ST.MARY'S UNVERSITY COLLEGE

FACULTY OF INFORMATICS

DEPARTMENT OF COMPUTER SCIENCE

BIRTH AND DEATH REGISTRATION AND ANALYSIS SYSTEM

PREPARED BY:-

1. FITSUM HAILEMELEKOT
2. WUBALEM TEGEGN
3. MESKEREM BEKELE

JULY 2011

SMUC

ADDIS ABABA

Birth And Death Registration And Analysis System

PREPARED BY:-

1. FITSUM HAILEMELEKOT
2. WUBALEM TEGEGN
3. MESKEREM BEKELE

ADVISOR:- ATO HAFTE ABERA

A SENIOR PROJECT DOCUMENT SUBMITTED TO THE

DEPARTMENT OF COMPUTER SCIENCE

FACULTY OF INFORMATICS

ST.MARY'S UNVERSITY COLLEGE

JULY 2011

SMUC

ADDIS ABABA

ST.MARY'S UNVERSITY COLLEGE

BIRTH AND DEATH REGISTRATION AND ANALYSIS SYSTEM

PREPARED BY:-

1. FITSUM HAILEMELEKOT
2. WUBALEM TEGEGN
3. MESKEREM BEKELE

FACULTY OF INFORMATICS

DEPARTMENT OF COMPUTER SCIENCE

Examination Board

______________	______________	__________
Advisor's Name	Signature	Date
______________	______________	__________
Examiner's Name	Signature	Date

List of Figures

Acknowledgement

We would like to thank our advisor Ato Haftey Abera. Special advice from the start to completion of the project (we honestly want to say that, our advisor was not only an advisor, but also a teacher for us. so we can say that we are lucky to be advised by him. Without his great advice and meaningful comments the project wouldn’t have been successfully completed. We also like to thank the employers of City Government of Addis Ababa Resident Identification and Civil status Documents Registration Services. For providing us all the necessary information and materials needed. We appreciate also his grateful explanation for questions we asked.

CHAPTER ONE

Project Overview

1. Introduction

Birth and Death registration system located in lideta sub sity wereda 4 kebele 05/08 is an integrated information system that generates primarily legal, administrative and statistical information that benefits individuals, households, communities, government institutions and non-governmental, regional and international organizations engaged in various socio-economic and other human development endeavors.

Today, every nation, whether developed or developing, has built such a system or is striving to have one as an integral part of their efforts to improve the social, economic and human development package. Developed countries in general have managed to have complete and comprehensive Birth and Death registration systems for years. Developing countries, on the other hand, have not succeeded in the development of such a comprehensive and complete Death and Birth -registration system.

1.1 Background

Birth and death registration system in Ethiopia is among the countries that have not installed national as well as regional civil Birth and Death registration system. One of the prior requirements for the establishment of Birth country is the installation of compulsory Birth and Death registration law.

1.2 Overview of the project

Birth and Death registration system starting In 1990 to be exercised using a computer and in 1995 is where there start called software's were used by people with a list amount of education on that area and also solution the software was new and was supposed to be revised edited, It takes a little bit of time both for the software to be fully success full and for the people to start learning and now more about it.

However the specified starting of birth and death certificate in Ethiopia was in Addis Ababa around 1935-1995 mainly in City Administration, Lideta, Bole, andLaftosub city and the data's that were stored were found at City Government of Addis Ababa Resident Identification and Civil status Documents Registration Services.

2. Statement of the problem and justification

Managing public data become difficult by using manual method. Some of the real challenging problems that created by the manual operations and the current system are listed below:-

- ❖ The system does not permit handling of two customers:
- ❖ Time wastage
- ❖ Poor data sharing
- ❖ sometimes they system gives wrong information about the users
- ❖ Resource wastage in compilation of the data
- ❖ Data duplication as a result of manually handling
- ❖ Poor documentation
- ❖ Poor service due to long appointment and delivery process
- ❖ Wastage of paper
- ❖ When someone wants a birth certificate the system automatically would issue death certificate. Because all tables are in one database.

2.1 Problems of the existing system

The explanation of problem of the existing system is given below using a PIECES framework

Performance: - on the system working giving Birth and death certificate together because of that you get one service the system register you on getting other services.

- ✓ The woreda residence should be registered first in order to get the service.

Throughput

- Receive necessary document from the client, which are needed by the Birth and death offices is takes extended time.
- Study & conform Birth and death registration procedures to be imported or exported is bulky.
- Gather information from the documents presented by the client and fill the declaration also requires more time and laborer.
- Receive full-required documents from the client takes more time.
- Performance of work units is humped due to lack of programmed transfer activities.
- Check the required documents and Give acceptance for full documents to the client consume more time.

Response time

Delay in the preparation and distribution of client files.

For instance *if the customer get another service the system register customer's profile as get all services because of this problem the officer should check the profile it will be Cause the response time will be long*.

Poor time management:- Often paper work is slow, which results in lag on client's services

Information

Output

- Inaccuracy in updating relevant information since manual processing is error prone.
- Information is not timely to its subsequent use.

  When a decision of some kind is needed, information on which to base that decision may take longer time to obtain.

- Redundant information.

  Customer and product information is stored in many places.

Input

- Clients Data is not captured in time to be useful.
- In the current system much process has to be performed to make data useful.
- Clients Data may not be accurately captured and processed.

- Clients Data is captured redundantly.

Stored data

- Clients Data is stored redundantly in multiple files. Those clients information found in application forms are also repeated in other forms.
- Since some data are not accurately captured they are not accurately stored.
- Clients Data is not securely stored. As the result, client's information is exposed to unauthorized person.

Economy

To figure out a way to increase the number of Birth and death getting certification and finally to make people responsible. Control/security

Control

- Redundantly stored data is inconsistent in different files.
- Processing errors are occurring.
- Decision making errors due to lack of credibility of reports.

Security

The authorized person /database administrator/ can also access client's information easily and which is violated to the rule of the country.

Efficiency

Time wastage

- Inefficiency due to poor time management.
- Extended time required to process clients information.
- At certain occasions, data is redundantly copied which creates repeated information at the cost of time.
- As data is redundantly copied, the information generated is also redundant.
- Requires more laborers to process client's data.

- Is tiresome action for Birth and death collectors and process client’s information less efficiently.

Materials wastage

- Effort and materials required for tasks is excessive.

Service

The system produces inaccurate results.

2.1.1 Alternative Solution

Depending on the problems identified under section 2.1.1 the following alternative solutions can be considered to address the problems.

- ➢ The system will check all customer information’s if the customers get with service.
- ➢ The system will register only the customer get service area
- ➢ Keep the customer data only for one customer give one id, if the customers get another service by using the first id.

3. Objective of the study

3.1. General objective are:-

- ➢ To develop software that can accurately substitute the current paper based on Birth and Death registration and analysis system.

3.2. Specific Objectives

Specific objectives are:-

- ➢ To create software that records birth and death registration using the data analysis….
- ➢ To assess the application of window based on Birth and Death registration and analysis system.

- To formulate suggestions and feasibilities on the use of window based Birth and Death registration system.
- To use tools and techniques during data collecting and system development.

4. Tools and Methodology

4.1 Data collection methodologies

The methodologies that are going to be used in collecting the system requirements will comprise mainly interview, data observation.

a) **Interview**: During interview we use deliberate sampling because we interview the domain expert these interview will target people, who work on the system, lideta subcity of wereda 4 kebele 05/08 office.

b) **Data observation**: Observation will take into consideration in order to trace the behavior of the user of current system and some data and document that are used the system.

c) **Document analysis:** Document analysis is one of the crucial method in which it shows exactly the real world of the woreda sub city unfortunately we have not got well organized document which describe how the woreda sub city done their activities.

4.2 Software development methodologies

In modeling the system, unified modeling language (UML) the object oriented approach used. Thus; it will form the estimated methodology that will be used in the software development, which implies the following activities:

- **During analysis Stage methodologies used are**:-

Use cases and scenarios will be elaborated in the functional requirement but class, sequence and activity diagram are used in RAD (Requirement Analysis Document)

- **During Design Stage methodologies used are:-**

Object, component, development, deployment and dynamic state chart diagrams are used.

4.3 Implementation tools

The approach that will be followed throughout the implementation of the proposed system will be Object Oriented software engineering approach. Necessarily, the implementation we are using is for the front end Vb.Net 2010 environment and the backend will be SQL Server 2008 which means the database engine.

Data modeling tools:-

Documentation tool: - for the documentation tool we are using we are using **Microsoft word** and for the drawing / diagram/ part we are using the **VISIO** software.

5. Scope of the Project

The main purpose of this project is to address the possibility of creating software that can be applied to record Birth and Death events in the context of Ethiopia. Using Birth and Death events as a model of illustration.

The scope of the project is limited to creating a desktop based Birth and Death registration for these two major events (birth and death).

It includes not limited to:-

- Registering new customer,
- Reporting
- Searching
- Providing certificate

It is expected to play a role in the government's transformation and development goal to the best of our knowledge.

6. Limitation of the project

The limitation of the project is:-

The current system is including birth death and marriage registration system, but we developed the birth and death registration system, we can't include the marriage registration system due to time limitation.

And also it includes the following limitations:-

- They don't have server computer
- they don't have Internet
- How the system does permit death and birth certificate individually
- How resident record should be control and manage

7. Application and beneficiary of the project

The beneficiary of the project is:-

- ✓ Woreda's sub city organ
- ✓ System developers
- ✓ Society of Woreda's sub cities
- ✓ Including us

Birth and Death registration has many uses and benefits for the individual citizen, society and the state provides

Certificates requested by Government Departments and Agencies in support of applications for services e.g. old age pensions, passports and drivers licenses. Evidence of birth and death is often needed for commercial or personal purposes, is a source of statistical information on population and social trends

- A birth certificate is necessary for personal identification and protection in the following ways :

- To prove the fact of birth this may become necessary for providing parentage and family relationship and settlement of property rights and also for establishing nationality.
- To establish the date of birth and proof of age which may be required for Admission to school, Admission to government, semi-government or private service, Employment in factories and other industries, Recruitment in armed forces, Right to vote, Taking and insurance policy, Claiming social security benefit under health schemes, Obtaining driving license/pilot licenses, Claiming right to marry after crossing the minimum marriage age and Other purposes where maximum or minimum age limits are prescribed.

A death certificate is necessary to prove the fact about death in the following ways:

- For circumstances of death some towns require registration certificate before disposal of dead body
- To prove time and date of death
- To establish the fact of death for relieving the individual from social, legal and official obligations
- To enable settlement of inheritance of property
- To enable family to collect insurance and social security benefits

CHAPTER TWO

BUSINESS AREA ANALYSIS AND REQUIREMENT DEFINITION

2.1. Introduction

Describing and modeling the major functions of the existing system provides away to identify problems in the existing system, to provide alternative solutions for the problem identified, to select the feasible solution among the alternative solution and finally to decide the functional requirements of the proposed new system.

This chapter presents the description of the existing Birth, Death and Marriage collection of woreda sub city level.

It presents the major functions of the existing system, documents, forms and business rules used and reports generated by the current Birth, Death and Marriage system along with their models using essential use case diagram, to capture domain concept using class responsibility collaborator, to capture user interface requirements using essential user interface prototype, to capture relationships between major user interface flow diagrams, user interface identification. In addition, it presents problems faced in the existing system, good practices to be preserved and the players in the existing system. Alternative solutions to address the problem in the existing system, with options analysis, are also part of the things discussed in this chapter.

2.2 Major Functions of the Existing system

The existing system performs the Birth Death and marriage through manual and developed system. The major activities performed at woreda levels are register birth, death and marriage and issues certificate to a person who needs birth, death and marriage certification.

2.3 Description of the existing system

Based on the analysis though observation and document analysis along with interviews we identified that the current woreda system basically has three major are services to be processed. These services are:

- Registration of personal Birth, Death and Marriage data.
- Display information.
- Issue certificate.

The main purpose of this system is to make register woreda residence Birth, Death and Marriage information. So as to facilitates the customers' legal certificate. In the existing system the following procedures should be encountered for Registration of personal Birth and Death data.

2.3.1 Registration

-Registration of customer data

Input

- customer information of each resident
- Information about birth date
- Information about what service he/she want

Process

This type of registration is performed by gathering data from the resident through interview. Each type of registration has its own way. When the residence come to woreda, the woreda check if he/she has registered or residence of that woreda if he/she has been registered give the service they want. Otherwise he/she will full fill the processes the system required. And also delegated person or unclear family of the deceased person can request the office, with one of the following original document from court, kebele, hospital, church, mosque and labor & social affairs bureau. By attaching passport or ID. Card copy with photograph is essential.

Output

The following are the result of the above processes.

- The resident will get birth certificate
- The resident will get death certificate

2.3.2 Reports generated in the existing system

There are different levels of reports generated by the existing system under this functionality the system will produce various reports, it includes major types and each report will be printable.

This system applied for lideta sub city woreda 4 Keble05/08

- For each day, month, year
- By Age ,sex, Ethnic group
- And adopted child

2.3.3 Forms and Documents used by the existing system

There are different forms and documents that are used to collect Interview, observation and document analysis.

2.3.4 Managing system users

This functionality allows adding new user and deleting existing user as required, in addition to this modifying the system user access right is also possible.

2.3.5 Players in the existing system

Customer's like new born childe, married couples, and died person family's are the main players in the existing system and the detail tasks activity of each is shown on the above analysis of existing system

2.4 Functional requirement

A functional requirement is that directly related to the function of the system. It can also be defined as a description or statement of function, feature or condition that a user seeks to have implemented in a system or it is a function or feature that must be include in an information system to satisfy the business needs and be acceptable to the users and solve the problem in the current system.

The functional requirement of our system describes the interaction between the system and its environment independent of the implementation. The development of Birth and death system is a system which facilitates and create both information management system of the birth and death in the following section we will describe the functional requirement of our system and the interaction with the environment.

2.4.1 Personal data registration

The functionality of personal data registration is to allow all the users of the system to register personal data in detail. Our system allows user to register personal data currently living in the woreda, transfer from other woreda, new born, passed away.

2.4.2 Service registration

The functionality of this system will enable to register all the service that the public got from the office. This includes birth certification and death certification. Our system finally generate report and update information for each as required.

2.4.3 Display information

This functionally will possess what the customers want the birth certificate or the death certificate.

2.4.4 Managing system users

This functionality allows adding new user and deleting existing user as required. In addition to this modifying the system user access right is also possible.

2.4.5 Report

Report is very essential to know the data exactly; the new system generates different report on different time. The contents of this report may include:

- ✓ About Birth information
- ✓ About death information
- ✓ Statistical data for the governmental and non-governmental organizations.

2.5 Non-Functional requirements

Non-Functional requirements describe user –visible aspects of the system that are not directly related with the functional behavior of the system .Non –functional requirements also specifies global constraints that must be satisfied by the software for it to be acceptable. The following are the non-functional requirements relevant to desk based information system to be deployed.

2. 5.1 Documentation

The system has a documentation that is used to support maintenance and user‘s guide .the documentation should be organized in a format that will be easy to understand for the system users .consequently, whether the user is an IT skilled or anyone who uses such kind of system, the system user may require this documentation for the purpose of accessing each functions of the system.

2. 5.2 Performance characteristics

The system should respond to the request within reasonable period of time. The system can support concurrent access .the system should be capable of processing queries quickly.

2.5.3 Error handling and extreme conditions

One advantage of automating some process is to make accurate and reliable process than human beings. Automated process is not always free from error rather it will have correcting mechanism to fix it .bearing in mind this, we identify possible errors during system development and some of them are avoided errors either come from the user operation or form the system operation our system is expected to handle error which comes from user’s wrong operation by using exception handling .for instance, if the user enters text data in the input place of numeric value ,our system should handle it and display message to correct the user input .in case of database failure, error due to wrong user input :the system has its own mechanisms to handle such types of errors due to the possibility to anticipate them in advance

In spite of the above truth, our system will not be free of error. But we should think for possible correction when the system encountered error.

2.5.4 Backup

The system will also have different modules for taking backup of the office data with CD ROM or External hard disk. And the backup of the system will of two type's differential or incremental backup depending on the choice of the user.

2.5.5 Quality issues

The system should be reliable and accurate so that it returns the correct result in all circumstances unless an error is encountered if an error occurs, the system will trap the error in the input and notify the user of the error to take appropriate correction .the purpose of this nonfunctional requirement focuses on the system reliability, robustness and availability.

Robustness:-the system will inform a response message for any unacceptable input by the user and for any of invalid option selected by the user.

Availability:-the system can be available at any time for the user and she/he can get the services throughout government work hours.

Reliability:- the system will be reliable and accurate in providing information to customer.

2.5.6 System modification

Modification requests either come from the user or developer. All the documentation of modeling and coding are done by following the standard naming and modeling convention that, any modification or upgrading can be done easily.

2.5.7 Security issues

The system should have a protection mechanism to prevent users 'intervention from an authorized modification to the posted data having direct access to the database due to its milt-user environment. In addition, the system will enforce authentication to admit every user in to the system before he/she starts any activity on the system. This protects the system form any unauthorized access.

2. 5.8 Resource issues

The system supported by different equipment including a server with secured operating system installed on it. The machine is envisioned to be stationed in a controlled machine room; procedures should be placed for the backup of important data. Furthermore, a redundant for access to the user, constantly synchronized to the master, may be appropriate given the critical need for access to the data.

At the development stage the following resources need to be considered carefully.

2.5.9 Software Requirements

- The system used SQL server 2008 for the database
- Vb.net 2010 software

2.5.10 System model

System modeling abstract the overall system under consideration .hence ,we are trying to present the scenarios, Scenarios and use case realization are just a textual and graphical sequence of events or an instance of a use case .These realization or scenarios are depicted as either a sequence or collaboration with diagram.

2.6. Use case modeling

Identify all actors and use case

A use case in software engineering and systems engineering is adscription of a systems behavior as it responds to a request that originates from outside of that system. In other words, a use case describes “who” can do “what” with the system in question .the use case technique is used to capture a system behavioral requirement by detailing scenario-driven threads through the functional requirement .

Actors:- officer, woreda database, Birth and Death Registration system .

Use case:-Represent sequence of interaction between the customer, officer and the system.

Use case list:

- Login
- customer personal data
- customer birth date
- customer death date
- Generate Report
- Print certificate

Description of use case

- Customer personal data:-used to allow the officer to the customer fulfill the requirement.
- Customer Birthday: Used to allow record officer to register the birthday of customer.

- Resident death: Used to allow record officer to register the death date of residents.
- Generate Report: used to the officer generate the information as user want.
- Print certificate: is used to the officer giving to the user the correct certificate.
- Login: user to the correct person (officer) login the system.

Description of use case

- Login: user to the correct person (officer) login the system.
- Customer personal data:-used to allow the officer to the customer fulfill the requirements.
- Customer Birthday: Used to allow record officer to register the birthday of residents.
- Customer death: Used to allow record officer to register the death date of residents.
- Generate Report: used to the officer generate the information as user want.
- Print certificate: is used to the officer giving to the customer the correct certificate.

Description of Actors

Customer:-A customer who lives in that wored or transferred

Officer:-A person who serves the customer and manipulate different data.

Woreda Data base: - is a system used to maintain woredacustomer detail information’s.

Relationship between actors and Use case

~ Customer Register Personal data
~ Officer provides information to resident.
~ Officer encodes resident personal data and their properties.
~ Officer login in and generate statistical report.
~ Woreda database maintains record and /or information of the customer.

Relationship between use cases

~ Login is a pre-condition for generate report

Use case diagram

The project team members have to show the functional requirement of the proposed system in the following essential use case diagram.
Use case diagram is used to describe the functionality of from external point of view. The use case model is one which is considered as functional model.

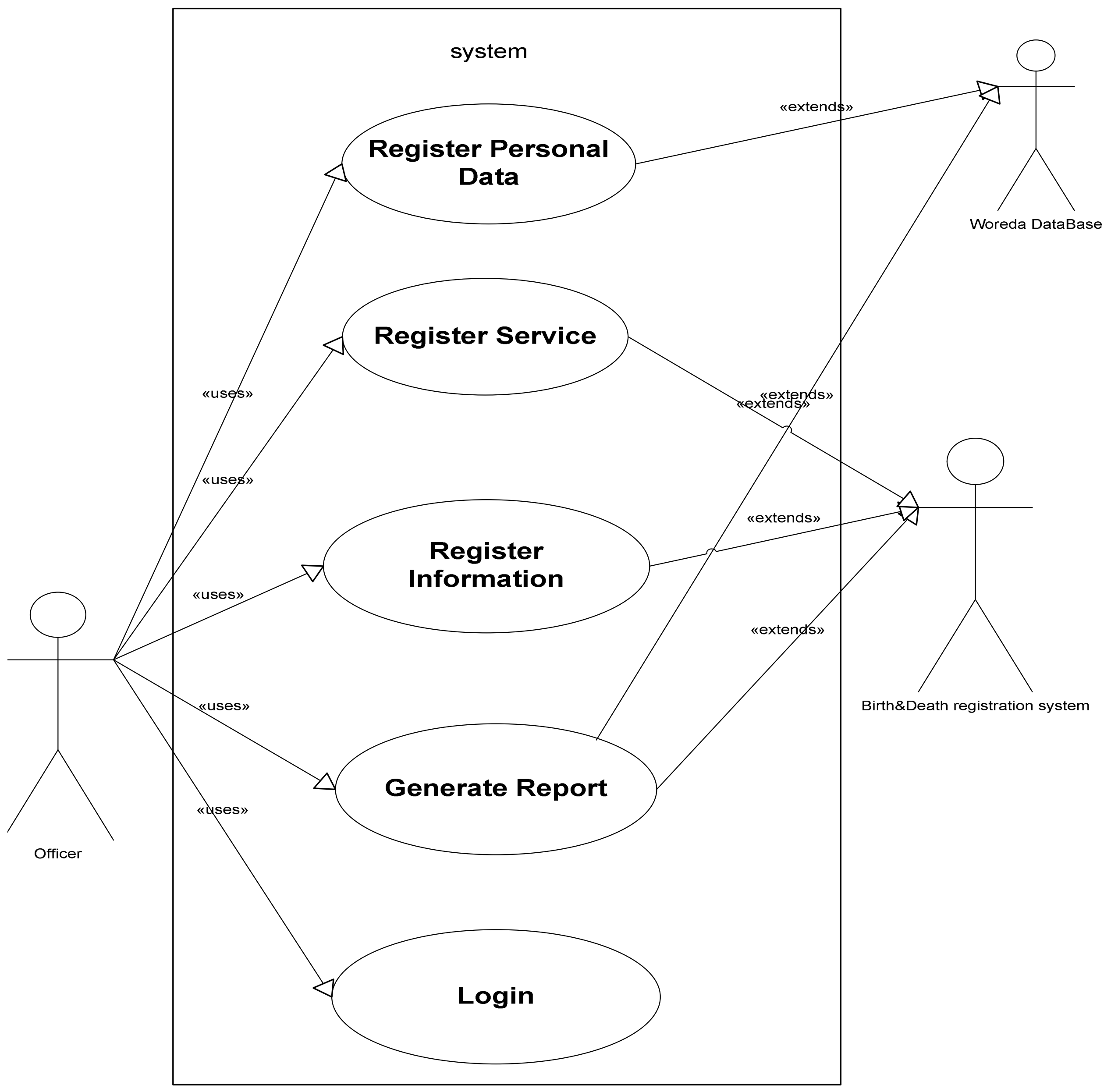


Figure 1 :- System use case diagram

2.6.1 Use case Description

The descriptions briefly explain how the functionalities precede using natural language in a step wise manner. It is for the all use case

.Use Case Id: KS01

Use case Name: login

Participating actor(s): Officer

Precondition(s): The Record Officer should have already been registered.

Main Flow of Event:

1) The Officer activities (click) to login service
2) The record system displays the login interface.
3) The Officer enters username and password and click on login button.
4) The KS verified the user name and password and displays the services that the Officer is allowed to do.
5) The Use case Ends.

Alternative flow of event:

1) If the record Officer omits one or both inputs.
   1.1) The KS notify the Record Officer to complete both inputs
2) The user name and /or password are invalid.
   2.1) the KS notifies the Record Officer invalidity of the User name and /or password and allows re-trial.

Post- conditions: the record officer is able to login the woreda system.

Include: None

Extend: None

Use Case Id: KS 02

Use case Name: Generate report

Participating actor(s): Officer,

Precondition(s): The Officer should be login.

Main Flow of Event:

1) The Officer click on the generate report service or button.
2) The KS opens the interface containing report types.
3) The Officer selects the report type and sets report criteria/fields.
4) The KS verifies completeness of the report details.
5) The wored system displays the report
6) The use case Ends.

Alternative flow of event:

3) If the report type and/or criteria are invalid/incomplete.
    1.2) The KS notifies the emptiness/incorrectness and prompts the Officer enter again.
4) If the verification fails to generate report.
    2.1) the KS notifies the failure and advises the Officer to try again.

Post- conditions: the officer is should be able to generate report.

Include: Login

Extend: None

Use Case Id: KS01

Use case Name: login

Participating actor(s): Officer

Precondition(s): The Officer should have already been registered.

Main Flow of Event:

6) The Officer activities (click) to login service

7) The record system displays the login interface.
8) The Officer enters username and password and click on login button.
9) The KS verified the user name and password and displays the services that the Officer is allowed to do.
10) The Use case Ends.

Alternative flow of event:

5) If the Officer omits one or both inputs.
    1.3) The KS notify the Officer to complete both inputs
6) The user name and /or password are invalid.
    2.1) the KS notifies the Officer invalidity of the User name and /or password and allows re-trial.

Post- conditions: the record officer is able to login the woreda system.

Include: None

Extend: None

Use Case Id: KS 03

Use case Name: Register customer data.

Participating actor(s): customer, woreda database, Officer.

Precondition(s): The customer should prepare requirement detail for registration.

Main Flow of Event:

1) The Officer and/or customer activate the “register customer data “service button.
2) The KS display the register personal data interface.
3) The Officer and/or the customer fills all the details and submits.
4) The KS verifies the register customer data interface.
5) The KS displays confirmation for the registration of the customer
6) The use case Ends.

Alternative flow of event:

1) If the Officer and/or customer fills incomplete/incorrect or invalid detail.

a. The KS notifies the incorrectness and prompts the Officer and/or the customer to enter again.

2) If the verification fails to generate report.

a. The KS notifies Officer and/or customer the failure and asks to re-entry filling the details.

Post- conditions: a customer will be able to register his personal data and save in the woreda database.

Include: None

Extend: None

Use Case Id: KS 04

Use case Name: Register Property.

Participating actor(s): woreda database, Officer.

Precondition(s): The customer should prepare requirement detail for registration.

Main Flow of Event:

7) The Officer activates the “register property “service/ button.
8) The KS display the” register property” interface.
9) The record Officer fills all the details and submits.
10) The KS verifies the requirement correctness and completeness of the details.
11) The KS displays confirmation for the registration of the resident
12) The use case ends.

Alternative flow of event:

3) If the Record officer fills incomplete/incorrect or invalid detail.

a. The KS notifies the incorrectness and prompts the Record officer and/or the resident to enter again.

Post- conditions: The Record officer should be able to register property database.

Include: None

Extend: None

Use Case Id: KS 05

Use case Name: Register Service.

Participating actor(s): Birth and Death registrations system, Record officer.

Precondition(s): The resident should be able to register.

Main Flow of Event:

13) The Officer activates (click) the service/ button.
14) The KS display the service interface.
15) The Officer fills all the details and submits.
16) The KS verifies the requirement correctness and completeness of the details.
17) The KS displays confirmation for the registration of the resident
18) The use case ends.

Alternative flow of event:

4) If the Officer fills incomplete/incorrect or invalid detail.
    a. The KS notifies the incorrectness and prompts the Officer and/or the customer to enter again.
5) If the verification fails
   The KS notifies Officer the failure asks to re-entry filling the details.

Post- conditions: The Officer should be able to register service delivered to him/her .

Include: “register customer data”

Extend: None

2.7 dynamic model

Represented in UML with sequence diagrams, collaboration diagram, we are going to use sequence diagrams and activity diagrams from the dynamic parts of UML diagrams.

2.7.1 Activity diagram

The description of the system in terms of activities is shown using activity diagram theactivity represents the execution of a set of operations. The diagram shown below depicts the activity diagram of woreda system.

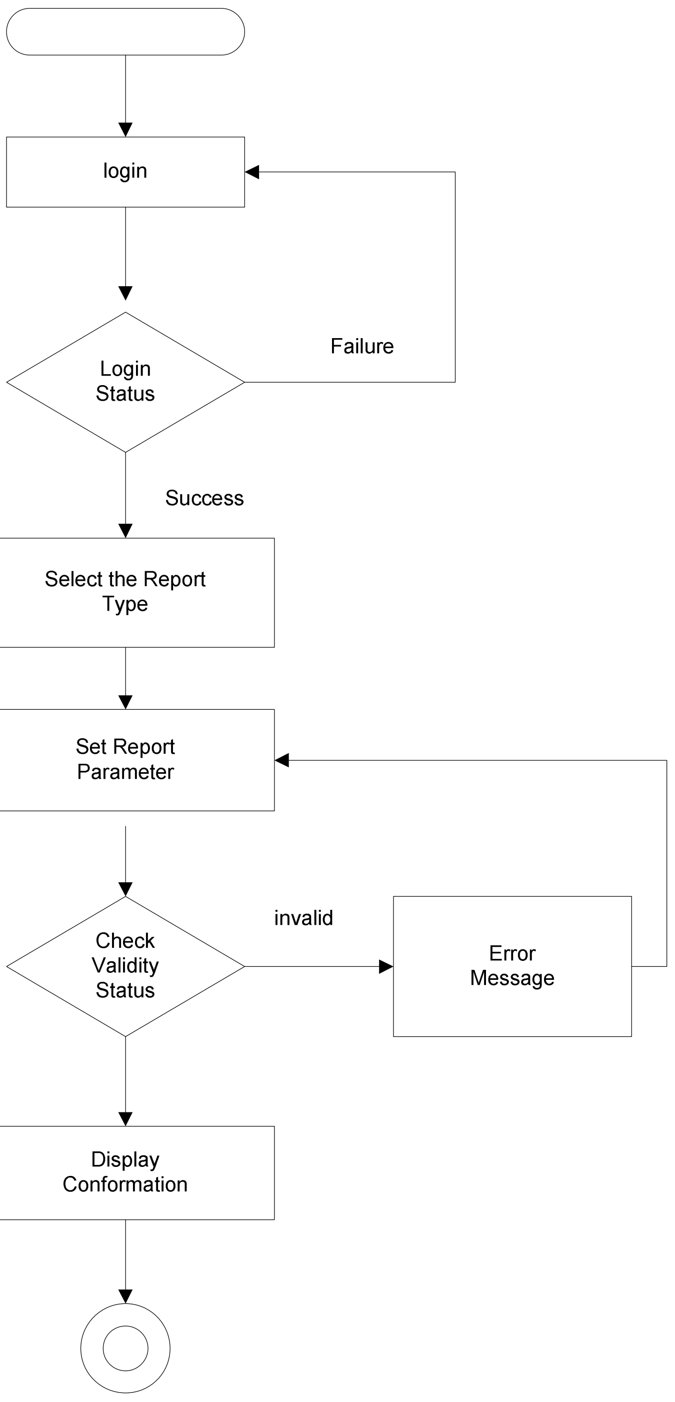


Figure 2:- Login page of the database user

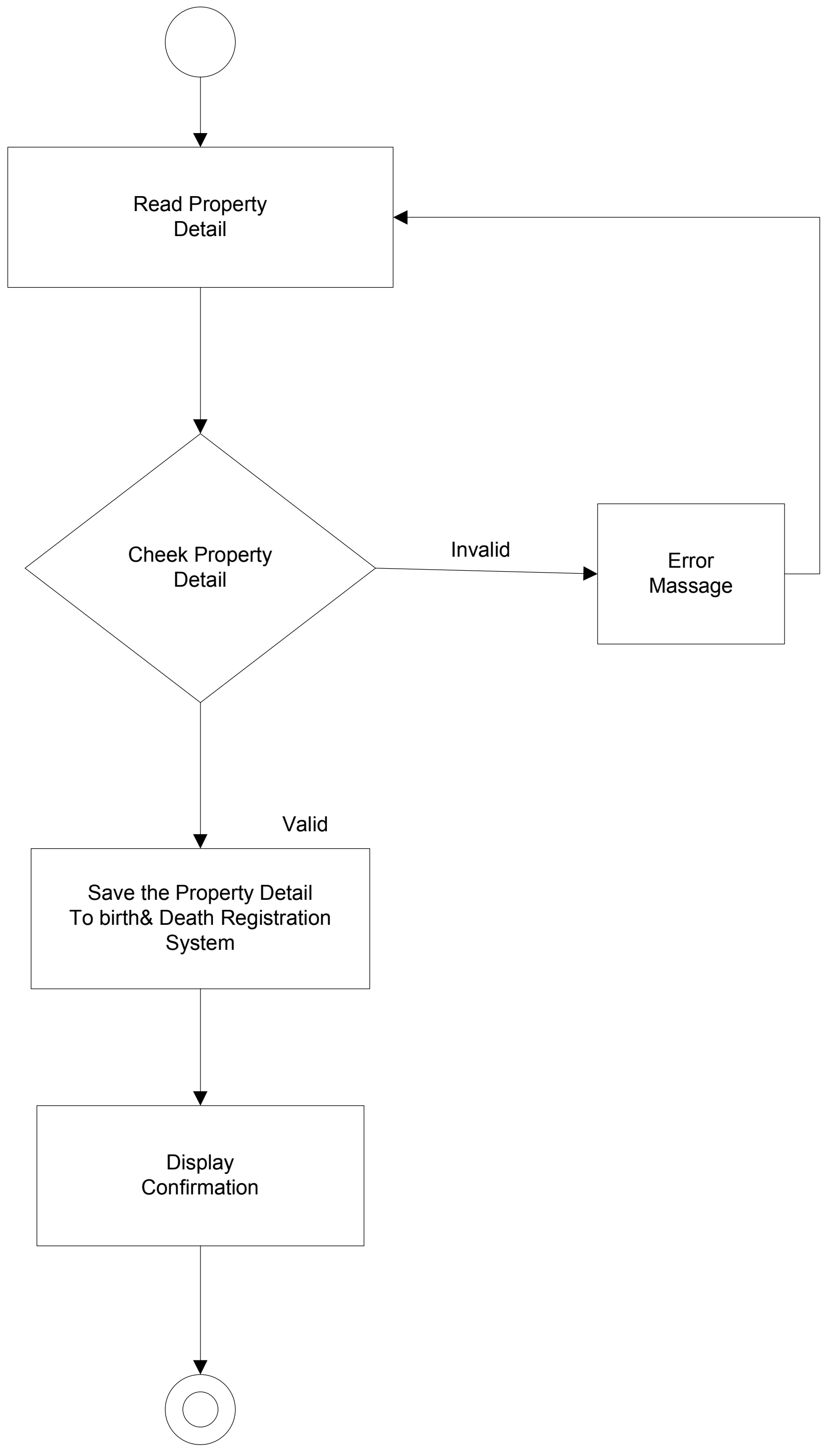


Figure 3:- Activity diagram for Registration of customer Data

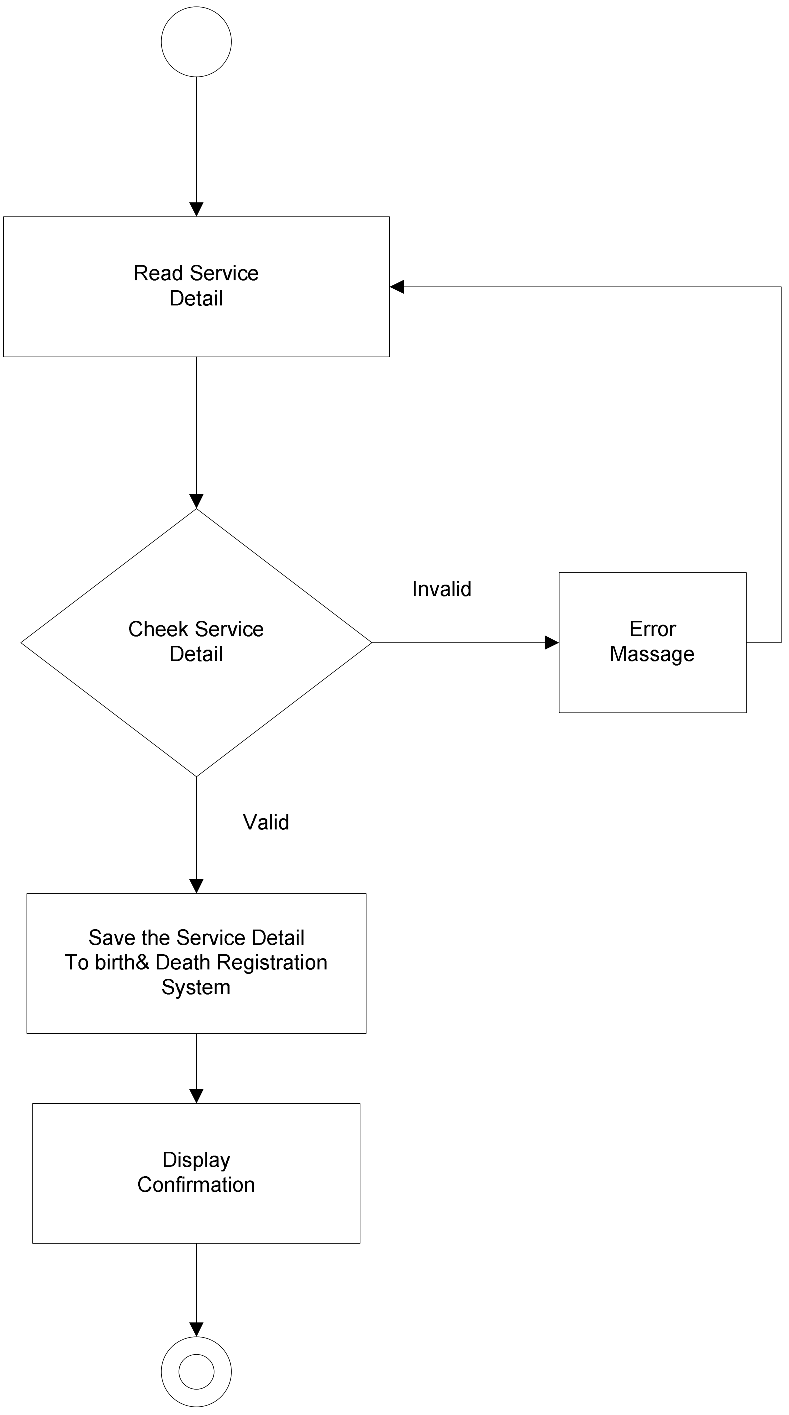


Figure 4:- Activity diagram for Registration of Service

2.7.2 Sequence diagram

The sequence diagram is used to formalize the behavior of the system and to visualize the communication among objects of the system it is the one that should the dynamic model for this work. The sequence diagram for most use case is shown in the coming diagrams the label below the figure describes to which use case the sequence diagram belong too.

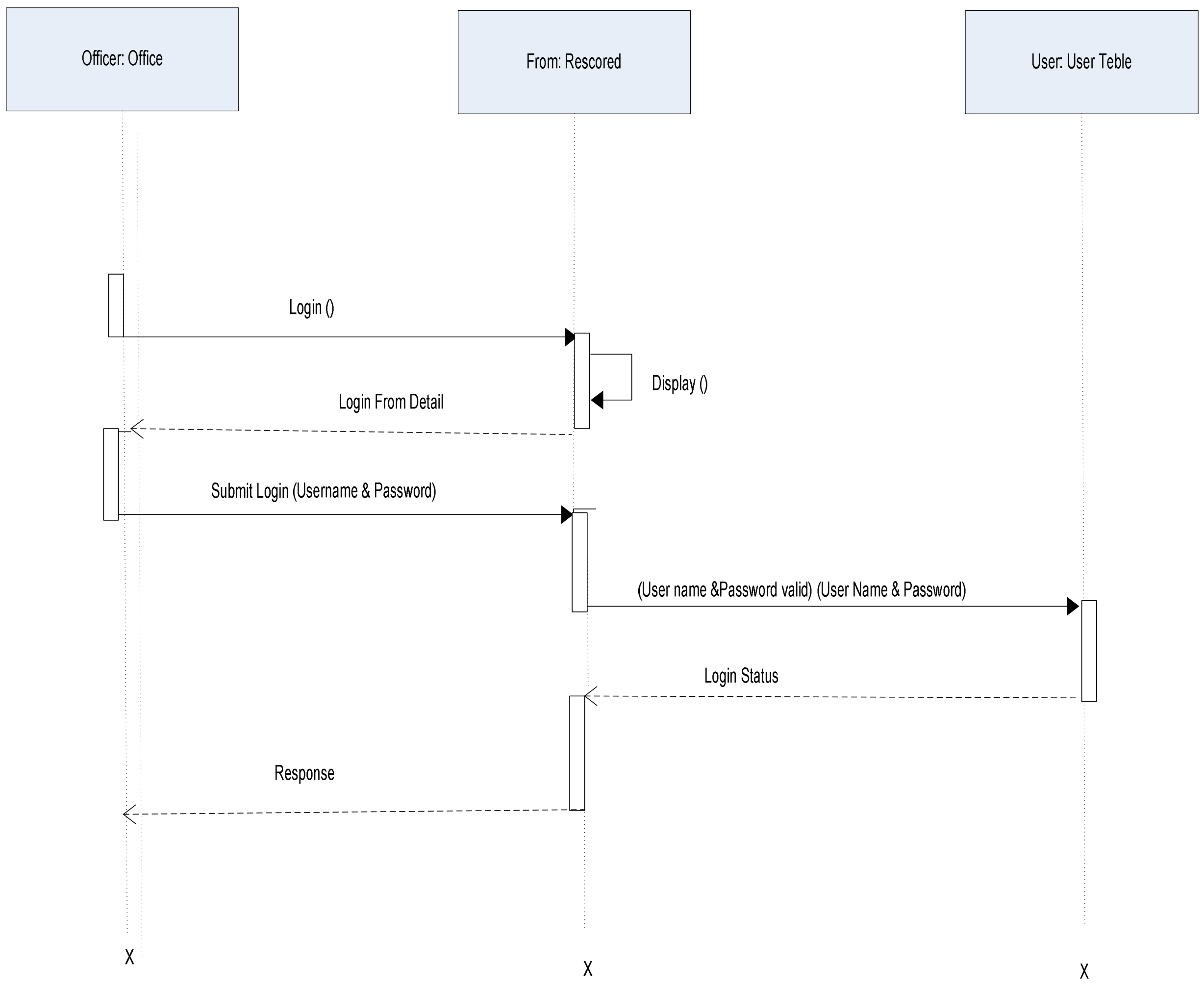


Figure 5:- Sequence Diagram for login

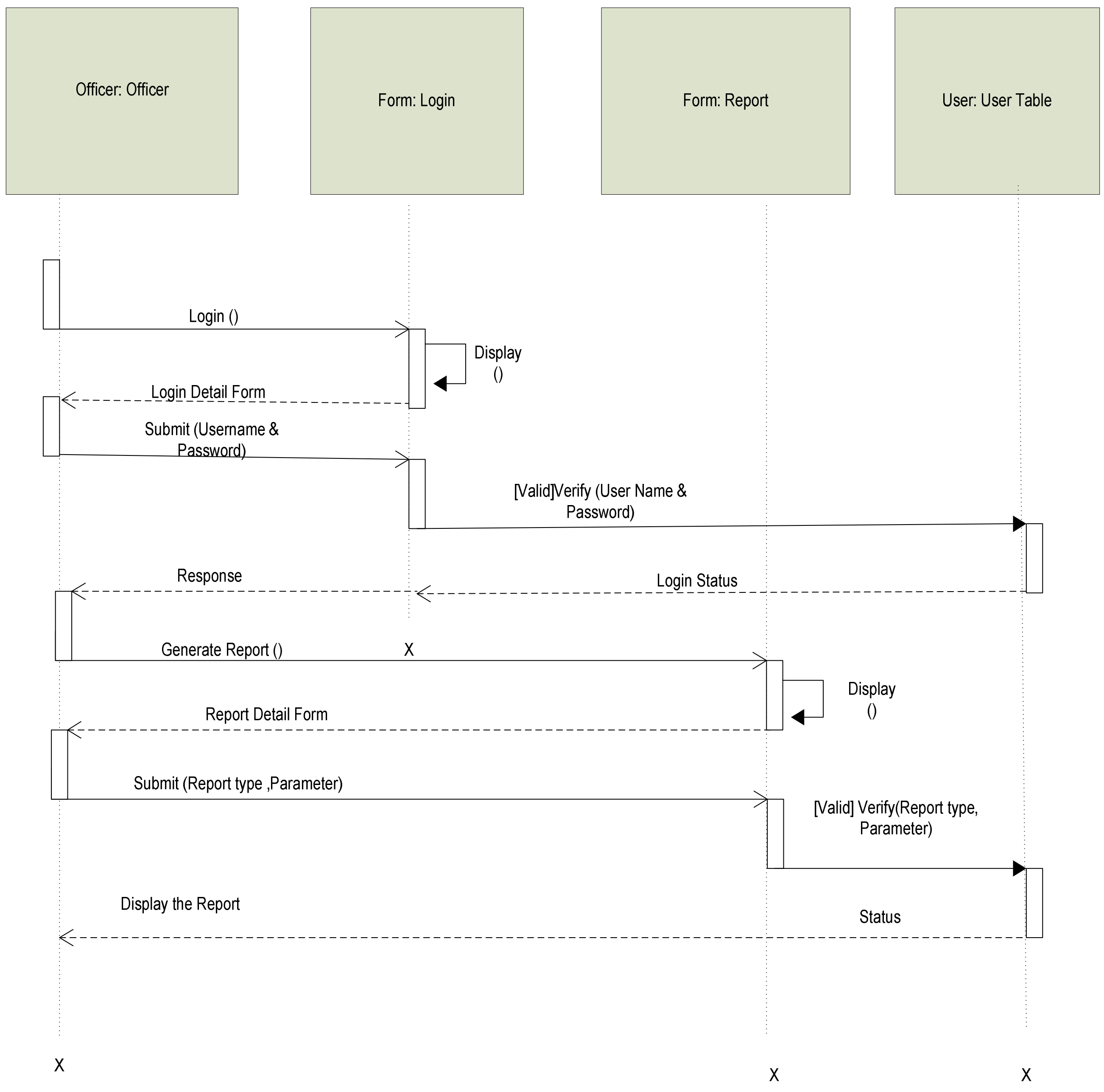


Figure 6:- Sequence Diagram for Generate Reports

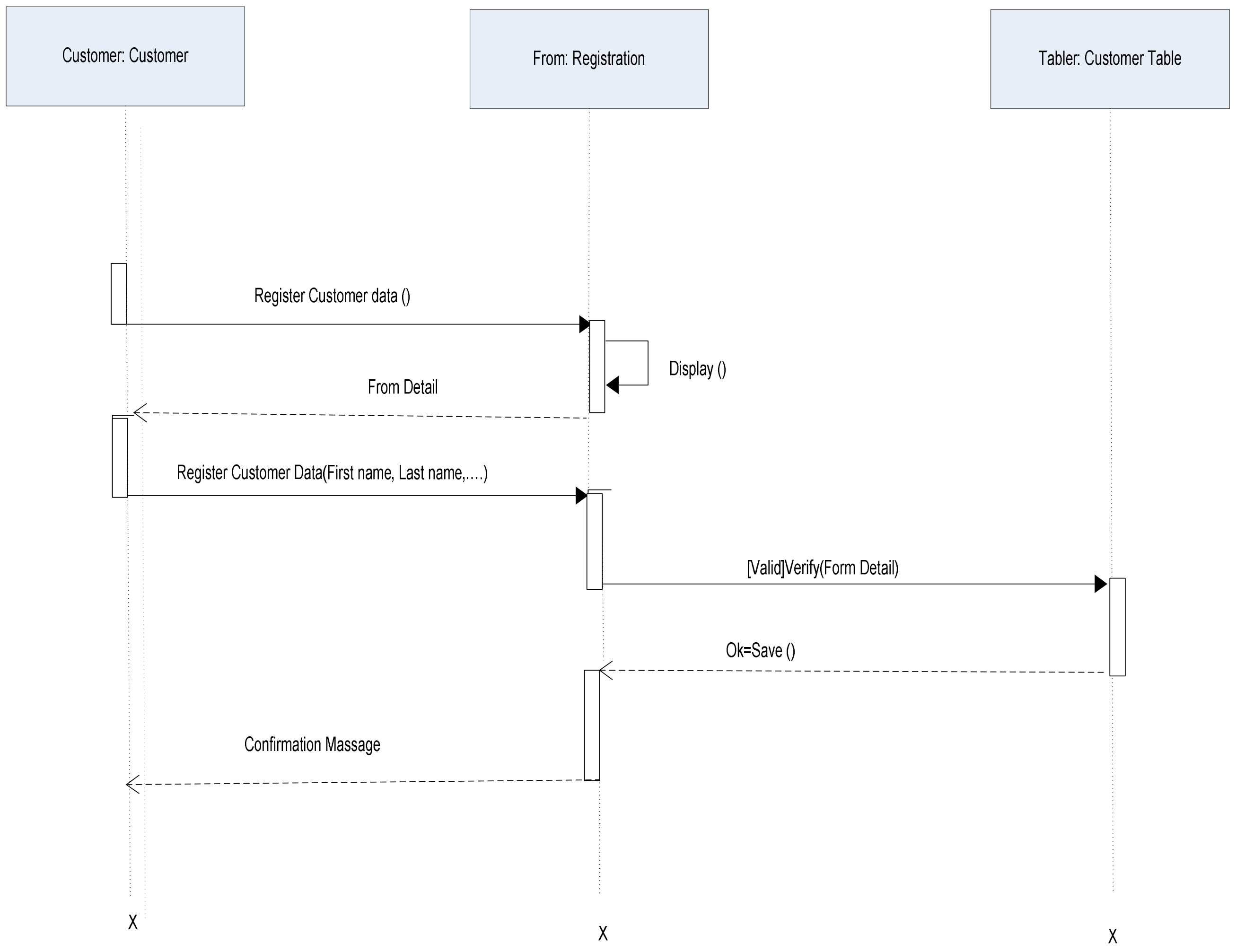


Figure 7:- Sequence Diagram for Registration of Customer Data

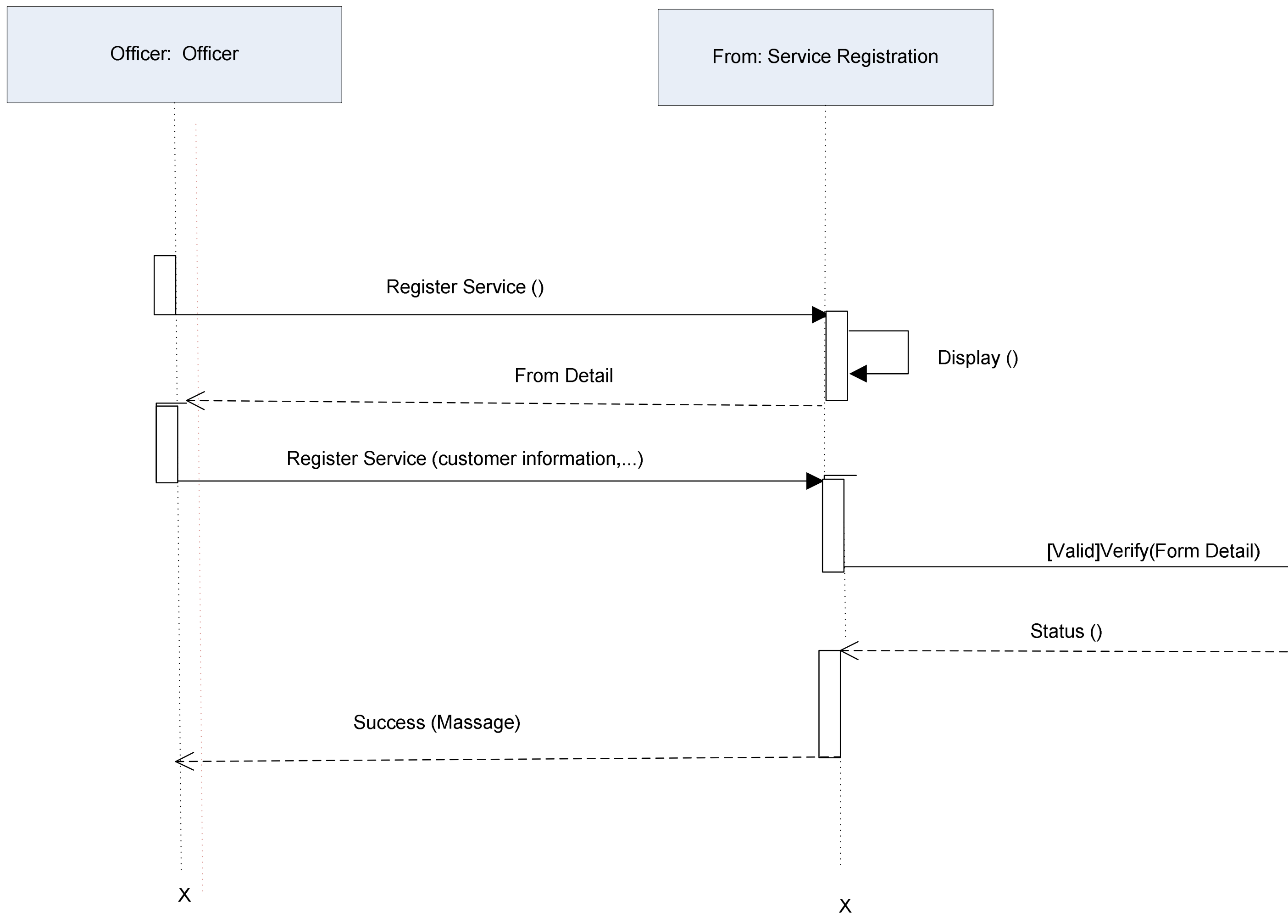

Officer: Officer
From: Service Registration
Register Service ()
Display ()
From Detail
Register Service (customer information,...)
[Valid]Verify(Form Detail)
Status ()
Success (Massage)
X
X

Figure 8:-Sequence Diagram for Registration of Service

CHAPTER THREE

System and object Design

3.1. Introduction

In this chapter object oriented design of our software will be covered. We will start the detail of chapter by describing what is mean by the title (system and object design).

System and object design is concerned with developing an object oriented models of a software system to implement the identified requirements.

Based on the concept of system and object design, our software architecture, models and interface design will be depicted using the following artifacts .class models, collaboration modeling, component modeling, deployment modeling, relational persistence modeling and user interface design of our system will be dealt deeply in this chapter.

3.1.1. Purpose of the System and Object Design

The purpose of this project is to design a system which solves, makes simplify and eases the day to day activity of Birth and Death registration system. Also to develop a support to users to make decision that could facilitate and improve the system will allow register customer data, reporter generating, giving certificate to the customer.

3.1.2 Design goals and objectives

The design goal and objectives of a given system is derived from the non-functional requirements of the system. The same way, we can drive the design goal of Desktop application Birth and Death registrations system form its non-functional requirements described in chapter two thus, the following are the design goals of the system. Which require the system to be designed in such a way that:

- It will provide reliable and accurate information.
- It will be secured and reliable enough to be protected from unauthorized use.
- It will have a full documentation used to support maintenance and user's guide.

- It will support a number of at a time.
- The system should be extendable.
- Minimize Cost

3.2 System Use Case Modeling

A System use case model is composed of a use diagram and documentation describing the use cases, actors and their association. A use case describes a sequence of action that provides a measurable value to an action and is drown as horizontal ellipse. An actor is a person, organization or external system that plays a role in one or more interaction with the system. Actors are drowning using stick figures. Association between actors is involved with an interaction described by use case. Associations are modeled as lines communicating use case and actions to one another.

Classes in Birth and Death registration system

Identify all Entities and Attributes

Customer		
Attribute	Data Type	Size
ID	Character (string)	Max
First Name	Character (string)	Max
Last Name	Character (string)	max
Middle Name	Character (string)	Max
Sex	Number	4
Birthday	Number	10
Wereda	Character (string)	Max
City	Character (string)	Max
House Number	Number	10
Telephone	Number	15
Religion	Character (string)	Max
Ethnic Group	Character (string)	Max

Nationality	Character (string)	Max

Officer		
Attribute	Data Type	Size
User Name	Character (string)	Max
Password	Character (string)	Max

Identify all Operation for each class

Customer	**Officer**
* Register customer ()	Login ()
	Logout ()
	Generate report ()

Identify All Relationship among Classes

* Officer Manages customer.

3.3. Class responsibility Collaborator

Domain modeling is the task of discovering the classes that presents the things and concepts pertinent to your problem space.

Class Responsibility Collaborator (CRC) is the UML artifact that is used to help identify the classes in the system .Accordingly, the CRC model consisting of the classes with their respective responsibility and collaborator are as follows.

Class	
Responsibility	Collaborator

Officer <<Actor>>	
Register Customer	Customer
Register Service	User interface
Generate report	User interface

Customer	
customer name	Customer
customer address	
customer id	

Main Menu<<UI>>	
Enable to search to different tasks	

Service	
Service name	Service
Service request	
Service obtained	

Officer<<UI>>	
Enables to search report model Enables to search service form Enables to search customer form	Officer

Report	
Enables to search different report forms	Officer
Login <<UI>>	
Enables to access to different user interface.	Account

Figure 9:- CRC Diagram

3.4. Class Diagram

3.5 Class Diagram with Operations

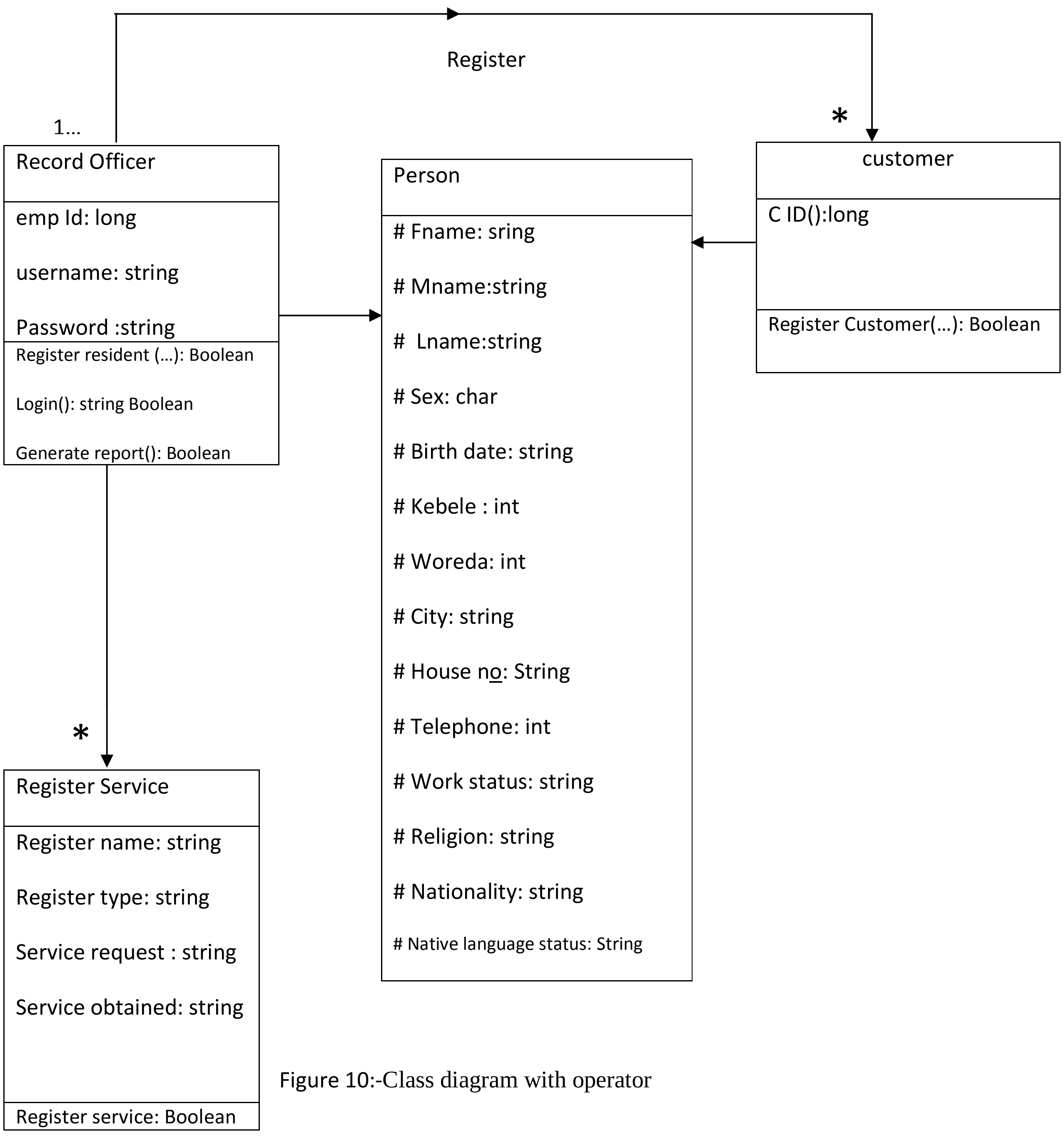


Figure 10:-Class diagram with operator

Actor identification

An actor represents anything or anyone that interact with the system. This may include

People external system and other organizations. The project team has identified the following actors

- Record Officer
- Resident
- Woreda Database

Use case Identifications

Use case is sequence of actions that provides a measurable value to an actor. Another

Way to look at it is that a use case describes a way in which a real world actors interacts

With the system. The project team has identified the following use cases.

- Login
- Register personal data
- Register service
- Register Information
- Generate Report

3.6 Class type architecture [layering]

Class type architecture is the concept of organizing a software designs into Layers/collections of classes or components that fulfill a common purpose, Implementing user interface or the business logic of the system. Class type Architectures provides a strategy for layering the classes, furthermore, class type Architectures provides guidance as to what other types of classes a given type of class Will interest with and that interactions will occur. This increase the extensibility, Maintainability and portability of the system. User interfuse layer:-implements the major user interface elements of the system. User Interface classes only with system, busyness/domain and controller /process

Classes. The major user interface classes are listed as follows. Security login screen, register personal data screens, registers Information screen, register service screen, account management screen Process layer/controller layer:-the purpose of a controller process class is to impairment business login that involves collaborating with several business/domain classes or even with other controller/process classes.

Control process classes interact only with system and business /domain class's login, register property, register service, register information.

Business/Domain Layer:-The business/domain class also called an analysis or entity class is a class that is usually identified during analysis is used to implement concepts pertinent to business domain such as resident focusing on the data aspect of the business objects, plus the behavior specific to individual object .Business domain classes interact only with system and persistence classes. Some of the basic business classes identified are Login, resident, register.

Persistence Layer:- Persistence classes encapsulates the capacity to store, retrieve and delete objects without revealing the underlying storage technology, this helps to isolate the application from changes to the permanent storage approach .persistence classes interact with only system classes and persistence stores all the business area class has persistence classes.

System layer: - the system layer provides features specific to operating system by isolating the software system from operating system by wrapping up OS-specific features. This will increase the portability of the software.

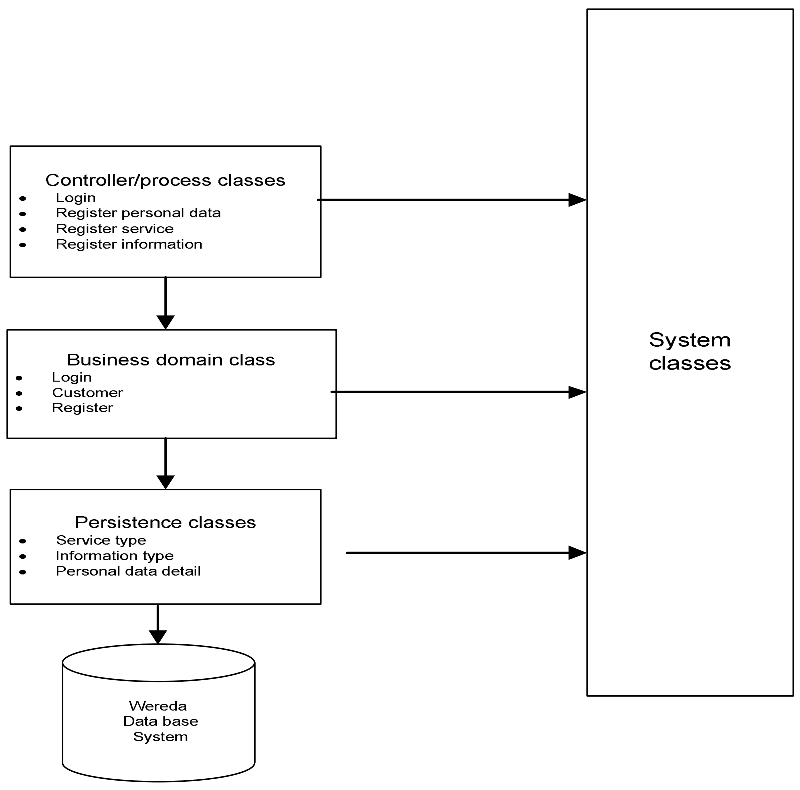


Figure 11:-Class type Architecture

3.7. Class modeling

Class modeling is a model that is used to model the static structure of software to be built in the design of class modeling inheritance, association and dependency are depicted. A class is representation of an object which includes attribute and methods. Attributes are information stored about an object while methods are what the object or the class does. The difference between the analysis and design of class modeling is that analysis focuses on the problem domain whereas the design deals with solution.

Association and dependency are also used to minimize the coding effort. The following figure shows a typical class element in class model.

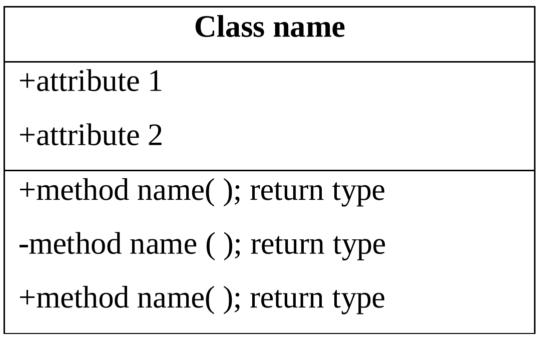

Class name
+attribute 1
+attribute 2
+method name( ); return type
-method name ( ); return type
+method name( ); return type

Figure 12:- Class model

The plus (+) and (-) sign are used to show the visibility of the methods and attribute.'+' sign indicate the visibilities of the methods of attribute are public while the '-'sign indicate the visibility of the method or attribute are private. The method name is the name of the method whereas attribute is the name for specific class. Arguments inside the bracket are the parameter that the method will

Use to pass. Return type is the value of return type which the method gives after processing.

3.8 Collaboration modeling

Collaboration diagrams shows the behavior of severed objects collaborating together to fulfill a common purpose. They depict a bird eye view of the interactions between objects. Collaboration diagram shows the message flow between objects in an object oriented application and imply the associations between classes.

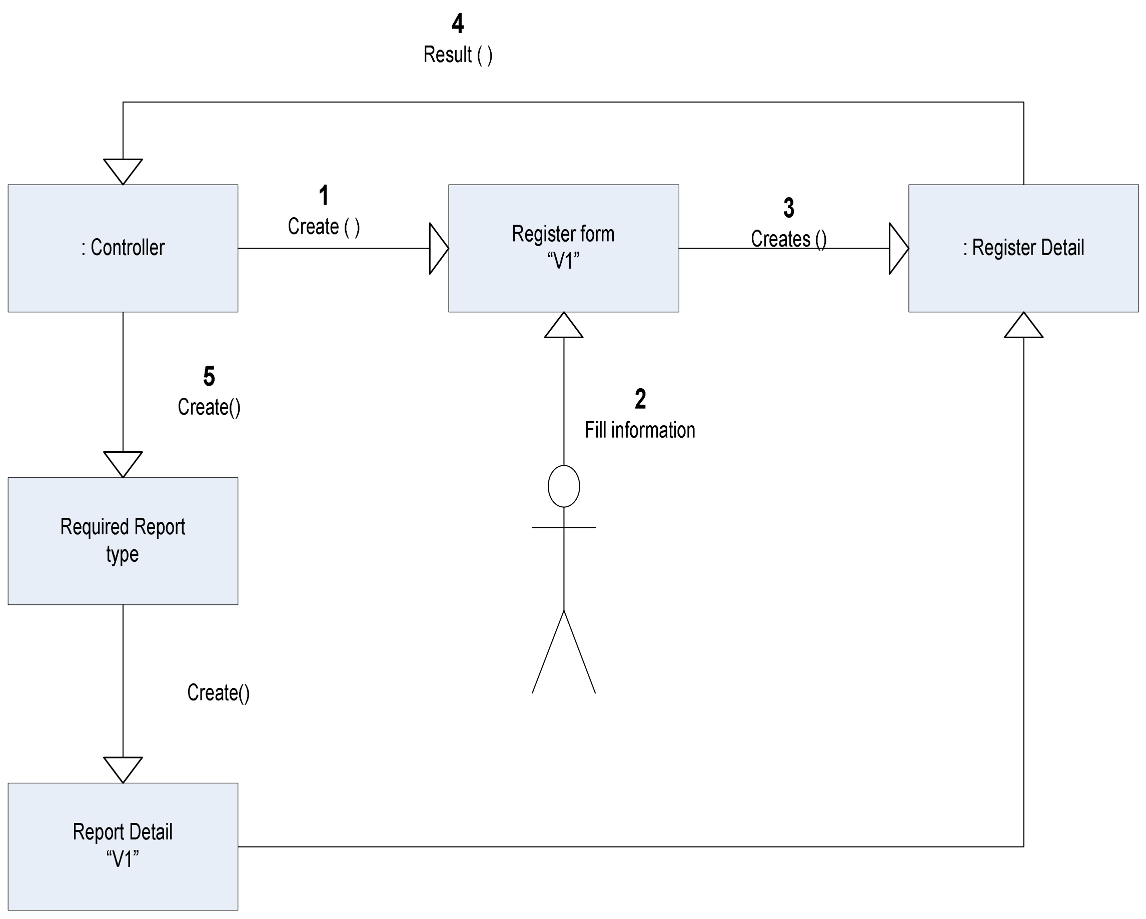


Figure 13:- Collaboration diagram for Register

3.9 Component diagram

Component diagram shows how the physical components of a system are organized. This diagram allows combining deployment nodes with components to show which components run on each node (i.e. hardware).

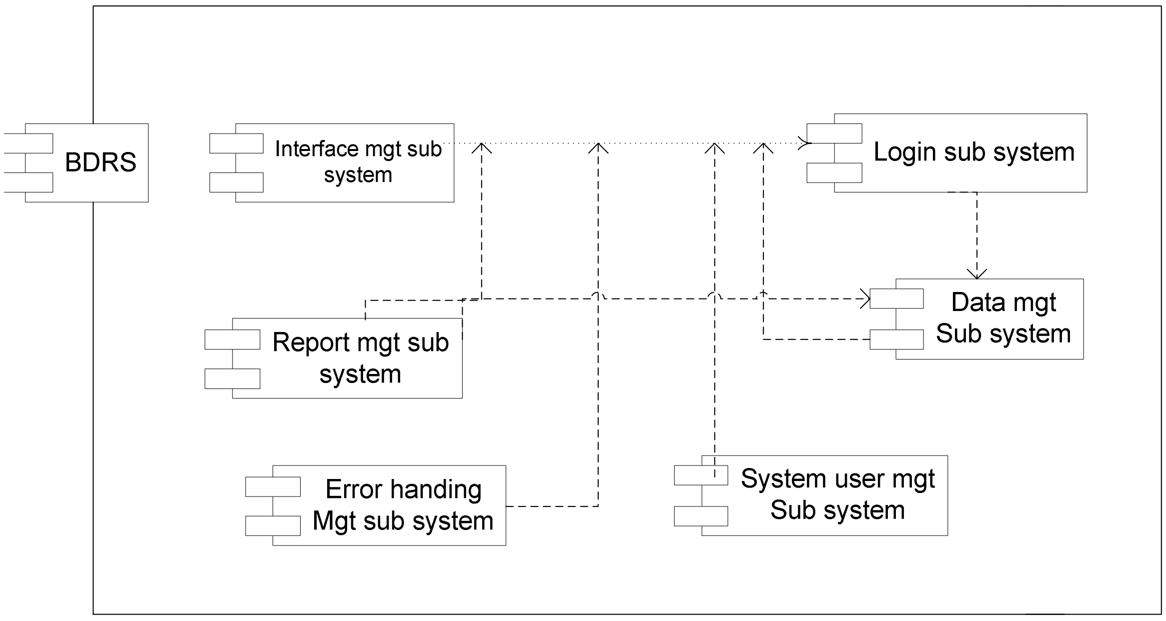


Figure 14:- UML Component diagram

3.10 deployment diagram

The deployment diagram shows the hardware of our system and the software which is installed on that hard ware and the middle used to connect the desperate machine to one another it also used to depict the relationship among run-time components and hardware modes. Components are self-contained entities that provide services to other components or actors.

The figure below shows deployment diagram.

Identify the component/sub systems (SW) (definition)

- User interface component
- Registration management component
- User management component
- Data management component

Identify the mode (HW) (definition)

- Client (user pc)
- Database server

Cardinality (definition)

- server communicate with database server several times

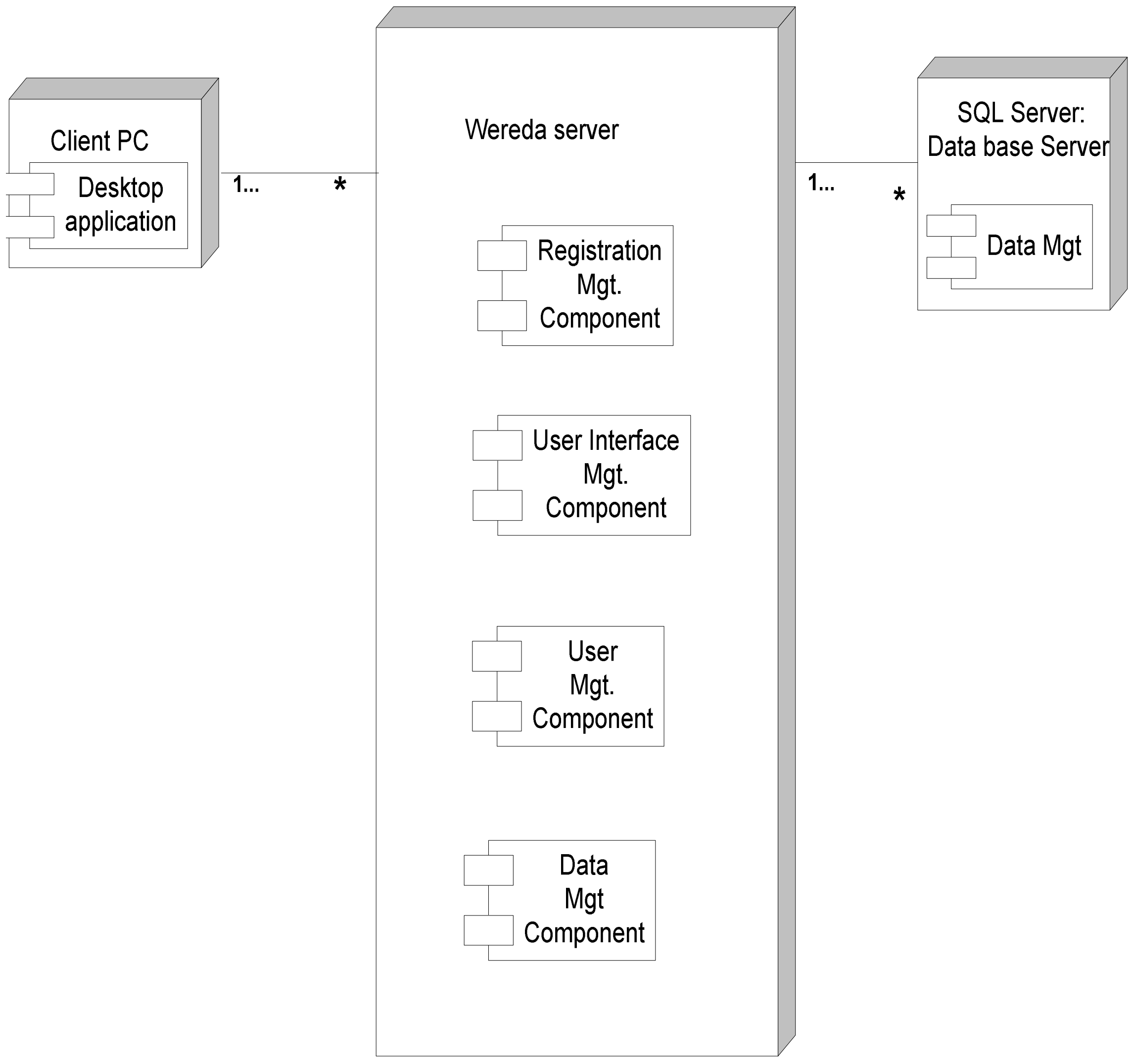


Figure 15:- Deployment diagram

3.11 persistence data modeling

It shows the persistence data stored by the system and data management infra-structure required.

In this system the data of all members will be stored. The persistence data of the new system is easy to display the data that are added by the users. The data base schema and database recording is identified to avoid the existence of problem in the system.

The conceptual data modeling of the system is built based on the information described in the requirement analysis the flow activities will be performed to

- Identify entity types
- Identify relationship and cardinality
- Identify entity and relationship attributes

Identifying entity type

In the requirement analysis phase activities of the system are explicitly described and requirement specifications are defined at this point the designers focus on producing a valid interpretation of the objects

These are the entities of the

- Employee
- Customer
- Register service

Data dictionary of entity types

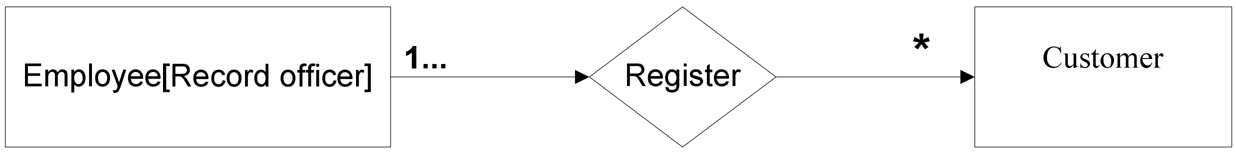
- Employee this refers to individual's information that uses or access the system. Those are record officer and database administrator.
- Register service: this refers to the service delivered by the kebele to the resident.

- User account: this entity refers to individual's information of user from accessing the system.
- Resident id: refers to uniquely identify the resident member.

3.11.1 Identifying relationship and cardinality

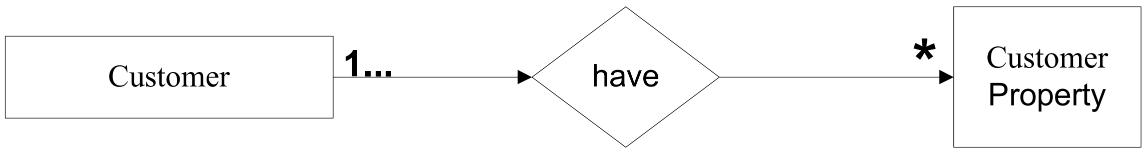
In this section the relationship among entities will be identified, after the cardinality of each relationship is determined as being one- to- one, one-to- many, and many- to –many.

According to requirement specification person will register resident member with one-to-many relationship with the verb' 'register''

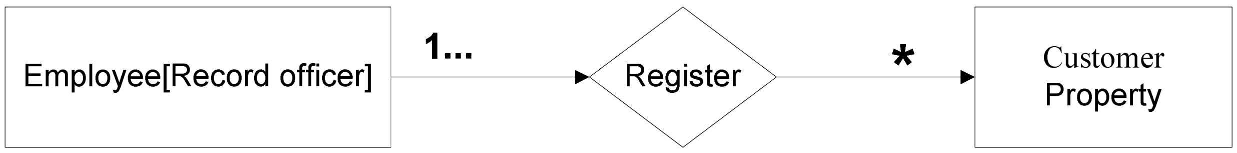


The relationship between resident and property will be joined using keyword '' have''

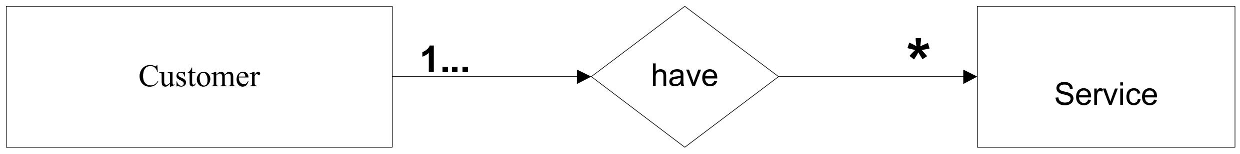
This entity will have a relationship of one from.



The relationship between employee and the register property will be joined with the word ''register' this entity has one-to- many relationships



The relationship between resident member and the service register will be joined be the word ''have''. This entity will have a relationship of one-to- many.



3.11.2 Identification of attributes

Those information required to capture the entities and relationships are identified. Using tables attributes with their description for each entity.

EMPLOYEE

Attributes name	**Description**
E Id	Unique number that identify each employee
First name	First name
Father name	Father name of the employee
Grandfather name	Grandfather name of the employee
Sex	Sex of the employee
Birth date	Birth date of the employee
Wereda	Employees Wereda
City	Employees City
Telephone no	Employees telephone no
House No	The house number of employee
Marital Status	Marital status of employee
Religion	Religion of employee
Nationality	Nationality of employee
Native language	The first language of the employee

CUSTOMER

Attributes name	Description
Customer Id(R ID)	Unique number that identify each resident
First name	First name of the resident
Father name	Father name of the resident
Grandfather name	Grandfather name of the resident
Sex	Sex of the resident
Birth date	Birth date of the resident
Wereda	Resident Wereda
City	Resident City
Telephone no	Resident telephone no
House No	The house number of resident
Marital Status	Marital status of resident
Religion	Religion of resident
Nationality	Nationality of resident
Native language	The first language of the resident

USER ACCOUNT

Attributes name	Description
First name	First name of the employee
Father name	Father name of the employee
Grand Father name	Grand Father name of the employee
User name	An alphabet used to access own page
Password	Secrete string which is known by the admin and member
Account type	The type of the position given to an employee

REGISTER SERVICE

Attributes name	Description
Service type	The type of service that the resident will have
Service name	The name of service obtained by the resident
Service obtained	The service in which the resident got
Serve request	The service in which the resident request.

REGISTER PROPERTY

Attribute Name	**Description**
Property name	The name of the property that the residence will have
Property type	The type of property the residence will have

3.12 Logical Database Design

In this section what has been modeled in conceptual model is mapped to logical model, to reduce data redundancy, file storage space required by the implementation base relation, attributes grouped into relations.

3.13 Normalization

In order to avoid redundancy and different kind of anomalies from relation, it is mandatory to implement the process of Normalization the relations, are analyzed based on their primary key, transitivity and functional dependency among each other.

UN normalized table for resident

CUSTOMER

Cid	First name	Last name	Middle name	Sex	City	S/city	Wereda	tel.no	nationality	birthdate	Service

First normalization (1NF)

No repeating of groups.

Tables should have only two dimensions since one resident can have several address this class should be listed in separate table address at this stage will have a one –to-many relationship so this is not advisable to put the one side and the many side in the same table, instead create on other table in 1NF by eliminating the repeating group as address shown below.

CUSTOMER

Cid	First name	Last name	Middle name	Sex	Nationality	Birth date	Address	service

Second Normalization (2NF)

* Eliminating Repeating

Note the multiple address values for each Resident value in the above table .address is not functionally dependent on resident id (Primary Key), so this not in second normal form .the following tables demonstrates second normal form.

CUSTOMER

<u>Cid</u>	First Name	Last Name	Middle Name	Sex	Service	Nationality	birth data

CUSTOMER ADDRESS

<u>Cid</u>	ADDRESS

Third Normalization (3NF)

* Remove transitive dependency

CUSTOMER SERVICE

<u>Cid</u>	First Name	Last Name	Middle Name	Sex	Nationality	Birth Date

SERVICE

Name of Service	Service Request	Service obtained	Service Delivery Date

PROPERTY

Name of property	Type of property	Date of registration

3.14 Physical Database Design

At this stage the logical database design is translated into physical design that will be Implemented using the target DBMS (database management system), using

My sql build the interface. To code the program using C #.

3.15 Access Control and Security

This section describes the user model of the system in terms of access matrix shown below.

It also describes security issues like selection of an authentication mechanism.

Authentication and authorization mechanism used to Verify or Validate a user on the Proposed DABDRS (Desktop application Birth and Death Registration system).The DABDRS accepts login information from user the system will validate the login Information from two different aspects. The information from two different aspects. The First one is whether the user has an account on the system or not (authentication).the Second one is even if the user has an account; it will check his/her permission(Authorization).this will help to deny an authorized access to the system

Actor/operation	Record Officer
Register Customer ( )	✓
Register Service ( )	✓
Login ( )	✓
Logout ( )	✓
Generate report ( )	✓
View information ( )	✓
Search ( )	✓
Delete user account ( )	✓

3.16 Hardware and software Consideration

Since the system is Birth and Death registration system that there will be a client side andServer side architecture. The servers have at least windows 2000 advanced server upTo the latest version.

The minimum requirement of the hardware and software for the client system is:-

- ✓ Operating System;- Windows xp professional, windows 7.
- ✓ RAM;2GB
- ✓ Processor; Inter Pentium IV With speed of 2.8 GHZ
- ✓ SQL DBMS.
- ✓ 40GB of Hard disk

CHAPTER FOUR

IMPLEMENTATION

4.1User Interface design

The design type is required to be suitable to the system‘s duties and to its user who will interact directly with the computers. The user interface is developed based on requirement elicitation.

The following are considered as significant criteria for development of user interface design:

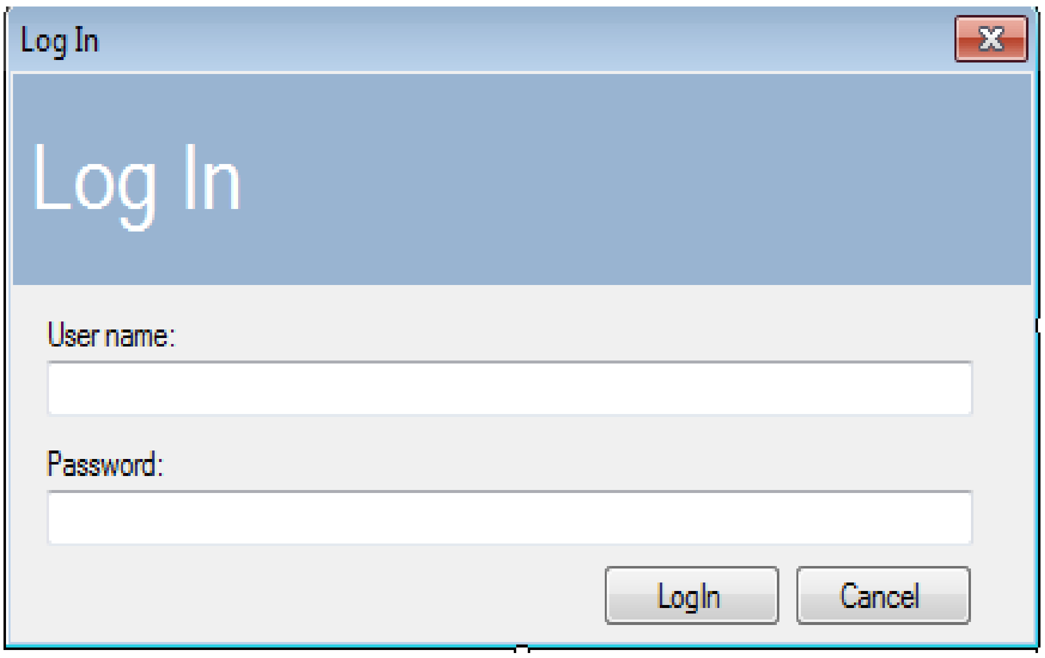
- ✓ Easy to use: easy even with non experienced users
- ✓ Easy to learn: easy for user to remember
- ✓ Processing and responding speed
- ✓ Easy o develop clearly and meaningfully

4.2User Interface flow

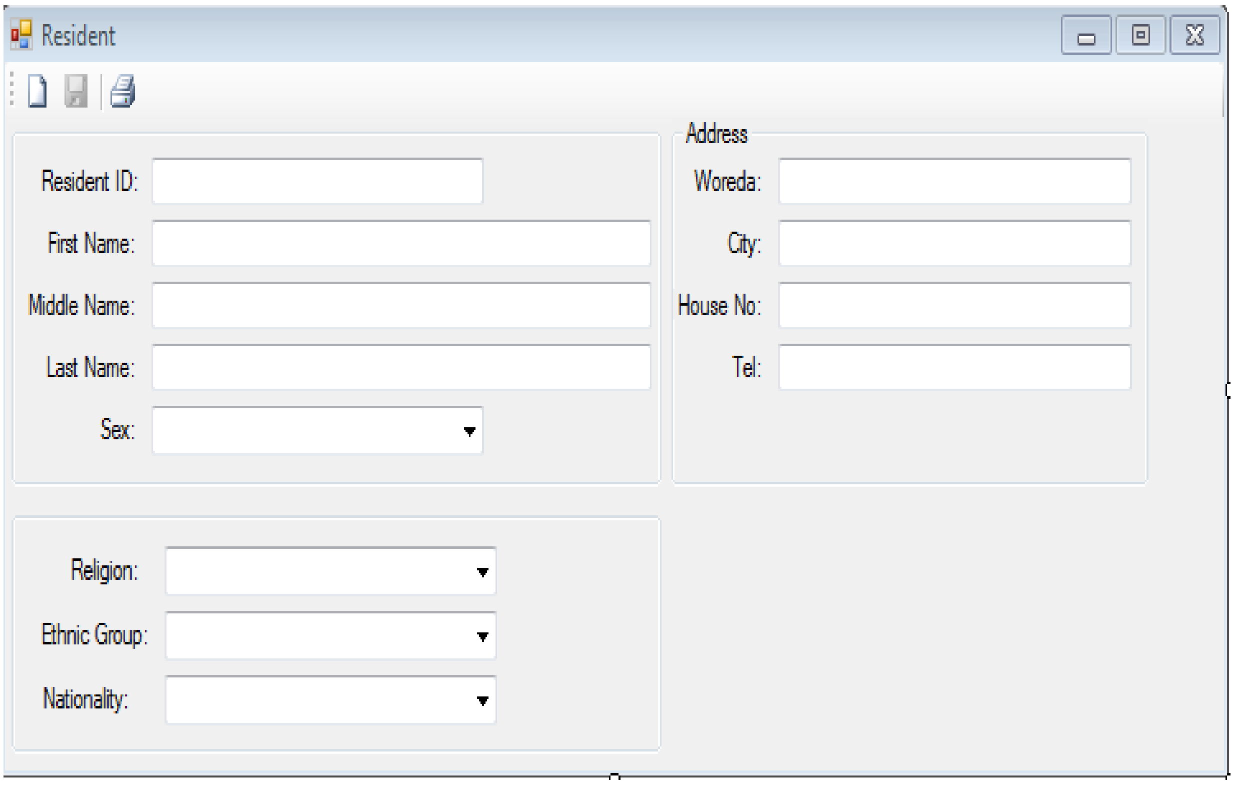
The design version of user interface flow is evolved from its requirement gathering equivalent adding some implementation issues. Such as determining their implementation choices (window, C#) how to interact is performed click or double click.

4.3 User Interface sample models

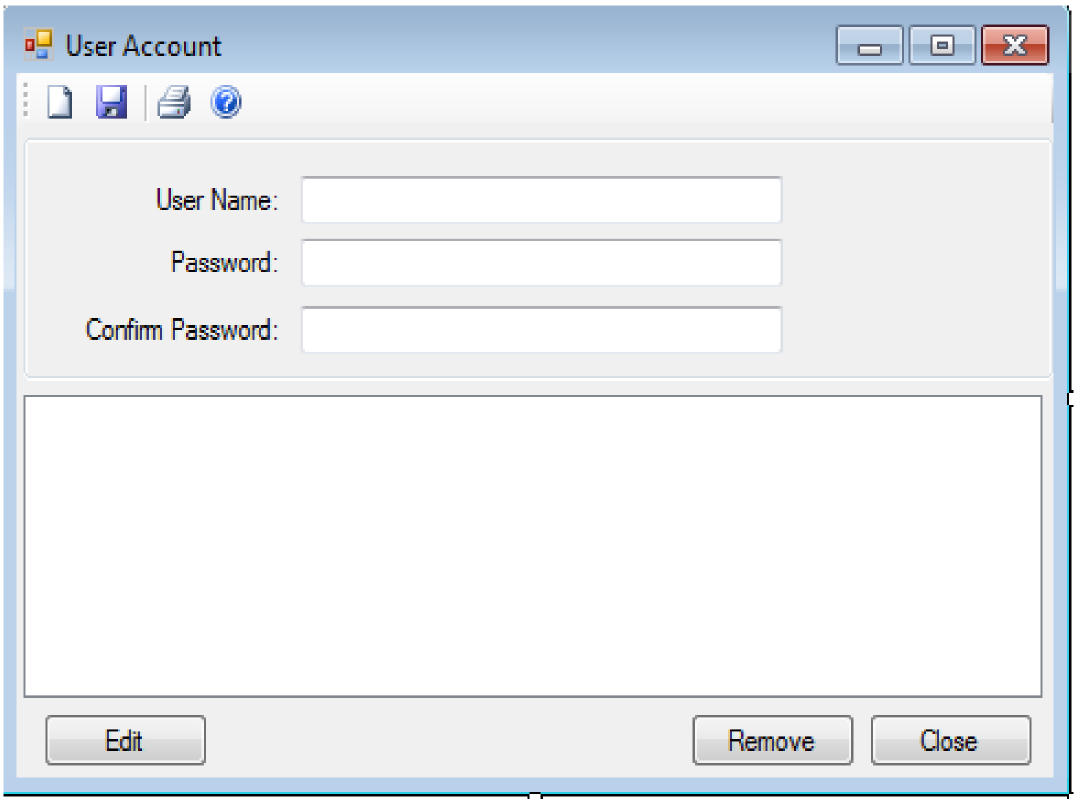
Birth and Death registration system



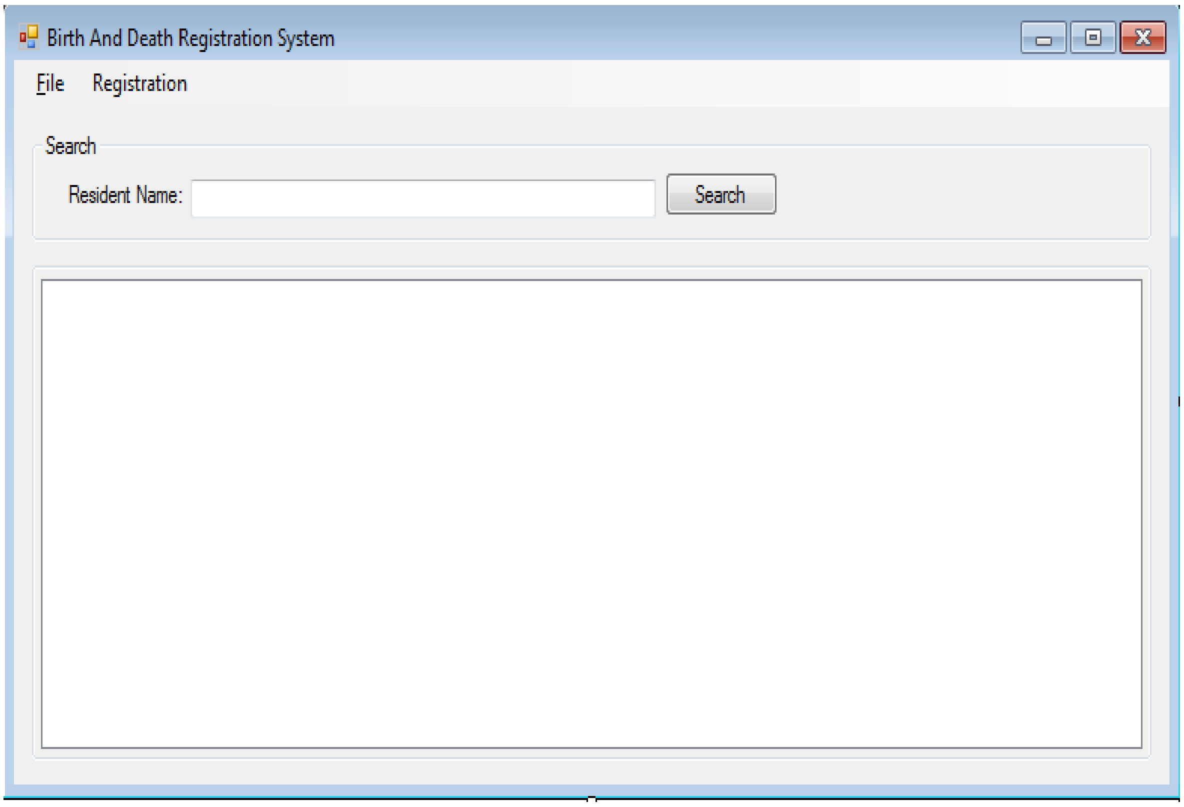
Birth and Death registration system



Birth and Death registration system



Birth and Death registration system



4.4 Sample codes

Log in

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace BDRS
{
    public partial class frmLogIn : Form
    {
        public frmLogIn()
        {
            InitializeComponent();
        }
    }
}
```

//Resident register

```
namespace BDRS
{
    partial class frmMain
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be
disposed; otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.menuStrip1 = new System.Windows.Forms.MenuStrip();
            this.fileToolStripMenuItem = new
System.Windows.Forms.ToolStripMenuItem();
            this.registrationToolStripMenuItem = new
System.Windows.Forms.ToolStripMenuItem();
            this.newRegistrationToolStripMenuItem = new
System.Windows.Forms.ToolStripMenuItem();
            this.searchResidentToolStripMenuItem = new
System.Windows.Forms.ToolStripMenuItem();
            this.groupBox1 = new System.Windows.Forms.GroupBox();
            this.searchCommand = new System.Windows.Forms.Button();
            this.searchTextBox = new System.Windows.Forms.TextBox();
            this.label1 = new System.Windows.Forms.Label();
            this.groupBox2 = new System.Windows.Forms.GroupBox();
            this.lv = new System.Windows.Forms.ListView();
            this.setupToolStripMenuItem = new
System.Windows.Forms.ToolStripMenuItem();
            this.mnuSetupConfigToolStripMenuItem = new
System.Windows.Forms.ToolStripMenuItem();
            this.mnuSetupUserAccountToolStripMenuItem = new
System.Windows.Forms.ToolStripMenuItem();
            this.exitToolStripMenuItem = new
System.Windows.Forms.ToolStripMenuItem();
            this.menuStrip1.SuspendLayout();
            this.groupBox1.SuspendLayout();
            this.groupBox2.SuspendLayout();
            this.SuspendLayout();
            //
            // menuStrip1
            //
            this.menuStrip1.Items.AddRange(new
System.Windows.Forms.ToolStripItem[] {
            this.fileToolStripMenuItem,
            this.registrationToolStripMenuItem,
            this.setupToolStripMenuItem});
            this.menuStrip1.Location = new System.Drawing.Point(0, 0);
            this.menuStrip1.Name = "menuStrip1";
            this.menuStrip1.Size = new System.Drawing.Size(767, 24);
            this.menuStrip1.TabIndex = 0;
            this.menuStrip1.Text = "menuStrip1";
            //
            // fileToolStripMenuItem
            //
            this.fileToolStripMenuItem.DropDownItems.AddRange(new
System.Windows.Forms.ToolStripItem[] {
            this.exitToolStripMenuItem});
            this.fileToolStripMenuItem.Name = "fileToolStripMenuItem";
            this.fileToolStripMenuItem.Size = new System.Drawing.Size(37,
20);
            this.fileToolStripMenuItem.Text = "&File";
            //
            // registrationToolStripMenuItem
            //
            this.registrationToolStripMenuItem.DropDownItems.AddRange(new
System.Windows.Forms.ToolStripItem[] {
            this.newRegistrationToolStripMenuItem,
            this.searchResidentToolStripMenuItem});
            this.registrationToolStripMenuItem.Name =
"registrationToolStripMenuItem";
            this.registrationToolStripMenuItem.Size = new
System.Drawing.Size(82, 20);
            this.registrationToolStripMenuItem.Text = "Registration";
            //
            // newRegistrationToolStripMenuItem
            //
            this.newRegistrationToolStripMenuItem.Name =
"newRegistrationToolStripMenuItem";
            this.newRegistrationToolStripMenuItem.Size = new
System.Drawing.Size(164, 22);
            this.newRegistrationToolStripMenuItem.Text = "New Registration";
            this.newRegistrationToolStripMenuItem.Click += new
System.EventHandler(this.newRegistrationToolStripMenuItem_Click);
            //
            // searchResidentToolStripMenuItem
            //
            this.searchResidentToolStripMenuItem.Name =
"searchResidentToolStripMenuItem";
            this.searchResidentToolStripMenuItem.Size = new
System.Drawing.Size(164, 22);
            this.searchResidentToolStripMenuItem.Text = "Search Resident";
            //
            // groupBox1
            //
            this.groupBox1.Controls.Add(this.searchCommand);
            this.groupBox1.Controls.Add(this.searchTextBox);
            this.groupBox1.Controls.Add(this.label1);
            this.groupBox1.Location = new System.Drawing.Point(12, 37);
            this.groupBox1.Name = "groupBox1";
            this.groupBox1.Size = new System.Drawing.Size(743, 56);
            this.groupBox1.TabIndex = 1;
            this.groupBox1.TabStop = false;
            this.groupBox1.Text = "Search";
            //
            // searchCommand
            //
            this.searchCommand.Location = new System.Drawing.Point(420, 22);
            this.searchCommand.Name = "searchCommand";
            this.searchCommand.Size = new System.Drawing.Size(75, 23);
            this.searchCommand.TabIndex = 2;
            this.searchCommand.Text = "Search";
            this.searchCommand.UseVisualStyleBackColor = true;
            this.searchCommand.Click += new
System.EventHandler(this.searchCommand_Click);
            //
            // searchTextBox
            //
            this.searchTextBox.Location = new System.Drawing.Point(105, 24);
            this.searchTextBox.Name = "searchTextBox";
            this.searchTextBox.Size = new System.Drawing.Size(309, 20);
            this.searchTextBox.TabIndex = 1;
            //
            // label1
            //
            this.label1.AutoSize = true;
            this.label1.Location = new System.Drawing.Point(21, 25);
            this.label1.Name = "label1";
            this.label1.Size = new System.Drawing.Size(83, 13);
            this.label1.TabIndex = 0;
            this.label1.Text = "Resident Name:";
            //
            // groupBox2
            //
            this.groupBox2.Controls.Add(this.lv);
            this.groupBox2.Location = new System.Drawing.Point(12, 99);
            this.groupBox2.Name = "groupBox2";
            this.groupBox2.Size = new System.Drawing.Size(743, 259);
            this.groupBox2.TabIndex = 2;
            this.groupBox2.TabStop = false;
            //
            // lv
            //
            this.lv.Location = new System.Drawing.Point(6, 13);
            this.lv.Name = "lv";
            this.lv.Size = new System.Drawing.Size(731, 240);
            this.lv.TabIndex = 0;
            this.lv.UseCompatibleStateImageBehavior = false;
            this.lv.View = System.Windows.Forms.View.Details;
            //
            // setupToolStripMenuItem
            //
            this.setupToolStripMenuItem.DropDownItems.AddRange(new
System.Windows.Forms.ToolStripItem[] {
            this.mnuSetupConfigToolStripMenuItem,
            this.mnuSetupUserAccountToolStripMenuItem});
            this.setupToolStripMenuItem.Name = "setupToolStripMenuItem";
            this.setupToolStripMenuItem.Size = new System.Drawing.Size(49,
20);
            this.setupToolStripMenuItem.Text = "Setup";
            //
            // mnuSetupConfigToolStripMenuItem
            //
            this.mnuSetupConfigToolStripMenuItem.Name =
"mnuSetupConfigToolStripMenuItem";
            this.mnuSetupConfigToolStripMenuItem.Size = new
System.Drawing.Size(152, 22);
            this.mnuSetupConfigToolStripMenuItem.Text = "Configuration";
            this.mnuSetupConfigToolStripMenuItem.Click += new
System.EventHandler(this.mnuSetupConfigToolStripMenuItem_Click);
            //
            // mnuSetupUserAccountToolStripMenuItem
            //
            this.mnuSetupUserAccountToolStripMenuItem.Name =
"mnuSetupUserAccountToolStripMenuItem";
            this.mnuSetupUserAccountToolStripMenuItem.Size = new
System.Drawing.Size(152, 22);
            this.mnuSetupUserAccountToolStripMenuItem.Text = "User Account";
            this.mnuSetupUserAccountToolStripMenuItem.Click += new
System.EventHandler(this.mnuSetupUserAccountToolStripMenuItem_Click);
            //
            // exitToolStripMenuItem
            //
            this.exitToolStripMenuItem.Name = "exitToolStripMenuItem";
            this.exitToolStripMenuItem.Size = new System.Drawing.Size(152,
22);
            this.exitToolStripMenuItem.Text = "E&xit";
            this.exitToolStripMenuItem.Click += new
System.EventHandler(this.exitToolStripMenuItem_Click);
            //
            // frmMain
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.ClientSize = new System.Drawing.Size(767, 370);
            this.Controls.Add(this.groupBox2);
            this.Controls.Add(this.groupBox1);
            this.Controls.Add(this.menuStrip1);
            this.MainMenuStrip = this.menuStrip1;
            this.Name = "frmMain";
            this.StartPosition =
System.Windows.Forms.FormStartPosition.CenterScreen;
            this.Text = "Birth And Death Registration System";
            this.menuStrip1.ResumeLayout(false);
            this.menuStrip1.PerformLayout();
            this.groupBox1.ResumeLayout(false);
            this.groupBox1.PerformLayout();
            this.groupBox2.ResumeLayout(false);
            this.ResumeLayout(false);
            this.PerformLayout();

        }

        #endregion

        private System.Windows.Forms.MenuStrip menuStrip1;
        private System.Windows.Forms.ToolStripMenuItem fileToolStripMenuItem;
        private System.Windows.Forms.ToolStripMenuItem
registrationToolStripMenuItem;
        private System.Windows.Forms.ToolStripMenuItem
newRegistrationToolStripMenuItem;
        private System.Windows.Forms.ToolStripMenuItem
searchResidentToolStripMenuItem;
        private System.Windows.Forms.GroupBox groupBox1;
        private System.Windows.Forms.Button searchCommand;
        private System.Windows.Forms.TextBox searchTextBox;
        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.GroupBox groupBox2;
        private System.Windows.Forms.ListView lv;
        private System.Windows.Forms.ToolStripMenuItem
setupToolStripMenuItem;
        private System.Windows.Forms.ToolStripMenuItem
mnuSetupConfigToolStripMenuItem;
        private System.Windows.Forms.ToolStripMenuItem
mnuSetupUserAccountToolStripMenuItem;
        private System.Windows.Forms.ToolStripMenuItem exitToolStripMenuItem;
    }
}
```

//User account

```
namespace BDRS
{
    partial class frmUserAccount
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be
disposed; otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.components = new System.ComponentModel.Container();
            System.Windows.Forms.Label userNameLabel;
            System.Windows.Forms.Label passwordLabel;
            System.ComponentModel.ComponentResourceManager resources = new
System.ComponentModel.ComponentResourceManager(typeof(frmUserAccount));
            this.toolStrip1 = new System.Windows.Forms.ToolStrip();
            this.newToolStripButton = new
System.Windows.Forms.ToolStripButton();
            this.saveToolStripButton = new
System.Windows.Forms.ToolStripButton();
            this.toolStripSeparator = new
System.Windows.Forms.ToolStripSeparator();
            this.printToolStripButton = new
System.Windows.Forms.ToolStripButton();
            this.helpToolStripButton = new
System.Windows.Forms.ToolStripButton();
            this.groupBox1 = new System.Windows.Forms.GroupBox();
            this.textBox1 = new System.Windows.Forms.TextBox();
            this.label1 = new System.Windows.Forms.Label();
            this.passwordTextBox = new System.Windows.Forms.TextBox();
            this.userAccountBindingSource = new
System.Windows.Forms.BindingSource(this.components);
            this.userNameTextBox = new System.Windows.Forms.TextBox();
            this.listView1 = new System.Windows.Forms.ListView();
            this.button1 = new System.Windows.Forms.Button();
            this.button2 = new System.Windows.Forms.Button();
            this.editCommand = new System.Windows.Forms.Button();
            userNameLabel = new System.Windows.Forms.Label();
            passwordLabel = new System.Windows.Forms.Label();
            this.toolStrip1.SuspendLayout();
            this.groupBox1.SuspendLayout();

((System.ComponentModel.ISupportInitialize)(this.userAccountBindingSource)).B
eginInit();
            this.SuspendLayout();
            //
            // userNameLabel
            //
            userNameLabel.AutoSize = true;
            userNameLabel.Location = new System.Drawing.Point(57, 25);
            userNameLabel.Name = "userNameLabel";
            userNameLabel.Size = new System.Drawing.Size(63, 13);
            userNameLabel.TabIndex = 0;
            userNameLabel.Text = "User Name:";
            //
            // passwordLabel
            //
            passwordLabel.AutoSize = true;
            passwordLabel.Location = new System.Drawing.Point(64, 51);
            passwordLabel.Name = "passwordLabel";
            passwordLabel.Size = new System.Drawing.Size(56, 13);
            passwordLabel.TabIndex = 2;
            passwordLabel.Text = "Password:";
            //
            // toolStrip1
            //
            this.toolStrip1.Items.AddRange(new
System.Windows.Forms.ToolStripItem[] {
            this.newToolStripButton,
            this.saveToolStripButton,
            this.toolStripSeparator,
            this.printToolStripButton,
            this.helpToolStripButton});
            this.toolStrip1.Location = new System.Drawing.Point(0, 0);
            this.toolStrip1.Name = "toolStrip1";
            this.toolStrip1.Size = new System.Drawing.Size(471, 25);
            this.toolStrip1.TabIndex = 0;
            this.toolStrip1.Text = "toolStrip1";
            //
            // newToolStripButton
            //
            this.newToolStripButton.DisplayStyle =
System.Windows.Forms.ToolStripItemDisplayStyle.Image;
            this.newToolStripButton.Image =
((System.Drawing.Image)(resources.GetObject("newToolStripButton.Image")));
            this.newToolStripButton.ImageTransparentColor =
System.Drawing.Color.Magenta;
            this.newToolStripButton.Name = "newToolStripButton";
            this.newToolStripButton.Size = new System.Drawing.Size(23, 22);
            this.newToolStripButton.Text = "&New";
            this.newToolStripButton.Click += new
System.EventHandler(this.newToolStripButton_Click);
```

```
            //
            // saveToolStripButton
            //
            this.saveToolStripButton.DisplayStyle =
System.Windows.Forms.ToolStripItemDisplayStyle.Image;
            this.saveToolStripButton.Image =
((System.Drawing.Image)(resources.GetObject("saveToolStripButton.Image")));
            this.saveToolStripButton.ImageTransparentColor =
System.Drawing.Color.Magenta;
            this.saveToolStripButton.Name = "saveToolStripButton";
            this.saveToolStripButton.Size = new System.Drawing.Size(23, 22);
            this.saveToolStripButton.Text = "&Save";
            //
            // toolStripSeparator
            //
            this.toolStripSeparator.Name = "toolStripSeparator";
            this.toolStripSeparator.Size = new System.Drawing.Size(6, 25);
            //
            // printToolStripButton
            //
            this.printToolStripButton.DisplayStyle =
System.Windows.Forms.ToolStripItemDisplayStyle.Image;
            this.printToolStripButton.Image =
((System.Drawing.Image)(resources.GetObject("printToolStripButton.Image")));
            this.printToolStripButton.ImageTransparentColor =
System.Drawing.Color.Magenta;
            this.printToolStripButton.Name = "printToolStripButton";
            this.printToolStripButton.Size = new System.Drawing.Size(23, 22);
            this.printToolStripButton.Text = "&Print";
            //
            // helpToolStripButton
            //
            this.helpToolStripButton.DisplayStyle =
System.Windows.Forms.ToolStripItemDisplayStyle.Image;
            this.helpToolStripButton.Image =
((System.Drawing.Image)(resources.GetObject("helpToolStripButton.Image")));
            this.helpToolStripButton.ImageTransparentColor =
System.Drawing.Color.Magenta;
            this.helpToolStripButton.Name = "helpToolStripButton";
            this.helpToolStripButton.Size = new System.Drawing.Size(23, 22);
            this.helpToolStripButton.Text = "He&lp";
            //
            // groupBox1
            //
            this.groupBox1.Controls.Add(this.textBox1);
            this.groupBox1.Controls.Add(this.label1);
            this.groupBox1.Controls.Add(passwordLabel);
            this.groupBox1.Controls.Add(this.passwordTextBox);
            this.groupBox1.Controls.Add(userNameLabel);
            this.groupBox1.Controls.Add(this.userNameTextBox);
            this.groupBox1.Location = new System.Drawing.Point(3, 21);
            this.groupBox1.Name = "groupBox1";
            this.groupBox1.Size = new System.Drawing.Size(468, 107);
            this.groupBox1.TabIndex = 1;
            this.groupBox1.TabStop = false;
            //
            // textBox1
            //
            this.textBox1.Location = new System.Drawing.Point(126, 76);
            this.textBox1.Name = "textBox1";
            this.textBox1.Size = new System.Drawing.Size(219, 20);
            this.textBox1.TabIndex = 5;
            // 
            // label1
            // 
            this.label1.AutoSize = true;
            this.label1.Location = new System.Drawing.Point(26, 79);
            this.label1.Name = "label1";
            this.label1.Size = new System.Drawing.Size(94, 13);
            this.label1.TabIndex = 4;
            this.label1.Text = "Confirm Password:";
            // 
            // passwordTextBox
            // 
            this.passwordTextBox.DataBindings.Add(new 
System.Windows.Forms.Binding("Text", this.userAccountBindingSource, 
"Password", true));
            this.passwordTextBox.Location = new System.Drawing.Point(126, 
48);
            this.passwordTextBox.Name = "passwordTextBox";
            this.passwordTextBox.Size = new System.Drawing.Size(219, 20);
            this.passwordTextBox.TabIndex = 3;
            // 
            // userAccountBindingSource
            // 
            this.userAccountBindingSource.DataSource = 
typeof(BDRS.Business.Entities.UserAccount);
            // 
            // userNameTextBox
            // 
            this.userNameTextBox.DataBindings.Add(new 
System.Windows.Forms.Binding("Text", this.userAccountBindingSource, 
"UserName", true));
            this.userNameTextBox.Location = new System.Drawing.Point(126, 
22);
            this.userNameTextBox.Name = "userNameTextBox";
            this.userNameTextBox.Size = new System.Drawing.Size(219, 20);
            this.userNameTextBox.TabIndex = 1;
            // 
            // listView1
            // 
            this.listView1.Location = new System.Drawing.Point(3, 134);
            this.listView1.Name = "listView1";
            this.listView1.Size = new System.Drawing.Size(463, 126);
            this.listView1.TabIndex = 2;
            this.listView1.UseCompatibleStateImageBehavior = false;
            // 
            // button1
            // 
            this.button1.Location = new System.Drawing.Point(387, 266);
            this.button1.Name = "button1";
            this.button1.Size = new System.Drawing.Size(75, 23);
            this.button1.TabIndex = 3;
            this.button1.Text = "Close";
            this.button1.UseVisualStyleBackColor = true;
            // 
            // button2
            //
            this.button2.Location = new System.Drawing.Point(306, 266);
            this.button2.Name = "button2";
            this.button2.Size = new System.Drawing.Size(75, 23);
            this.button2.TabIndex = 4;
            this.button2.Text = "Remove";
            this.button2.UseVisualStyleBackColor = true;
            //
            // editCommand
            //
            this.editCommand.Location = new System.Drawing.Point(12, 266);
            this.editCommand.Name = "editCommand";
            this.editCommand.Size = new System.Drawing.Size(75, 23);
            this.editCommand.TabIndex = 5;
            this.editCommand.Text = "Edit";
            this.editCommand.UseVisualStyleBackColor = true;
            //
            // frmUserAccount
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.ClientSize = new System.Drawing.Size(471, 293);
            this.Controls.Add(this.editCommand);
            this.Controls.Add(this.button2);
            this.Controls.Add(this.button1);
            this.Controls.Add(this.listView1);
            this.Controls.Add(this.groupBox1);
            this.Controls.Add(this.toolStrip1);
            this.Name = "frmUserAccount";
            this.StartPosition =
System.Windows.Forms.FormStartPosition.CenterScreen;
            this.Text = "User Account";
            this.Load += new System.EventHandler(this.frmUserAccount_Load);
            this.toolStrip1.ResumeLayout(false);
            this.toolStrip1.PerformLayout();
            this.groupBox1.ResumeLayout(false);
            this.groupBox1.PerformLayout();

((System.ComponentModel.ISupportInitialize)(this.userAccountBindingSource)).E
ndInit();
            this.ResumeLayout(false);
            this.PerformLayout();

        }

        #endregion

        private System.Windows.Forms.ToolStrip toolStrip1;
        private System.Windows.Forms.ToolStripButton newToolStripButton;
        private System.Windows.Forms.ToolStripButton saveToolStripButton;
        private System.Windows.Forms.ToolStripSeparator toolStripSeparator;
        private System.Windows.Forms.ToolStripButton printToolStripButton;
        private System.Windows.Forms.ToolStripButton helpToolStripButton;
        private System.Windows.Forms.GroupBox groupBox1;
        private System.Windows.Forms.TextBox textBox1;
        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.TextBox passwordTextBox;
        private System.Windows.Forms.BindingSource userAccountBindingSource;
        private System.Windows.Forms.TextBox userNameTextBox;
        private System.Windows.Forms.ListView listView1;
        private System.Windows.Forms.Button button1;
        private System.Windows.Forms.Button button2;
        private System.Windows.Forms.Button editCommand;
    }
}
```

// Search

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using BDRS.Business.Entities;
using BDRS.Business;

namespace BDRS
{
    public partial class frmMain : Form
    {
        BDRSRepository repo = new BDRSRepository();
        public frmMain()
        {
            InitializeComponent();
            FillList();
        }

        private void newRegistrationToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            frmResident objFrm = new frmResident();
            objFrm.ShowDialog();
        }

        private void FillList()
        {
            repo = new BDRSRepository();
            IEnumerable<Resident> list = repo.GetResidents();
            ListViewItem item = new ListViewItem();

            lv.Items.Clear();
            SetUpColumns();

            foreach (var i in list)
            {
                item = new ListViewItem();
                item.Text = string.Format("{} {} {}", i.First_Name,
i.Middle_Name, i.Last_Name);
                item.SubItems.Add(i.SexId == 0 ? "Female" : "Male");
                item.SubItems.Add(i.Tel);
                lv.Items.Add(item);
            }
        }
        private void searchCommand_Click(object sender, EventArgs e)
        {
            repo = new BDRSRepository();
            IEnumerable<Resident> list =
repo.SearchResident(searchTextBox.Text.Trim());
            ListViewItem item = new ListViewItem();

            lv.Items.Clear();
            SetUpColumns();

            foreach (var i in list)
            {
                item = new ListViewItem();
                item.Text = string.Format("{} {} {}", i.First_Name,
i.Middle_Name, i.Last_Name);
                item.SubItems.Add(i.SexId == 0 ? "Female" : "Male");
                item.SubItems.Add(i.Tel);

                lv.Items.Add(item);
            }
        }

        private void SetUpColumns()
        {
            lv.Columns.Clear();
            lv.Columns.Add("Full Name", 300);
            lv.Columns.Add("Sex", 100);
            lv.Columns.Add("Telephone", lv.Width - 404);
        }

        private void mnuSetupUserAccountToolStripMenuItem_Click(object
sender, EventArgs e)
        {
            frmUserAccount frm = new frmUserAccount();
            frm.ShowDialog();
        }

        private void mnuSetupConfigToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            frmConfig frm = new frmConfig();
            frm.ShowDialog();
        }

        private void exitToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }
    }
}
```

Database connection

```
usingSystem.Data;

usingSystem.Drawing;

usingSystem.Linq;

usingSystem.Text;

usingSystem.Windows.Forms;

usingBDRS.Business;

usingBDRS.Business.Entities;


namespace BDRS

{

publicpartialclassfrmAdvancedSearch : Form

    {

BDRSRepository repo = newBDRSRepository();


publicfrmAdvancedSearch()

        {

InitializeComponent();

searchByComboBox.Items.Clear();

searchByComboBox.Items.Add("By First Name");

searchByComboBox.Items.Add("By Father Name");

searchByComboBox.Items.Add("By Mather Name");

searchByComboBox.Items.Add("By Birth Date");

searchByComboBox.Items.Add("By House #");

searchByComboBox.SelectedIndex = 0;

        }


privatevoidsearchCommand_Click(object sender, EventArgs e)

        {
```

```
switch(searchByComboBox.SelectedIndex)
            {
case 0: FillList(repo.SearchResident(r =>r.First_Name.StartsWith(keyTextBox.Text)));
break;

case 1: FillList(repo.SearchResident(r =>r.Middle_Name.StartsWith(keyTextBox.Text)));
break;

case 2: FillList(repo.SearchResident(r =>r.Mother_Name.StartsWith(keyTextBox.Text)));
break;

case 3: FillList(repo.SearchResident(r =>r.BirthData.Equals(keyTextBox.Text))); break;

case 4: FillList(repo.SearchResident(r =>r.HouseNo.StartsWith(keyTextBox.Text))); break;

            }

        }


privatevoidSetUpColumns()

        {

lv.Columns.Clear();

lv.Columns.Add("Full Name", 300);

lv.Columns.Add("Sex", 100);

lv.Columns.Add("Telephone", lv.Width - 404);

        }


privatevoidFillList(IEnumerable<Resident>listResult)

        {

repo = newBDRSRepository();

IEnumerable<Resident> list = listResult.ToList();

ListViewItem item = newListViewItem();


lv.Items.Clear();

SetUpColumns();
```

```
foreach (var i in list)
            {
item = newListViewItem();
item.Tag = i.Record_Id;
item.Text = string.Format("{0} {1} {2}", i.First_Name, i.Middle_Name, i.Last_Name);
item.SubItems.Add(i.SexId == 0 ? "Female" :"Male");
item.SubItems.Add(i.Tel);

lv.Items.Add(item);
            }
        }

privatevoidprintBirthCommand_Click(object sender, EventArgs e)
        {
if (lv.Items.Count> 0)
            {
if (lv.SelectedItems.Count> 0)
                {
frmPrintBirthfrm = newfrmPrintBirth();
                    frm.id = (int)lv.SelectedItems[0].Tag;
frm.ShowDialog();
                }
            }
        }

privatevoidprintDeathCommand_Click(object sender, EventArgs e)
        {
if (lv.Items.Count> 0)
            {
```

```
if (lv.SelectedItems.Count> 0)
                {
frmPrintDeathfrm = newfrmPrintDeath();
                    frm.id = (int)lv.SelectedItems[0].Tag;
frm.ShowDialog();
                }
            }
        }
    }
}
```

CHAPTER FIVE

CONCLUSION AND RECOMENDATION

5.1 Conclusion

From the start day of presenting the problems, faced in woreda Birth and Death registration system we are delegated to take this project as our BSC degree fulfillment.

An effort has been made to study birth and death registration system as partial fulfillment of BSC degree in computer science. In doing the study the team has tried to follow object oriented system analysis and design methodology.

Since the success and failure of any system depend on the gathering the right information through different fact finding techniques and user involvement the team has made the best effort together requirement, after detail review and study of the existing system of Birth and Death registration and analysis system model has been design to reflect the new system that is supposed to solve problem designing the BDRS helps to maintain standalone based registration system for both customer and administrative organ of the woreda subcity.

It facilitates activities such as registration of personal data, property service is delivered by the woreda sub city, information delivery and finally generate report beside this it avoides the problems which are related to the above activities.

In order to solve the problems existed in the past team has tried to propose a solution at least reduce the existing problem and model the proposed system using different tools and methodologies.

The team believe that different tools and methodologies helped as a lot in capturing real user requirements and model the right system for the user of proposed system for their day to day activity.

5.2 Recommendation

The birth and death registration system that we have developed is not enough for the future use, so we recommended that the system should be developed using web based basis because it is better to access the system using web from anywhere. The system should generate the requested certificate individually. In addition to this all the sub city system user's /employers/ should be trained about:-

- The basics of computers
- How to Surfing the web /Internet /…

References

1) Mulualem Sintayehu, Serawit Chalachew,Zelalem Ashenafi: Human Resource Management Support System for SMUC Augest 2008.Addis Ababa, Ethiopia.

2) Mohammed Ali, Shimelis Tamiru, and Bezawit Tadele: Web based Keble Information System August 2010 SMUC. Addis Ababa, Ethiopia.

3) City Governments of Addis Ababa policy, 1995e.c

4) Kelly R. (2002). Civil Registration: Vital Change -Birth, Marriage and Death Registration in The 21th Century. Presented to the Parliament of England by the Economic Secretary to the Treasury by Command of Her Majesty Cm 5355 January 2002. .St Clements House, 2-16 Coalgate, Norwich NR3 1BQ.

5) http://www.google.com