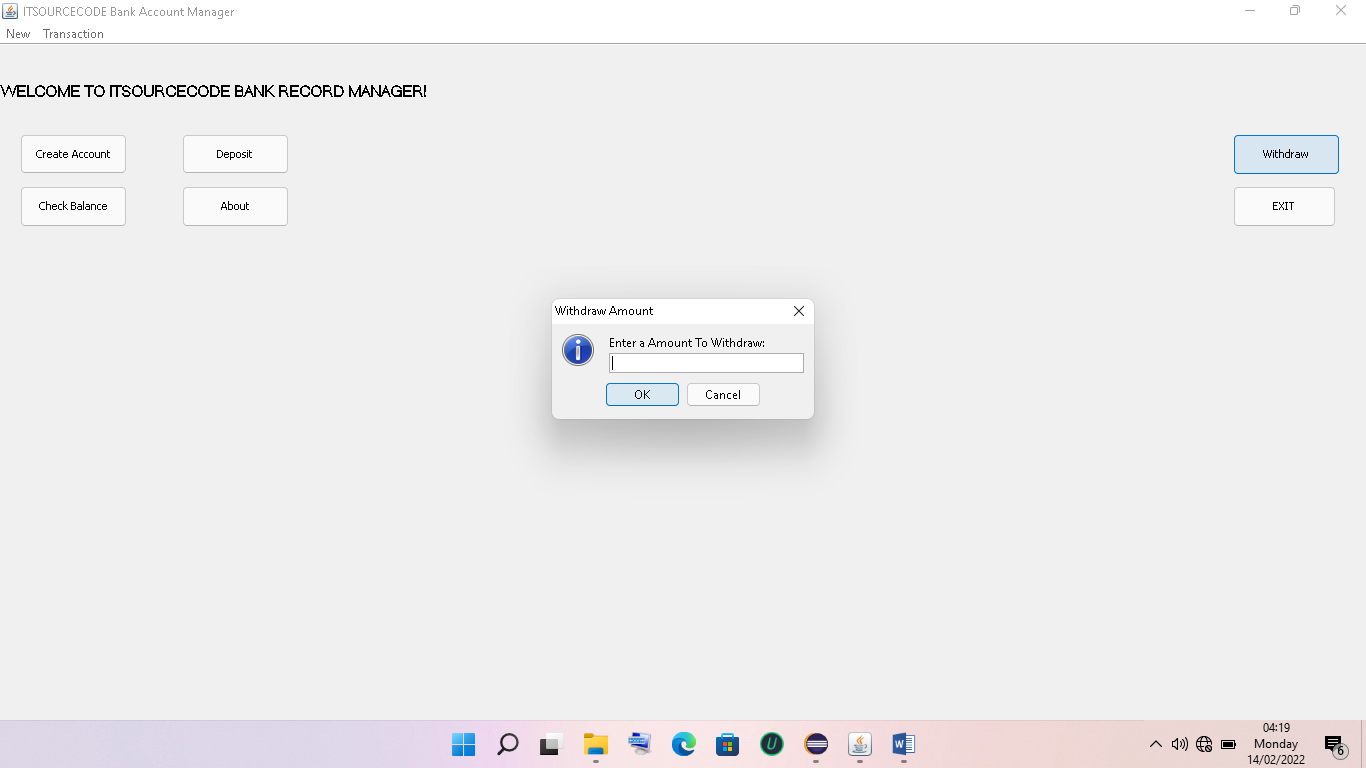
**2.Deposit Module**

****

**3.Check Balance Module**

****

**4.Widthdraw**

****

**DATA FLOW DIAGRAM**

**DATA FLOW DIAGRAM: -**The data flow diagram is also known as “bubble chart” has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design so it is the starting point of specification down to the lowest level of detail. A DFDs consists of a series if bubbles joined by lines. The bubbles represent data transformation and the lines represent the data flow in the system.

  **DFD SYMBOLS:**

  A system defined a source or destination of data.

  An arrow identifies data flow, data in motion.

  A circle represents the process that transforms incoming data flow to outgoing data flow.

  An open rectangular is data store-data at rest or a temporary repository of data.

**SYSTEM DATA FLOW DIAGRAM**

**E-R DIAGRAM**

ER-modeling is a data modeling technique used in software engineering to produce a conceptual data model of a information system. Diagrams created using this ER-modeling technique are called Entity-Relationship Diagrams, or ER diagrams or ERDs. So you can say that Entity Relationship Diagrams illustrate the logical structure of databases.

Dr. Peter Chen is the originator of the Entity-Relationship Model. His original paper about ER-modeling is one of the most cited papers in the computer software field. Currently the ER model serves as the foundation of many system analysis and design methodologies, computer-aided software engineering (CASE) tools, and repository systems.

The original notation for ER-Diagrams uses rectangles to represent entities, and diamonds to represent relationships.

There are three basic elements in ER-Diagrams:

* Entities are the "things" for which we want to store information. An entity is a person, place, thing or event.
* Attributes are the data we want to collect for an entity.
* Relationships describe the relations between the entities.

ERDs show entities in a database and relationships between tables within that database. It is essential to have ER-Diagrams if you want to create a good database design. The diagrams help focus on how the database actually works.

Entity (Instance)

An  instance  of a  physical object in the real world.

Entity Class

: Group of objects of the same type.

–

E.g. Entity Class “Student”, Entities “John”, “Trish” etc

Attributes

Properties  of  Entities that describe  their characteristics.

Types:

Simple

: Attribute that is not divisible, e.g. age.

Composite

: Attribute composed of several simple attributes,

e.g. address (house number, street, district)

Multiple

: Attribute with a set of possible values for the same

entity, e.g. Phone (home, mobile etc.) or email

Key

: Uniquely Ids the Entity e.g. PPSN, Chassis No.

Each simple attribute associated with a VS that may be assigned to that attribute for each individual entity,

e.g. age = integer

**DATA STRUCTURES AND DATABASE SPECIFICATIONS**

**“ACCOUNT\_INFO” Table: -**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Type** | **Constraints** |
| Account\_No | Int | Primary Key |
| Branch\_No | Varchar(7) | References Branch\_Info(Branch\_No) |
| Branch\_Name | VARCHAR(50) | Not Null |
| Account\_H\_Type | Varchar(15) | Not Null |
| No\_Account\_H | Varchar(5) | Not Null |
| Saluation\_F | Varchar(5) | Not Null |
| Name\_P\_O\_F | Varchar(50) | Not Null |
| Fa\_Name\_F | Varchar(50) | Not Null |
| Gender\_F | Varchar(6) | Not Null |
| DOB\_F | DateTime | Not Null |
| Age\_F | Varchar(3) | Check(Age\_F>=0 and Age\_F<100 o:p=""> |

Occupation\_F

Varchar(15)

Not Null

Photo\_F

Image

Not Null

Sign\_F

Image

Not Null

Address\_F

Varchar(100)

Not Null

Ph\_No\_F

Varchar(11)

Not Null

Mob\_No\_F

Varchar(14)

Not Null

Saluation\_S

Varchar(5)

Name\_P\_O\_S

Varchar(50)

Fa\_Name\_S

Varchar(50)

Gender\_S

Varchar(6)

DOB\_S

DateTime

Age\_S

Varchar(3)

Check(Age\_S>=0 and Age\_S<100 o:p="">

Occupation\_S

Varchar(15)

Photo\_S

Image

Sign\_S

Image

Address\_S

Varchar(100)

Ph\_No\_S

Varchar(11)

Mob\_No\_S

Varchar(14)

Saluation\_T

Varchar(5)

Name\_P\_O\_T

Varchar(50)

Fa\_Name\_T

Varchar(50)

Gender\_T

Varchar(6)

DOB\_T

DateTime

Age\_T

Varchar(3)

Check(Age\_T>=0 and Age\_T<100 o:p="">

Occupation\_T

Varchar(15)

Photo\_T

Image

Sign\_T

Image

Address\_T

Varchar(100)

Ph\_No\_T

Varchar(11)

Mob\_No\_T

Varchar(14)

Account\_Type

Varchar(25)

Not Null

Witness\_Name

Varchar(50)

Not Null

Witness\_Sign

Image

Not Null

Nominee\_Rel

Varchar(10)

Not Null

Nominee\_Name

Varchar(50)

Not Null

Nominee\_Sign

Image

Not Null

Opening\_Bal

Varchar(10)

T\_Date

DateTime

Not Null

**“Branch\_Info” Table: -**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Type** | **Constraints** |
| Branch\_No | Varchar(7) | Primary Key |
| Branch\_Name | Varchar(50) |  |

**“Deposit\_Info” Table: -**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Type** | **Constraints** |
| Account\_No | Int | References Account\_Info(Account\_No) |
| Branch\_No | Varchar(7) |  |
| Depositor\_Name | Varchar(150) |  |
| Account\_H\_Type | Varchar(15) |  |
| Deposit\_Amt | Varchar(10) |  |
| Deposit\_Date | DateTime |  |

**“Fixed\_Info” Table: -**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data type** | **Constraints** |
| Account\_No | Int | References Account\_Info(Account\_No) |
| Branch\_No | Varchar(7) |  |
| Depositor\_Name | Varchar(50) |  |
| Account\_H\_Type | Varchar(15) |  |
| Time\_Span | Varchar(5) | References Rate\_Of\_Interest\_Info(Time\_Span) |
| ROI | Varchar(5) |  |
| Start\_Date | DateTime |  |
| Mature\_Date | Varchar(15) |  |
| Deposit\_Amt | Varchar(10) |  |
| Mature\_Amt | Varchar(10) |  |

**“Login\_Info” Table: -**

|  |  |  |
| --- | --- | --- |
| Field Name | Data type | Description |
| UserName | Varchar(20) | Primary Key |
| UserPassWord | Varchar(15) |  |

|  |  |  |
| --- | --- | --- |
| Field Name | Data type | Description |
| Time\_Span | Varchar(5) | Primary Key |
| ROI | Varchar(5) |  |

**“Withdrawl\_Info” Table: -**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Type** | **Constraints** |
| Account\_No | Int | References Account\_Info(Account\_No) |
| Branch\_No | Varchar(7) |  |
| Withdrawee\_Name | Varchar(150) |  |
| Account\_H\_Type | Varchar(15) |  |
| Withdrawl\_Amt | Varchar(10) |  |
| Withdrawl\_Date | DateTime |  |

**“Loan\_Info” Table: -**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Type** | **Constraints** |
| Account\_No | Int | References Account\_Info(Account\_No) |
| Branch\_No | Varchar(7) |  |
| Acc\_Holder\_Name | Varchar(50) |  |
| Account\_Type | Varchar(15) |  |
| Account\_Sub\_Type | Varchar(15) |  |
| Time\_Span | Varchar(5) | References Rate\_Of\_Interest\_Info(Time\_Span) |
| ROI | Varchar(5) |  |
| Issue\_Date | DateTime |  |
| Due\_Date | Varchar(15) |  |
| Loan\_Sanctioned | Varchar(10) |  |
| No\_Installments | Varchar(5) |  |
| EMI | Varchar(10) |  |
| Total\_Loan\_Ret | Varchar(10) |  |

TESTING AND DEBUGGING

INTRODUCTION:-

The implementation phase of software development is concerned with translating design specification into source code. The preliminary goal of implementation is to write source code and internal documentation so that conformance of the code to its specifications can be easily verified, and so that debugging, testing and modifications are eased. This goal can be achieved by making the source code as clear and straight forward as possible. Simplicity, clarity and elegance are the hallmark of good programs, obscurity, cleverness, and complexity are indications of inadequate design and misdirected thinking.

Source code clarity is enhanced by structured coding techniques, by good coding style, by, appropriate supporting documents, by good internal comments, and by feature provided in modern programming languages.

The implementation team should be provided with a well-defined set of software requirement, an architectural design specification, and a detailed design description. Each team member must understand the objectives of implementation.

TERMS IN TESTING FUNDAMENTAL

1. Error

The term error is used in two ways. It refers to the difference between the actual output of software and the correct output, in this interpretation, error is essential a measure of the difference between actual and ideal. Error is also to used to refer to human action that result in software containing a defect or fault.

2. Fault

Fault is a condition that causes to fail in performing its required function. A fault is a basic reason for software malfunction and is synonymous with the commonly used term Bug.

3. Failure

Failure is the inability of a system or component to perform a required function according to its specifications. A software failure occurs if the behavior of the software is the different from the specified behavior. Failure may be caused due to functional or performance reasons.

a. Unit Testing

The term unit testing comprises the sets of tests performed by an individual programmer prior to integration of the unit into a larger system.

A program unit is usually small enough that the programmer who developed it can test it in great detail, and certainly in greater detail than will be possible when the unit is integrated into an evolving software product. In the unit testing the programs are tested separately, independent of each other. Since the check is done at the program level, it is also called program teasing.

b. Module Testing

A module and encapsulates related component. So can be tested without other system module.

c. Subsystem Testing

Subsystem testing may be independently design and implemented common problems are sub-system interface mistake in this checking we concentrate on it.

There are four categories of tests that a programmer will typically perform on a program unit.

1)      Functional test

2)      Performance test

3)      Stress test

4)      Structure test

1) Functional Test

Functional test cases involve exercising the code with Nominal input values for which expected results are known; as well as boundary values (minimum values, maximum values and values on and just outside the functional boundaries) and special values.

2) Performance Test

Performance testing determines the amount of execution time spent in various parts of the unit, program throughput, response time, and device utilization by the program unit. A certain amount of avoid expending too much effort on fine-tuning of a program unit that contributes little to the over all performance of the entire system. Performance testing is most productive at the subsystem and system levels.

3) Stress Test

Stress test are those designed to intentionally break the unit. A great deal can be learned about the strengths and limitations of a program by examining the manner in which a program unit breaks.

4) Structure Test

Structure tests are concerned with exercising the internal logic of a program and traversing particular execution paths. Some authors refer collectively to functional performance and stress testing as “black box” testing. While structure testing is referred to as “white box” or “glass box” testing. The major activities in structural testing are deciding which path to exercise, deriving test date to exercise those paths, determining the test coverage criterion to be used, executing the test, and measuring the test coverage achieved when the test cases are exercised.

DEBUGGING

Defect testing is intended to find areas where the program does not confirm to its specifications. Tests are designed to reveal the presence of defect in the system.When defect have been found in the program. There must be discovered and removed. This is called “Debugging”.

FUTURE SCOPE OF THE PROJECT

This project can be handled in future by doing various modifications like: -

  We can go further for Online Banking.

  We can establish and start various Branches and available help centers for Account Holder’s Queries.

  We can also deal through internet by creating web pages and a banking website for internet dealing.

  To attract Account Holder’s we can offer various offers during festivals months.

  We can also deal in various types of Banking Transactions.

  To have more and more customer satisfaction we will emphasize more and more on our dealings.

BIBLIOGRAPHY AND REFERENCES

BOOK: -                         Black Book on Visual Basic .Net 2003 By Steven Holzner

WEBSITES: -                 WWW.VBTUTORIALS.COM

                                         WWW.VBSOURCECODE.COM

                                          WWW.LOGICATWORK.INFO