BACHELOR OF SCIENCE IN INFROMATION TECHNOLOGY

PROGRAMMING PROGECT: BIT 2206

SAVINGS AND CREDIT COOPERATIVE MANAGEMENT SYSTEM FOR LOCA SAVINGS AND CREDIT COOPERATIVE

BY

NJOROGE MORRIS KIMANI

16/05387

PROGECT SYSTEM DESIGN SPECIFICATION IN PARTIAL FULFILMENT FOR THE REQUIREMENTS FOR THE AWARD OF A DEGREE IN INFORMATION TECHNOLOGY

PRESENTED TO: Dr. LUCY MBURU

Declaration

I declare that this project is my original work and has not been presented in any other college or university for the award of a Diploma or a Degree.

Student

Name………………………………………………. ……. Date……………………………………….

Signature……………………………………………….

Table of Contents

Contents

Abstract

The motivation behind undertaking this project was the inefficiency of the manual system being used by the SACCO which resulted in many errors in the data that was in their possession and also degrading the quality of service they were giving their members.

Therefore, the goal of this project is to automate their manual systems hence eliminating errors that occur during data collection and computations and overall improvement in service delivery to the members of the SACCO.

1.0 Introduction

A Software Design Document that provides a description of a software product written by a software designer to give a software development team guidance to the architecture of the software product or how the software is to be built. Within this document is a narrative and graphical documentation in the form of use case models, class diagrams, sequence diagrams, activity diagrams and other supporting information that will prove useful to the development of this project.

This Design Document provides in-depth details for the design of the SACCO Management system for LOCA SACCO.

At this stage (system design) the primary objective is producing a design that will meet the expectations of a high-quality information system which are:

(i) Security: It can defend itself against unauthorized personnel trying to access it.
(ii) Reliable: Always available when needed.
(iii) Can integrate easily with other systems.
(iv) Easy to use.
(v) Provides timely and useful data.

In this document, emphasis will be laid on four key areas: data, interfaces, architecture and components of the software under production.

1.1 Purpose

The purpose of this document is to give insight on the design of the system as candidly as possible to enable development of the software to take off with complete knowledge of what is being built and how.

1.2 Scope

This Software Design Document will cover the basic but critical parts of the system hence acting as a proof of concept that it is feasible for largescale use with emphasis being on data collection, manipulating and storage of information. The system will be used in conjunction with existing systems.

This document contains a complete description of the design of the SACCO Management System. Its basic structure is a Client-Server paradigm. Basic forms will be created and coded in Visual Basic.

Staff that will be designated to use the system will be able to make changes to the data that has already been entered into the system.

1.3 System Design Constraints

The user requirements' specifications have directly constrained the system design. This describes the functions that are required by the user which must be implemented as part of the design without leaving out the environmental constraints resulting from the hardware and software environment implementation. These include:

- Integration of data from the current into the new system.
- Implementing an interface between two different applications will be difficult to implement.
- Hardware limitation such as no ownership of a computing device.

1.4 Design Goals and Objectives

The goal of the SACCO Management System is to design a system which delivers the functions required by LOCA SACCO to support its business. The importance of software design is:

- Allows proper understanding of the system hence requirements can be met.
- Prevents redundancy and increases reusability.
- Helps to mitigate unknown risks.
- The resulting system will be stable with very lo probabilities of failure.

Objectives

- **Security:** The system should be able to implement access control methods on who can view which data.
- **Ease of use:** The system should be user –friendly.
- **Flexibility:** The system should enable new requirements of the SACCO to be incorporated without too much complexity.
- **Availability:** The system should always be available for use at any point in time the SACCO needs to utilize it.

1.5 Document Overview

The Software Design Document has been divided into sections and subsections which are:

- **Section 2, Architectural Design:** It specifies all the design entities that collaborate to perform all the functions included in the system.
- **Section 3:** contains File and Data Structure Design
- **Section 4:** Contains Normalization whose purpose is to organize data in a way that optimizes logical storage of data.
- **Section 5:** It is the conclusion of the System Design Document. It indicates how the software will be meeting its user requirements.

2.0 System Architecture

2.1 Introduction

Systems Architecture is a response to the conceptual and practical difficulties of the description and the design of complex systems. It can be viewed as a blueprint of a software system. Here, a software developer analyzes the requirements of the system, determines the components the system will need as well as support the project even after its production is complete.

2.2 Client-Server Model

It is a distributed application structure that partitions tasks between providers of a service (server) and service requesters (clients). It is composed of the following:

(i) **Servers:** they offer services to subsystems. There are many types of servers, examples include: print servers which avail printing services, database servers which provide access to databases and file servers which offer file management services.

(ii) **Clients:** They are software that request services from a server.

(iii) **Network:** It is a connection between two or more computing devices for the purpose of information and resource sharing. It is used by clients to request and obtain services from a server.

Clients access the services provided by a server through remote procedure calls and must be aware of which services a particular server offers. In the case of a server, it must be able to know how many requests it can handle.

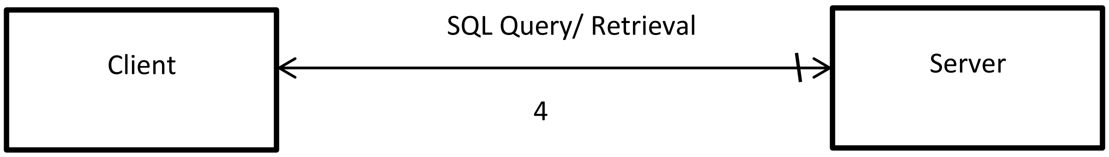
2.3 Design Approach

To solve a problem, it is easier if the said problem is first broken down into smaller and more manageable proportions to better understand it. This is the best design approach there is in developing software also known as modularization. This is breaking down a complex system into smaller subsystems.

Top-down approach design will be implemented as it describes the overall control architecture of the application before designing the individual modules.

2.4 Architectural Design

It's a concept that focuses on the components of a system and unifies them into a functional fully integrated whole. The proposed system will be developed using a two-tier architectural program. The module of the system being run on workstation will hold the bulk of the system as it will house both the user interface and the application logic while the server will run the database only.



2.5 Logical Design

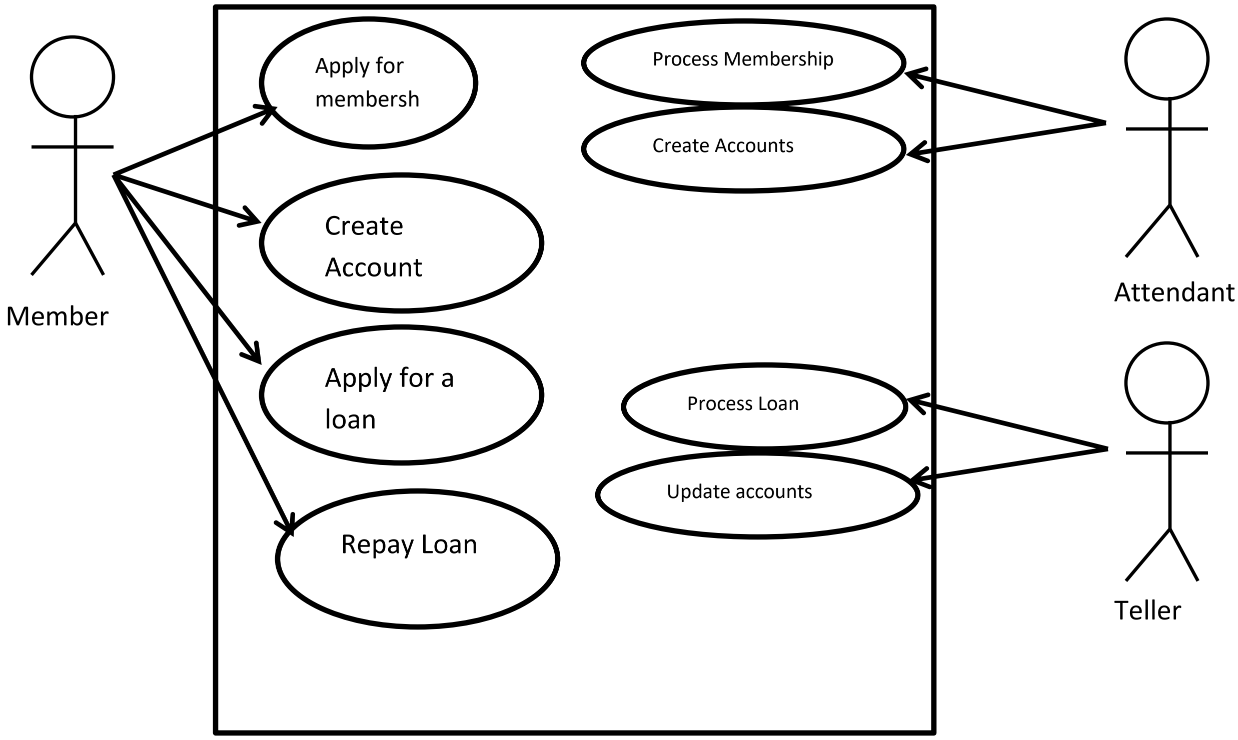
Its main concern is the processes being performed by the system. It describes what is to be done and how it will be done and is not in any way associated with hardware and software of the system.

2.6 Use Case Diagrams

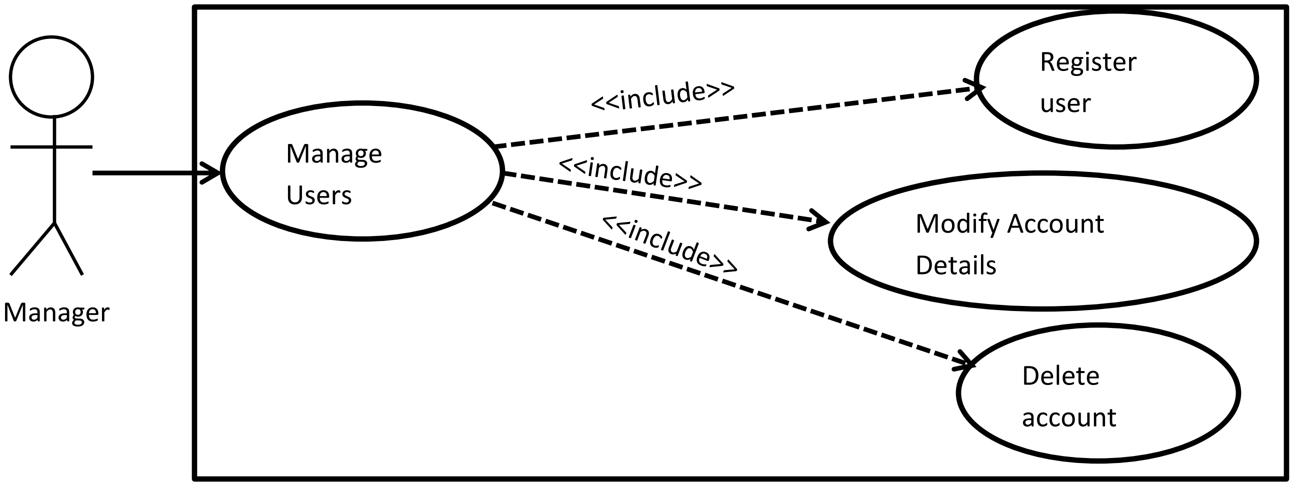
They are at times referred to as behavior diagrams used to describe a set of actions (use cases) that a system (subject) should or can perform in collaboration with one or more external users of the system (actors). There are three main actors in this system namely:

- **Manager:** Holds an administrative role in the system with the full ability to oversee the various users and accounts.
- **Attendant:** This user will be responsible for adding new members into the system and helping them (or already existing ones) to create new accounts.
- **Teller:** This actor will be charged with the responsibility of controlling the accounts of the members. This includes issuing loans, repaying loans and depositing money into the various accounts that are present.
- **Member:**

Below is a use case diagram depicting the attendant, member and teller actors and their roles:



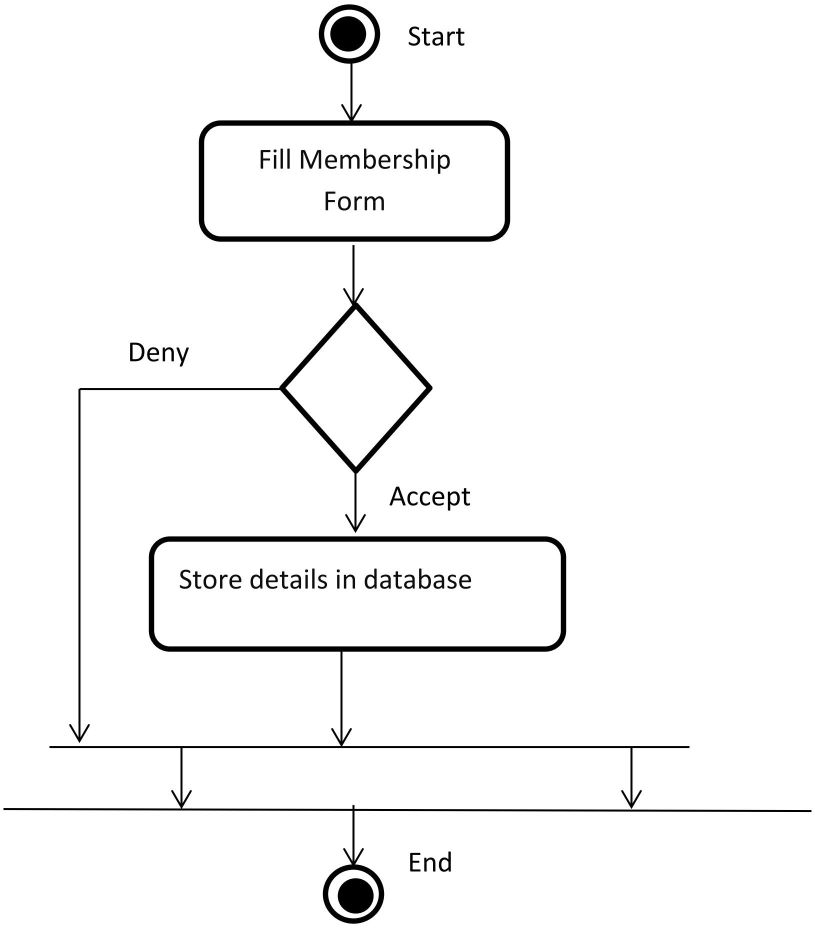
The use case diagram below represents the administrative user and their rolves



2.7 Activity Diagrams

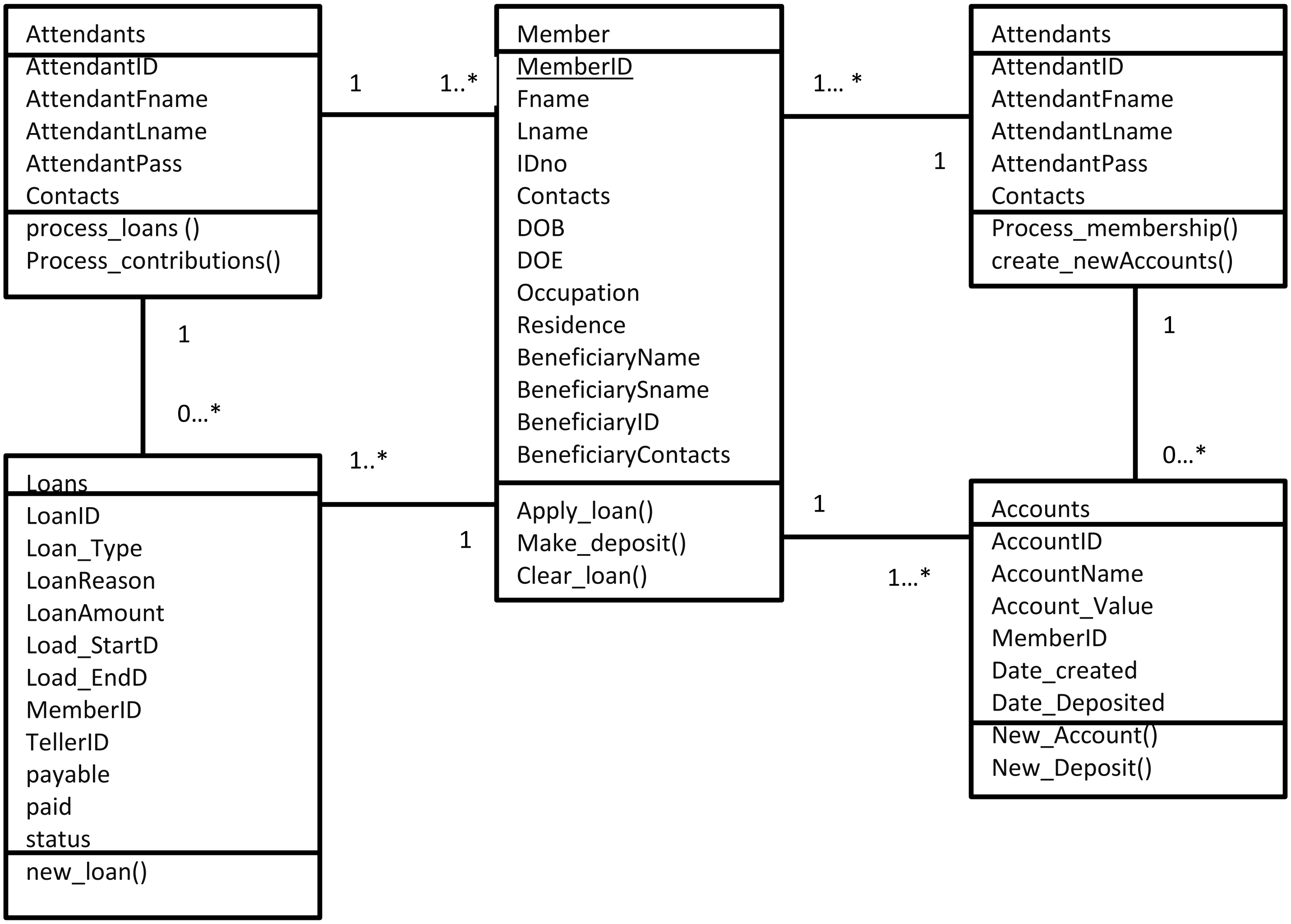
They are basically similar to flowcharts and depict flow from one activity to another which can be described as an operation of the system. The activity could either be sequential, branched or concurrent. Its purpose is to capture the dynamic nature of a system.

Below is an activity diagram of adding a member into the system:



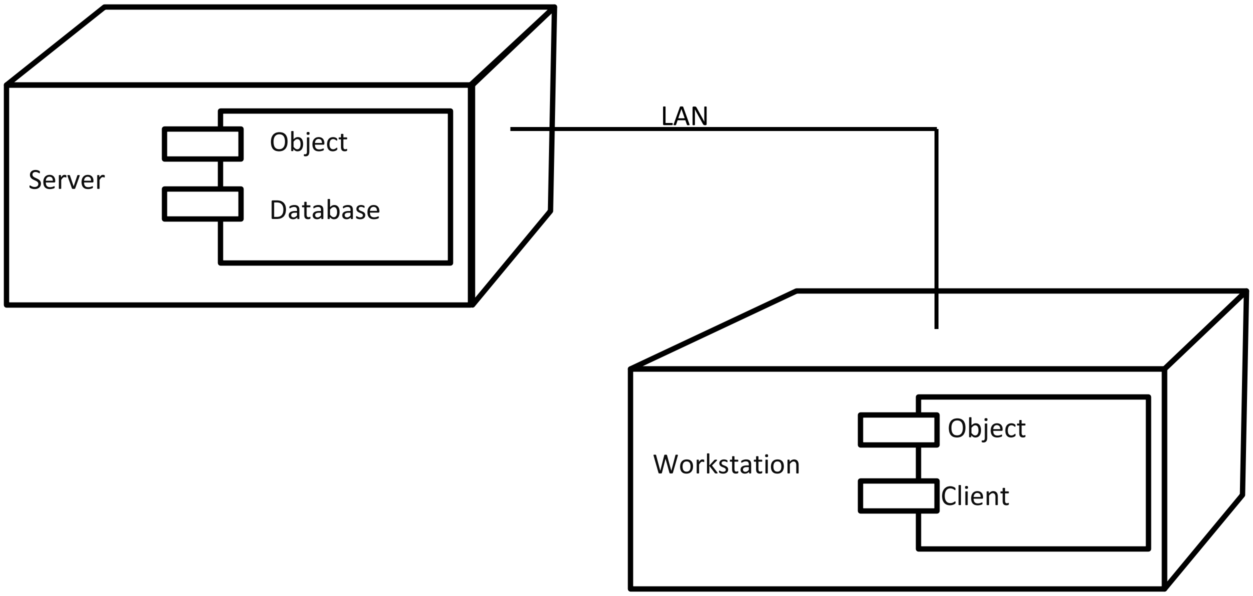
2.8 Class Diagrams

They depict a static state of the system by use of classes, their attributes, methods/operations and relationships between objects.



2.9 Deployment Diagrams

They are used to visualize the topology of the physical components of a computer system where the software components are deployed.



3.0 Database Design

3.1 Introduction

A database is an organized collection of data stored in a way that minimizes duplication of information and provides consistency of data and access for users. The standard application program interface for many relational databases is Structured Query Language (SQL). These are commands/statements that are used to create databases, insert and modify information into the database, generate reports and even delete a database.

The database for this system will be implemented by use of MySQL 5.6.

Reasons for choice of MySQL 5.6

- On-demand scalability
- It is globally recognized as one of the most secure database management systems globally.
- Assurance of 24/7 availability
- Low total cost of ownership due its easy management which saves time during trouble-shooting.
- Meets the needs of demanding applications while providing optimum speed and unique memory caches for enhanced performance.

A good database must have the following qualities:

- **Atomicity:** If one part of a transaction fails, the whole transaction fails and the state of the database remains unchanged.
- **Consistency:** any data entered into the database is valid according to all the defined rules.
- **Isolation:** An incomplete transaction remains invisible to other transactions until it is complete.
- **Durability:** Once a transaction is commited to the database, it will remain so even in the event of a failure or errors in the database.

3.2 Normalization

It is a process of decomposing tables to eliminate data redundancy and unwanted characteristics such as insert, update and delete anomalies. There are several steps involved where data is put into tabular form and duplicated data is eliminated form related tables.

The table below show the whole process from the un-normalized for to the third normal form.

UNF	**1st NF**	**2nd NF**	**3RD NF**
MemberID	**MemberID (PK)**	**MemberID (PK)**	**MemberID (PK)**
ID/Passport_No	**AccountID (PK)**	ID/Passport_No	ID/Passport_No
Contacts	**Loan_ID (PK)**	Contacts	Contacts
Names	**AttendantID (PK)**	First_Name	First_Name
Date_of_Birth	**TellerID (PK)**	Surname	Surname
Date_Of_Application	ID/Passport_No	Date_of_Birth	Date_of_Birth
Beneficiary_Names	Contacts	Date_Of_Application	Date_Of_Application
Beneficiary_Contacts	First_Name	Beneficiary_First_Name	Beneficiary_First_Name
AccountID	Surname	Benefiiary_Surname	Benefiiary_Surname
Account_Name	Date_of_Birth	Beneficiary_Contacts	Beneficiary_Contacts
Account_Value	Date_Of_Application		
Loan_ID	Beneficiary_First_Name	**AccountID (PK)**	**AccountID (PK)**
Loan_Type	Benefiiary_Surname	Account_Name	Account_Name
Loan_Amount	Beneficiary_Contacts	Account_Value	Account_Value
LoanReason	Account_Name		**MemberID (FK)**
Loan_Start_Date	Account_Value	**Loan_ID (PK)**	
Laon_Pay_Date	Loan_Type	Loan_Type	**Loan_ID (PK)**
Loan_Status	Loan_Amount	Loan_Amount	Loan_Type
Loan_Payable	LoanReason	LoanReason	Loan_Amount
Loan_Paid	Loan_Start_Date	Loan_Start_Date	LoanReason
AttendantID	Laon_Pay_Date	Laon_Pay_Date	Loan_Start_Date
AttendantFname	Loan_Status	Loan_Status	Laon_Pay_Date
AttendantLname	Loan_Payable	Loan_Payable	Loan_Status
AttendantPass	Loan_Paid	Loan_Paid	Loan_Payable
Attendant_Contacts	AttendantFname		Loan_Paid
TellerID	AttendantLname	**AttendantID (PK)**	**MemberID (FK)**
Teller_Fname	AttendantPass	AttendantFname	**TellerID (FK)**
Teller_Sname	Attendant_Contacts	AttendantLname	
TellerPass	Teller_Fname	AttendantPass	**AttendantID (PK)**
Teller_Contacts	Teller_Sname	Attendant_Contacts	AttendantFname
	TellerPass		AttendantLname
	Teller_Contacts	**TellerID (PK)**	AttendantPass
		Teller_Fname	Attendant_Contacts
		Teller_Sname	
		TellerPass	**TellerID (PK)**
		Teller_Contacts	Teller_Fname
			Teller_Sname
			TellerPass
			Teller_Contacts

3.3 Database Description

Members Table			
Field Name	**Data Type**	**Size**	**Remark**
MemberID	Integer	40	Primary Key
Fname	Varchar	40	
Lname	Varchar	40	
IDno	Varchar	40	
Contacts	Varchar	40	
DOB	Date		
DOE	Date		
Occupation	Varchar	40	
Residence	Varchar	40	
BeneficiaryName	Varchar	40	
BeneficiaryName	Varchar	40	
BeneficiarySname	Varchar	40	
BeneficiaryID	Varchar	40	
BeneficiaryContacts	Varchar	40	

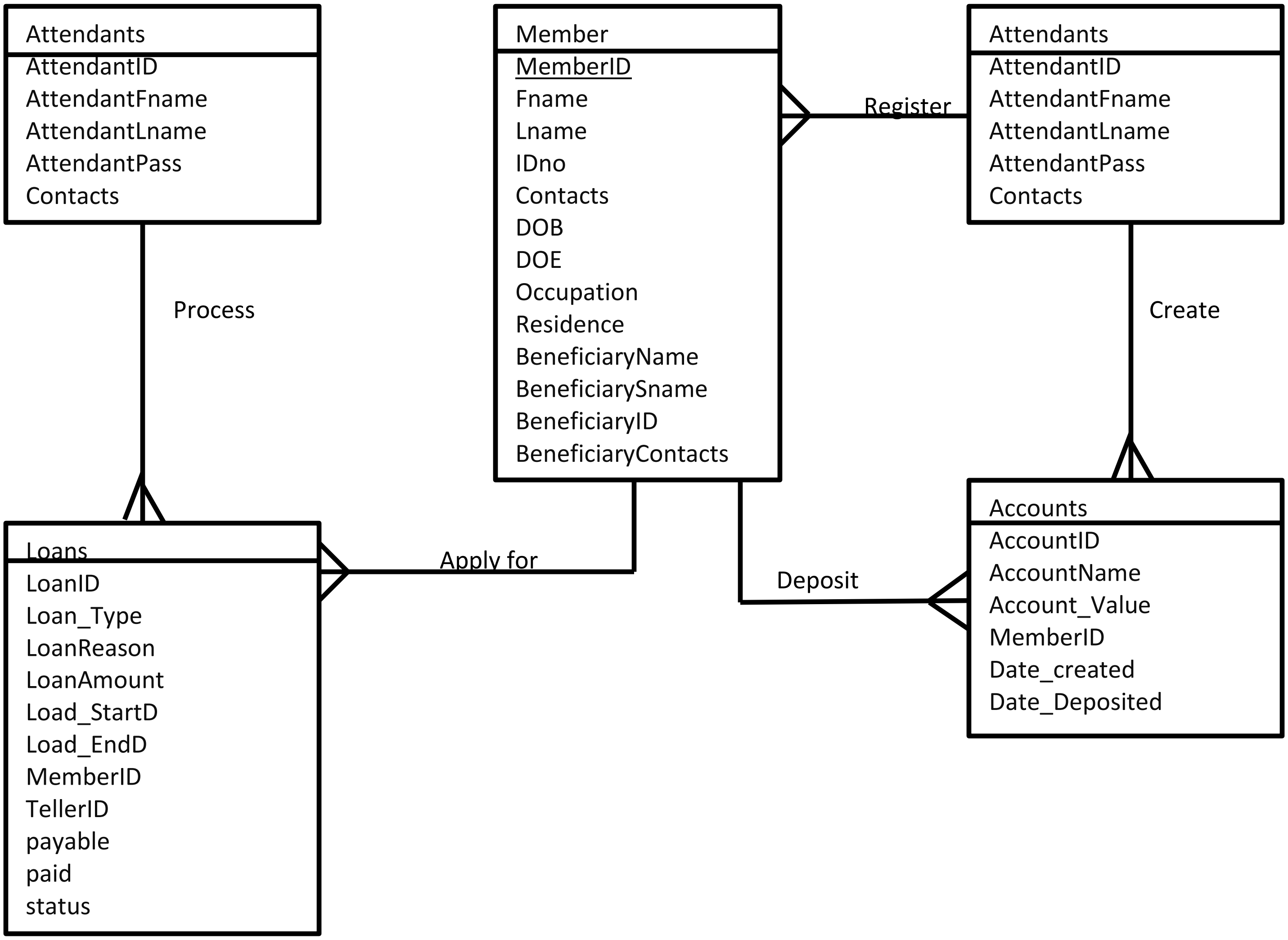
Loans Table			
Field Name	**Data Type**	**Size**	**Remark**
LoanID	Int	40	Primary Key
Loan_Type	Varchar	40	
LoanReason	Varchar	200	
Loan amount	Double	10,2	
Load_Start_Date	Date		
Load_End_Date	Date		
MemberID	Int	40	Foreign Key
TellerID	Int	40	Foreign Key
payable	Decimal	10,2	
paid	Decimal	10,2	
status	varchar	40	

Accounts Table			
Field Name	**Data Type**	**Size**	**Remark**
AccoutnID	Int	40	Primary Key
AccountName	varchar	40	
Account_VAlue	decimal	10,2	
MemberID	Int	40	Foreign Key
Date_Created	Date		
Date_Deposited	Date		

Tellers Table			
Field Name	**Data Type**	**Size**	**Remark**
TellerID	int	40	Primary Key
TellerFname			
TellerLname			
TellerPass			
Contacts			

Attendants Table			
Field Name	**Data Type**	**Size**	**Remark**
AttendantID	int	40	Primary Key
AttendantFname			
AttendantLname			
AttendantPass			
Contacts			

3.3 Entity Relationship Diagram



4.0 User Interface Design

4.1 Internal Machine Interfaces

A client-server network will first need to be created before the system is installed in both the client and server. The database will be installed in the server and the user interface in the client machines. Records added or updated by the user will automatically reflect on server.

4.2 External System Interfaces

Resources such as printers will be available on the network while computer accessories such as a keyboard and mouse will only be usable on one computer only. The server and workstations should have antivirus installed to protect them against malware, viruses and other malicious software that will threaten their operations. The antivirus should be frequently updated online hence internet access will be necessary.

4.3 Human Interfaces

All the workstations will allow the users to interact with the system after they have been authenticated to do so. The server however will not be accessible to them. This is because it will have a command-line interface which only an expert will be able to handle. It will also minimize the possibility of unknown errors from occurring.

4.4 Interface Design

This depicts how users will be able to interact with the system to perform the actions they require. They will do so via forms made using Visual Basic.

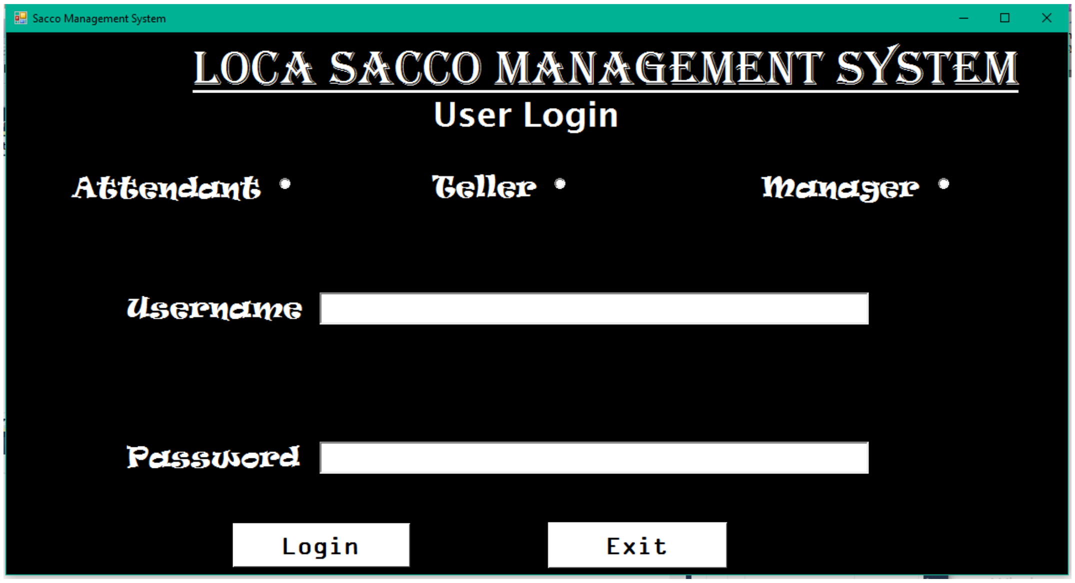
4.4.1 Description of the user interface

- **Command buttons:** They will issue commands to the system such as: open, save, edit, delete, search etc.
- **Command line:** The user can write SQL statements to query the database.
- **Pop-ups:** They will serve as error messages telling the user what error is occurring and why.

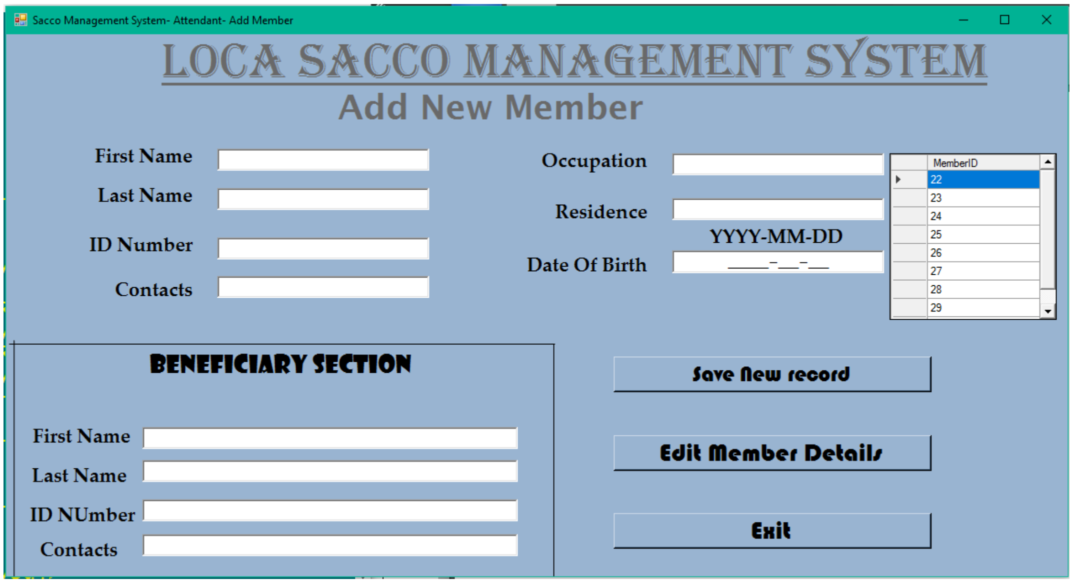
4.4.2 Description of the user interface

Screen images depict what the user will see when performing certain tasks in the system. Below are some of those designs that will be used in the system.

4.4.2.1 Login Form

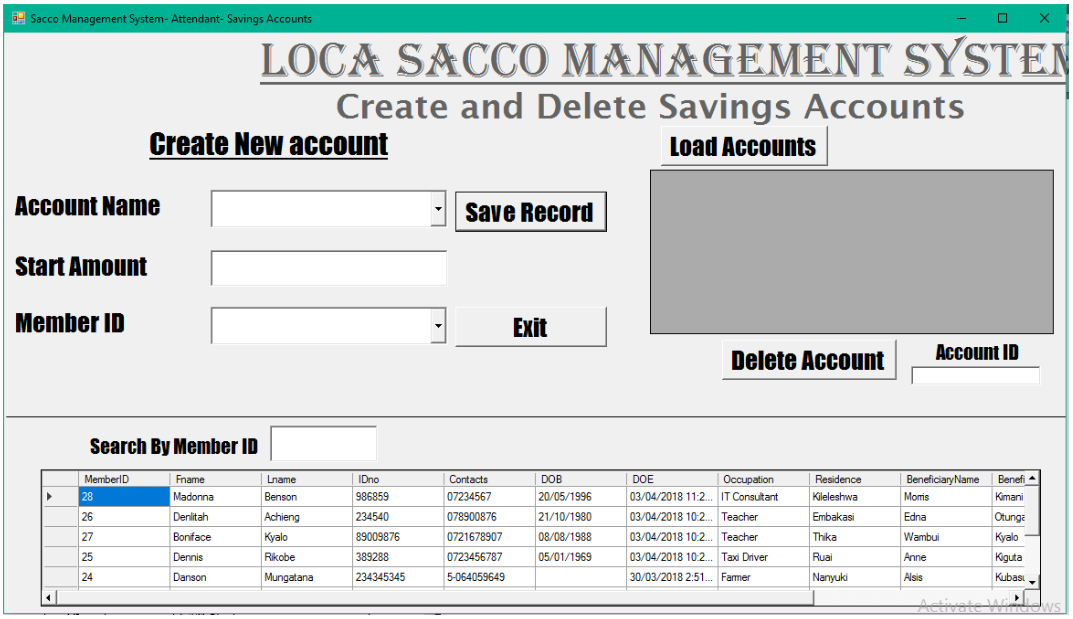


4.4.2.2 Add New Member Form



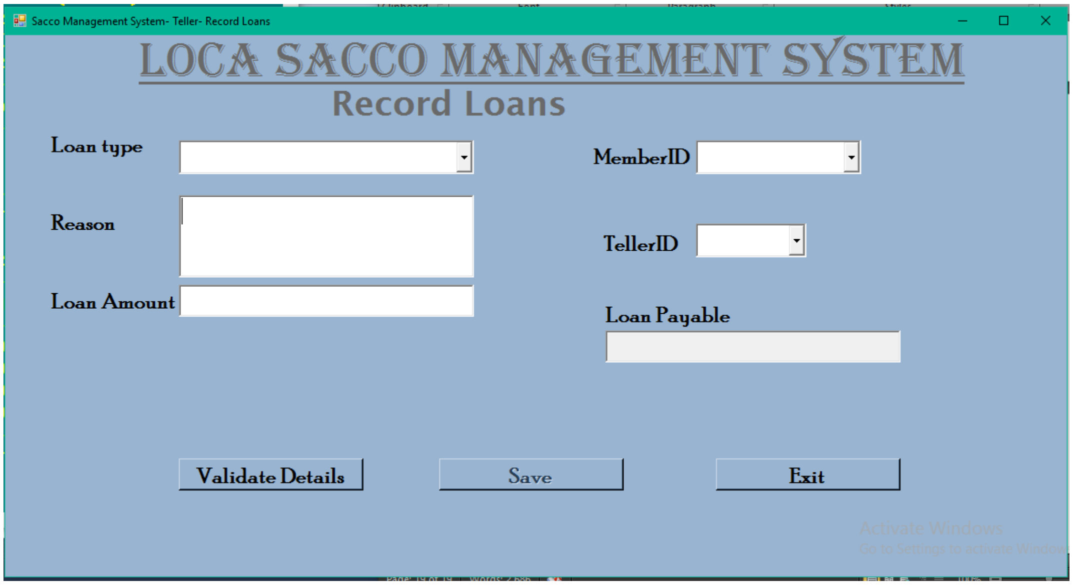
MemberID
22
23
24
25
26
27
28
29

4.4.2.3 Add New/Remove Member Account Form

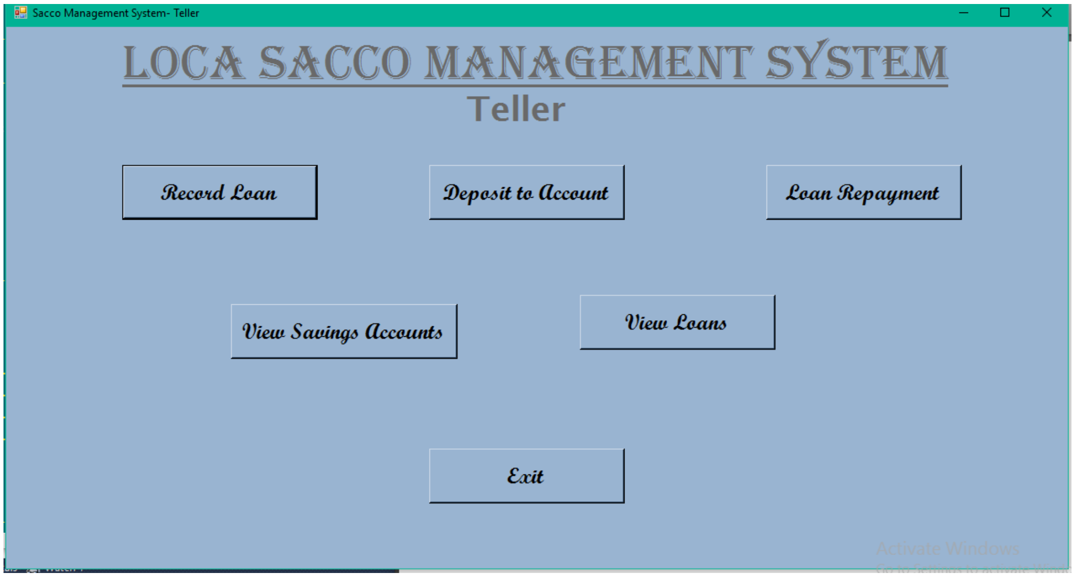


	MemberID	Fname	Lname	IDno	Contacts	DOB	DOE	Occupation	Residence	BeneficiaryName	Benefi
▶	28	Madonna	Benson	986859	07234567	20/05/1996	03/04/2018 11:2...	IT Consultant	Kileleshwa	Morris	Kimani
	26	Denlitah	Achieng	234540	078900876	21/10/1980	03/04/2018 10:2...	Teacher	Embakasi	Edna	Otunga
	27	Boniface	Kyalo	89009876	0721678907	08/08/1988	03/04/2018 10:2...	Teacher	Thika	Wambui	Kyalo
	25	Dennis	Rikobe	389288	0723456787	05/01/1969	03/04/2018 10:2...	Taxi Driver	Ruai	Anne	Kiguta
	24	Danson	Mungatana	234345345	5-064059649		30/03/2018 2:51...	Farmer	Nanyuki	Alsis	Kubasu

4.4.2.3 Loans Form


Sacco Management System- Teller- Record Loans
LOCA SACCO MANAGEMENT SYSTEM
Record Loans
Loan type
MemberID
Reason
TellerID
Loan Amount
Loan Payable
Validate Details
Save
Exit
Activate Windows
Go to Settings to activate Windows

4.4.2.4 Teller's Main Menu Form



4.4.3 Components Available

- **Save Button:** Used to save records.
- **Search button/textbox:** Searches for a specified record from within the database.
- **Validate button:** Used to cross-check data entered with that in the database.
- **Load table button:** Populates tables with data from the database.
- **Delete Button:** Removes unwanted information from the database.
- **Update Button:** Saves a record that has been successfully edited.
- **Exit button:** Closes down a form and automatically opens the previous form or closes down the interface completely. Hence restart is required.

4.4.4 Software Context

The system will include three main modules: the Teller, Attendant and Manager module. Each module will have specified functionality with each being different from the other or sharing some degree of similarity in some sub-modules.

4.4.5 Expected Software Response

The system is expected to run smoothly without any fatal errors that may cause its operation to halt completely or crash. This is because the system has undergone vigorous testing by the system designer and will be handed to qualified team of testers. But in case of any unexpected errors during testing, they will be corrected before implementation by the client.

4.4.6 Packaging and installation

The system on completion will be packaged neatly into a compact disc and copyrighted. It will be accompanied by its documentation.

5.0 Conclusion

The system will be able to fulfil the described user requirements and tests plan documents will be designed to help verify this argument.

6.0 References

1. Frank B. Watts (2004) Engineering documentation control handbook
2. En.tekstenuitleg.net/articles/software/database-design-tutorial.html
3. www.tutorialspoint.com
4. Alan Dennis, Barbara Harley Wixom, Ronerta Marie, Roth Hardcover (2012) Systems Analysis and Design 5th Edition