BANK MANAGEMENT SYSTEM

USING PYTHON

Submitted in partial fulfilment of the requirements for the award of Bachelor of Science degree in Computer Science

By

Santhosh Kumar. B (39290089)

Vignesh. A (39290124)

DEPARTMENT OF COMPUTER SCIENCE

SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI,

CHENNAI – 600119

MARCH 2022

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE

BONAFIDE CERTIFICATE

This is to verify that this project report is the bonafide work of Santhosh Kumar. B(39390089) and Vignesh. A (39290124) who carried out the project entitled as "BANK MANAGEMENT SYSTEM USING PYTHON" under our supervision from Dec 2021 to March 2022.

Internal Guide

DR . MRS. M. MALINI DEEPIKA

Head of the Department

DR . L. LAKSHMANAN M.E, PH.D

Submitted for Viva voce Examination held on_____

Internal Examiner **External Examiner**

DECLARATION

I Santhosh Kumar B (39290089) and Vignesh A (39290124) hereby declare that the Project Report entitled “Bank Management System using python” done by me under the guidance of Dr . M . Malini Deepika is submitted in partial fulfilment of the requirements for the award of Bachelor of Science degree in Computer Science.

DATE: **01.03.2022**

PLACE: **Chennai, Pondicherry** **SIGNATURE OF THE CANDIDATE**

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to the Board of Management of SATHYABAMA for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks **to Dr . T. Sasikala M.E., Ph . D, Dean, School of Computing Dr . L. Lakshmanan M.E., Ph.D. , and Dr . S. Vigneshwari M.E., Ph.D.** Heads of the Department of Computer Science and Engineering for providing necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide Dr./Mr./Ms for his valuable guidance, suggestions and constant encouragement paved the way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the Department Of Computer Science and Engineering who were helpful in many ways for the completion of the project.

ABSTRACT

This is a simple GUI based system which is very easy to understand and use .In this project , it contains all the basic functions of bank .Creating a new account ,Transactions like withdrawal and deposit amount to the account ,Getting the balance details ,Closing an account Updating the account details.

In this project, he /she can use all those available features easily without any restruction. It is too easy to use, he/she can check the total bank account records easily. Talking about the features of the bank management system, A user can create an account by providing the name of the account holder, number, selecting amount type (Saving account or current account) and providing an initial amount more than or equal to 500.

Then the user can also deposit and withdraw money just by providing his/her account and entering the amount. For certain purpose, he /she can also check for the balance inquiry which displays the account number and amount. He / She can also view the entire account holder’s list. Another feature s that he/she can modify their account detail and type if they want to.

This simple GUI based Bank Management system provides the simplest management of bank account and transaction. In short, this project mainly focuses on CRUD operation. There’ s an external database connection file used in this project to save user’s data permanently.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATION

GUI	Graphical user Interface
ICT	Information Communication Technology
SOA	Service Oriented Architecture
OS	Operating System
API	Application Programming Interface
CMS	Cash Management Service
IDE	Integrated Development Environment
MVC	Model View Controller
TCP	Transmission Control Protocol
ER	Entity Relationship

CHAPTER 1

1. INTRODUCTION

1.1 OVERVIEW OF PROJECT

A program performs a task in the computer. But, in order to be executed, a program must be written in the machine language of the processor of a computer. Unfortunately, it is extremely difficult for humans to read or write a machine language program. This is because a machine language is entirely made up of sequences of bits. However, high level languages are close to natural languages like English and only use familiar mathematical characters, operators and expressions.

A high-level program is translated into machine language by translators like compiler or interpreter. Python is a high-level programming language that is translated by the python interpreter. An interpreter works by translating line-by-line and executing. Python was developed by Guido Van-Rossum in 1990, at the National Research Institute for Mathematics and Computer Science in Netherlands. The Bank Management system is a web-based application used for paying financial institutions for the services they provide to the Bureau of the fiscal service. BMS also provides analytical tools to review, and approve compensation, budgets, and outflows. Bank Management System project is written in python. The project file contains a python script(main.py) and a database file .This a simple console based system which is very easy to understand and use. Talking about the system, it contains all the basic functions which include creating a new account, view account holders record, withdraws and deposit amount, balance inquiry, closing an account and edit account details. In this mini project, there is no such login system. This means he/she can use all those available features easily without any restriction. It is too easy to use, he/she can check the total bank account records easily. Talking about the features of the Bank Management System, a user can create an account by providing the name of the account holder, number, selecting amount type (Saving account or Current account) and providing an initial amount more than or equal to 500. Then the user can also deposit and withdraw money just by providing his/her account and entering the amount. For certain purpose, he/she can also check for the balance inquiry which displays the account number and amount. He/she can also view all the account holder’s list. Another feature is that he/she can modify their account detail and type if they want to.

CHAPTER 2

LITERATURE SURVEY

In MD. Faizan(2012), Information and communication technology (ICT) has helped to drive increasingly intense global Competition. In the world history the most of the countries are most developed because of they are financially very clear for how to use the high amount of money in the developing process in own country. We also use the SOA architecture for providing the scalable and reliable service therefor we studied related to the SOA architecture to know how we use to implementation process in our project using Service Oriented Architectures (SOA).we also refer the paper who give the case study information about Scandinavian bank and a Swiss bank This two banks are working on the basis of service oriented architecture for providing the service for the customer. SOA provides potential for greater organizational agility (and thereby competitiveness).

In MD. Aquil Amwar (2012), in the second paper we learn which type of problems are created in banking system during the different types of transactions. Here discuss about if any region the transaction may be fail then how to avoid it and fixed it. We also studied about Firms in Italy defaulted more against banks with high levels of past losses. This `selective' default increases where legal enforcement is weak. Poor enforcement thus can create a systematic transaction risk by encouraging banking users to defaulted masse once the continuation value of their bank relationships comes into doubt. In banking sector the security also must and when we talk about money or property this case is more sensational then we found the security is the major thing to do in banking system. In our project we provide the security questions when customer login with account to prevent the fraud and provide the best security in the bank management system. The study used constructs adopted from Technology Acceptance Model along with constructs of perceived service quality, perceived credibility and perceived risk to empirically establish the influence on satisfaction and continuance usage intentions. The study confirmed that after adoption of the technology, the customer finds satisfaction in the quality parameters of the service

CHAPTER 3

3. AIM AND SCOPE OF PRESENT INVESTIGATION

3.1 AIM OF THE PROJECT

To develop a software for solving financial applications of a customer in banking environment in order to nurture the needs of an end banking user by providing various ways to perform banking tasks. Also to enable the users workspace to have additional functionalities which are not provided under a conventional banking software.

3.2 SCOPE AND OBJECTIVE

Banking Management System can be used by bank employees and /or customers depending on the bank policies. It can be used by several employees at the same time with the required rights. It can be accessed using any general web browser with graphical interface. Objectives are goals, and it is toward these results that all activities are directed. Objectives may change over time, but they are looked upon as firm and binding contracts once formulated. Bank objectives are usually stated in short, concise terms and limited to ten to twelve items. A few items from a list of one bank s objectives follow:

1. Our business is selling financial services in Oregon and in selected regional, national, and international markets. We will extend our business into areas that provide sound expansion opportunities meeting predetermined profit criteria.
2. We will strive for stability in earning growth, acquiring high-quality investments, and pursuing sound and innovative tactics ‘. Through strategic planning and strong management, we will aggressively expand income sources while remaining in control of costs.
3. Management will provide continuity of policies and directions. Changes will be implemented quickly and in a manner that considers both individual and corporate needs.

3.3 SYSTEM REQUIREMENTS

3.3.1 Hardware Requirements

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware. The minimal hardware requirements are as follows,

1. Processor : AMD
2. RAM : 8 GB
3. Processor : 2.4 GHz
4. Main Memory : 8GB RAM
5. Hard Disk Drive : 1tb
6. Keyboard : 104 Keys

3.3.2 Software Requirements

Software requirements deals with defining resource requirements and prerequisites that needs to be installed on a computer to provide functioning of an application. The minimal software requirements are as follows.

1. Front end : python
2. IDE : pycharm
3. Operating System : Windows 10

3.4 SOFTWARE USED:

3.4.1 Python Language

Python is an object-oriented programming language created by Guido Rossum in 1989. It is ideally designed for rapid prototyping of complex applications. It has interfaces to many OS system calls and libraries and is extensible to C or C++. Many large companies use the Python programming language include NASA, Google, YouTube, BitTorrent, etc. Python programming is widely used in Artificial Intelligence, Natural Language Generation, Neural Networks and other advanced fields of Computer Science. Python had deep focus on code readability & this class will teach you python from basics.

3.4.2 Python Characteristics

• It provides rich data types and easier to read syntax than any other programming languages

• It is a platform independent scripted language with full access to operating system API's

• Compared to other programming languages, it allows more run-time flexibility

• It includes the basic text manipulation facilities of Perl and Awk

• A module in Python may have one or more classes and free functions

• Libraries in Pythons are cross-platform compatible with Linux, Macintosh, and Windows

• For building large applications, Python can be compiled to byte-code

• Python supports functional and structured programming as well as OOP

• It supports interactive mode that allows interacting Testing and debugging of snippets of code

• In Python, since there is no compilation step, editing, debugging and testing is fast.

3.4.3 Applications of Python Programming

Web Applications:

You can create scalable Web Apps using frameworks and CMS (Content Management System) that are built on Python. Some of the popular platforms for 8 creating Web Apps are: Django, Flask, Pyramid, Plone , Django CMS. Sites like Mozilla, Reddit, Instagram and PBS are written in Python.

Scientific and Numeric Computing:

There are numerous libraries available in Python for scientific and numeric computing. There are libraries like: SciPy and NumPy that are used in general purpose computing. And, there are specific libraries like: Earthy for earth science, Astray for Astronomy and so on. Also, the language is heavily used in machine learning, data mining and deep learning.

Creating software Prototypes:

Python is slow compared to compiled languages like C++ and Java. It might not be a good choice if resources are limited and efficiency is a must. However, Python is a great language for creating prototypes. For example: You can use Pygmy (library for creating games) to create your game's prototype first. If you like the prototype, you can use language like C++ to create the actual game.

Good Language to Teach Programming:

Python is used by many companies to teach programming to kids and newbies. It is a good language with a lot of features and capabilities. Yet, it's one of the easiest languages to learn because of its simple easy-to-use syntax.

3.5 PYCHARM:

PyCharm is a dedicated Python Integrated Development Environment (IDE) providing a wide range of essential tools for Python developers, tightly integrated to create a convenient environment for productive Python, web, and data science development.

PyCharm is available in three editions:

- *Community* (free and open-sourced): for smart and intelligent Python development, including code assistance, refactorings, visual debugging, and version control integration.
- *Professional* (paid) : for professional Python, web, and data science development, including code assistance, refactorings , visual debugging, version control integration, remote configurations, deployment, support for popular web frameworks, such as Django and Flask, database support, scientific tools (including Jupyter notebook support), big data tools.

- *Edu* (free and open-sourced): for learning programming languages and related technologies with integrated educational tools.

Supported languages :

To start developing in Python with PyCharm you need to download and install Python from python.org depending on your platform.

PyCharm supports the following versions of Python:

- **Python 2:** version 2.7
- **Python 3:** from the version 3.6 up to the version 3.11

Besides, in the Professional edition, one can develop Django, Flask, and Pyramid applications. Also, it fully supports HTML (including HTML5), CSS, JavaScript, and XML: these languages are bundled in the IDE via plugins and are switched on for you by default. Support for the other languages and frameworks can also be added via plugins (go to Settings | Plugins or PyCharm | Preferences | Plugins for macOS users, to find out more or set them up during the first IDE launch).

Supported platforms:

PyCharm is a cross-platform IDE that works on Windows, macOS, and Linux. Check the system requirements**:**

Requirement	Minimum	Recommended
RAM	4 GB of free RAM	8 GB of total system RAM
CPU	Any modern CPU	Multi-core CPU. PyCharm supports multithreading for different operations and processes making it faster the more CPU cores it can use.
Disk space	2.5 GB and another 1 GB for caches	SSD drive with at least 5 GB of free space
Monitor resolution	1024x768	1920×1080

Operating system	Officially released 64-bit versions of the following: • Microsoft Windows 8 or later • macOS 10.14 or later • Any Linux distribution that supports Gnome, KDE , or Unity DE. PyCharm is not available for some Linux distributions, such as RHEL6 or CentOS6, that do not include GLIBC 2.14 or later. Pre-release versions are not supported.	Latest 64-bit version of Windows, macOS, or Linux (for example, Debian, Ubuntu, or RHEL)

You can install PyCharm using Toolbox or standalone installations. If you need assistance installing PyCharm, see the installation instructions: Install PyCharm .

3.6 Tkinter:

This Tkinter tutorial introduces you to the exciting world of GUI programming in Python.

Tkinter is pronounced as tea-kay-inter. Tkinter is the Python interface to Tk, which is the GUI toolkit for Tcl/Tk.

Tcl (pronounced as tickle) is a scripting language often used in testing, prototyping, and GUI development. Tk is an open-source, cross-platform widget toolkit used by many different programming languages to build GUI programs.

Python implements the Tkinter as a module. Tkinter is a wrapper of C extensions that use Tcl/Tk libraries.

Tkinter allows you to develop desktop applications. It’s a very good tool for GUI programming in Python.

T k inter is a good choice because of the following reasons:

- Easy to learn.
- Use very little code to make a functional desktop application.
- Layered design.
- Portable across all operating systems including Windows, macOS, and Linux.
- Pre-installed with the standard Python library.

1.T k inter Fundamentals

- Tkinter Hello, World! – show you how to develop the first Tkinter program called Hello, World!
- Window – learn how to manipulate various attributes of a Tkinter window including title, size, location, resizability, transparency, and stacking order.
- Tk Themed Widgets – introduce you to Tk themed widgets.
- Setting options for a widget – learn various ways to set options for a widget.
- Command Binding – learn how to respond to events using command bindings.
- Event Binding – show you how to use the bind() method to bind an event of a widget.
- Label – learn how to use the Label widget to show a text or image on a frame or window.
- Button – walk you through the step of creating buttons.
- Entry – learn how to create a textbox using the Entry widget.

2. Layout Managements

Geometry managers allow you to specify the positions of widgets inside a top-level or parent window.

- pack – show you how to use the pack geometry manager to arrange widgets on a window.
- grid – learn how to use the grid geometry manager to place widgets on a container.
- place – show you how to use the place geometry manager to precisely position widgets within its container using the (x, y) coordinate system.

3. **Ttk & Tkinter Widgets**

Tkinter provides you with some commonly used widgets, which allow you to start developing applications more quickly.

- Text – show a multi-line text input field.
- Scrollbar – learn how to link a scrollbar to a scrollable widget e.g., a Text widget.
- Scrolled Text – show you how to create a scrolled text widget that consists of Text and vertical scrollbar widgets.
- Separator – use a separator widget to separate fields.
- Checkbox – show how to create a checkbox widget.
- Radio Button – learn how to use radio buttons to allow users to select one of a number of mutually exclusive choices.
- Combo box – walk you through the steps of creating a combo box widget.

- List box – show you how to display a list of single-line text items on a Lis tbox.
- Paned Window – show you how to use the Paned Window to divide the space of a frame or a window.
- Slider – learn how to create a slider by using the Tkinter Scale widget.
- Spin box – show you how to use a Spin box.
- Size grip – guide you on how to use the Size grip widget to allow users to resize the entire application window.
- Label Frame – show you how to group related widgets in a group using the Label Frame widget.
- Progress bar – show you how to use the progress bar widget to give feedback to the user about the progress of a long-running task.
- Notebook – guide you on how to use the Notebook widget to create tabs.
- Tree view – walk you through the steps of creating tree view widgets that display tabular and hierarchical data.
- Frame – learn how to use the Frame widget to group other widgets.

4. Tkinter Examples

- Tkinter example – show you how to build a simple application that converts a temperature from Fahrenheit to Celsius.

5. Object-Oriented Programming with Tkinter

- Creating an object-oriented window – learn how to define an object-oriented window.
- Creating an object-oriented frame – show you how to define an object-oriented Frame.
- Developing a full Tkinter object-oriented application – show you how to develop a full Tkinter object-oriented application.
- Switching between frames – guide you on how to switch between frames in a Tkinter application.

6. Dialogs and Menus

- Displaying a message box – show you how to display various message boxes including information, warning, and error message boxes.
- Displaying a Yes/No Dialog – show you how to use the askyes no() function to display a yes/no dialog.
- Display an OK/Cancel Dialog – show you how to use the askok cancel() function to display an OK/Cancel dialog.
- Display a Retry/Cancel Dialog – show you how to use the askretry cancel() function to display a Retry/Cancel dialog.

- Show an Open File Dialog – display an open file dialog to allow users to select one or more files.
- Displaying the Native Color Chooser – show you how to display the native color-chooser dialog.
- Menu – learn how to add a menu bar and menus to a window.
- Menu button – show you how to the Menu button widget.
- Option Menu – Walk you through the steps of creating an Option Menu widget that provides a list of options in a drop-down menu.

7. Tkinter Themes and Styles

- Changing the ttk theme – how to change the default ttk theme to the new one.
- Modifying ttk style – show you how to change the appearance of widgets by modifying or extending the ttk style.
- Understanding ttk elements – help you understand ttk elements and how to use them to change the appearance of widgets.
- Modifying the appearance of a widget based on its states – show you how to dynamically change the appearance of a widget based on its specific state.

8. Tkinter Asynchronous Programming

- Scheduling a task with the after() method – how to use the after() method to schedule a task that will run after a timeout has elapsed.
- Developing multithreading Tkinter Applications – show you how to use the threading module to develop a multithreading Tkinter application.
- Displaying a progress bar while a thread is running – walk you through the steps of connecting a progress bar with a running thread.

9. Advanced Tkinter Programming

- Tkinter MVC – structure a tkinter application using the MVC design pattern.
- Tkinter validation – show you how to use the Tkinter validation to validate user inputs.
- Tkinter & Matplotlib – show you how to display a bar chart from the matplotlib in Python.

3.7 Pickle

Pickle is used for serializing and de-serializing Python object structures, also called marshalling or flattening. Serialization refers to the process of converting an object in memory to a byte stream that can be stored on disk or sent over a network. Later on, this character stream can then be retrieved and de-serialized back to a Python object. Pickling is not to be confused with compression! The former is the conversion of an object from one representation (data in Random Access Memory (RAM)) to another (text on disk), while the latter is the process of encoding data with fewer bits, in order to save disk space.

Pickling is useful for applications where you need some degree of persistency in your data. Your program's state data can be saved to disk, so you can continue working on it later on. It can also be used to send data over a Transmission Control Protocol (TCP) or socket connection, or to store python objects in a database. Pickle is very useful for when you're working with machine learning algorithms, where you want to save them to be able to make new predictions at a later time, without having to rewrite everything or train the model all over again.

If you want to use data across different programming languages, pickle is not recommended. Its protocol is specific to Python, thus, cross-language compatibility is not guaranteed. The same holds for different versions of Python itself. Unpickling a file that was pickled in a different version of Python may not always work properly, so you have to make sure that you're using the same version and perform an update if necessary. You should also try not to unpickle data from an untrusted source. Malicious code inside the file might be executed upon unpickling.

Why Pickle?: In real world sceanario, the use pickling and unpickling are widespread as they allow us to easily transfer data from one server/system to another and then store it in a file or database.

Precaution: It is advisable not to unpickle data received from an untrusted source as they may pose security threat. However, the pickle module has no way of knowing or raise alarm while pickling malicious data.

Only after importing pickle module we can do pickling and unpickling. Importing pickle can be done.

CHAPTER 4

4. EXPERIMENTAL OR MATERIAL METHODS

4.1 DESIGN METHODOLOGY

4.1.1Existing System:

The existing bank system is slow as every task is being performed by the human being and comparing the computer task speed with a computer is not fair. The complexity of this system is increased when an increase in the number of customers and with that there will be a number of transactions will be performed now everything needs to log in to a file for reference in the future which is simply not the kind of scenario we need at this time.

Some other drawbacks of the existing system:

- Less security of customer and bank information.
- Require more physical work and manpower.
- All the manual entry and editing will take more time.
- No level of clearance for the different levels of employees.
- Safety of paper documents from the disaster.
- No backup of the information.

4.1.2Proposed System:

The by looking at disadvantages these are pretty serious for any banking system as they are capable of bringing down the whole system. By digitalization in the banking system, it will not only achieve its goals and also will give some benefits like less manual calculation will be required.

4.2 MODULE DESCRIPTION

Module 1: New Account

A person can open a bank account either by visiting a bank's branch or through the bank's website. Here is all you need to know. There are broadly two modes of opening a bank account - online and offline. One can either go to a nearby branch and request to open a new bank account or directly go to a bank's website to start the procedure online.

Different types of account:

Savings Account, Current Account, Recurring Deposit Account, Fixed Deposit Account, DEMAT Account, NRI Account.

- First i need to enter the account number.
- Then enter the name of the account holder.
- Next select the which type of account savings or current account.
- And enter the initial deposit amount.

Now the new account was created.

Module 2: Transaction

The transactions are doing every day. It manages all the transactions like new account entry, deposit as well as withdraw entry, transaction of money for various processes. A bank transaction is a record of money that has moved in and out of your bank account. When you have costs associated with your business - for example, rent for office space - the payments for these will come out of your bank account as transactions. The formation of your asset accounts, capital accounts and liability accounts all rely on bank transactions. A transaction account, also called a checking account, chequing account, current account, demand deposit account, or share draft account at credit unions, is a deposit account held at a bank or other financial institution. It is available to the account owner "on demand" and is available for frequent and immediate access by the account owner or to

others as the account owner may direct. Access may be in a variety of ways, such as cash withdrawals, use of debit cards, cheques (checks) and electronic transfer. In economic terms, the funds held in a transaction account are regarded as liquid funds. In accounting terms, they are considered as cash.

Transaction accounts are known by a variety of descriptions, including a current account (British English), chequing account or checking account when held by a bank, share draft account when held by a credit union in North America. In the United Kingdom, Hong Kong, India and a number of other countries, they are commonly called current or cheque accounts. Because money is available on demand they are also sometimes known as demand accounts or demand deposit accounts. In the United States, NOW accounts operate as transaction accounts.

Transaction accounts are operated by both businesses and personal users. Depending on the country and local demand economics earning from interest rates varies. Again depending on the country the financial institution that maintains the account may charge the account holder maintenance or transaction fees or offer the service free to the holder and charge only if the holder uses an add-on service such as an overdraft.

Module 3: Close Account

Many salaried people hold multiple bank accounts as they change jobs or shift to new cities. In such cases, some banks convert their zero balance salary accounts into regular savings accounts after a couple of months, as they notice no salary credits in this period. So, you are expected to maintain a minimum average balance in those non-salary savings accounts. It is better to close any dormant account to save on charges that would be levied for not maintaining a minimum average balance as specified by the respective bank.

Maintaining too many bank accounts can be difficult, here are some steps you can follow to close a bank account. Having a limited number of bank accounts is good but too many bank account can be a trouble for you. Because you have to maintain minimum balance requirements on each of them. So it is advisable to close bank accounts that are not used actively. If you among them who have an unwanted bank account then you should close it. Do you know how to close a bank account?

Here are some steps which you can follow to close your bank account. But before you go for it don't forget to delink your bank account from any of the payments platforms or service apps like Paytm, Uber, Swiggy etc.

Things you need to know

Here are a few things you need to know before closing your bank account.

- Once you close your account you cannot re-open it again
- Before proceeding with the account closure you should make the balance to zero
- In case if there are any pending dues, then you should clear it before closing the account
- Before closing an account you should take the complete bank statement of your account for future use.

Steps to close a bank account

Here are some steps which can guide you to close a bank account.

Visit Bank

You cannot close your bank account online. You need to visit your home branch where you opened the account. So you need to walk into the home branch where you have an account and request them for account closure.

Account closure form

All banks provide an account closure form, which you can procure from the bank's branch or website. If you have a joint account, then all account holders will have to give their consent by signing the closure form.

Fill the complete details

After you receive the account closure form. You need to fill the complete details on it:

- Name of the account holder
- Account number
- Contact number
- Signature of the account holder
- Reasons for closing the account.

Submit required document

After filling up the closure form you need to submit it to the bank with the following:

- **Cheque Book:** You need to return the cheque book along with remaining cheque leaves to the respective bank branch at the time of closing their account.
- **Passbook:** You should also handover your passbook to the bank at the time of closing their SBI account.
- **Debit Card:** The account holder should also return their debit card which is used to withdraw money from ATM.
- **ID proof:** Some bank may even ask you for ID proof and address proof before closing your account.

Closure charges

Some banks charge for the account closure. For example, SBI Bank's don't charge for the closure within 14 days of the opening of an account. Any closure of the SBI bank account after 14 days but before 1 year are subject to some closure charges.
Keep these things in mind and don’t let unwanted bank accounts lie idle as there is no benefit in making yourself overburdening in gathering information and statements from too many banks. So close unwanted account they serve no good to your financial life.

Module 4: Deposit Amount

A deposit is a financial term that means money held at a bank. A deposit is a transaction involving a transfer of money to another party for safekeeping. However, a deposit can refer to a portion of money used as security or collateral for the delivery of a good.

- First we need to enter the account number
- Then we get the details of our account.
- Next we want to enter the Deposit amount.
- Then we want to select the which type of account.
- Next select save option in the display.

Module 5: Withdrawal Amount

A withdrawal involves removing funds from a bank account, savings plan, pension or trust. Some accounts don't function like simple bank accounts and carry fees for the early withdrawal of funds.

- First, we need to enter the account number
- Then we get the details of our account.
- Next, we want to enter the withdrawal amount.
- Then we want to select the which type of account.
- Next select save option in the display.

Module 6: Balance Enquiry

The Balance Inquiry process is associated with customer accounts and is used to check the amount remaining on a customer's store credit voucher, gift card, or gift certificate.

- First, we need to enter the account number
- we get the details of our account with balance.

Module 7: Update Details

Details such as bank name, account number, etc., which uniquely identify a bank account, and are used when making or receiving a payment, now especially electronically. In recent use often in the context of the dishonest acquisition of another's bank details in order to perpetrate fraud.

- First, we need to enter the account number
- Next enter the account holder name and the account type.

- we get the details of our account
- now we can update the details.

Module 8: Account list

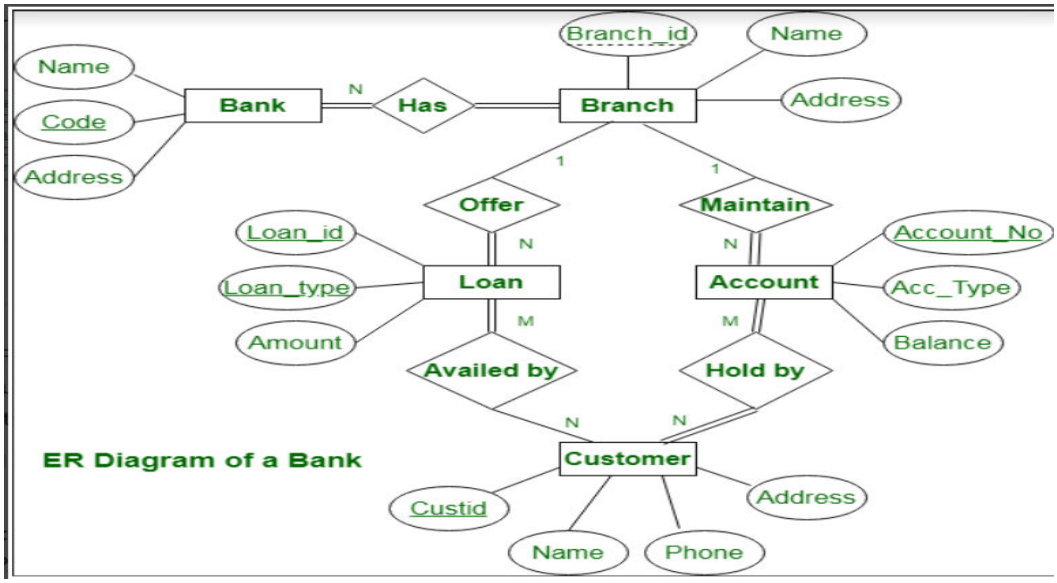
Here we can see the how many account are here with account number, account holder name, account type and balance amount.

4.3 ARCHITECTURE DIAGRAM

ER diagram is known as Entity-Relationship diagram. It is used to analyze to structure of the Database. It shows relationships between entities and their attributes. An ER model provides a means of communication.
ER diagram of Bank has the following description :

- Bank have Customer.
- Banks are identified by a name, code, address of main office.
- Banks have branches.
- Branches are identified by a branch_no., branch_name, address.
- Customers are identified by name, cust-id, phone number, address.
- Customer can have one or more accounts.
- Accounts are identified by account_no., acc_type, balance.
- Customer can avail loans.
- Loans are identified by loan_id, loan_type and amount.
- Account and loans are related to bank’s branch.

4.3.1 ER Diagram of Bank Management System :



This bank ER diagram illustrates key information about bank, including entities such as branches, customers, accounts, and loans. It allows us to understand the relationships between entities.

Entities antheir **Attributes** are :

- **Bank Entity :** Attributes of Bank Entity are Bank Name, Code and Address.
  Code is Primary Key for Bank Entity.
- **Customer Entity :** Attributes of Customer Entity are Customer id, Name, Phone Number and Address.
  Customer id is Primary Key for Customer Entity.
- **Branch Entity :** Attributes of Branch Entity are Branch id, Name and Address.
  Branch id is Primary Key for Branch Entity.
- **Account Entity :** Attributes of Account Entity are Account number, Account type and Balance.
  Account number is Primary Key for Account Entity.
- **Loan Entity :** Attributes of Loan Entity are Loan id, Loan Type and Amount.
  Loan id is Primary Key for Loan Entity.

Relationships are :

- **Bank has Branches => 1 : N**
  One Bank can have many Branches but one Branch cannot belong to many Banks, so the relationship between Bank and Branch is one to

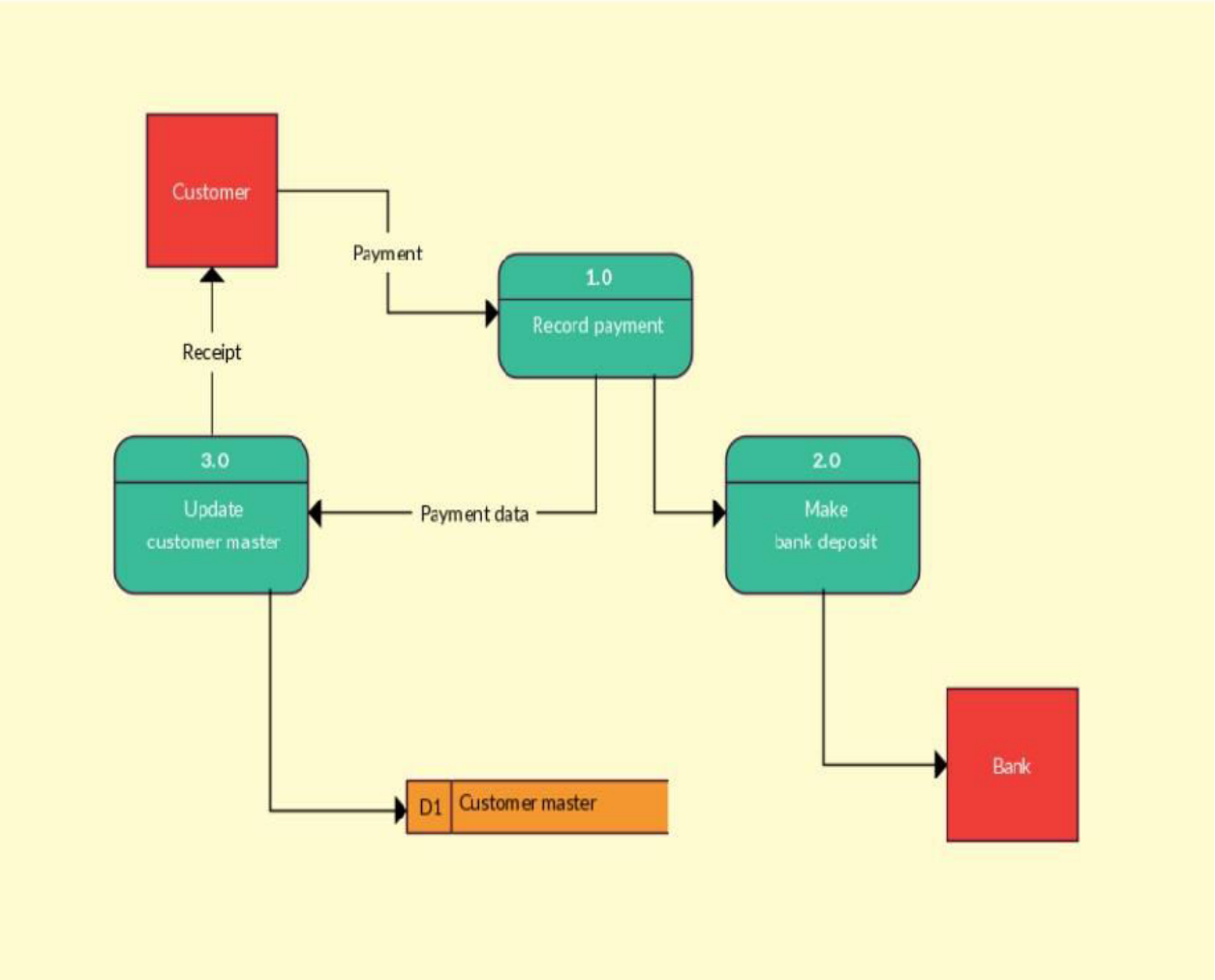
many relationship.

- **Branch maintain Accounts => 1 : N**
  One Branch can have many Accounts but one Account cannot belong to many Branches, so the relationship between Branch and Account is one to many relationship.

- **Branch offer Loans => 1 : N**
  One Branch can have many Loans but one Loan cannot belong to many Branches, so the relationship between Branch and Loan is one to many relationship.

- **Account held by Customers => M : N**
  One Customer can have more than one Accounts and also One Account can be held by one or more Customers, so the relationship between Account and Customers is many to many relationship.

- **Loan availed by Customer => M : N**
  (Assume loan can be jointly held by many Customers).
  One Customer can have more than one Loans and also One Loan can be availed by one or more Customers, so the relationship between Loan and Customers is many to many relationship.



EXPLANATION

If need to create a new account. Select the new account option. In that option enter the new account number. Then enter the name of the account holder. Next select the which type of account savings or current account. And enter the initial deposit amount. Now the new account was created. If I want to put the money in the account means, Click the Deposit option then we need to enter the account number Then we get the details of our account. we want to enter the Deposit amount.

Then select the which type of account. Next select save option in the display. If I need to withdrawal the amount means, Select the withdrawal option then, we need to enter the account number Then we get the details of our account. Next, we want to enter the withdrawal amount. Then we want to select the which type of account.

Next select save option in the display. Now I need to check the balance means, Select the option check balance then, we need to enter the account number we get the details of our account with balance. Now I need to change my details means, First, we need to enter the account number Next enter the account holder name and the account type. we get the details of our account now we can update the details.

If I need to check the how many accounts in our bank means, we can see the option account list. In that option we can see how many accounts are in the bank with account number, account holder name, account type and balance amount.

CHAPTER 5

5. RESULTS AND PERFORMANCE ANALYSIS

5.1 HOME PAGE: This is the total system of banking

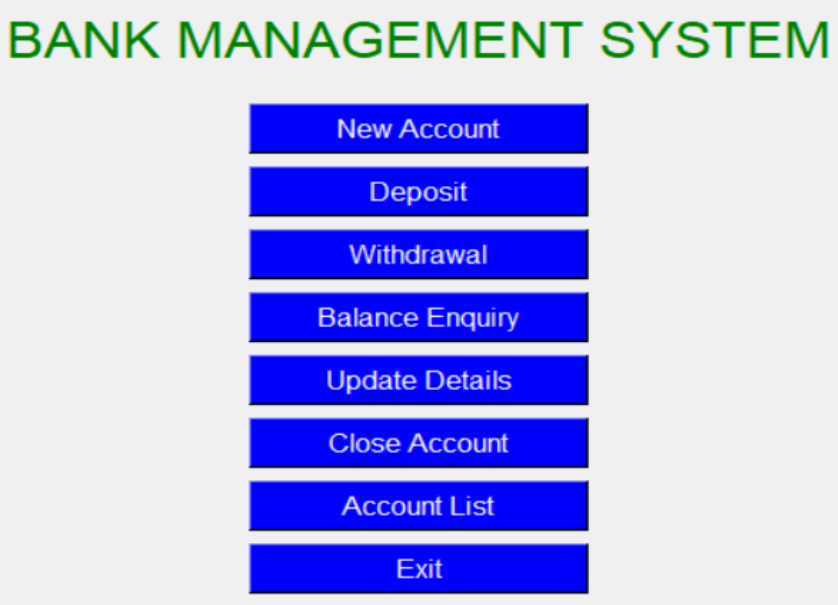


Fig 5.1: Home page

5.2 NEW ACCOUNT: This is how the customers are created with a bank account.

Fig5.2:create new account

5.3 TRANSACTION: This is how the transactions are doing every day. It manages all the transactions like new account entry, deposit as well as withdraw entry transaction of money for various processes.

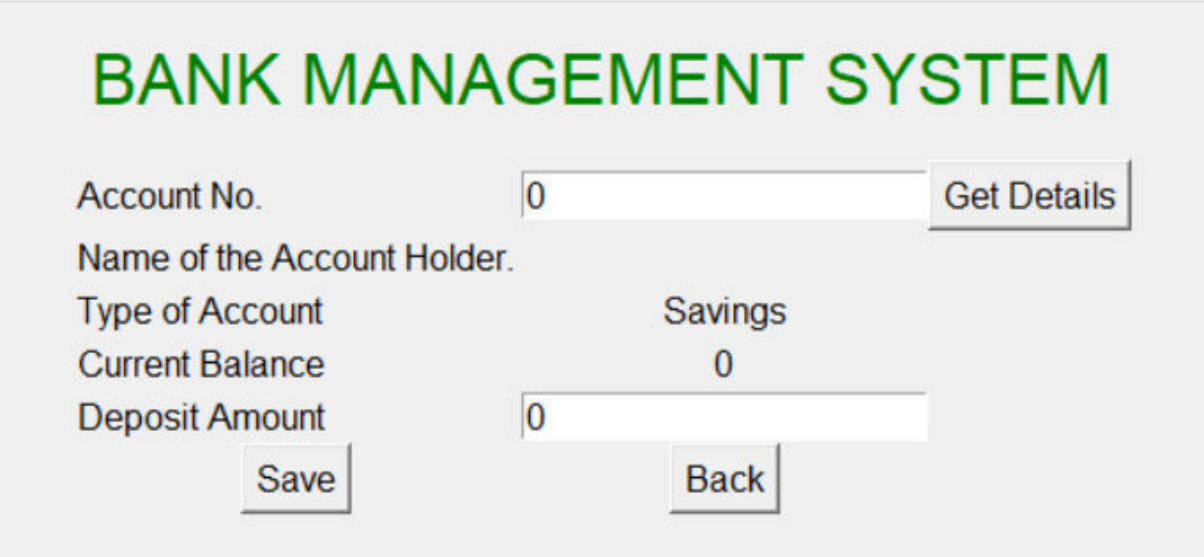


Fig 5.3: Transaction method

5.4 UPDATATION: This is how a edit account details.

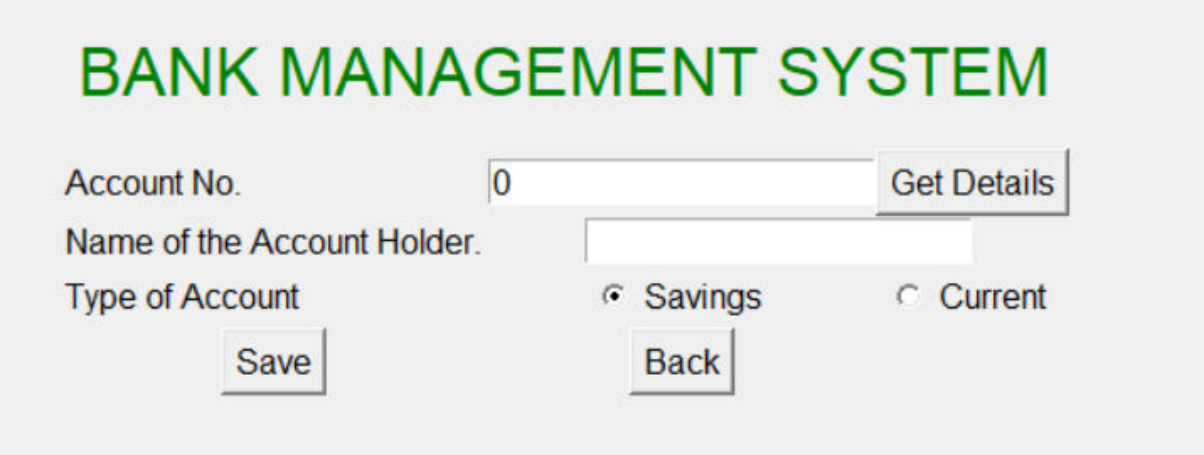


Fig 5.4: for update the details

5.5 BALANCE ENQUIRY: Reads the balance of a particular customer.

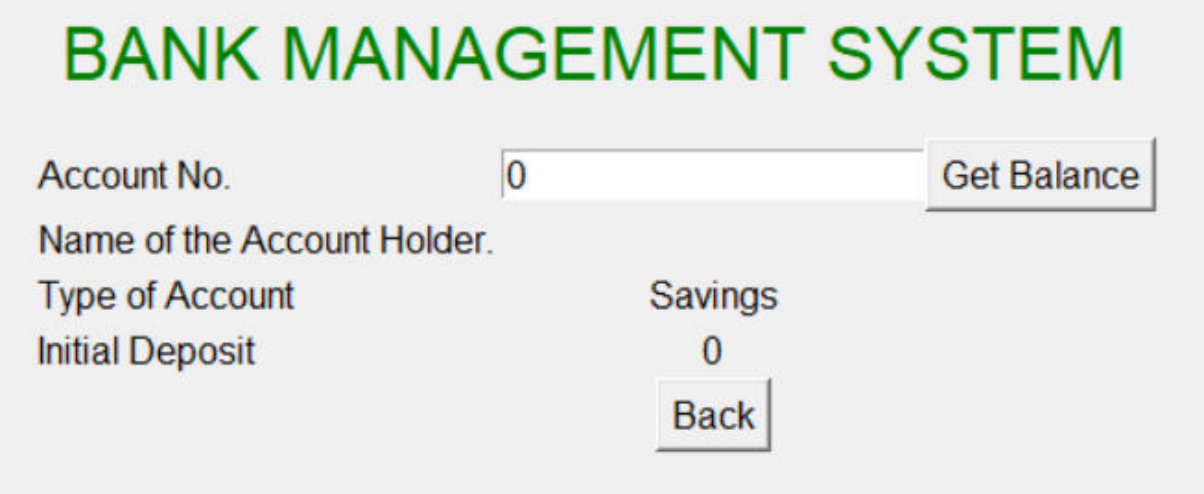


Fig 5.5: check the balance

5.6 CLOSE ACCOUNT: How to close an account from the banking organization.

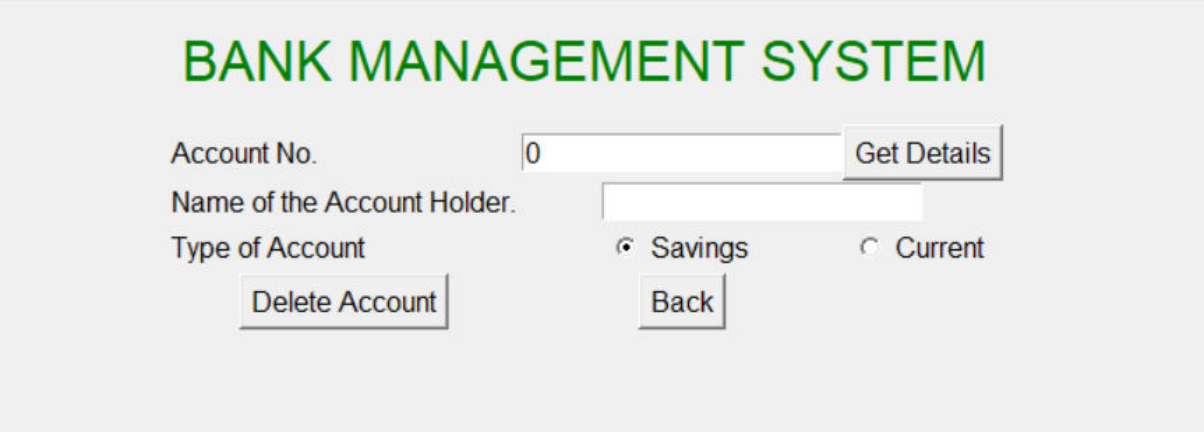


Fig 5.6: to close the account

5.7 REPORTS: This is how to get all the account details.

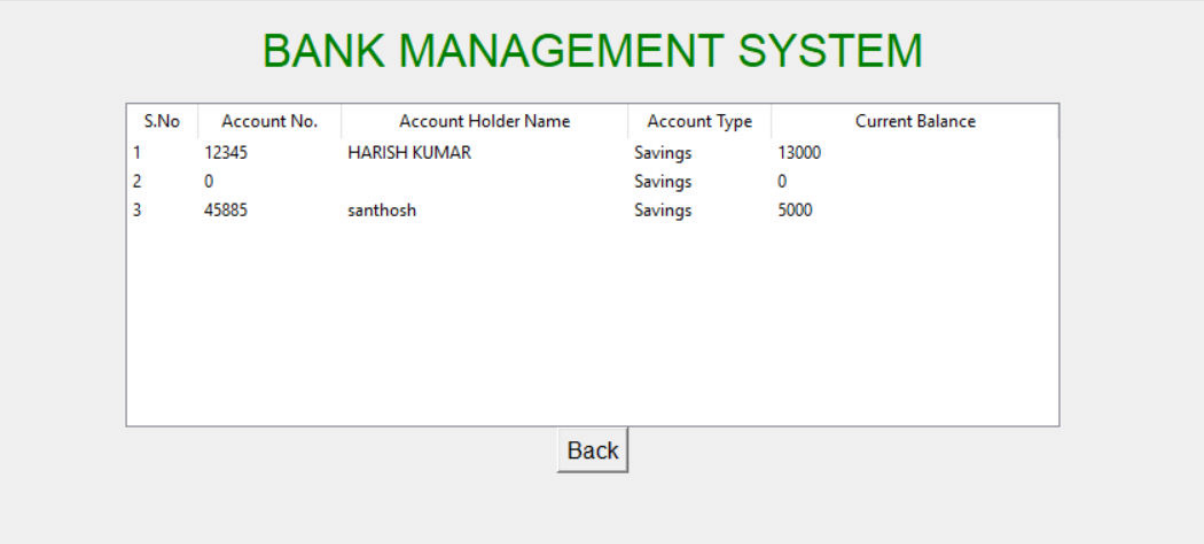


Fig 5.7: to know the all account details

BANK MANAGEMENT SYSTEM undertaken as a project is based on relevant technologies. The main aim of this project is to develop software for bank management system. This project is to develop software for bank management system.

This project has been developed to carry out the processes easily and quickly, which is not possible with the manuals systems, which are overcome by this software.

project is developed using VB language and. Hence it provides the complete solution for the current management system.

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

Bank is the place where customers feel the sense of safety for their property. In the bank, customers deposit and withdraw their money. Smooth and efficient management affects the satisfaction of the customers and staff members, indirectly. And of course, it encourages management committee in taking some needed decision for future enhancement of the bank. Now a days, managing a bank is tedious job up to certain limit. So software that reduces the work is essential. Thus, considering above necessities, the software for bank management has became necessary which would be useful in managing the bank more efficiently.

Our software will perform and fulfill all the tasks that any customer would desire. It is developed as a software program for managing the entire bank process related to customer accounts to keep each every track about their property and their various transaction processes efficiently. Hereby, our main objective is the customer’s satisfaction considering today’s faster world.

In the recent years, computers are included in almost all kind of works and jobs everyone come across in the routine. The availability of the software's for almost every process or every system has taken the world in its top-gear and fastens the day-to-day life.

So, we have tried our best to develop the software program for the Bank Management System where all the tasks to manage the bank system are performed easily and efficiently.

Thus, above features of this software will save transaction time and therefore increase the efficiency of the system. Requirements definition and management is recognized as a necessary step in the delivery of successful system s and software projects, discipline is also required by standards, regulations, and quality improvement initiatives. Creating and managing requirements is a challenge of IT, systems and product development projects or indeed for any activity where you have to manage a contractual relationship.

6.2 FUTURE ENHANCEMENT

For any system, present satisfaction is important, but is also necessary to see and visualizes the future scope. It is necessary for any system as the limitations that cannot be denied by anybody. These limitations, can be overcome by better technologies.

In my project, records of the customers are transactions are maintained. It will be helpful for the organization and customer.

REFERENCES

1. Python For Desktop Applications: How to develop, pack and deliver Python applications with TkInter by Tran Duc Loi
2. Python in A Nutshell: A Desktop Quick Reference, Third Edition by Alex Martelli , Anna Ravenscroft , Steve Holden
3. The Python Language Reference Manual (version 3.2) by Guido van Rossum, and Fred L. Drake, Jr. (Editor)
4. Dr. Geeta Sharma, “Study of Internet Banking Scenario in India”, International Journal of Emerging Research in Management &Technology, ISSN: 2278-9359, Volume 5, Issue 5, 2016, pp.43-48.
5. Anju Dagar, “Online Banking : Benefits and Related Issues”, International Journal of Commerce, Business and Management (IJCBM), ISSN: 2319–2828,Vol. 3, No. 5, 2014, pp.715-719.
6. Ebubeogu Amarachukwu Felix, “ Bank Customers Management System”, International Journal of Scientific & Technology Research Volume 4, Issue 08, 2015, pp.326- 343.
7. Mahmood Shah, “E-Banking Management: Issues, Solutions, and Strategies”, 2009.
8. Muhammad Abdus Sattar Titu and Md. Azizur Rahman, “ Online Banking System-Its Application in Some Selected Private Commercial Banks in Bangladesh”, IOSR Journal of Business and Management (IOSR-JBM) e-ISSN: 2278- 487X, Volume 9, Issue 4,2013, pp.37-44.
9. Bahman Saeidipour, Hojat Ranjbar and Saeed Ranjbar, “Adoption of Internet banking”, IOSR Journal of Business and Management (IOSR-JBM) e-ISSN: 2278-487X, Volume 11, Issue 2 2013, pp.46-51.
10. D.Amutha, “A Study of Consumer Awareness towards eBanking”, Int J Econ Manag Sci, ISSN: 2162-6359, Volume 5, issue 4, 2016, pp.1-4.

APPENDIX

SORCE CODE

```
import tkinter

from tkinter import ttkfrom BankDetails import Account

from tkinter import messagebox
from tkinter import font
import pickle
import os
import ctypes
import pathlib
user32 = ctypes.windll.user32
xpos = int((user32.GetSystemMetrics(0) - 800) / 2)
ypos = int((user32.GetSystemMetrics(1) - 600) / 2)
oldlist=[]
recordpos=0
wn = tkinter.Tk()
wn.geometry('800x600+{}+{}'.format(xpos, ypos))
wn.title('Bank Management System')
ft = font.Font(size=12)
accno = tkinter.IntVar()
accname = tkinter.StringVar()
acctype = tkinter.StringVar()
amount = tkinter.IntVar()
tamount = tkinter.IntVar()
currentscreen=0
def writeAccount():
    account = Account()
    account.createAccount(accno.get(), accname.get(), acctype.get(), amount.get())
    writeAccountsFile(account)
```

```
    messagebox.showinfo('BMS', 'Account created')
    clearvalues()
def displayAll():
    showframe(7)
    file = pathlib.Path("accounts.data")
    for i in table.get_children():
        table.delete(i)
    if file.exists():
        infile = open('accounts.data', 'rb')
        mylist = pickle.load(infile)
  #tempList.sort(key=lambda e: e[1], reverse=True)
        for i, item in enumerate(mylist, start=1):
            table.insert("", "end", values=(i, item.accNo,item.name,item.type,item.deposit))
        infile.close()
    else:
        messagebox.showinfo('BMS',"No records to display")
def showframe(num):
    clearvalues()
    mainframe.pack_forget()
    eval('frm{}'.format(num)).pack()
def getbalancedetails():
    file = pathlib.Path("accounts.data")
    if file.exists():
        infile = open('accounts.data', 'rb')
        mylist = pickle.load(infile)
        infile.close()
        found = False
        for item in mylist:
            if item.accNo == accno.get():
                amount.set(item.deposit)
                accname.set(item.name)
                acctype.set(item.type)
                found = True
    else:
```

```
        messagebox.showinfo('BMS',"No Account were open ")
    if not found:
        messagebox.showinfo('BMS',"No existing record with this number")
def trans(ttype):
    global  oldlist
    if ttype==1:
        oldlist[recordpos - 1].deposit += tamount.get()
    else:
        if tamount.get()<=int(amount.get()):
            oldlist[recordpos-1].deposit-=tamount.get()
        else:
            messagebox.showinfo('BMS','Insufficient Balance')
            rew os.remove('accounts.data')
    outfile = open('newaccounts.data', 'wb')
    pickle.dump(oldlist, outfile)
    outfile.close()
    os.rename('newaccounts.data', 'accounts.data')
    clearvalues()
    messagebox.showinfo('BMS',"updated....")
def deleteAccount():
    newlist = []
    for item in oldlist:
        if item.accNo != accno.get():
            newlist.append(item)
    os.remove('accounts.data')
    outfile = open('newaccounts.data', 'wb')
    pickle.dump(newlist, outfile)
    outfile.close()
    os.rename('newaccounts.data', 'accounts.data')
    clearvalues()
    messagebox.showinfo('BMS',"Account Deleted.....")
def modifyAccount():
    global recordpos,oldlist
    file = pathlib.Path("accounts.data")
```

```
    if file.exists():
        infile = open('accounts.data', 'rb')
        oldlist = pickle.load(infile)
        infile.close()
        recordpos=0
        for item in oldlist:
            recordpos+=1
            if item.accNo == accno.get():
                accname.set(item.name)
                acctype.set(item.type)
                amount.set(item.deposit)
                break
        else:
            messagebox.showinfo('BMS',"No such account exists")
    else:
        messagebox.showinfo('BMS',"No accounts were open")
def updatedetails():
    oldlist[recordpos-1].name=accname.get()
    oldlist[recordpos-1].type =acctype.get()
    os.remove('accounts.data')
    outfile = open('newaccounts.data', 'wb')
    pickle.dump(oldlist, outfile)
    outfile.close()
    os.rename('newaccounts.data', 'accounts.data')
    clearvalues()
    messagebox.showinfo('BMS',"updated....")
def writeAccountsFile(account):
    file = pathlib.Path("accounts.data")
    if file.exists():
        infile = open('accounts.data', 'rb')
        oldlist = pickle.load(infile)
        oldlist.append(account)
        infile.close()
        os.remove('accounts.data')
```

```
    else:
        oldlist = [account]
    outfile = open('newaccounts.data', 'wb')
    pickle.dump(oldlist, outfile)
    outfile.close()
    os.rename('newaccounts.data', 'accounts.data')
def showlink(opt):
    frm=links.index(opt)
    if frm<6:
        showframe(frm+1)
    elif frm==6:
        displayAll()
    elif frm==7:
        wn.destroy()
def goback(frm):
    eval('frm{}'.format(frm)).pack_forget()
    mainframe.pack()
def clearvalues():
    accno.set(0)
    accname.set('')
    acctype.set('Savings')
    amount.set(0)
    tamount.set(0)
mainframe = tkinter.Frame(wn)
links = ['New Account', 'Deposit', 'Withdrawal', 'Balance Enquiry', 'Update Details', 'Close Account
'Account List', 'Exit']
ft1=font.Font(size=24)
tkinter.Label(wn,font=ft1,fg='green',text='BANK MANAGEMENT SYSTEM').pack(pady=15)
for item in links:
  tkinter.Button(mainframe, width=20, text=item, font=ft, bg='blue', fg='white',
   command=lambda txt=item: showlink(txt)).pack(pady=4)
frm2 = tkinter.Frame(wn)
tkinter.Label(frm2, text='Account No.', font=ft, justify=tkinter.LEFT, anchor="w").grid(sticky=tkinter.W,
row=0, column=0)
```

```
tkinter.Label(frm2, text='Name of the Account Holder.', font=ft, justify=tkinter.LEFT, anchor="w").grid(
sticky=tkinter.W, row=1, column=0)

tkinter.Label(frm2, text='Type of Account', font=ft, justify=tkinter.LEFT,
anchor="w").grid(sticky=tkinter.W, row=2,  column=0)

tkinter.Label(frm2, text='Current Balance', font=ft, justify=tkinter.LEFT,
anchor="w").grid(sticky=tkinter.W, row=3,  column=0)tkinter.Entry(frm2, textvariable=accno,
font=ft).grid(row=0, column=1)tkinter.Button(frm2, text='Get Details', font=ft,
command=modifyAccount).grid(row=0, column=2)tkinter.Label(frm2, textvariable=accname,
font=ft).grid(row=1, column=1, columnspan=2)tkinter.Label(frm2, textvariable=acctype,
font=ft).grid(row=2, column=1)tkinter.Label(frm2, textvariable=amount, font=ft).grid(row=3,
column=1)tkinter.Label(frm2, text='Deposit Amount', font=ft, justify=tkinter.LEFT,
anchor="w").grid(sticky=tkinter.W, row=4,  column=0)

tkinter.Entry(frm2, textvariable=tamount, font=ft).grid(row=4, column=1, columnspan=1)

tkinter.Button(frm2, text='Save', font=ft, command=lambda :trans(1)).grid(row=5, column=0)

tkinter.Button(frm2, text='Back', font=ft, command=lambda: goback(2)).grid(row=5, column=1)

frm3 = tkinter.Frame(wn)

tkinter.Label(frm3, text='Account No.', font=ft, justify=tkinter.LEFT, anchor="w").grid(sticky=tkinter.W,
row=0, column=0)

tkinter.Label(frm3, text='Name of the Account Holder.', font=ft, justify=tkinter.LEFT,
anchor="w").grid(sticky=tkinter.W, row=1, column=0)

tkinter.Label(frm3, text='Type of Account', font=ft, justify=tkinter.LEFT,
anchor="w").grid(sticky=tkinter.W, row=2,  column=0)

tkinter.Label(frm3, text='Current Balance', font=ft, justify=tkinter.LEFT,
anchor="w").grid(sticky=tkinter.W, row=3, column=0)

tkinter.Entry(frm3, textvariable=accno, font=ft).grid(row=0, column=1)

tkinter.Button(frm3, text='Get Details', font=ft, command=modifyAccount).grid(row=0, column=2)

tkinter.Label(frm3, textvariable=accname, font=ft).grid(row=1, column=1, columnspan=2)

tkinter.Label(frm3, textvariable=acctype, font=ft).grid(row=2, column=1)

tkinter.Label(frm3, textvariable=amount, font=ft).grid(row=3, column=1)

tkinter.Label(frm3, text='Amount to Withdraw', font=ft, justify=tkinter.LEFT,
anchor="w").grid(sticky=tkinter.W, row=4,  column=0)tkinter.Entry(frm3, textvariable=tamount,
font=ft).grid(row=4, column=1, columnspan=1)

tkinter.Button(frm3, text='Save', font=ft, command=lambda :trans(2)).grid(row=5, column=0)

tkinter.Button(frm3, text='Back', font=ft, command=lambda: goback(3)).grid(row=5, column=1)

frm4 = tkinter.Frame(wn)

tkinter.Label(frm4, text='Account No.', font=ft, justify=tkinter.LEFT, anchor="w").grid(sticky=tkinter.W,
row=0, column=0)

tkinter.Label(frm4, text='Name of the Account Holder.', font=ft, justify=tkinter.LEFT, anchor="w").grid(
sticky=tkinter.W, row=1, column=0)

tkinter.Label(frm4, text='Type of Account', font=ft, justify=tkinter.LEFT,
anchor="w").grid(sticky=tkinter.W, row=2,  column=0)
```

```
tkinter.Label(frm4, text='Initial Deposit', font=ft, justify=tkinter.LEFT, anchor="w").grid(sticky=tkinter.W,
row=3,  column=0)tkinter.Entry(frm4, textvariable=accno, font=ft).grid(row=0, column=1)

tkinter.Label(frm4, textvariable=accname, font=ft).grid(row=1, column=1, columnspan=2)

tkinter.Label(frm4, textvariable=acctype, font=ft).grid(row=2, column=1)

tkinter.Label(frm4, textvariable=amount, font=ft).grid(row=3, column=1)

tkinter.Button(frm4, text='Get Balance', font=ft, command=getbalancedetails).grid(row=0, column=2)

tkinter.Button(frm4, text='Back', font=ft, command=lambda: goback(4)).grid(row=4, column=1)

frm5 = tkinter.Frame(wn)

tkinter.Label(frm5, text='Account No.', font=ft, justify=tkinter.LEFT, anchor="w").grid(sticky=tkinter.W,
row=0,column=0)

tkinter.Label(frm5, text='Name of the Account Holder.', font=ft, justify=tkinter.LEFT,
anchor="w").grid(sticky=tkinter.W, row=1, column=0)

tkinter.Label(frm5, text='Type of Account', font=ft, justify=tkinter.LEFT,
anchor="w").grid(sticky=tkinter.W, row=2  column=0)

tkinter.Entry(frm5, textvariable=accno, font=ft).grid(row=0, column=1, columnspan=1)

tkinter.Button(frm5, text='Get Details', font=ft, command=modifyAccount).grid(row=0, column=2)

tkinter.Entry(frm5, textvariable=accname, font=ft).grid(row=1, column=1, columnspan=2)

tkinter.Radiobutton(frm5, text='Savings', variable=acctype, value='Savings', font=ft).grid(row=2,
column=1)

tkinter.Radiobutton(frm5, text='Current', variable=acctype, value='Current', font=ft).grid(row=2,
column=2)

tkinter.Button(frm5, text='Save', font=ft, command=updatedetails).grid(row=4, column=0)

tkinter.Button(frm5, text='Back', font=ft, command=lambda: goback(5)).grid(row=4, column=1)

frm6 = tkinter.Frame(wn)

tkinter.Label(frm6, text='Account No.', font=ft, justify=tkinter.LEFT, anchor="w").grid(sticky=tkinter.W,
row=0, column=0)

tkinter.Label(frm6, text='Name of the Account Holder.', font=ft, justify=tkinter.LEFT, anchor="w").grid(

    sticky=tkinter.W, row=1, column=0)

tkinter.Label(frm6, text='Type of Account', font=ft, justify=tkinter.LEFT,
anchor="w").grid(sticky=tkinter.W, row=2,column=0)

tkinter.Entry(frm6, textvariable=accno, font=ft).grid(row=0, column=1, columnspan=1)

tkinter.Button(frm6, text='Get Details', font=ft, command=modifyAccount).grid(row=0, column=2)

tkinter.Entry(frm6, textvariable=accname, font=ft).grid(row=1, column=1, columnspan=2)

tkinter.Radiobutton(frm6, text='Savings', variable=acctype, value='Savings', font=ft).grid(row=2,
column=1)

tkinter.Radiobutton(frm6, text='Current', variable=acctype, value='Current', font=ft).grid(row=2,
column=2)

tkinter.Button(frm6, text='Delete Account', font=ft, command=deleteAccount).grid(row=4, column=0)
```

```
tkinter.Button(frm6, text='Back', font=ft, command=lambda: goback(6)).grid(row=4, column=1)

frm7=tkinter.Frame(wn)

cols = ('S.No', 'Account No.','Account Holder Name','Account Type', 'Current Balance')

table = ttk.Treeview(frm7, columns=cols, show='headings')

colsize=(50,50,100,200,100)

for i in range(5):

    table.column('#' + str(i), width=colsize[i],minwidth=50, stretch=1)

    table.heading(i, text=cols[i])

table.grid(row=1, column=0, columnspan=4)

tkinter.Button(frm7, text='Back', font=ft, command=lambda: goback(7)).grid(row=2,
column=0,columnspan=4)

frm1 = tkinter.Frame(wn)

tkinter.Label(frm1, text='Account No.', font=ft, justify=tkinter.LEFT, anchor="w").grid(sticky=tkinter.W,
row=0,column=0)

tkinter.Label(frm1, text='Name of the Account Holder.', font=ft, justify=tkinter.LEFT,
anchor="w").grid(sticky=tkinter.W, row=1, column=0)

tkinter.Label(frm1, text='Type of Account', font=ft, justify=tkinter.LEFT,
anchor="w").grid(sticky=tkinter.W, row=2,  column=0)

tkinter.Label(frm1, text='Initial Deposit', font=ft, justify=tkinter.LEFT, anchor="w").grid(sticky=tkinter.W,
row=3,column=0)

tkinter.Entry(frm1, textvariable=accno, font=ft).grid(row=0, column=1, columnspan=2)

tkinter.Entry(frm1, textvariable=accname, font=ft).grid(row=1, column=1,
columnspan=2)tkinter.Radiobutton(frm1, text='Savings', variable=acctype, value='Savings',
font=ft).grid(row=2, column=1)tkinter.Radiobutton(frm1, text='Current', variable=acctype,
value='Current', font=ft).grid(row=2, column=2)tkinter.Entry(frm1, textvariable=amount,
font=ft).grid(row=3, column=1, columnspan=2)

tkinter.Button(frm1, text='Save', font=ft, command=writeAccount).grid(row=4, column=0)

tkinter.Button(frm1, text='Cancel', font=ft, command=lambda: goback(1)).grid(row=4, column=1)

mainframe.pack()

wn.mainloop()
```