COMPUTERIZATION OF BENEFIT PAYMENT PROCESS IN THE NSITF CONTRIBUTORY PENSION SCHEME

(A CASE STUDY OF TRUSTFUND PENSIONS PLC)

BY

BALOGUN, MOHAMMED SHEREEF
PGD/MCS/05/1186

DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE,
FEDERAL UNIVERSITY OF TECHNOLOGY, MINNA.

FEBRUARY, 2008

COMPUTERIZATION OF BENEFIT PAYMENT PROCESS IN THE NSITF CONTRIBUTORY PENSION SCHEME

(A CASE STUDY OF TRUSTFUND PENSIONS PLC)

BY

BALOGUN, MOHAMMED SHEREEF
PGD/MCS/05/1186

A PROJECT SUBMITTED TO THE
DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE,
SCHOOL OF SCIENCE AND SCIENCE EDUCATION,
FEDERAL UNIVERSITY OF TECHNOLOGY,
MINNA, NIGER STATE

IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE AWARD OF POSTGRADUATE DIPLOMA (PGD/MCS) IN COMPUTER SCIENCE

FEBRUARY, 2008

ACKNOWLEDGEMENT

I express my profound gratitude to the Almighty Allah for giving me the opportunity and knowledge to complete this project, and the course work. I express my deep appreciation and gratitude to the Head of Department, Dr. N. I. Akinwande and all the lecturers that contributed to my success in the programme. I express my gratitude to Mallam Shehu Usman for supervising my project.

I appreciate the co-operation of my course mates, my colleagues at work, especially Farouk Alogba; who was very instrumental to the success of the project and I finally express my love, appreciation and profound gratitude to my family for their encouragements, inspiration and understanding throughout the course.

Shereef Balogun
Mathematics and Computer Science Department

ABSTRACT

Information and Communication Technology, especially computers are being used in the contemporary world in industries, military, agriculture, medicine, public service, educational institutions etc. to carry out transactions, store information, record data, provide information, perform range of tasks and even aid in making decisions. With this development, manual operations are being systematically replaced with the modern computerized system. This development does not exempt the pension and gratuity benefits processing and payment system. At present, pensioners' benefits are calculated manually by benefit officers. This process is very tedious, tiring, time consuming and prone to a lot of errors due to multiple human interventions in the entire process. The proposed pension and gratuity processing and payment system is a Windows application which entails the storage and processing of retired employees' data with the use of computers. Computers are fast, reliable and efficient in handling human limitations which used to impede speedy processing of records/retirements benefits. The project is based on the computerization of pension benefit payment process in Trustfund Pensions Plc. To ensure the following among other benefits: Reduction in man-hours spent on benefit processing; Accuracy in the calculation/computation of pensioners'/claimants benefit; Reduction in cost of processing benefits; Higher level of data integrity in Pensioners' data; Possibility of electronic interface with payment banks for issuing payment advice and reconciliation of accounts.

also important for political inclusion, empowerment and the development of democracy.

Prior to the enactment of the Pension Reform Act (PRA) 2004, Nigeria Social Insurance Trust Fund (NSITF) was solely responsible for the management of the contributory pension scheme in the country, which was a form of social security. The scheme was in operation for the private sector only and it operated fully from 1994 to 2004, when the new Act was enacted in which the pension industry was opened up to competition and reforms. Consequently, NSITF was mandated to continue handling the processing and payment of its current pensioners and those that would qualify in the future. NSITF is to provide qualifying claimants who have worked and contributed to the scheme some form of benefits. The benefits are a form of contingencies which a claimant may qualify to access from the pool of funds contributed by all contributors to the NSITF Scheme.

The Act also mandated NSITF to set up a privately owned PFA (Pension Fund Administration) to compete with other PFAs in the emerging pensions industry and also to manage the accumulated pension funds of current NSITF contributors for a transitional period of five years, before moving such funds to other PFAs at employees' request. Against this background, NSITF incorporated Trustfund Pensions Plc., to carry on the business of pension fund administration.

In a Social Security System, benefits are taken as rights due to claimants who qualify and apply for payment and an obligation to the administrators. It is mandatory for the administrators to pay claimants who qualify for benefits as at when due. Social Security benefits are earned benefits, devoid of any kindness, help or favour on the part of the administering authority.

The project is based on the automation of pension benefit payment process with particular interest in how it was being done in the private sector by the organization statutorily responsible for this task, which is Nigeria Social Insurance Trust Fund (NSITF); whose benefit processing and payment responsibilities have now been transferred to Trustfund Pensions Plc. The project work looked into the present system of processing the retired workers' benefits and aimed at designing a computerized system that ensured effortless and quick retrieval of accurate information for accurate and speedy processing and payment of appropriate benefits to the retired claimants. Presently, pensioners' benefits are calculated manually by benefit officers in Trustfund branches spread across the federation. This process is very tedious, tiring, time consuming and prone to a lot of errors due to multiple human interventions in the entire process.

1.2 Significance of Benefits in Social Security Schemes

In a Social Security Scheme, benefits are taken as rights due to claimants who qualify and apply for payment and an obligation to the Social Security organization. It is mandatory for the Social Security organization to pay claimants who qualify for benefits as at when due. Social Insurance benefits are earned benefits, devoid of any kindness, help or favour on the part of the administering authority.

It enhances human wellbeing with particular regard to people's ability to cope with life-risks. But the universality of social security benefits is also attributable to the fact that, by reducing inequality and social deprivation, social security provisions keep social discontent and conflict at a manageable level and, thereby, act as one of the capitalist society's main integrative mechanism. Social security provisions enhance social capital, and generate substantial economic payoffs.

Social security provisions (benefits) are, essentially and primarily, a societal (or public) responsibility, though this is without prejudice to the fact that the burden may be shared between government and other stakeholders at the level of implementing particular programmes.

1.3. Definition of Terms and Concepts

The meaning or explanation of some terms and concepts used in the project are as follows:

The Fund: means Nigeria Social Insurance Trust Fund or Trustfund Pensions Plc.

Monthly Insurable Income: means the monthly total emolument of the employee in respect of which contributions are based. Total emolument in a month is defined as the sum of basic salary, housing allowance and transport allowance paid in that month.

Member: means a person who is registered with the Fund as an employee.

Claimant: means a person who has qualified for a benefit and has duly applied for same.

Dependant or next-of-kin: is defined as a person nominated by a member at the time of registration or by the latest amendment to such records. Where the member dies intestate, the dependant shall be the beneficiary who is so authorized in a Letter of Administration or who is otherwise so determined by the Administrator-General.

Contribution: means the combined amount payable by both the employee and the employer at the rate prevailing at that time.

Life Time Average Insurable Income: means the average monthly earnings calculated on the entire contribution period.

ILO: International Labour Organisation

NPF: National Provident Fund

1.4. Scope of the Project

Due to the magnitude of the matter, this project will attempt to cover all necessary areas in the processing of claimant's benefits as it was done by NSITF and presently being done by Trustfund, especially the contingencies that are found to be most common that is, **Retirement Pension and Retirement Grant**. There shall be a database consisting of at least three normalized tables viz; Employers, Employees and Schedule. The Employers table shall contain records of members' employers described in the following fields: employer number, employer name, employer address, employer location and some other necessary data items that may be required.

The Employees table shall contain records of members registered with the Fund described in the following fields: employee number, employer number, surname, firstname, othernames, date of birth, gender, date employed, contact address, and some other necessary data items that may be required for example, provision will be made to update members' records at retirement. New information like present contact address, telephone, email, bank, bank account number etc will be collected from members and their records updated appropriately. The Schedule table shall contain records of monthly contributions remitted by members described in the following fields: employee number, employer number, month code, year, employer contribution, employee contribution, total contribution, NPF balance and some other necessary data items that may be required. There shall also be some other necessary supporting tables. The project will provide outputs inform of notification or advice to claimants, listing of processed benefits, list of employees, list of employers, details of contributions by members. It will not cover online notification of paying banks. The project will give details of the new system implementation.

1.4.1 Data Collection Method

The methods of data collection are interviews, on-site observations and ready literature. To enable the researcher have proper understanding of the procedures and processes involved in the processing of benefits of claimants, some of the personnel/officers in charge of benefit processing in NSITF and Trustfund were contacted and interviewed. Some booklets and hand-outs were also provided to enable the researcher get the details of existing laws and regulations.

1.5. Aims and Objectives of the Project

1.5.1 Project Aim

The main aim of the project work is to study and analyze the existing system of processing claimant's benefit and use the resultant information/facts elicited to design a computerized system that will process the claimant's data and compute appropriate pension/grant for the members of the NSITF Contributory Pension Scheme without going through the present approach. Which is manual, very tedious, tiring, time consuming and prone to a lot of errors because the process is fraught with multiple human interventions.

1.5.2 Project Objectives

a) The primary objective of the project is to study the existing means of processing NSITF retirees' benefits and develop a program to manage existing data of members and automate NSITF benefit processing.

b) To reduce the rate of error in computation of claimants' benefits and reduce benefit processing time.

c) To provide a better means of administration of member's records.

CHAPTER TWO

REVIEW OF RELATED LITERATURE

2.1 Introduction

Although this project work centered on computerizing the pension benefit processing and payment to qualifying members of the NSITF Contributory Pension Scheme, it will not be appropriate if the researcher fails to tell the readers some things about the subject matter, especially on the concept of Social Security Schemes, significance of benefits in social security schemes, social security schemes in Nigeria, benefit administration, contingency types, qualifying/eligibility conditions, and computation of benefits. It is in the light of the foregoing that this chapter is dedicated to the review of some related literature on the topic.

2.2 Social Security Schemes

The essential function of social security is to protect the individual against the crises and uncertainties of life, particularly those crises and uncertainties which threaten seriously or actually disrupt his access to income to meet essential needs, including the needs of dependants, if any. (Akeredolu-Ale, 2004).

The following are the commonest and most universal of such threats or risks:

- Unemployment
- Retirement/Old Age
- Disability/Invalidity/Sickness
- Work Injury
- Maternity
- Widowhood
- Death

ILO's concept of 'social security' probably represents the most comprehensive scope that a national social security scheme should have, since the organization defines 'social security' as 'the life protection which society provides for its members through a series of public measures against the economic and social distress that otherwise would be caused by the stopping or substantial reduction in earnings, resulting from sickness, maternity, employment injuries, unemployment, invalidity, old-age and death (as well as) the provision of medical care and subsidies for families and children'.

On the whole, social security is a critical factor in ensuring the economic and overall well-being of the ordinary citizen, even in the event of their running into any of the many crises and uncertainties of life. And this is why the ILO has been mandated by the international community to promote and foster among all member-states of the United Nations the establishment of social security schemes and measures to guarantee the provision of a basic income to all who need such protection, in addition to the provision of comprehensive medical care. Needless to say, the need to provide adequate social security services tends to be more urgent in countries where poverty, unemployment and inequality are still at a very high level, which, incidentally, also tend to be the countries that usually lack the resources to meet the people's need for social protection.

Social security, in general, refers to state provision of financial aid to alleviate poverty, as well as to make possible the maintenance of the beneficiary's income level even during periods of crisis.

Social Security is very important for the well being of workers, their families, and the entire community. It is a basic human right and a fundamental means for creating peace and social inclusion. It is an indispensable part of government social policy and an important tool to prevent and alleviate poverty. It can, through national solidarity and fair burden sharing, contribute to human dignity, equality and social justice. It is also important for political inclusion, empowerment and the development of democracy.

The idea of such payments arose in Europe in the later part of the 19th century: for example, the introduction of compulsory social insurance in Germany in 1883; the introduction in Britain of non-contributory (i.e. fully tax-funded) old-age pension from 1909 and of compulsory health and unemployment insurance from 1911.

Even though social security, in all but name, had begun in the United States much earlier, dating back to the Civil War Pensions, that country's earliest social security legislation in the more contemporary situation, the Social Security Act was passed in 1935 and it was to enable the Federal Government to cope with the severe employment and income disruptions caused by the Depression of 1929. France, Italy and Canada had joined the tradition by enacting their first social-security legislations in 1910, 1919 and 1927, respectively, while Japan introduced its own first social-security law in 1941. Thus, by early 1940s, the world's leading market economies today, the so-called G7 nations, had all inaugurated their national social security programmes.

Similarly, all the other 21 member-countries of the Organization for Economic Cooperation and Development (OECD) have also put in place functioning social security systems which serve as the welfare-state components of their economies.

As would be expected, the regions of the world and the countries within each region differ enormously as regards the scope of social security measures and the provisions that they make available to their citizens; the most significant difference being that between the industrialized countries of Europe and America (e.g. , all the 28 members of the Organization for Economic Cooperation and Development, OECD), on the one hand, and the so-called developing countries of Africa, Asia, Latin America and the Caribbean, on the other.

For example, according to the most recent ILO assessment (2000), these two broad groups of countries differ markedly in terms of the three dimensions of coverage identified and applied by ILO, namely, (i) the range of protection provided (ii) the level of protection and (iii) the categories of people covered. That assessment found that, even though most countries (166 out of the 175 surveyed) claimed to have one or more social security schemes which, allegedly, cover the contingencies of old age, disability (invalidity) and survivors, the reality is that more than half of the world's labour force and their dependants are not yet covered by such existing schemes. As the ILO report points out, the problem of exclusion is particularly acute in Sub-Saharan Africa, as it is in South Asia, where the estimated coverage of the existing fragmentary schemes in is about 5-10 percent of the labour force. And in mot cases, that privileged group consists exclusively of people working for an employer and excludes the self-employed.

2.2.1 Summary Accounts of the state of social security in selected countries:

There are many examples of countries which have created, developed and sustained for a very long period national social security schemes which can be regarded as typically comprehensive, some more so than others. For example;

GERMANY

Type(s) of Social Security Programme

- Social Insurance System
- Special schemes in most cases for certain categories of self-employed persons, miners, civil servants, farmers, seamen, railway men etc.
- Both social-insurance and social-assistance programmes for Unemployment Benefits.
- Both universal and social-assistance programmes for Family allowances.

Risks/Benefits Covered

- Old-Age, Disability, Survivors, Sickness, /Maternity, work Injury, Unemployment, and Family Allowances.
- Old-Age Benefits consist of Old-Age Pension, with additional compensation for low-income persons with 35 years of coverage.

Participants/Beneficiaries

- In general, both employees and certain categories of self employed persons, including apprentices.
- Self-employed persons are specifically excluded from certain benefits, e.g. Sickness and Maternity; most categories of self-employed persons are excluded from Work Injury.

Sources of Funds

- Contributions by insured persons and contribution by employers.
- Government subsidy to cover difference in contributions and payments (of benefit) due.
- No employee contribution towards Work-Injury Benefits.
- Government bears total cost of Family-allowance Benefits.

Special Features

- Unemployment Benefits extend to self-employed persons, home workers, apprentices and trainees but excludes persons in irregular employment.
- Universal Child-Benefit and Means-tested Child Allowance and Child-Rearing Allowance, the total cost of which is borne by the Government.

UNITED KINGDOM

Type(s) of Social Security Programme

- Both Social Insurance and Social Assistance for all risks/benefits except Family Allowance
- Universal Child Benefits and tax credits for Family Allowances

Risks/Benefits Covered

- Old-Age, Disability, Survivors, Sickness, Maternity, Work Injury, Unemployment, and Family Allowances.
- Old-Age Benefits consist of the Basic Sate (Flat Rate) Retirement Pension, Dependants' Supplement, State Second Pension, S2P (for people with lower income and/or incomplete contributions), Old-Person's pension (non-contributory retirement pension) and the Pensions Credit, which has replaced the former means-tested support)

Participants/Beneficiaries

- In general, all employees and self employed persons who satisfy the qualifying conditions.
- Self-employed persons excluded from Work Injury Benefits.

Sources of Funds

- Contributions by insured persons and contribution by employers.
- Government subsidy to cover difference between contributions and payments (of benefit) due.
- Government bears total cost or near-total cost of all means-tested and other non-contributory benefits e.g. Family Allowances, medical benefits, etc.

Special Features

- Medical Benefits cover all persons resident in the United Kingdom without any other qualifying conditions.
- Present scope of provisions reflects a recent major reform of the UK's social security system by Tony Blair's Labour administration

UNITED STATES

Type(s) of Social Security Programme

- Social Insurance System but with some element of Social Assistance, especially, the separate Federal and State programmes providing medical benefits for the indigent, the Federal and State family-assistance programmes, and the Old-Age Supplemental income benefit, which is means-tested.

Risks/Benefits Covered

- Old-Age, Disability, Survivors, Sickness, Maternity, Work Injury, Unemployment, and Family Allowances.
- Old-Age Benefits include the Old-age Pension (available to all insured, normally at 62), Dependants' Allowance, and Old-Age Supplemental Income benefits (means-tested).
- No maternity allowances. Replaced by statutory maternity pay by employers.

Participants/Beneficiaries

- In general, all employees and self employed persons who fulfill the qualifying conditions.
- Proven need or indigence is the main qualifying conditions for non-contributory/social-assistance programme, e.g., medical benefits for the uninsured, the family-assistance programme, the means-tested Old-age supplemental Income Benefit.
- Work Injury Benefits cover only employees, not Self-employed persons.

Sources of Funds

- Contributions by insured persons and contribution by employers.
- Government bears total cost of means-tested Old-Age Benefit (the Old-Age Supplemental Income Benefit), the total cost of hospitalization benefits for certain non-insured elderly.
- Federal tax revenue is used for the administration of state unemployment compensation programmes and the state tax-revenue is used for the payment of the unemployment benefits themselves.

Special Features

- Steady growth of social security in the US since 1935, e.g. Total benefit payment only had increased very substantially from slightly less that $1.0 billion in 1950 to $248.0 billion in 1990 and $408.0 billion in 2000.
- Social security has become the most prevalent source of income for the elderly in the US; the vast majority of elderly household heads receive social security.
- As high as 7 percent of America's huge Gross Domestic Product is committed annually to social-security retirement pensions.
- Unemployment Benefits programme has a very wide coverage: employees of firms in industry and commerce; employees of non-profit organizations with four or more workers; almost all state and local government workers, domestics, and more than 75 percent of farm employees.

- United States' social security programme (with benefits payment alone standing at $408.0 billion, as at 2000 and $450.0 billion as at 2002) is, most probably, the largest single government programme in the world.
- Social security is a major public-policy focus and issue in the US.

NORWAY

Type(s) of Social Security Programme

- Mainly Social Insurance System
- Both Social Insurance and Universal, e.g., for Work Injury, Unemployment, and Family Allowances

Risks/Benefits Covered

- Old-Age, Disability, Survivors, Sickness, Maternity, Work Injury, Unemployment, and Family Allowances.
- Old-Age Benefits include the Universal Old-Age Pension, payable in full for 40 years coverage; an income-tested Dependant's supplement; and an Earnings-Related Old-Age Pension

Participants/Beneficiaries

- In general, both employees and self employed persons who fulfill the qualifying conditions.
- All persons resident in Norway are entitled to family allowances appropriate to their family circumstances

Sources of Funds

- Contributions by insured persons and contribution by employers.
- National Government bears the total cost of Family-Allowance benefits. For the social security programme as a whole, it covers

any deficit that may arise when total contributions fall short of total benefits-payment due

Special Features

- Both Universal and Earnings-Related Pensions apply to both employees and self-employed persons
- Retirement from gainful or paid employment is not required for Old-Age Pensions but payments are adjusted.
- Several benefits cover all persons resident in Norway.
- Work-injury Programme covers self-employed persons, though on a voluntary basis.
- Government bears total cost of Family Allowances.

JAPAN

Type(s) of Social Security Programme

- Social-Insurance System for all programmes except Family Allowance
- Employer-Liability and Social Assistance System for Family Allowances.

Risks/Benefits Covered

- Old-Age, Disability, Survivors, Sickness, Maternity, Work Injury, Unemployment, and Family Allowances.
- Old-Age Benefits include a flat-rate benefit and Dependants' Supplement, for all residents under the National Pension programme and an Earnings-Related benefit under the Employee Pension Insurance Programme.

Participants/Beneficiaries

- In general, all employees and self employed persons who satisfy the qualifying conditions.
- All residents with one or more children below the age of 12 are entitled to Family Allowances appropriate to their family circumstances

Sources of Funds

- Contributions by insured persons and contribution by employers.
- Family Allowances funded by Employers and Government

Special Features

- Retirement from gainful or paid employment not required for Old-Age Pension, though benefits-payments are adjusted.
- Survivors' benefits not payable to widowers.
- Health Insurance Contribution and benefits cover all persons resident in Japan.
- Family/Children's allowances, though universal, are income-tested and payable only for children below the age of 12.

Some notable African countries that are already providing effective and adequate social security services to their citizenry are South Africa, Namibia, Tunisia, Libya and Ghana

2.3 Pension Scheme(s) in Nigeria

Historically, traditional means of social protection existed in Nigeria, mostly through collective solidarity within the framework of extended family system. However, with modern industrialization, the traditional system of social protection virtually collapsed. This explains the rationale behind governments' interest in setting up some minimum protection schemes like public service pension scheme, private/public sector occupational schemes, and the National Provident Fund (NPF).

The pension system in Nigeria before the enactment of the Pension Act 2004 was dual in nature. The system in force in the public sector differed markedly from the one in existence in the private sector.

2.3.1 The Public Sector System

The pension system in the public sector was PAYGO. It was not funded and the resources for meeting the pension obligations were drawn from the Consolidated Revenue Fund of the Federation through annual budgetary provisions. Unlike in several other countries operating PAYGO system, there was no arrangement for funding through a payroll tax. Thus, it was not contributory. Public sector employees came to regard the provision of pension benefits as a deferred element of their employment compensation package.

On the benefits side, the system was characterized by two components of payments. On retirement, retirees were paid a lump sum benefit in the form of gratuity. The gratuity payment was based on the number of years of service and the terminal compensation package of the retiree. In addition to the lump sum payment, monthly pension payments were guaranteed for life. The rate of payment depended on the length of service.

2.3.2 Private Sector Scheme

The pension scheme operational in the private sector evolved from an initial provident fund scheme which was merely a savings scheme. The broad objectives of establishing the National Provident Fund (NPF) through an Act of Parliament in 1961 was to provide poverty alleviation measure as required by convention 102 of the International Labour Organization (ILO), to which treaty Nigeria is a signatory. In 1962, the scheme was amended by another Act of Parliament to cover only employees in the private sector who were not covered by the public service pension scheme. The Scheme was meant for the protection of private sector employees, from financial difficulties in the event of either old age, cessation of employment, invalidity or death.

Most employers then did not have such provisions in their employment policies.

The act establishing the Scheme provided for monthly contributions by members at the rate of 6% of basic salary subject to a maximum of ₦8.00 per month or a total of ₦96.00 per annum. The amount was paid in equal proportion of ₦4.00 by the employer and the employee.

Within the life span of thirty-three (33) years of its operation (1961 – 1994), the NPF suffered severe operational inadequacies and constraints. As a result, its impact was hardly felt by stakeholders and regrettably, it developed a poor public image that was difficult to manage. The major shortcomings of the defunct NPF were as follows:

(a) Stagnant contribution rate of ₦96.00 per annum for about thirty-three years.

(b) A hyper-inflationary economic environment, which adversely affected the resource base of the Scheme.

(c) Evasion of the Scheme by many employers.

(d) Poor record keeping of members' accounts, contributions and benefit.

(e) Unviable and ineffective investment policies occasioned by the Trustees Investment Act, which limited the investment horizon of the Organization.

(f) Inefficient benefit administration coupled with meager benefit payments.

(g) Management ineptitude and poor service delivery.

The desire to transcend the limitation of the National Provident Fund (NPF) which was established by the 1961 Act of Parliament, led to the establishment of Nigeria Social Insurance Trust Fund (NSITF) through decree 73 of 1993.

2.3.3 Transformation of NPF into the NSITF Scheme

In 1992, the Federal Government accepted the recommendations of the Technical Committee on Privatization and Commercialization (TCPC) now the Bureau of Public Enterprises (BPE) and converted the provident fund scheme run by the NPF to a limited social insurance scheme. Government approved the partial commercialization of the organization and changed its name to the Nigeria Social Insurance Trust Fund (NSITF) to reflect its new role and objective. In 1993, the Federal Government promulgated the NSITF Decree, which repealed the NPF Act of 1992 (as amended) and vested all the assets and liabilities of the NPF Management Board in the new NSITF via Decree 73 of 1993.

Nigeria Social Insurance Trust Fund (NSITF) became operational on July 1, 1994 as a social security system with a limited coverage including pension, invalidity, death, accident and disability benefits. It is limited in perspective because it caters only for its members. All registered members of the former NPF scheme became automatic members of the new scheme and their contributions were similarly transferred to their contributions records using the existing membership numbers under the new scheme, to ensure continuity. The NSITF operated on a tripartite arrangement of: employers (represented by NECA), workers (represented by NLC) and the government.

Unlike the public sector scheme, this was a contributory scheme, with contributions from both employers and employees. The funds were managed centrally by the NSITF. This scheme was in operation for ten years and was generally adjudged to have performed reasonably well. Benefit payments were made as and when due. Accumulated funds were well managed. Apart from this central scheme, there are several in-house pension scheme arrangements in the private sector.

In spite of the establishment of both the National Provident Fund (NPF) and its successor, the Nigeria Social Insurance Trust Fund (NSITF) and in spite of the social-protection services which were run by NPF from 1961 to 1994 and the ones which NSITF developed from 1994, Nigeria is still in the category of countries where social security coverage remains limited so far below ten percent (10%) of the labour force. And it is also significant that existing schemes in Nigeria have been targeted exclusively at workers who that not less than ninety per cent (90%) of the country's total labour force comprising those employed in the informal sector, all self-employed persons and vulnerable groups, are not covered by any of the existing social-protection schemes including those to which government extends substantial budgetary support.

The vast excluded section includes all the nation's farmers and all other rural producers, self-employed artisans in the urban and rural areas (motor mechanics, carpenters, masons, other construction workers, small-scale transport entrepreneurs and their workers, petty traders and shopkeepers, all other domestic workers; and the dependants of all these groups). (Akeredolu-Ale, 2004)

2.3.4 The 2004 Pension Reforms

The public sector pension scheme was an unfunded pay-as-you-go (PAYGO) system beset with inefficiencies and tale of woes. As a result of the crisis occasioned by mal-administration, government felt that the solution was to reform the pension schemes of both the public and private sectors.

The Pension Act 2004 introduced a unified economy-wide pension scheme. The major change introduced by the reforms was the replacement of the PAYGO-DB (Pay As You GO-Defined Benefit) system previously operational in the public sector by a FF-DC (Fully Funded-Defined Contribution) system, which is now applicable to both the public and the private sector. The other change introduced by the reforms in the institutionalization of individual retirement accounts. The public pension system has therefore become privatized.

The reformed system provides for the emergence of Pension Fund Administrators (PFAs) with whom individual retirement accounts will be placed form investment and management, Pension Fund Custodians (PFCs) with whom the custody of the contributed funds is entrusted and the National Pension Commission which serves as the regulatory body in the new Pension Industry.

Prior to the enactment of the Pension Reform Act 2004, both the public and private sectors operated a defined benefits scheme which in the case of the public sector was largely unfunded. The private sector schemes, although better funded and managed, had limited coverage and preferred to pay pension benefits in lump sum (gratuity). The NSITF scheme, statutorily designed to have a wider coverage of the private sector, suffered from a low compliance rate and provided inadequate benefits payment.

The Pension Reform Act was enacted to address these and other challenges. It introduced the following major changes:

- A defined contributions scheme thus ensuring a funded status for all pension schemes under the Act
- Establishment of a Retirement Savings Account (RSA) for all eligible employees to be managed by licensed Pension Fund Administrators (PFAs)
- Separation of pension fund administration and custody. Pension fund assets will be held by specially licensed Pension Fund Custodians (PFCs)

- Liberalization of the industry
    - Allows private sector management of public sector pension schemes for the first time
    - Discontinued the NSITF defined benefits scheme for the private sector
- Stricter regulatory environment with the establishment of the National Pension Commission (PENCOM), the first regulatory agency created specifically for the pensions industry in Nigeria

The Act also mandated NSITF to set up a privately owned PFA to compete with other PFAs in the emerging pensions industry and also to manage the accumulated pension funds of current NSITF contributors for a transitional period of five years, before moving such funds to other PFAs at employees' request.

2.5 Benefit Payment Administration

The extract below is from the NSITF Benefit Manual produced in 2003 to amplify the provisions of NSITF Act No. 73 of 1993 and NSITF (General) Regulations, 1994 relating to Benefits administration:

"In a Social Insurance Scheme, benefits are taken as rights due to claimants who qualify and apply for payment and an obligation to the Fund. It is mandatory for the Fund to pay claimants who qualify for benefits as at when due. Social Insurance benefits are earned benefits, devoid of any kindness, help or favour on the part of the administering authority.

One of the major criticisms of the defunct NPF was the meager benefits paid to claimants. Unlike the NPF, which was merely a savings scheme that provided for lump sum payments of what was contributed plus a fixed interest as benefits, the NSITF scheme pays grants and monthly pensions, as benefits. There are seven types of benefits which members of the NSITF scheme are entitled to apply for subject to satisfying the qualifying conditions. The benefits are:

- Invalidity Pension
- Invalidity Grant
- Retirement Pension
- Retirement Grant
- Survivors Grant
- Survivors Pension
- Funeral Grant

Benefit claims were paid by NSITF through its Branch Offices nationwide. All branches are under strict instructions to ensure that benefit claims are processed and paid within two (2) weeks of receipt through designated banks, and in the event of a detected fault in claim, to advise the claimant within the same period, if not a shorter period.

Furthermore, efforts were intensified to ensure that contributors to the NSITF Scheme get real value and enhanced benefits at the end of their active service in paid employment. To achieve this, in line with dictates of the law that established the Fund, the NSITF Management commenced payment of 80% of National Minimum Wage in November 2002. The implication is that, the minimum pension payable to a beneficiary is ₦4,400.00 every month. This is in addition to the lump sum of ₦105,600.00 which is also paid as a grant, at first payment. These minimum payments are guaranteed to qualifying claimants." (NSITF Benefit Manual, 2003)

2.5.1 Contingencies and Eligibility Conditions

Currently, the following contingencies are covered and appropriate benefits are paid to qualifying claimants in the NSITF Scheme:

- Invalidity Pension
- Invalidity Grant
- Retirement Pension
- Retirement Grant
- Survivors Grant
- Survivors Pension
- Funeral Grant

Retirement Pension

- The claimant must have attained the age of 60 years or more as confirmed by the registration records,
- He/She must have retired from employment or shown to the satisfaction of the Fund that he/she is no longer employed in any work for which remuneration is equal to or exceeds the National Minimum Wage,
- Contributions must have been paid or credited in respect of the claimant for not less than one hundred and twenty (120) months of insurable employment.

Retirement Grant

- The claimant must have attained the age of 60 years or more.
- She/he must have retired from employment or shown to the satisfaction of the Fund that he/she is no longer employed in any

work for which remuneration is equal to or exceeds the National Minimum Wage.

- Contributions must have been paid or credited in respect of the claimant for less than 120 months but not less than 12 months of insurable employment.

Invalidity Pension

- A member shall be eligible for Invalidity Pension/Gratuity if he/she meets the following conditions:
    a. She/he is permanently incapable of working by reason of some specific disease or bodily/mental disablement, in any occupation such that he/she would be incapable of carrying out as much as 1/3 of the work that a fit person of similar training and previous occupation can perform.
    b. The period of invalidity shall be one in which a person has been certified as incapable of work by a Medical Board appointed by the Fund.
    c. Effective date of invalidity shall be the date invalidity occurred or cessation of employment on grounds of invalidity, whichever is later
    d. Invalidity Pension will be payable monthly for life or until the invalidity ceases.
    e. A claimant must have at least 36 months contribution in total, with 12 months of same within the preceding 36 months.
    f. The "final average insurable earning" shall be derived by determining the average monthly insurable earnings of an

employee paid during the last 36 months immediately preceding the months in which the period of incapacity commenced.

Invalidity Grant

- A member shall be eligible for invalidity grant if he/she meets the following conditions:
  - He/she is permanently incapable of working by reason of some specific disease or bodily/mental disablement, in any occupation such that he/she would be incapable of carrying out as much as one third (1/3) of the work that a fit person of similar training and previous occupation can perform;
  - The period of invalidity shall be one in which a person has been certified as incapable of work by a Medical Board appointed by the Fund;
  - Contributions must have been paid or credited in respect of the member for not less than 12 months, but not enough to qualify for invalidity pension;
  - The grant is payable in lump sum.

Survivor's Pension

- Survivors Pension is payable to the dependant(s) upon the death of a member:
  a. Who was receiving retirement or invalidity pension and has received same for a period of less than 60 months.
  b. Who would have satisfied the eligibility conditions of Retirement, Invalidity or Transitional Basic Pension by virtue of his/her contributions.

c. Who at the time of his/her death had paid a minimum of 120 months contribution, or paid a minimum of 36 months contribution with 12 months of the same paid within the preceding 36 months.

Survivor's Grant

- Survivors Grant is payable to the dependant (s) of a member who had contributed for at least 12 months but failed to meet the qualifying conditions for the survivor's pension.

Funeral Grant

Funeral Grant is payable to the dependants of a deceased member who at the time of death:

- Was receiving Invalidity or Retirement pension, or
- Had paid contribution of 60 months or more.

2.5.2 Computation of Retirement Pension & Gratuity

According to the Benefit Operational Manual 2003 as produced by NSITF, Retirement Pension and Gratuity are computed in the following seven major steps:

(i) determine the credit months earned by the member;

(ii) determine the pension rate;

(iii) determine best final average insurable income (AII);

(iv) compute Preliminary Pension = Pension Rate x AII;

(v) Compare with the Minimum pension. The Final Pension is the **higher** of: preliminary pension and the minimum pension;

(vi) compute the Gratuity which is 24 months Final Pension;

Worked Examples of Computation of Retirement Pension & Grant

Example 1:

Mr. Okonkwo, for whom NSITF contributions have been paid by his employer for 120 months, resigned his appointment before attaining the age of 60 years. The average of his best 36 months Insurable Income paid was ₦24, 000.00 per annum.

Calculate his benefits from the Fund on attaining the age of 60 years.

Solution:

(i) Qualifying age for retirement benefits = 60 years or above.

(ii) Total number of contributions made by Mr. Okonkwo = 120 months

(iii) Pension rate = 30%

(iv) Average of the best 36 months insurable income

(v) Pension due = $\frac{30 \times \text{₦}\ 24,000.00}{100}$ = ₦ 7,200.00 per annum

= ₦600.00 per month.

Mr. Okonkwo should have been entitled to a pension of ₦ 7,200.00 per annum, but with the minimum pension rule his pension shall be ₦ 4,400.00 (80% of ₦ 5,500.00) per month, (which is the present national minimum wage per month) on attaining the age of 60 years or above.

In addition to the retirement pension, Mr. Okonkwo should have been entitled to a gratuity of ₦ 14,400.00 but with the minimum pension rule, his gratuity is ₦ 105,600.00 (24 months pension entitlement).

Entitlements:

A claimant shall be entitled to:

(a) PENSION

(i) A periodic monthly payment (pension) for life to qualified members from the date of application.

(ii) Retirement pension is calculated on Average Insurable Income (A.I.I.) of the claimant during the best 36 months of contributions paid.

(iii) Retirement pension is fixed at 30% of final average monthly earnings for the 120 months of insurable employment, and the percentage of retirement pension increases by 1.5% for every additional 12 months of contribution such that every month of contribution in excess of 120 months attracts an additional pension right of 0.125% of a claimant's Average Insurable Income.

(iv) The minimum retirement pension shall be the equivalent of 80% of the National Minimum Wage obtainable at any time.

(v) The maximum retirement pension shall be 65% of the average insurable income [or four (4) times the National Minimum wage which ever is lower].

(b) GRATUITY: a claimant who qualifies for a pension shall also be paid Gratuity (lump sum) equal to the value of 24 months pension entitlement.

CHAPTER THREE

SYSTEM ANALYSIS AND DESGN

3.1 Introduction

System Analysis is the process of undertaking detailed study of an existing system by deploying system analysis tools like on-site observation/investigation, questionnaire, interviews etc., with a view to eliciting useful information/facts from present system of operations, highlighting the inherent weaknesses and strengths, with a view to proffering solutions to strengthen the weaknesses and improving on the strength. This process of system analysis sometimes leads to development of an entirely new system.

System analysis will enable the researcher to study and know the functionalities and shortcomings of the existing system and enable him to know the feasibility of a new system or areas of improvement of an existing system.

3.2 System Analysis of the existing System

From investigations carried out, it was apparent that the present system of processing NSITF benefits of claimants is manual, time consuming and requires many personnel. The following procedures take place during processing of a benefit claim:

- An applicant seeking to claim a benefit from the Fund is interviewed by the Benefit Officer at the branch office. If the officer is satisfied that the member is eligible for any claim, appropriate application forms are issued to the member. Thereafter, the issuance is entered into a register maintained for that purpose.

- The outcome of the interview process is documented vide the use of the appropriate Interview Form and same is enclosed in the claimant's dossier as the first document in the dossier. Thereafter, all documents in the dossier are paged numbered.

- On receipt of a completed application form and other documents, the Benefits Unit checks and confirms the completeness of the information and documents required before accepting the application form. In all cases, the applicants is expected to submit original copies of the Certificate of Service and Death Certificate which must be sighted by the Branch Manager, who thereafter certifies having sighted the original on the photocopy to be utilized in processing the application. No application form is accepted until all the required information and documents are provided. Consequently, the register for application forms is updated.

- A dossier is subsequently opened for each applicant by the Benefits Unit to contain the application forms, documents and other transactions.

- A request for the photocopy of the basic card and extraction of the NPF balance is made to the ICT Department by the Benefits Unit *not later than the following day* after receiving the application.

    - While awaiting the receipt of the basic card, the Benefits Unit will ascertain:

(i) the applicant's credit months from the ledger,

(ii) where the information is not available in the ledger, manual extractions are made from NSITF Form 05 (Schedule of Monthly Contributions) by the Records Officer duly certified by him/her and passed to the Benefits Unit.

- After receiving the photocopy of the basic card and confirming the qualification of the member for the claim, the Benefits Unit computes the entitlement of the applicant.
- Where an application is not successful, the Benefits Unit takes immediate steps to formally inform the applicant accordingly, stating the reasons why the application fails and when it could be represented. The applicant is also requested to inform the Branch office about any changes in the information earlier provided where such changes occur before the new due date.
- Where an application is to be reconsidered on a future date, the dossier is marked for B.U. (bring up) on the expected date by an appropriate officer designated for that purpose. The applicant is not expected to fill fresh forms nor submit fresh documents, except to either update any information submitted in the earlier documents or provide any new material facts.
- The Benefits Unit determines the average insurable earnings of successful applicants and computes the benefits payable to the members. The processed applications are batch listed and referred

to the Branch Manager for approval after which they are referred to the Internal Audit for auditing.

- Audited and passed applications are sent back to Finance Unit for payment while those that are queried are sent to the Branch Manager with batch lists in all cases.

- As payments are made to beneficiaries only through designated banks, the Finance Unit prepares the appropriate Branch "Benefits Payment Schedule" advising the banks of the approved payees.

- From the Benefits Payment Schedule duly acknowledged by the bank, the Benefit unit informs the claimant of the payments and :

(i) Issues the claimant with a letter:

(a) Introducing him/her to the bank to collect his/her entitlements for a grant. [Letter to state claimant, BP schedule s/n on schedule and amount. Letter of introduction carries the claimant's photograph];

(b) Requesting that an account be opened for the claimant for pension payment.

(ii) For a pensioner, an identity card (ID) duly signed by the Branch Manager is issued simultaneously.

(iii) All I. D. cards bear:

(a) the recent passport photograph of the claimant,
(b) the claimant's thumbprint.

- The pensioner shall furnish the branch office with his/her account number within fourteen (14) days of receiving the letter of introduction failing which the payment of monthly pension shall not commence.

- The Finance Unit shall authorize the bank to credit the account of pensioners with the value of their respective monthly pensions as per the approved Benefits Payment Schedule for the current month not later than the 20th day of every month.

  (i) The payment of monthly pension to any claimant continues without any interruption for 60 months.

  (ii) The payment of monthly pension shall cease only in the following cases:

    a) the cessation of invalidity in the case of invalidity pension;

    b) the death of the pensioner

  (iii) Where a pensioner receives pension for less than 60 months before his/her death, the balance from 60 months pension shall be payable to the dependant(s) in lump sum as survivor's pension on application.

(i) After 60 months of receipt of pension, a pensioner shall be required to physically present him/herself to the Branch Manager nearest to his place of residence in April and October each year for confirmation that he/she is still alive, and for invalids, that the state of invalidity persists. In the event of any doubt for invalids the matter shall be referred to the Medical Board.

(ii) In the event of a pensioner in the category mentioned above being physically incapable of presenting him/herself to the office, the Branch Manager shall be obligated to personally visit the pensioner for such confirmation in his/her home.

- The Finance unit in the branch office shall obtain from the bank, a statement showing paid claimants in every month with the following details:

  (i) name of the claimant,
  (ii) membership number,
  (iii) claimant's bank account number (for pensioners),
  (iv) type of benefit paid,
  (v) amount paid,
  (vi) date of payment,
  (vii) reference number of the Benefit Payment Schedule.

- The Finance Unit in the Branch office prepares a monthly bank reconciliation statement and renders requisite returns to the Head Office both duly countersigned by the Branch Manager. Any discrepancy in the bank statement must be promptly resolved with the bank at the branch level.

Any subsequent communication with a claimant must be duly filed in the dossier maintained for him/her. The dossiers shall be secured by the Records Unit in every branch office and arranged alphabetically according to the respective contingencies to facilitate easy access.

3.3 Limitations of the existing System

- Processing is cumbersome and time consuming due to the manual nature.
- The system is prone to errors – especially in computation of benefits due to claimants.
- Too many personnel involved.
- Documents can be lost or misplaced as a result of moving the the documents from one office to another.
- The results produced by the system undergo several manual test before standing reliable.

Having highlighted the general limitations of the existing manual system, it is therefore suggested or recommended that the way forward should be in the computerization of the process of benefit payment. This consequently leads to the introduction of the proposed system.

3.4 Proposed System of Benefit Processing

The present approach is very tedious, tiring, time consuming and prone to a lot of errors because the process is fraught with multiple human interventions.

The proposed system intends to alleviate all of the present problems bedeviling benefit payment processing. All the identified limitations of the present system will be addressed by the proposed system. The proposed system will be computer based. A program will be developed to manage the records of members, determine their qualification for benefit and compute the benefit as appropriate.

3.5 Benefits of Proposed System

As expected, the feasibility study undertaken revealed the workability and benefits of the proposed system over the existing one. The following are the identified benefits of the proposed system amongst others:

1. Reduction in cost of processing benefit claims
2. Reduction in time taken for processing benefit claims.
3. Assurance of correct computation of benefits to claimants.
4. More efficient handling of members' records.
5. Less number of personnel required for benefit processing.
6. Avoidance of unnecessary duplication of data
7. Spontaneous response to enquiries.
8. Production of accurate, reliable and timely reports/results.

Just as it is normal for every system that has benefits must have some limitations too, the proposed system is not an exception; the following are the currently identified limitations of the proposed system:

- The system can only be used for managing the benefit payment process of the NSITF scheme. It cannot be used for managing benefit processing in the new pension scheme.
- As stated in the scope of the project, the proposed system can only handle processing of retirement pension and gratuity for now. Perhaps,

in the future, other contingencies could be taken up for computerization.

- The proposed system does not have a direct interfacing with the payment banks for online advice for crediting claimants' account. This feature is possible and could also be taken up in future.

3.6 Project Requirement

This section of the project lists the functional, non-functional and configuration requirements of the proposed system of benefit payment processing:

Functional Requirements

1. The system should allow update, addition and deletion of employee and employer records.
2. The system should allow update, addition and deletion of contributions records.
3. The system should allow validation of claimants.
4. The system should allow determination of qualification of claimant for retirement pension and grant.
5. The system should allow computation of qualified claimant's retirement pension and grant.
6. The system should be able to generate the following reports and more:

   (i) Notice of successful validation

   (ii) Notice of successful qualification for benefits.

(iii) List of processed claimants' benefits

(iv) Bank Advice Slip

Non-Functional Requirements

1. The system should be user friendly

2. The system should provide an interface to control user access.

Configuration Requirements

System Components	**Requirements**
Database	MySQL
System Architecture	3-tier - Data Access Layer, Business Logic Layer and Presentation Layer
Programming Language	C# - C Sharp

3.6.1 Choice of programming language

In choosing a programming language for implementing the proposed system, the researcher had to consider the limited time available to develop the application and ready availability of a programming language. The researcher had a choice between VB.NET and C# as available languages and the time to development had to be considered. **C#** was selected for its **rapid application development** features.

C# (pronounced "C sharp") is a modern programming language that is designed for building a variety of applications that run on the .NET Framework. C# is simple, powerful, type-safe, and object-oriented. The many innovations in C# enable rapid application development. It is for all the above reasons that C# was chosen as the programming language of choice for the project.

3.7 Design of Proposed System

In this section, entities that make up the Benefit Payment system database are listed, described and the ER diagram is shown:

List of Entities

EMPLOYER

EMPLOYEE

SCHEDULE

MONTHS

BRANCHES

PENSIONERS

Description of Entities

EMPLOYER: this entity stores detailed information about employees' employers. The information includes employer number, employer name, address, location etc.

EMPLOYEE: this entity stores information about an employee. Such information includes employee number, employee name, gender, date of birth etc.

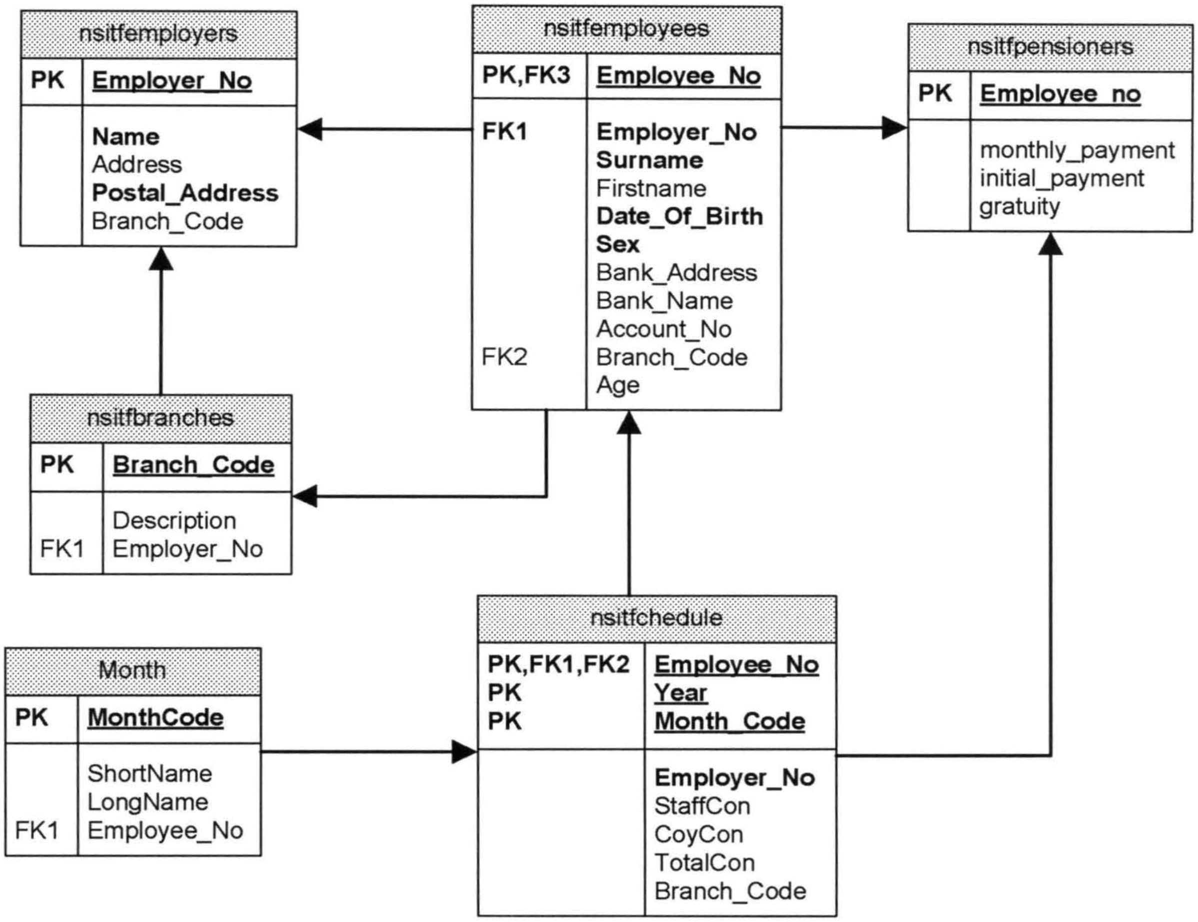
SCHEDULE: this entity stores information about the monthly contributions made on behalf of employees jointly by employee and employer.

MONTHS: this entity stores month codes and descriptions for use in the SCHEDULE table.

BRANCHES: this entity stores branch codes and descriptions for use in the EMPLOYER table.

PENSIONERS: this entity stores details of processed benefits of claimants. It is derived from both the EMPLOYEE table and the SCHEDULE table.

ENTITY RELATIONSHIP (ER) DIAGRAM



nsitfemployers	
PK	**Employer_No**
	Name Address **Postal_Address** Branch_Code

nsitfemployees	
PK,FK3	**Employee_No**
FK1	**Employer_No**
	Surname Firstname **Date_Of_Birth** **Sex** Bank_Address Bank_Name Account_No
FK2	Branch_Code
	Age

nsitfpensioners	
PK	**Employee_no**
	monthly_payment initial_payment gratuity

nsitfbranches	
PK	**Branch_Code**
	Description
FK1	Employer_No

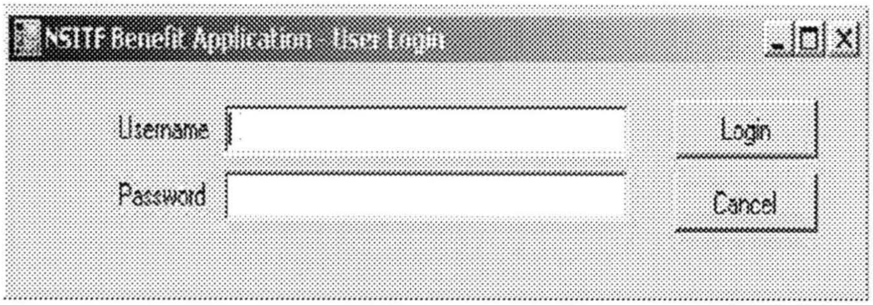
Month	
PK	**MonthCode**
	ShortName LongName
FK1	Employee_No

nsitfchedule	
PK,FK1,FK2	**Employee_No**
PK	**Year**
PK	**Month_Code**
	Employer_No StaffCon CoyCon TotalCon Branch_Code

3.7.1 Design of Application Interface

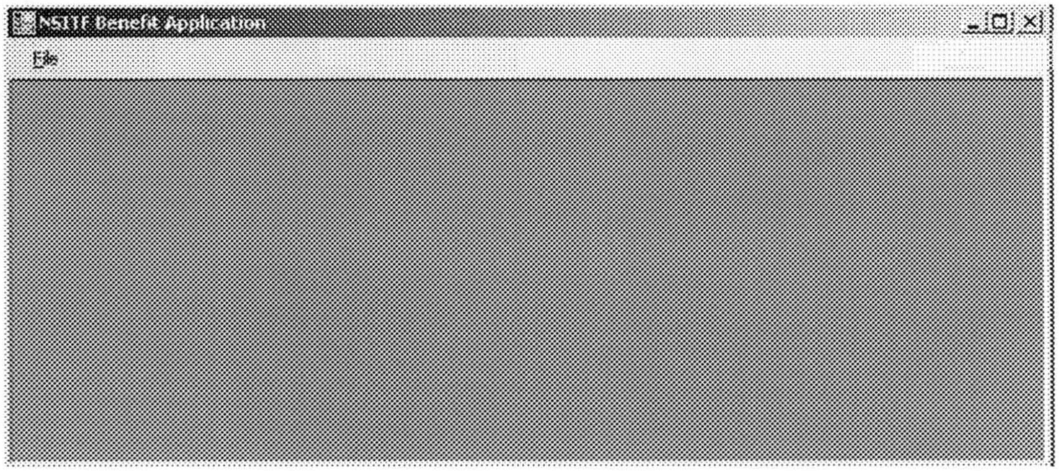
The application interface is based on Windows Forms. There is an opening dialog box for user log-on to control access to the application.

Log-in box

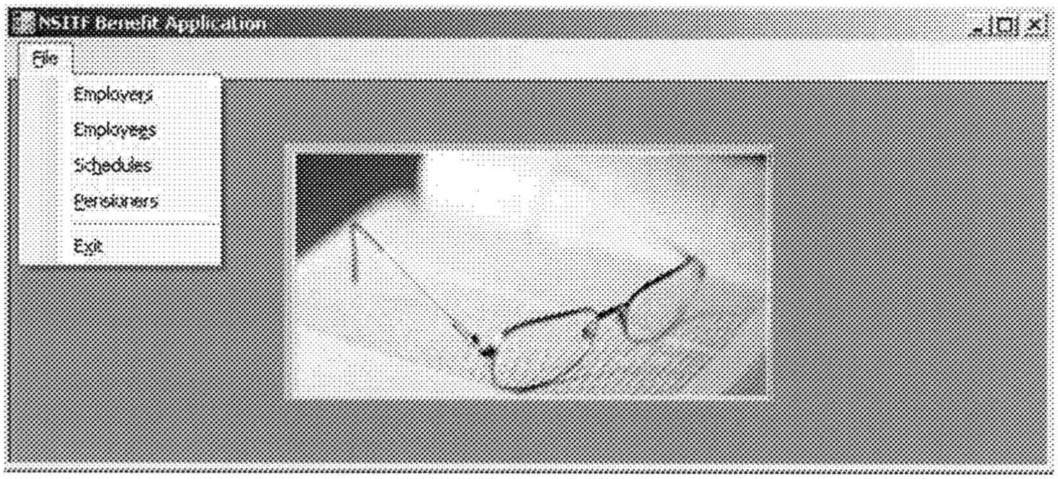


Thereafter, the application window is opened. The window contains one menu item in its menu bar. That is **File.** This main menu item has a drop-down menu which contains other options that control the functionalities of the system. The options are: Employers, Employees, Schedule, Pensioners and Exit.

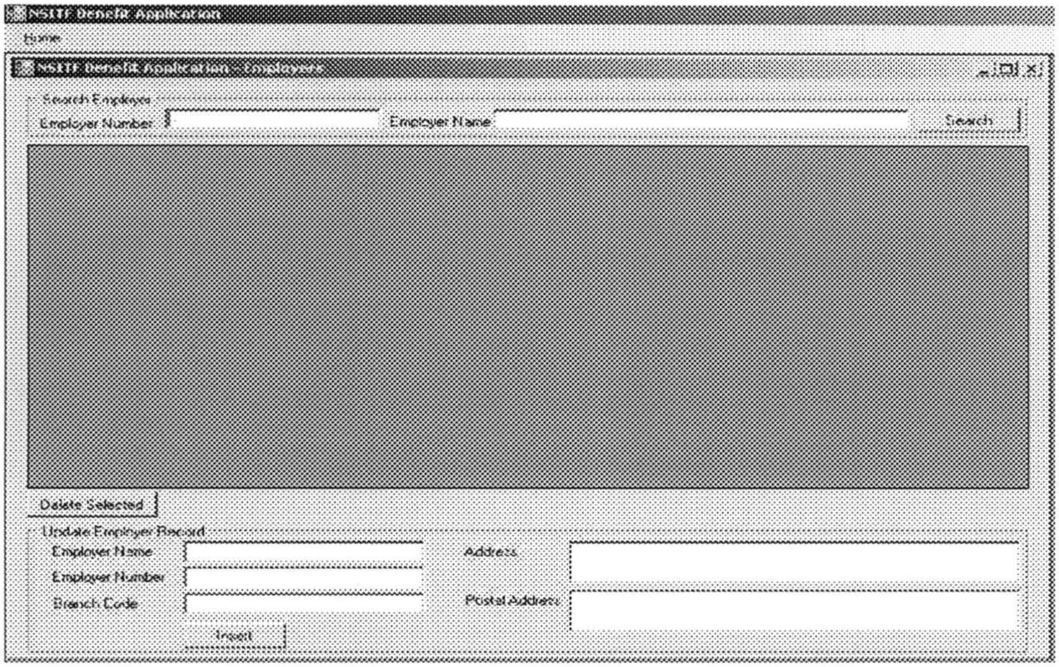
Main Menu - File



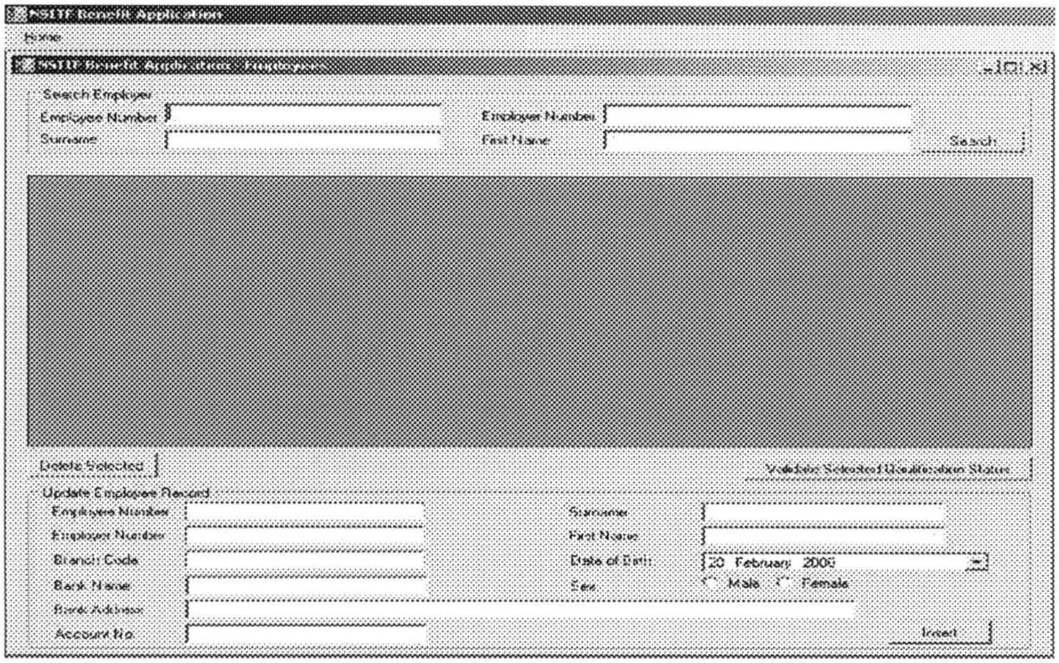
Drop-down Menu – Employers, Employees, Schedule, Pensioners and Exit



Employers: This option controls the functionality of viewing, editing, /updating and deleting Employer's record.

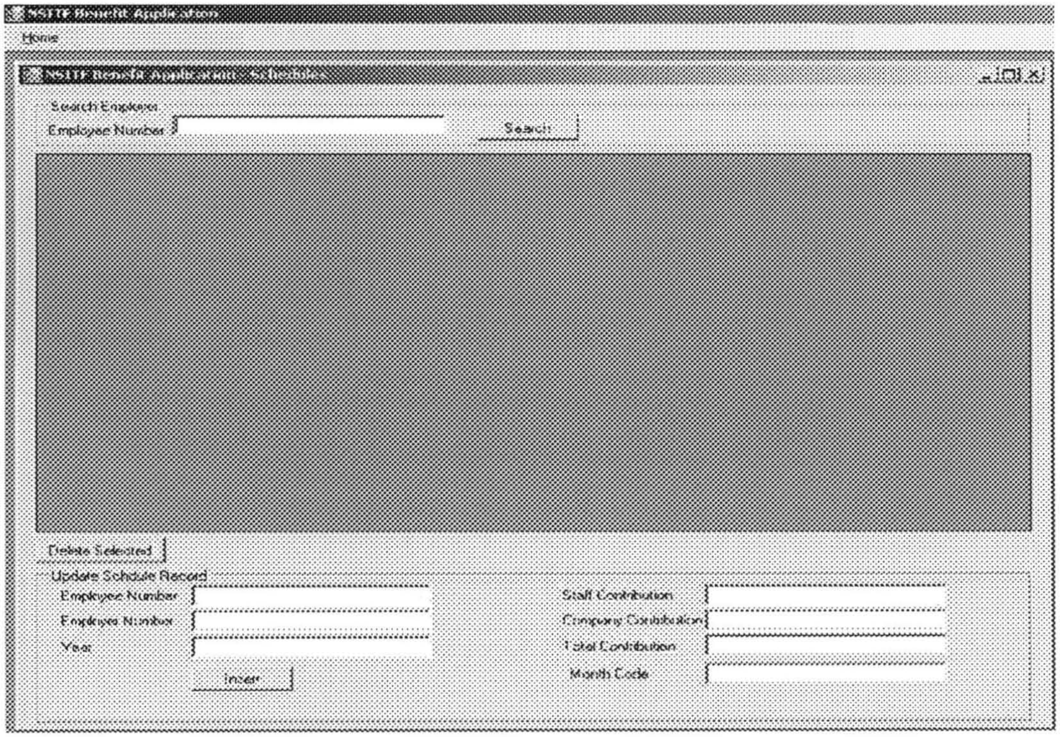


Employees: This option controls the functionality of viewing, editing/updating and deleting Employer's record. It also has the control for Validating Qualification for and computation of Benefits.



Date of Birth: 20 February 2006

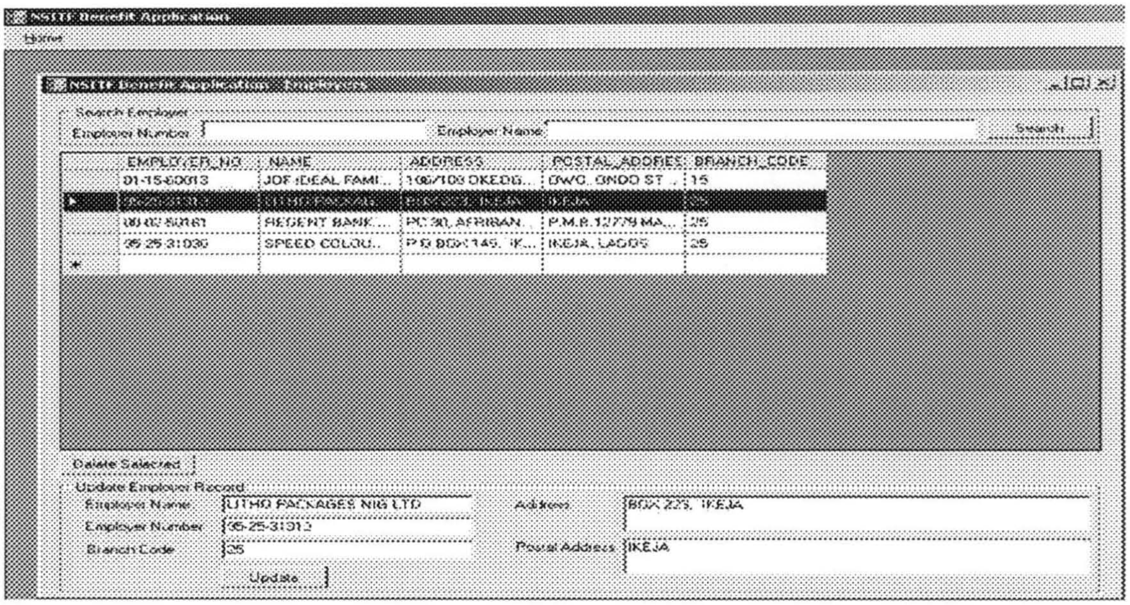
Schedule: This option controls the functionality of viewing, editing/updating and deleting Employer's contributions record.



3.7.2 Design of Forms

Apart from the forms that represent the application interface, several other forms were designed for viewing records and performing other functionalities available in the application. The following are some of the forms designed:

Form to View and Select Employer to perform operation on:



NSITF Benefit Application
Home
NSITF Benefit Application - Employers
Search Employer
Search
Delete Selected
Update

Employer Number:

Employer Name:

	EMPLOYER_NO	NAME	ADDRESS	POSTAL_ADDRES	BRANCH_CODE
	01-15-60013	JOF IDEAL FAMI...	106/108 OKEDG...	OWO, ONDO ST...	15
▶	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	00-02-50161	REGENT BANK...	PC 30, AFRIBAN...	P.M.B.12779 MA...	25
	95-25-31030	SPEED COLOU...	P.O BOX 145, IK...	IKEJA, LAGOS	25
*					

Update Employer Record

Employer Name: LITHO PACKAGES NIG LTD

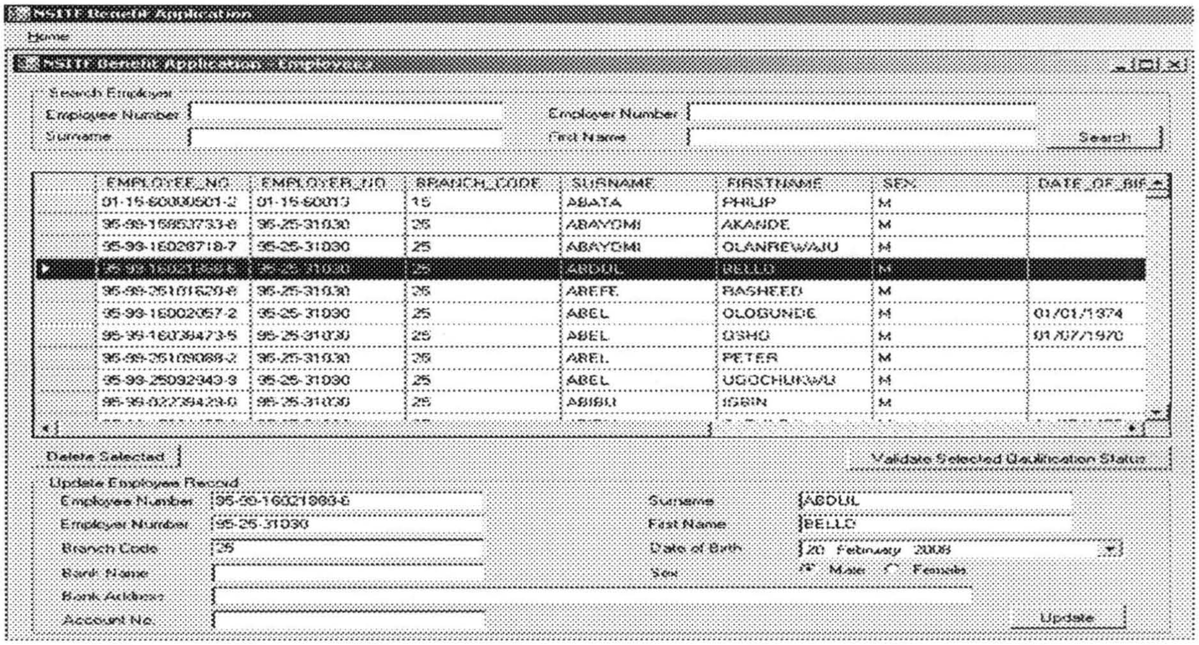
Employer Number: 95-25-31913

Branch Code: 25

Address: BOX 225, IKEJA

Postal Address: IKEJA

Form to View and Select Employee to perform operation on:



NSITF Benefit Application
Home
NSITF Benefit Application - Employees
Search Employer
Search
Delete Selected
Validate Selected Qualification Status
Update

Employee Number:

Employer Number:

Surname:

First Name:

	EMPLOYEE_NO	EMPLOYER_NO	BRANCH_CODE	SURNAME	FIRSTNAME	SEX	DATE_OF_BIR
	01-15-60000601-2	01-15-60013	15	ABATA	PHILIP	M	
	95-99-15853733-8	95-25-31030	25	ABAYOMI	AKANDE	M	
	95-99-16026718-7	95-25-31030	25	ABAYOMI	OLANREWAJU	M	
▶	[illegible]	[illegible]	[illegible]	ABDUL	BELLO	M	
	95-99-25101620-8	95-25-31030	25	AREFE	BASHEED	M	
	95-99-16002057-2	95-25-31030	25	ABEL	OLOGUNDE	M	01/01/1974
	95-99-16039473-5	95-25-31030	25	ABEL	OSHO	M	01/07/1970
	95-99-25109068-2	95-25-31030	25	ABEL	PETER	M	
	95-99-25092343-3	95-25-31030	25	ABEL	UGOCHUKWU	M	
	95-99-02239429-0	95-25-31030	25	ABIBU	ISBIN	M	

Update Employee Record

Employee Number: 95-99-16021983-6

Employer Number: 95-25-31030

Branch Code: 25

Bank Name:

Bank Address:

Account No.

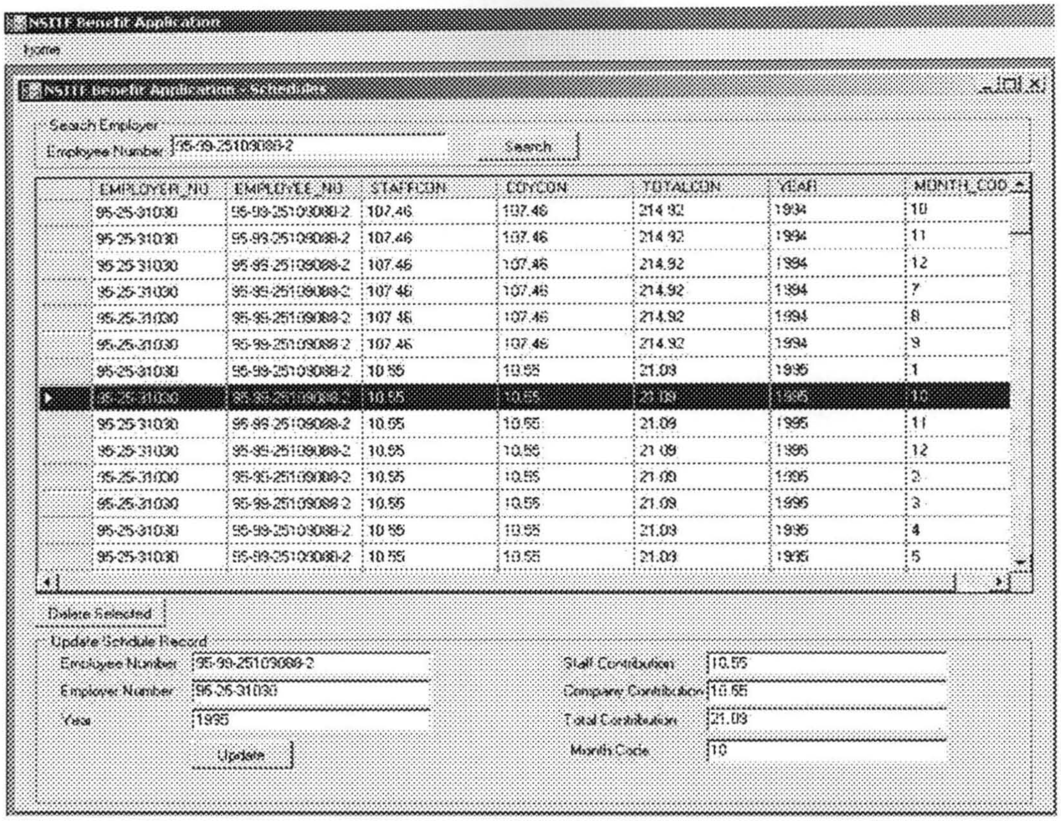
Surname: ABDUL

First Name: BELLO

Date of Birth: 20 February 2008

Sex: ● Male ○ Female

Form to View and Select Contribution to perform operation on:


NSITF Benefit Application
Home
NSITF Benefit Application - Schedules
Search
Delete Selected
Update

Search Employer

Employee Number: 95-99-25109088-2

EMPLOYER_NO	EMPLOYEE_NO	STAFFCON	COYCON	TOTALCON	YEAR	MONTH_COD
95-25-31030	95-99-25109088-2	107.46	107.46	214.92	1994	10
95-25-31030	95-99-25109088-2	107.46	107.46	214.92	1994	11
95-25-31030	95-99-25109088-2	107.46	107.46	214.92	1994	12
95-25-31030	95-99-25109088-2	107.46	107.46	214.92	1994	7
95-25-31030	95-99-25109088-2	107.46	107.46	214.92	1994	8
95-25-31030	95-99-25109088-2	107.46	107.46	214.92	1994	9
95-25-31030	95-99-25109088-2	10.55	10.55	21.09	1995	1
95-25-31030	95-99-25109088-2	10.55	10.55	21.09	1995	10
95-25-31030	95-99-25109088-2	10.55	10.55	21.09	1995	11
95-25-31030	95-99-25109088-2	10.55	10.55	21.09	1995	12
95-25-31030	95-99-25109088-2	10.55	10.55	21.09	1995	2
95-25-31030	95-99-25109088-2	10.55	10.55	21.09	1995	3
95-25-31030	95-99-25109088-2	10.55	10.55	21.09	1995	4
95-25-31030	95-99-25109088-2	10.55	10.55	21.09	1995	5

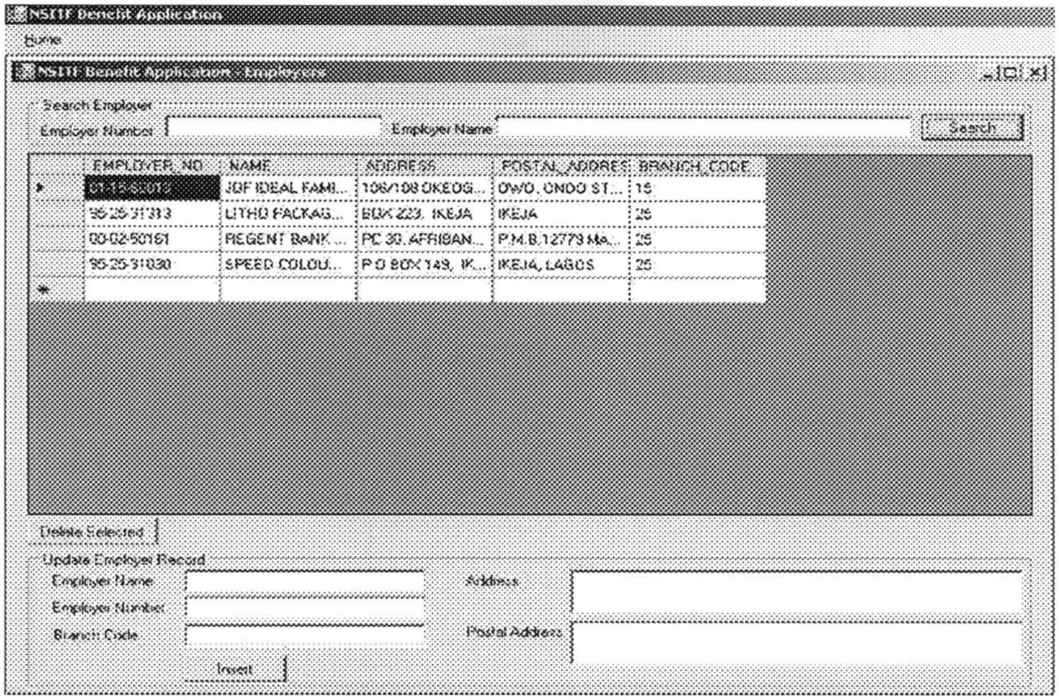
Update Schedule Record

Employee Number	95-99-25109088-2	Staff Contribution	10.55
Employer Number	95-25-31030	Company Contribution	10.55
Year	1995	Total Contribution	21.09
		Month Code	10

3.7.3 Design of Reports

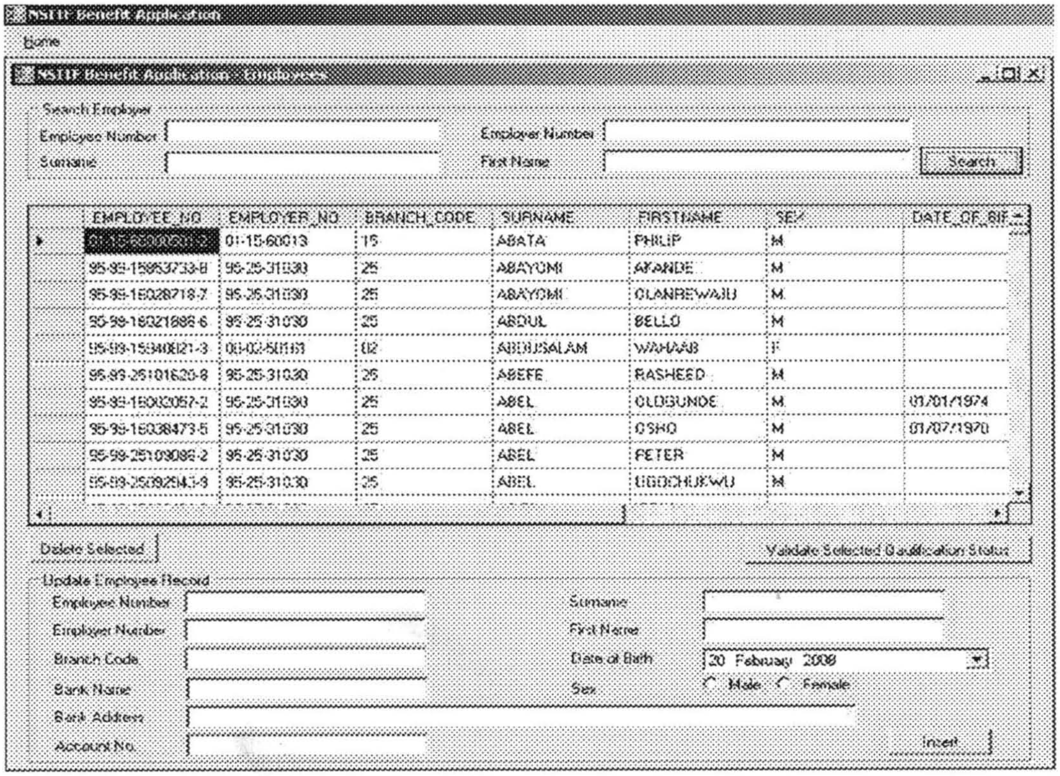
Considering the nature of the application, most of the expected reports are meant for display. For example, when a user enters a known employer number in the Employees interface, a display of all the employees belonging to that employer is presented in a data grid. Likewise, when a user of the system does not know the Employer number, a wide search of the entire table of employers can be done at the Employers Interface, where the search button can be used to display all available employers. Some of such reports are:

Employers List:



	EMPLOYER_NO	NAME	ADDRESS	POSTAL_ADDRES	BRANCH_CODE
▸	01-15-60013	JDF IDEAL FAMI...	106/108 OKEOG...	OWO, ONDO ST...	15
	95-25-31313	LITHO PACKAG...	BOX 223, IKEJA	IKEJA	25
	00-02-50161	REGENT BANK ...	PC 30, AFRIBAN...	P.M.B.12779 Ma...	25
	95-25-31030	SPEED COLOU...	P O BOX 149, IK...	IKEJA, LAGOS	25
*					

Employees List:



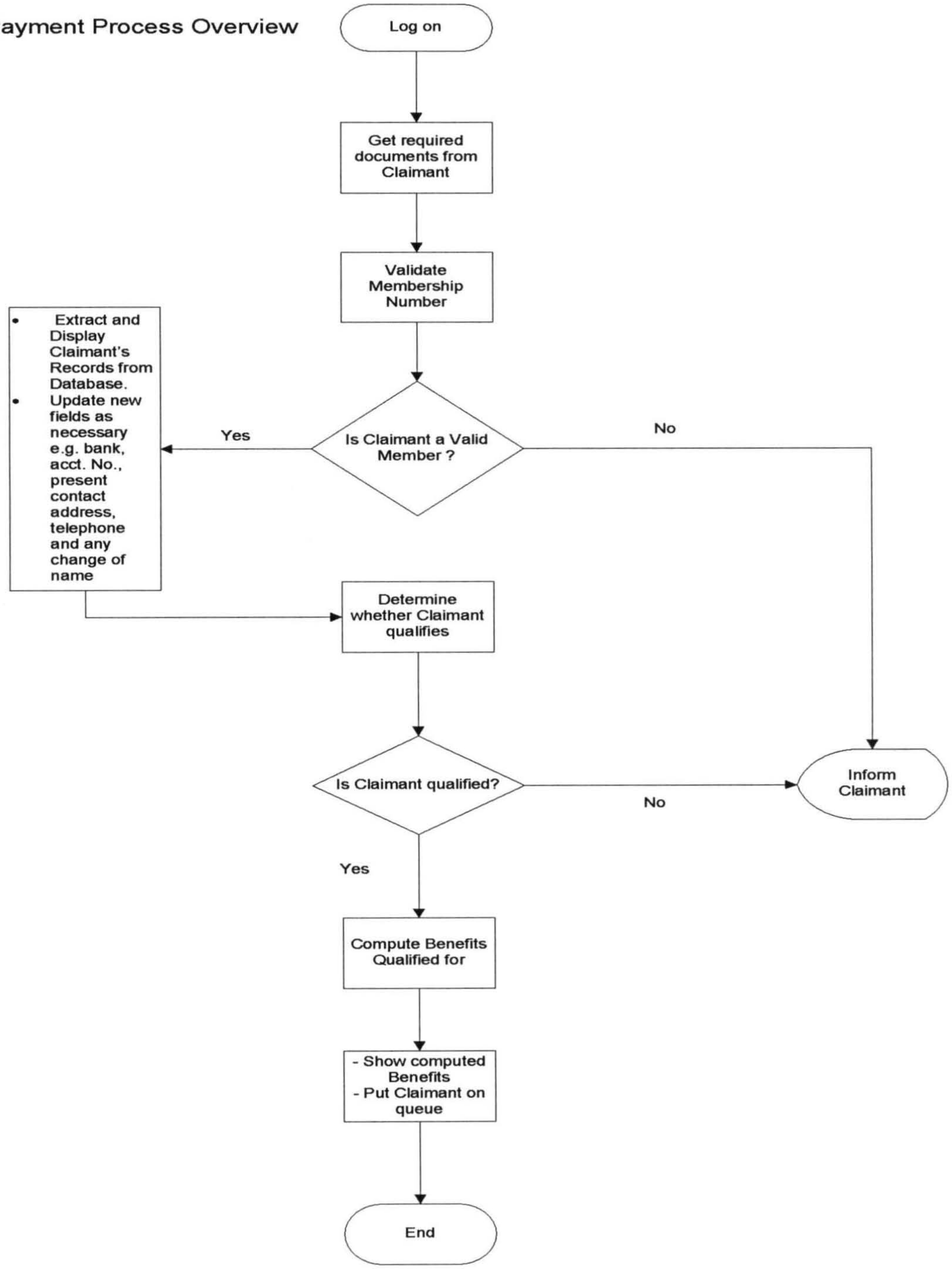
	EMPLOYEE_NO	EMPLOYER_NO	BRANCH_CODE	SURNAME	FIRSTNAME	SEX	DATE_OF_BIR
▸	01-15-60000581-2	01-15-60013	15	ABATA	PHILIP	M	
	95-99-15853733-8	95-25-31030	25	ABAYOMI	AKANDE	M	
	95-99-16028718-7	95-25-31030	25	ABAYOMI	OLANREWAJU	M	
	95-99-16021888-6	95-25-31030	25	ABDUL	BELLO	M	
	95-99-15340821-3	00-02-50161	02	ABDUSALAM	WAHAAB	F	
	95-99-25101620-8	95-25-31030	25	ABEFE	RASHEED	M	
	95-99-16002057-2	95-25-31030	25	ABEL	OLOGUNDE	M	01/01/1974
	95-99-16038473-5	95-25-31030	25	ABEL	OSHO	M	01/07/1970
	95-99-25109088-2	95-25-31030	25	ABEL	PETER	M	
	95-99-25092943-3	95-25-31030	25	ABEL	UGOCHUKWU	M	

Date of Birth: 20 February 2008

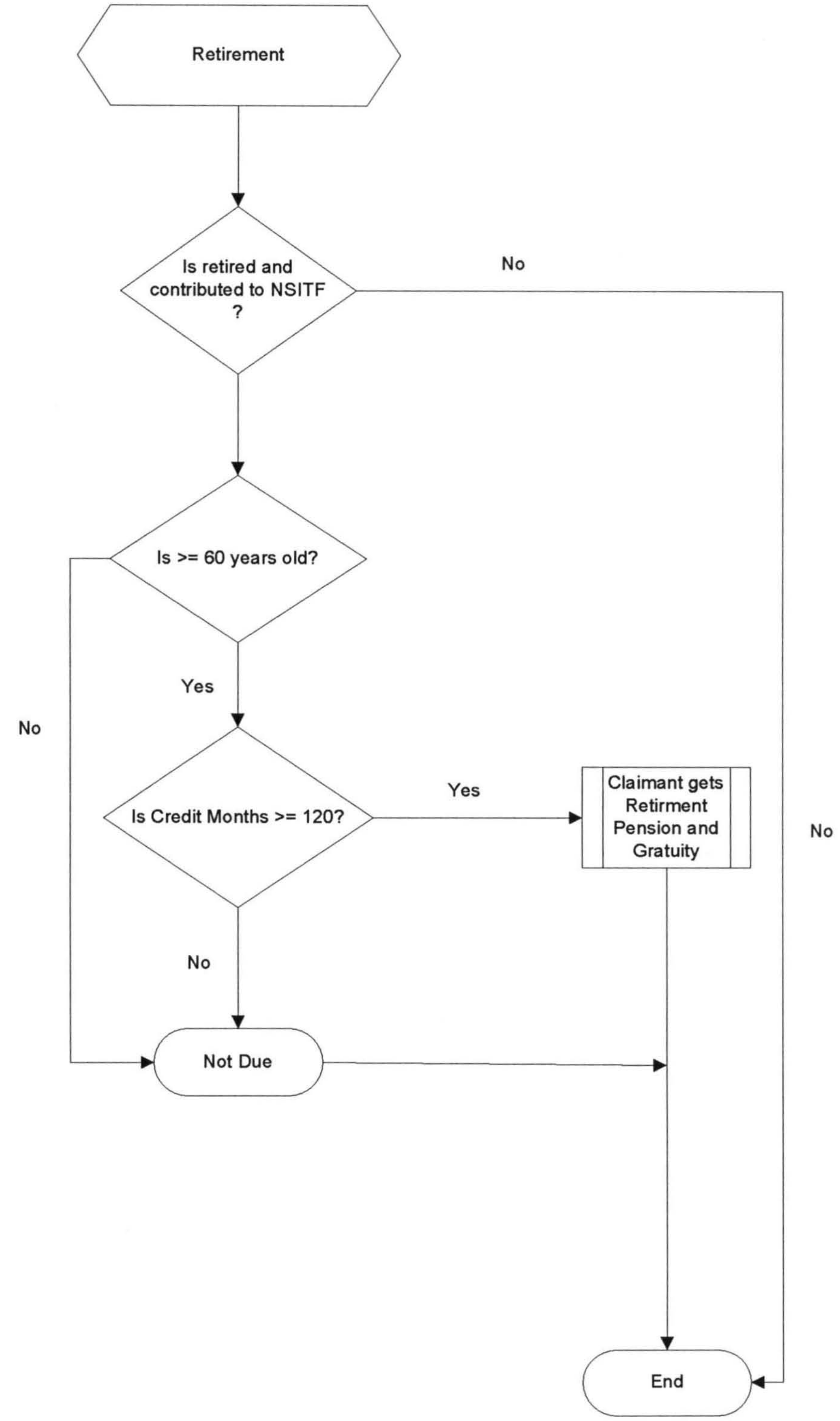
3.8 Program flowchart

Find below the flowchart for process overview of proposed Benefit Payment Processing System and program flowchart for determining qualification for benefit.

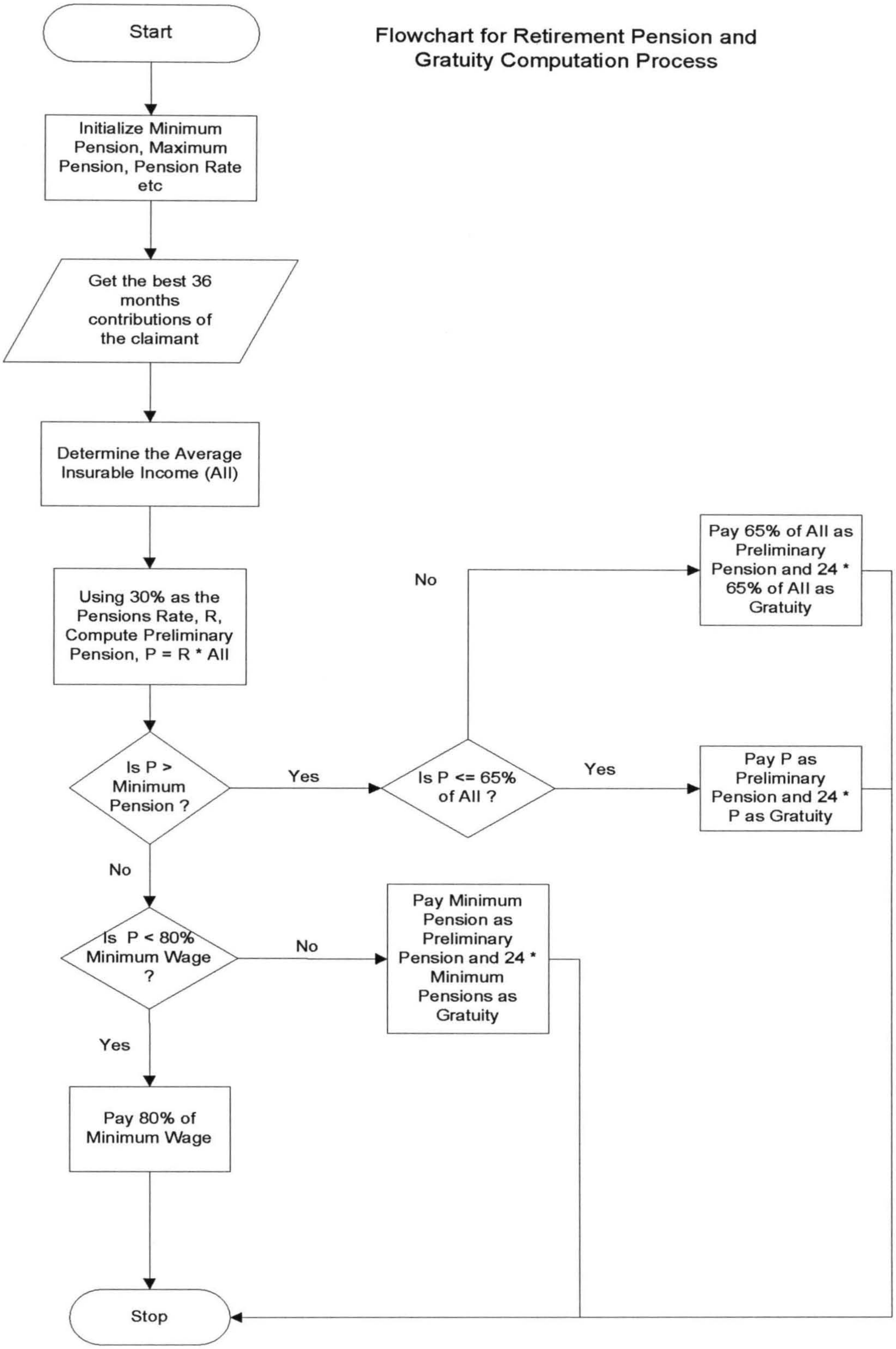
Benefit Payment Process Overview


Log on
Get required documents from Claimant
Validate Membership Number
Extract and Display Claimant's Records from Database.
Update new fields as necessary e.g. bank, acct. No., present contact address, telephone and any change of name
Yes
Is Claimant a Valid Member ?
No
Determine whether Claimant qualifies
Is Claimant qualified?
No
Inform Claimant
Yes
Compute Benefits Qualified for
- Show computed Benefits
- Put Claimant on queue
End

Flowchart to determine qualification


Retirement
Is retired and contributed to NSITF ?
No
Is >= 60 years old?
Yes
No
Is Credit Months >= 120?
Yes
Claimant gets Retirement Pension and Gratuity
No
No
Not Due
End

Flowchart for Retirement Pension and Gratuity Computation Process


Start
Initialize Minimum Pension, Maximum Pension, Pension Rate etc
Get the best 36 months contributions of the claimant
Determine the Average Insurable Income (AII)
Using 30% as the Pensions Rate, R, Compute Preliminary Pension, P = R * AII
Is P > Minimum Pension ?
Yes
Is P <= 65% of AII ?
No
Pay 65% of AII as Preliminary Pension and 24 * 65% of AII as Gratuity
Yes
Pay P as Preliminary Pension and 24 * P as Gratuity
No
Is P < 80% Minimum Wage ?
No
Pay Minimum Pension as Preliminary Pension and 24 * Minimum Pensions as Gratuity
Yes
Pay 80% of Minimum Wage
Stop

CHAPTER FOUR

SYSTEM IMPLEMENTATION

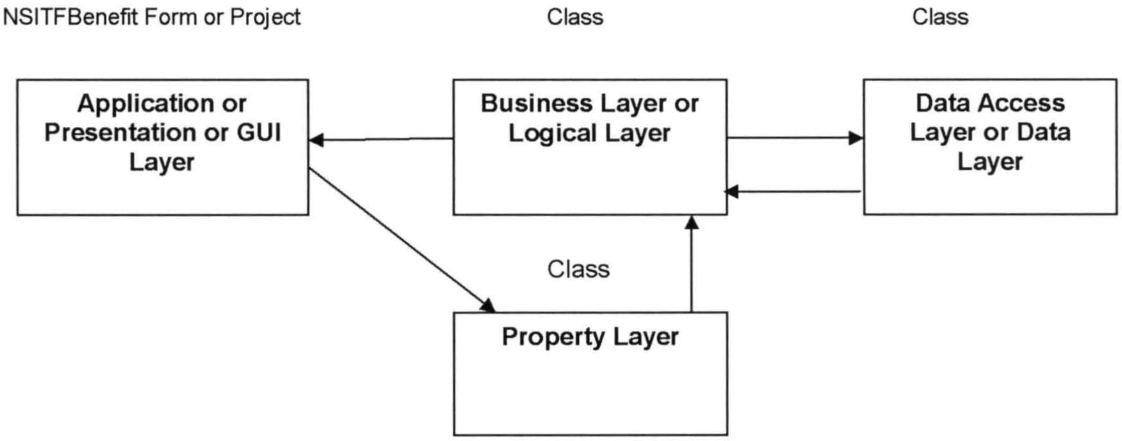
4.1 Introduction

The application is based on an exiting database of NSITF contributors and contributions made to NSITF under the NSITF scheme for private sector employees. The database contains about 40 million records of contributions. This project intends to develop an application for managing the records on the database, receive claimants' request for benefit and process appropriate benefit of qualifying claimants. The benefit payment processing system is a one-stop point for receiving claimants' claims and processing their claims.

4.2 Application Development

In developing the application, a 3-tier layer was adopted to modularize the different components of the system.

The Three-tier (layer) is an architecture in which the user interface, business process (business rules) and data storage and data access are developed and maintained as independent modules.


NSITFBenefit Form or Project
Class
Class
Application or Presentation or GUI Layer
Business Layer or Logical Layer
Data Access Layer or Data Layer
Class
Property Layer

- **Application/Presentation/GUI layer** is the form where the designs were done using the controls like textbox, labels, command buttons, data grid etc.
- **Business/Logical layer** is the class where the functions or procedures which get the data from the application layer and passes through the data access layer were written or coded.
- **Data/Data Access layer** is also the class which gets the data from the business layer and sends it to the database or gets the data from the database and sends it to the business layer.
- **Property layer** is the sub layer of the business layer in which the properties of data gotten from the application layer are defined before being sent to the business layer. These properties help to sustain the value in an object so that we can get these values till the object is destroyed.

The good reasons for the separation of the user interface from business logic and database access are as follows:

- Reusability of the business logic component resulted in quick development. For example, there is a module that handles adding, updating, deleting and finding members in the system. As this component is developed and tested, it can be used in any other project that might involve maintaining members' records.
- Transformation of the system was easy. Since the business logic is separate from the data access layer, changing the data access layer won't affect the business logic module much. Let's say if we are moving from MySQL Server data storage to Oracle there shouldn't be any changes required in the business layer component and in the GUI component.
- Change management of the system is easy. Let's say if there is a minor change in the business logic, we don't have to install the entire system in individual user's PCs. E.g. if Pension Rate is changed from 30% to 35% we only need to update the business logic component without affecting the users and without any downtime.

4.3 Implementation of the Benefit Payment System

The application program developed with C# will be compiled into an executable format, for ease of implementation. Some of the other requirements for implementing the systems are:

- A Computer System with the following minimum specifications:
    - Pentium 4 Processor
    - 512 MB RAM
    - 80 GB HDD
    - Windows XP Operating System
    - VGA monitor.
- MySQL Database
- MySQL ODBC Driver
- .Net Framework

4.3.1 Integrating the Application with the Database back-end

The database back-end deployed in the application is **MySQL**. Databases are an inevitable part of most of the real world applications. The deployed database is used for two basic actions: inserting/modifying data and querying (requesting information) from existing data. The back-end database is accessed by users indirectly through the application rather than by low level manipulation of the data (e.g. through SQL commands).

The back-end database stores data but does not include end-user application elements such as stored queries, forms or macros. MySQL provides connectors and drivers (ODBC, JDBC, etc.) that allow applications to make use of MySQL as a preferred data management server. Therefore, in integrating the application to the database backend, MySQL connectors and drivers were used. The driver used is MySQL ODBC 3.51 Driver. It is available at no cost like the database itself.

The following connection string was used (embedded) within the application to connect to the database:

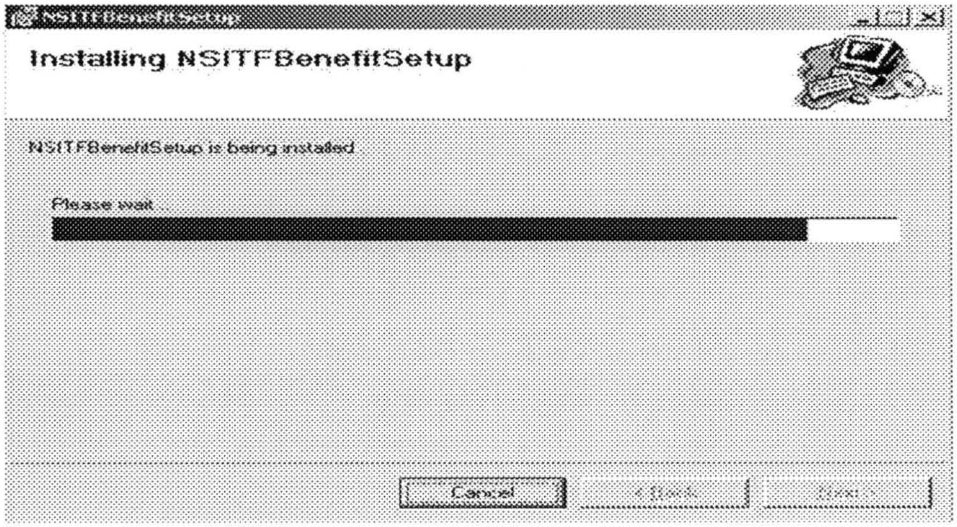
```
<appSettings> <add key="connectionStringBenefit"
        value="server=localhost;uid=root;password=password;database=nsitf;port=3
        306" />
</appSettings>
```

This connection assumes the existence of MySQL database (nsitf) on the local machine (localhost), with access parameters like user id/username - uid (root), password (password) and port (3306).

4.4 Application Deployment

Deployment is the process by which a finished application is distributed to be installed on other computers. The application will be deployed with the traditional setup method using the **Windows Installer** technology.

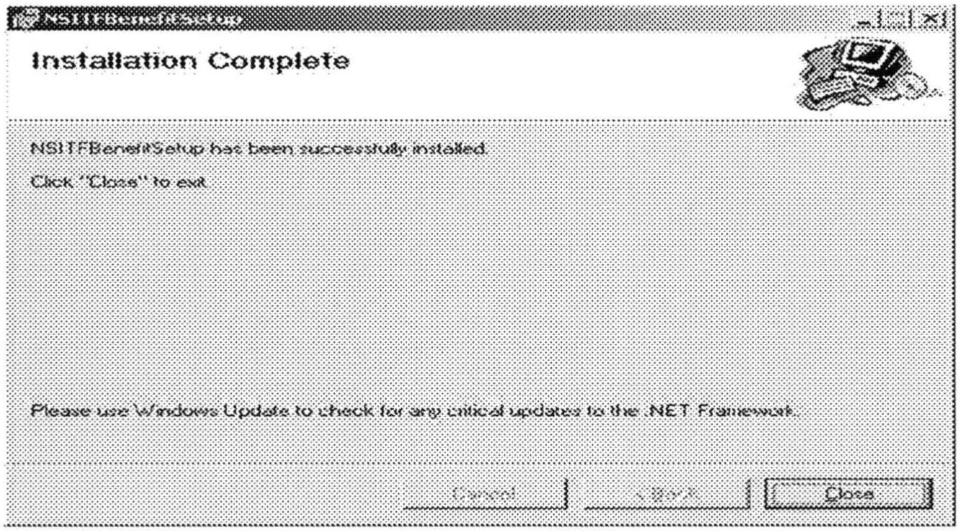
To deploy the application with Windows Installer, a Setup project was added to the NSITFBenefit solution in Visual Studio which created the setup.exe file to be distributed. With this deployment method, the application can be installed for multiple users on the same computer. In the deployment, the application is packaged into a setup.exe file that can be distributed to users, so that each user individually runs the setup.exe file to install the application. When the setup.exe is run, these are the screens displayed:



Installing NSITFBenefitSetup

NSITFBenefitSetup is being installed.

Please wait...



NSITFBenefitSetup

Installation Complete

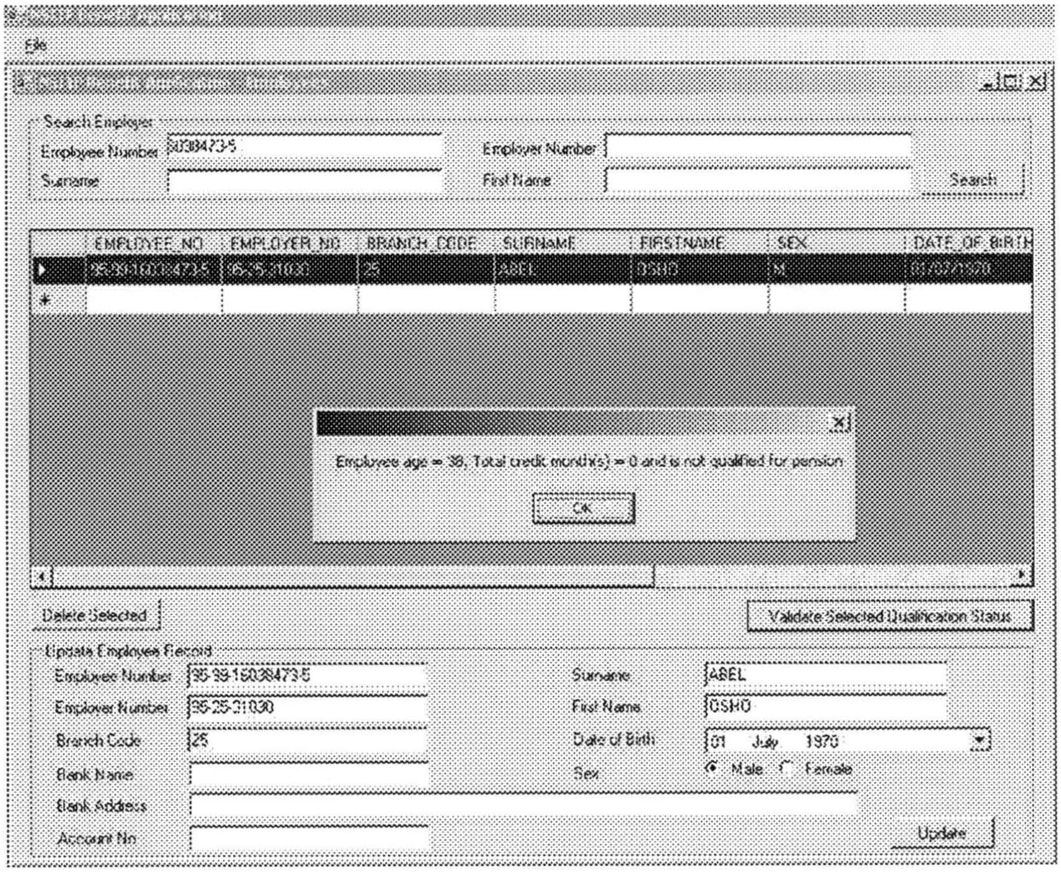
NSITFBenefitSetup has been successfully installed.

Click "Close" to exit.

Please use Windows Update to check for any critical updates to the .NET Framework.

4.5 Program Output

Several outputs can be gotten from the system. Some examples of outputs from the program are shown below:



Search Employee

Employee Number: 5038473-5

Employer Number:

Surname:

First Name:

	EMPLOYEE_NO	EMPLOYER_NO	BRANCH_CODE	SURNAME	FIRSTNAME	SEX	DATE_OF_BIRTH
▶	35-39-16038473-5	35-25-31030	25	ABEL	OSHO	M	01/07/1970
*							

Employee age = 38, Total credit month(s) = 0 and is not qualified for pension

Update Employee Record

Employee Number: 35-39-16038473-5

Employer Number: 35-25-31030

Branch Code: 25

Bank Name:

Bank Address:

Account No:

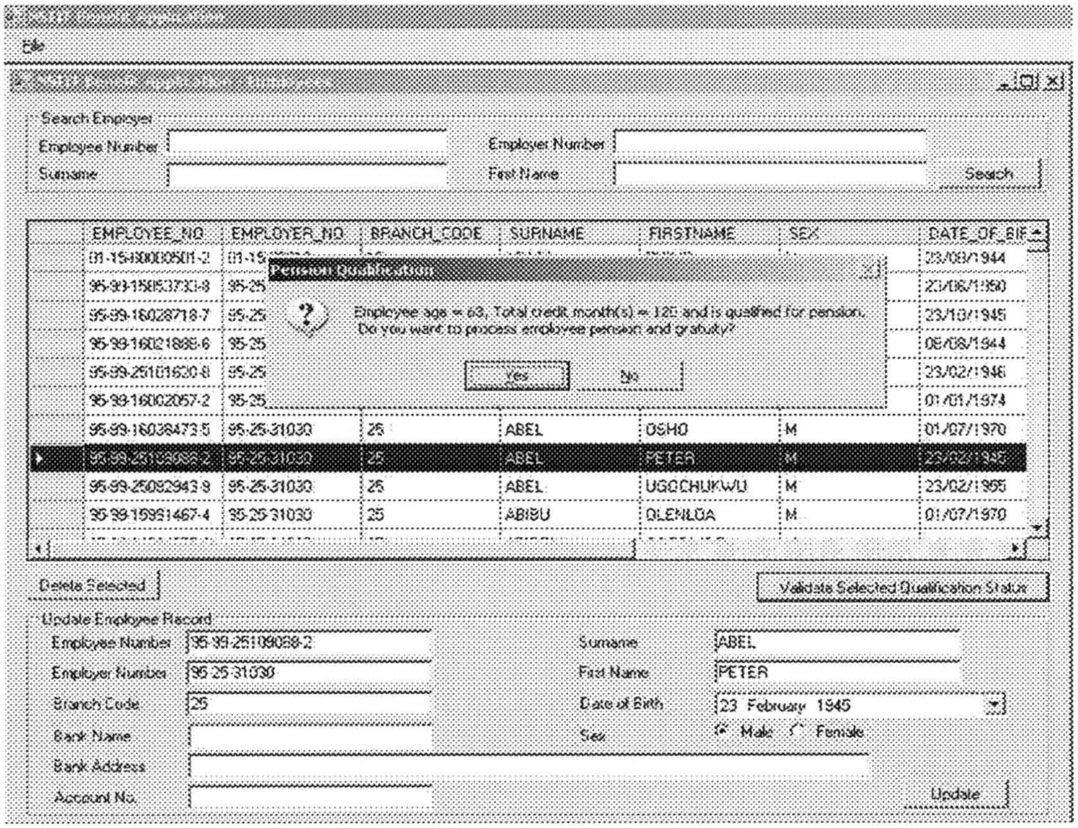
Surname: ABEL

First Name: OSHO

Date of Birth: 01 July 1970

Sex: ● Male ○ Female

- **When a Claimant's age is < 60 and has no Contributions**



Search Employer

Employee Number:

Employer Number:

Surname:

First Name:

	EMPLOYEE_NO	EMPLOYER_NO	BRANCH_CODE	SURNAME	FIRSTNAME	SEX	DATE_OF_BIF
	01-15-60000501-2	01-15-					23/08/1944
	95-99-15853733-8	95-25					23/06/1950
	95-99-16028718-7	95-25					23/10/1945
	95-99-16021888-6	95-25					08/08/1944
	95-99-25101620-8	95-25					23/02/1946
	95-99-16002057-2	95-25					01/01/1974
	95-99-16038473-5	95-25-31030	25	ABEL	OSHO	M	01/07/1970
▶	95-99-25109088-2	95-25-31030	25	ABEL	PETER	M	23/02/1945
	95-99-25092943-9	95-25-31030	25	ABEL	UGOCHUKWU	M	23/02/1955
	95-99-15991467-4	95-25-31030	25	ABIBU	OLENLOA	M	01/07/1970

Pension Qualification

Employee age = 63, Total credit month(s) = 120 and is qualified for pension. Do you want to process employee pension and gratuity?

Update Employee Record

Employee Number: 95-99-25109088-2

Employer Number: 95-25-31030

Branch Code: 25

Bank Name:

Bank Address:

Account No.:

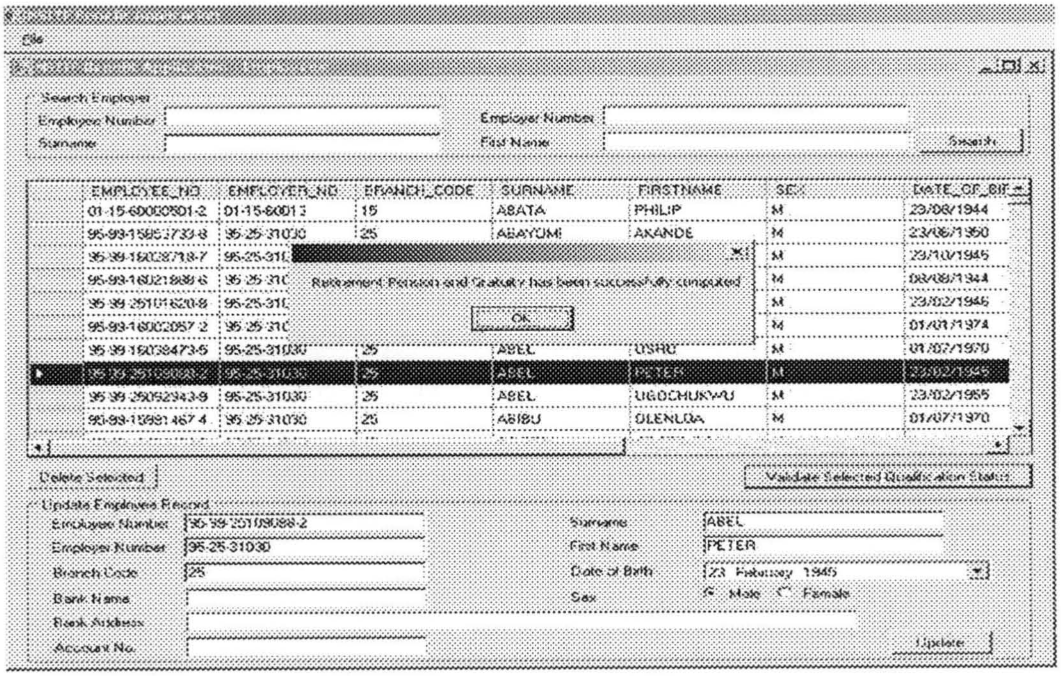
Surname: ABEL

First Name: PETER

Date of Birth: 23 February 1945

Sex: ● Male ○ Female

- **When a Claimant's age >= 60 and has >= 120 months contributions. Claimant is qualified and benefit can be computed**



Search Employer

Employee Number:

Employer Number:

Surname:

First Name:

	EMPLOYEE_NO	EMPLOYER_NO	BRANCH_CODE	SURNAME	FIRSTNAME	SEX	DATE_OF_BIF
	01-15-60000501-2	01-15-60013	15	ABATA	PHILIP	M	23/08/1944
	95-99-15853733-8	95-25-31030	25	ABAYOMI	AKANDE	M	23/06/1950
	95-99-16028718-7	95-25-310				M	23/10/1945
	95-99-16021888-6	95-25-310				M	08/08/1944
	95-99-25101620-8	95-25-310				M	23/02/1946
	95-99-16002057-2	95-25-310				M	01/01/1974
	95-99-16038473-5	95-25-31030	25	ABEL	OSHO	M	01/07/1970
▶	95-99-25109088-2	95-25-31030	25	ABEL	PETER	M	23/02/1945
	95-99-25092943-9	95-25-31030	25	ABEL	UGOCHUKWU	M	23/02/1955
	95-99-15991467-4	95-25-31030	25	ABIBU	OLENLOA	M	01/07/1970

Retirement Pension and Gratuity has been successfully computed

Update Employee Record

Employee Number: 95-99-25109088-2

Employer Number: 95-25-31030

Branch Code: 25

Bank Name:

Bank Address:

Account No.:

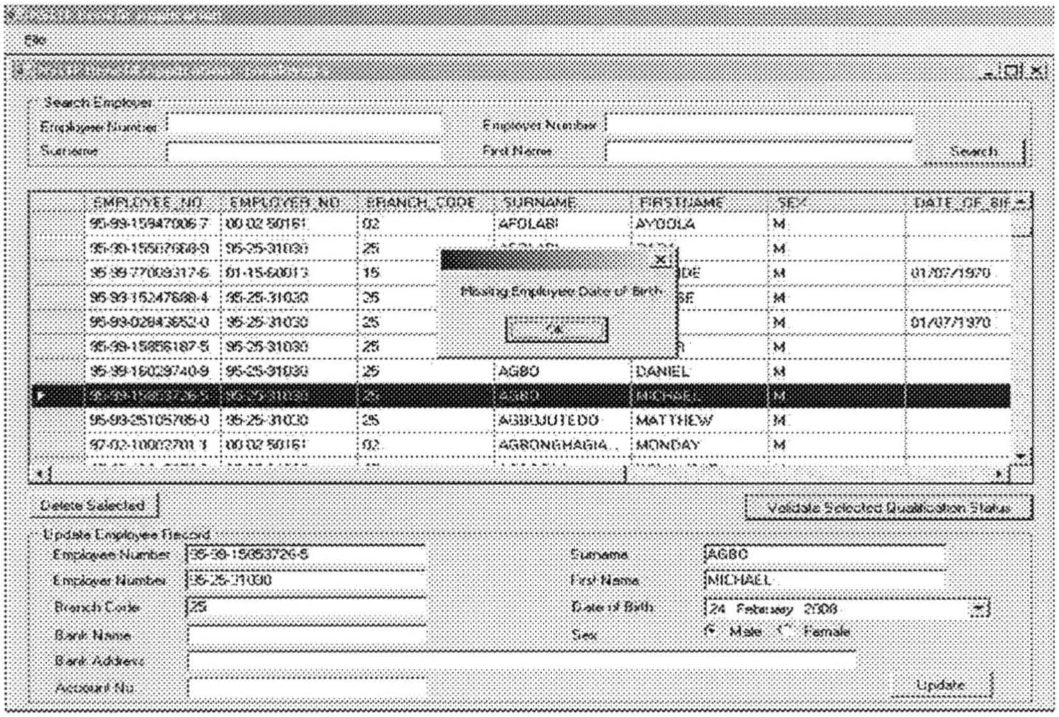
Surname: ABEL

First Name: PETER

Date of Birth: 23 February 1945

Sex: ● Male ○ Female

- **When retirement pension and gratuity is successfully computed.**



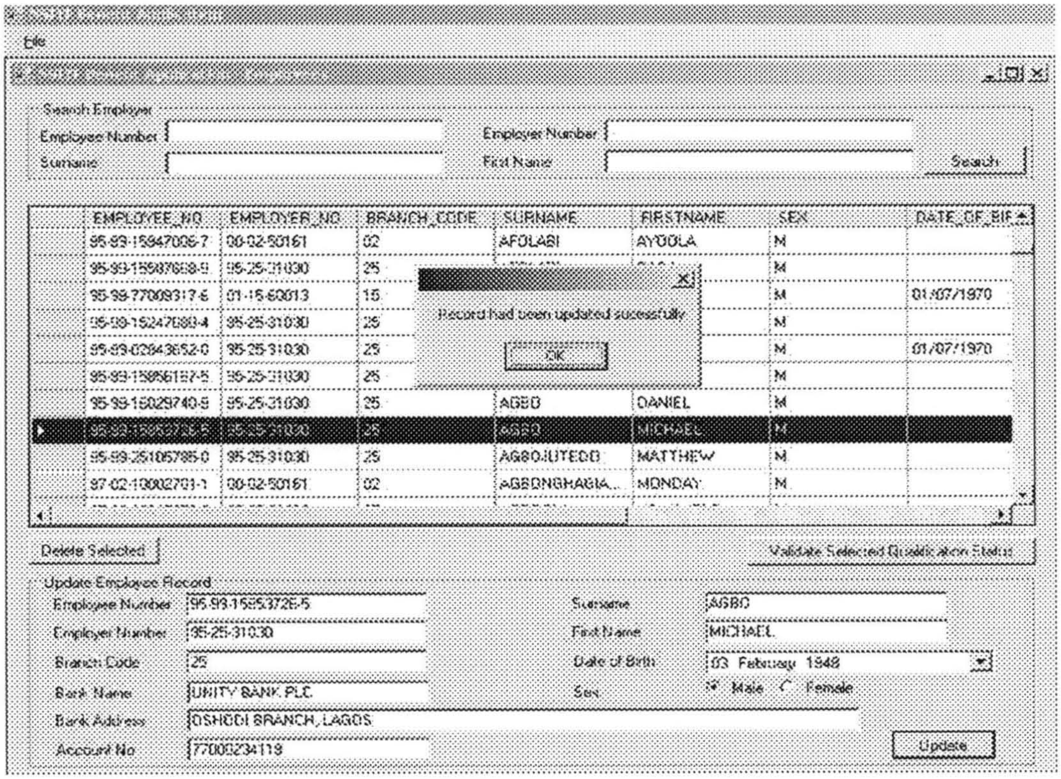
	EMPLOYEE_NO	EMPLOYER_NO	BRANCH_CODE	SURNAME	FIRSTNAME	SEX	DATE_OF_BIR
	95-99-15947006-7	00-02-50161	02	AFOLABI	AYODLA	M	
	95-99-15587668-9	95-25-31030	25			M	
	95-99-77009317-6	01-15-60013	15		DE	M	01/07/1970
	95-93-15247698-4	95-25-31030	25		SE	M	
	95-99-02843652-0	95-25-31030	25			M	01/07/1970
	95-99-15856187-5	95-25-31030	25			M	
	95-99-16029740-9	95-25-31030	25	AGBO	DANIEL	M	
▶	95-99-15853726-5	95-25-31030	25	AGBO	MICHAEL	M	
	95-93-25105785-0	95-25-31030	25	AGBOJUTEDO	MATTHEW	M	
	97-02-10002701-1	00-02-50161	02	AGBONGHAGIA...	MONDAY	M	

Missing Employee Date of Birth

Update Employee Record

Employee Number: 95-99-15853726-5
Employer Number: 95-25-31030
Branch Code: 25
Bank Name:
Bank Address:
Account No:
Surname: AGBO
First Name: MICHAEL
Date of Birth: 24 February 2008
Sex: ● Male ○ Female

- **When a claimant's date of birth record is missing. This is required to be updated.**



	EMPLOYEE_NO	EMPLOYER_NO	BRANCH_CODE	SURNAME	FIRSTNAME	SEX	DATE_OF_BIR
	95-99-15947006-7	00-02-50161	02	AFOLABI	AYODLA	M	
	95-99-15587668-9	95-25-31030	25			M	
	95-99-77009317-6	01-15-60013	15			M	01/07/1970
	95-99-15247080-4	95-25-31030	25			M	
	95-99-02843652-0	95-25-31030	25			M	01/07/1970
	95-99-15856187-5	95-25-31030	25			M	
	95-99-16029740-9	95-25-31030	25	AGBO	DANIEL	M	
▶	95-93-15853726-5	95-25-31030	25	AGBO	MICHAEL	M	
	95-99-25106785-0	95-25-31030	25	AGBOJUTEDO	MATTHEW	M	
	97-02-10002701-1	00-02-50161	02	AGBONGHAGIA...	MONDAY	M	

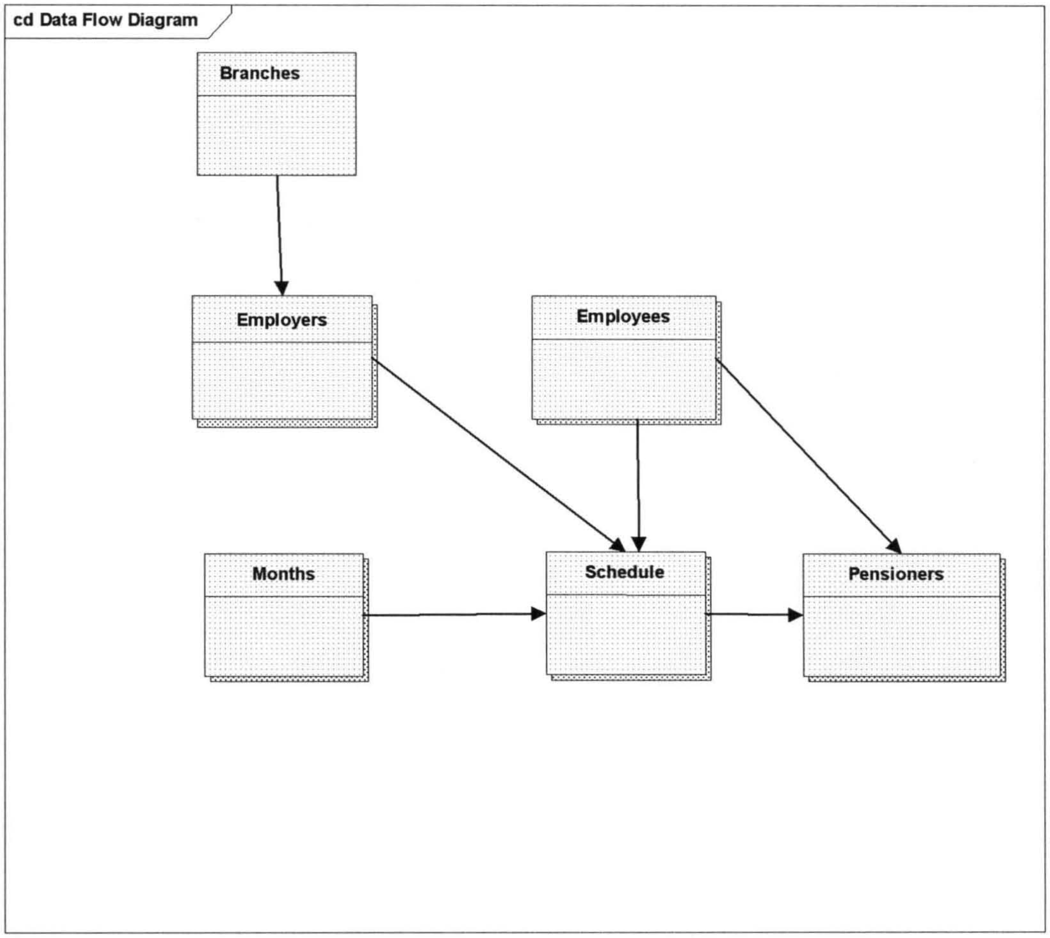
Record had been updated sucessfully

Update Employee Record

Employee Number: 95-93-15853726-5
Employer Number: 95-25-31030
Branch Code: 25
Bank Name: UNITY BANK PLC
Bank Address: OSHODI BRANCH, LAGOS
Account No: 7700234119
Surname: AGBO
First Name: MICHAEL
Date of Birth: 03 February 1948
Sex: ● Male ○ Female

- **Date of birth field successfully updated. Other missing fields can be updated at this point too.**

4.6 Data Flow Diagrams



CHAPTER FIVE

CONCLUSION

5.1 Summary

The project is based on the computerization of NSITF pension benefit payment process Trustfund Pensions Plc. The system investigation was conducted at the Benefit Unit of NSITF-Fund Division and the Abuja Branch office. Manuals on Benefit Processing, NSITF Act 1993, Pension Reform Act 2004 and other regulations were studied. Responsible benefit officers were interviewed. The investigations were to enable the researcher have better understanding of the current system to be able to properly proffer better solution on benefit processing. The investigation revealed that the present method of processing pensioners' benefits by manual calculation is very tedious, tiring, time consuming and prone to a lot of errors due to multiple human interventions in the entire process. The project work looked into the present system of processing the retired workers' benefits and focused on designing a computerized system that ensured effortless and quick retrieval of accurate information for accurate and speedy processing and payment of appropriate benefits to the retired claimants.

A cost effective, user friendly, easy to use Windows Forms application solution was developed using C# - a robust, object-oriented, programming language with rapid application development feature and MySQL - an open source (free) database backend. These tools were implemented in a 3-tier application architecture comprising a Presentation Layer, Business Logic

Layer and Data Access Layer. Records of members are efficiently managed with the Relational Database Management System (MySQL) and due to the limitation in time and the scope defined for the project, only one contingency:– Retirement Pension & Gratuity, was considered and few reports available. But overall, the system is user-friendly, easy to use and stable.

5.2 Conclusions

This implementation of a computerized Benefit Payment Processing System for the NSITF Contributory Scheme has shown that the problems presently encountered in the manual processing system can be overcome with the use of computers, deploying this application and improving on it by implementing some or all of the recommendations above.

The project work reveals that, Trustfund will benefit from its deployment in the following areas:

- Reduction in man-hours spent on benefit processing.
- Better accuracy in the calculation/computation of pensioners'/claimants benefit.
- Higher level of data integrity in Pensioners' data.
- Possibility of electronic interface with payment banks for issuing payment advice and reconciliation of accounts.

5.3 Recommendations

Since no one can claim to be an epitome of knowledge and since no system is perfect, the researcher accepts the fact that there are still some rooms for improvement and areas of further study on the subject. As indicated highlighted in the limitations of the new system, the following recommendations are hereby suggested for implementation in the future:

1. Computation of other contingencies like Survivor Pension, Survivors grant, Invalidity pensions, Invalidity grant, Emigration grant and others could be incorporated.
2. Computation of Retirement Grant for members 60 years and above with less than 120 months contributions.
3. Possible on-line interfacing of the system with the paying bank(s).
4. Pictures of the claimants could be integrated to further serve to control impersonation.
5. The system could be developed into a Web Application. This would allow its deployment and running via simple web browsers to all branches, so that, the computerized processing can be decentralized in a way. But controls should be put in place to prevent fraud. There could be different levels of authority/privileges and task segregation; where a supervising officer checks what his subordinate does before final processing is done.

REFERENCES

1. **Akeredolu-Ale, Dayo** (1996), Liberalization and the Welfare State Project in Africa, An Explanatory Comment, 'paper delivered as Visiting Professor, Center for Health and Social Policy Studies, University of Bergen, Norway, June 4, 1996. Unpublished.
2. **Akeredolu-Ale, Dayo** (2004), Laying the Foundation for a Comprehensive Social Security Scheme, Contribution in The Quest For Social Security. A book in commemoration of the 10th anniversary of Nigeria Social Insurance Trust Fund (NSITF), NSITF, Abuja.
3. **Akeredolu-Ale, Dayo** (2007), Economic Reform and Social Security in Nigeria: Lessons from More Mature Market Economies, A commissioned paper presented at the National Conference on Social Security, organized by the Nigeria Social Insurance Trust Fund (NSITF), October 15 – 17, 2007, Abuja. Unpublished.
4. **Akeredolu-Ale, E. O. and Aribiah, O.** (2001), Public Policy and Old-Age Security in Nigeria. Social Policy in Nigeria (SPIN) Monograph Series, No. 2/2001, Centre for Social Policy, Ibadan.
5. **Graeme C. S. and Graham C. W. (**2005), Data Modeling Essentials**,** Third Edition. Morgan Kaufman Publishers, San Francisco.
6. **Harvey M. Deitel et. Al.** (2005), C# How to Program, Deitel & Associates Inc.
7. **ILO** (2000), Social Security Pensions: Development and Reforms, Geneva: International Labour Organization Office (Volume edited by Gillion, Colin et. al.)

8. **International Social Security Association** (2007), Improving Operational Capacities, Social Security Documentation, African Series Number 28.

9. **Jay Greenspan and Brad Bulger** (2001), MySQL/PHP Database Applications, M & T Books.

10. **Nigeria Social Insurance Trust Fund** (2003)**,** Benefit Administration Manual, NSITF, Abuja.

11. **Obienu Chinedu** (2007), The Constitutional Imperatives for Social Security Scheme in Nigeria. A paper presented at the National Conference on Social Security, organized by the Nigeria Social Insurance Trust Fund (NSITF), October 15 – 17, 2007, Abuja. Unpublished.

12. **Steve Suehring (**2002), MySQL Bible, Wiley Publishing Inc., New York.

13. **United States of America** (2007), Social Security Programs Throughout the World, US Social Security Administration, Office of Policy.

14. **Yusufu, Mohammed D.** (2001), The Fear of Retirement Among Nigerians, Social Policy in Nigeria (SPIN) Monograph Series, No. 1/2001, Centre for Social Policy, Ibadan.

APPENDIX

Program Listing

```
using System;
using System.Collections.Generic;
using System.Windows.Forms;

namespace NSITFBenefit
{
    static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            //Application.Run(new FrmMain());
            Application.Run(new FrmLogin());
        }
    }
}


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

namespace NSITFBenefit
{
    public partial class FrmLogin : Form
    {
        private static string username = "admin";
        private static string password = "password";

        public FrmLogin()
        {
            InitializeComponent();
        }

        private void btnLogin_Click(object sender, EventArgs e)
        {
            Form frmMain = new FrmMain();

            if (txtUserName.Text == "" && txtPassword.Text == "")
            {
                MessageBox.Show("Username and password are required",
"Login Error", MessageBoxButtons.OK);
            }
            else
            {
                try
                {
                    if (txtUserName.Text == "admin" &&
txtPassword.Text == "password")
                    {
                        frmMain.Show();
                        this.Hide();
                    }
                    else
                    {
                        string exceptionMessage;
                        exceptionMessage = "Invalid username or
password.\n";

                        MessageBox.Show(exceptionMessage, "Login
Error");
                    }
                }
                catch (Exception ex)
                {
                    string exceptionMessage;
                    exceptionMessage = "Error occured while trying to
log in.\n";
                    exceptionMessage += "Actual Reason: " +
ex.Message;
                    MessageBox.Show(exceptionMessage, "Login Error");

                }
            }

        }

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

        private void FrmLogin_Load(object sender, EventArgs e)
        {
            this.Text = "NSITF Benefit Application - User Login";
        }

    }
}


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

namespace NSITFBenefit
{
    public partial class FrmMain : Form
    {
        public FrmMain()
        {
            InitializeComponent();
        }
        private void miFileExit_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }

       private void miFileEmployers_Click(object sender, EventArgs e)
        {
            FrmEmployers frmEmployers = new FrmEmployers();

            foreach (Form frm in this.MdiChildren)
            {
                if (frm.Name.ToLower() == "FrmEmployers".ToLower())
                {
                    frm.Select();

      frm.WindowState = System.Windows.Forms.FormWindowState.Normal;
                    return;
                }

            }

            frmEmployers.MdiParent = this;
            frmEmployers.Show();
        }

        private void miFileEmployees_Click(object sender, EventArgs
e)
        {
            FrmEmployees frmEmployees = new FrmEmployees();

            foreach (Form frm in this.MdiChildren)
            {
                if (frm.Name.ToLower() == "FrmEmployees".ToLower())
                {
                    frm.Select();
       frm.WindowState = System.Windows.Forms.FormWindowState.Normal;
                    return;
                }

            }

            frmEmployees.MdiParent = this;
            frmEmployees.Show();
        }

        private void miFileSchedules_Click(object sender, EventArgs
e)
        {
            FrmSchedules frmSchedules = new FrmSchedules();

            foreach (Form frm in this.MdiChildren)
            {
                if (frm.Name.ToLower() == "FrmSchedules".ToLower())
                {
                    frm.Select();
       frm.WindowState = System.Windows.Forms.FormWindowState.Normal;
                    return;
                }

            }
            frmSchedules.MdiParent = this;
            frmSchedules.Show();
        }

        private void miFilePensioners_Click(object sender, EventArgs
e)
        {
            FrmPensioners frmPensioners = new FrmPensioners();

            foreach (Form frm in this.MdiChildren)
            {
                if (frm.Name.ToLower() == "FrmPensioners".ToLower())
                {
                    frm.Select();
       frm.WindowState = System.Windows.Forms.FormWindowState.Normal;
                    return;
                }

            }

            frmPensioners.MdiParent = this;
            frmPensioners.Show();
        }
    }
}


using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using Benefit.BLL;
using Benefit.VOL;

namespace NSITFBenefit
{
    public partial class FrmEmployees : Form
    {
        public FrmEmployees()
        {
            InitializeComponent();
        }

        private void btnSearch_Click(object sender, EventArgs e)
        {
            string employeeNo = txtEmployeeNo.Text;
            string employerNo = txtEmployerNo.Text;
            string surname = txtSurname.Text;
            string firstName = txtFirstName.Text;

            ResetForm();
            btnUpdateRecord.Text = "Insert";

            DataSet dsGetItems;

            try
            {
               dsGetItems = new BenefitBL().GetEmployees(employerNo,
employeeNo, surname, firstName);
                if (dsGetItems != null)
                {
                    dgvEmployees.DataSource = dsGetItems.Tables[0];
                }
                else
                {
                    dgvEmployees.DataSource = null;
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message, "Error");
            }
        }

       private void btnUpdateRecord_Click(object sender, EventArgs e)
        {
            if (txtUSurname.Text != "" && txtUEmployeeNo.Text != ""
&& btnUpdateRecord.Text.ToLower() == "update")
            {
                EmployeeInfo employeeInfo = new EmployeeInfo();
                BenefitBL benefitBL = new BenefitBL();

                employeeInfo.EmployeeNo = txtUEmployeeNo.Text;
                employeeInfo.EmployerNo = txtUEmployerNo.Text;
                employeeInfo.BranchCode = txtUBranchCode.Text;

                employeeInfo.Surname = txtUSurname.Text;
                employeeInfo.FirstName = txtUFirstName.Text;
                employeeInfo.DateOfBirth =
(dtpUDateOfBirth.Value.Date == DateTime.Now.Date) ? "" :
DateTime.Parse(dtpUDateOfBirth.Text).ToString("yyyy-MM-dd");
                if (rdMale.Checked)
                {
                    employeeInfo.Sex = "M";
                }
                else if (rdFemale.Checked)
                {
                    employeeInfo.Sex = "F";
                }

                employeeInfo.BankName = txtUBankName.Text;
                employeeInfo.BankAddress = txtUBankAddress.Text;
                employeeInfo.AccountNo = txtUAccountNo.Text;

                try
                {
                    if (benefitBL.UpdateEmployee(employeeInfo))
                    {
            MessageBox.Show("Record had been updated sucessfully");
                        ResetForm();
                        btnUpdateRecord.Text = "Insert";
                    }
                    else
                    {
                     MessageBox.Show("Unknown error during update");
                    }
                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message, "Error");
                }
            }
            else if (txtUSurname.Text != "" && txtUEmployeeNo.Text !=
"" && btnUpdateRecord.Text.ToLower() == "insert")
            {
                EmployeeInfo employeeInfo = new EmployeeInfo();
                BenefitBL benefitBL = new BenefitBL();

                employeeInfo.EmployeeNo = txtUEmployeeNo.Text;
                employeeInfo.EmployerNo = txtUEmployerNo.Text;
                employeeInfo.BranchCode = txtUBranchCode.Text;

                employeeInfo.Surname = txtUSurname.Text;
                employeeInfo.FirstName = txtUFirstName.Text;
                employeeInfo.DateOfBirth =
(DateTime.Parse(dtpUDateOfBirth.Text).Date == DateTime.Now.Date) ? ""
: DateTime.Parse(dtpUDateOfBirth.Text).ToString("yyyy-MM-dd");
                if (rdMale.Checked)
                {
                    employeeInfo.Sex = "M";
                }
                else if (rdFemale.Checked)
                {
                    employeeInfo.Sex = "F";
                }

                employeeInfo.BankName = txtUBankName.Text;
                employeeInfo.BankAddress = txtUBankAddress.Text;
                employeeInfo.AccountNo = txtUAccountNo.Text;

                try
                {
                    if (benefitBL.InsertEmployee(employeeInfo))
                    {
            MessageBox.Show("Record had been inserted sucessfully");
                        ResetForm();
                        btnUpdateRecord.Text = "Insert";
                    }
                    else
                    {
            MessageBox.Show("Unknown error during insert");
                    }
                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message, "Error");
                }
            }
        }

        private void ResetForm()
        {
            txtUEmployeeNo.Text = "";
            txtUEmployerNo.Text = "";
            txtUBranchCode.Text = "";

            txtUSurname.Text = "";
            txtUFirstName.Text = "";
            dtpUDateOfBirth.Text = "";
```

```
      rdFemale.Checked = false;
      rdMale.Checked = false;

      txtUBankName.Text = "";
      txtUBankAddress.Text = "";
      txtUAccountNo.Text = "";
  }

  private void dgvEmployees_Click(object sender, EventArgs e)
  {
      BenefitBL benefitBL = new BenefitBL();
      EmployeeInfo employeeInfo;
      int currentRow = 0;
      string employeeNo = "";

      try
      {
          if (dgvEmployees != null)
          {
      currentRow = dgvEmployees.SelectedCells[0].RowIndex;
              employeeNo =
dgvEmployees.Rows[currentRow].Cells[0].Value.ToString();
          }
      }
      catch (Exception ex)
      {
          ex.ToString();
      }

      try
      {
          if (employeeNo != null && employeeNo != "")
          {
              employeeInfo = benefitBL.GetEmployee(employeeNo);
              if (employeeInfo != null &&
employeeInfo.EmployeeNo != "")
              {
                  txtUEmployeeNo.Text =
employeeInfo.EmployeeNo;
                  txtUEmployerNo.Text =
employeeInfo.EmployerNo;
                  txtUBranchCode.Text =
employeeInfo.BranchCode;

                  txtUSurname.Text = employeeInfo.Surname;
                  txtUFirstName.Text = employeeInfo.FirstName;
                  dtpUDateOfBirth.Value =
(employeeInfo.DateOfBirth == "") ? DateTime.Now :
DateTime.Parse(employeeInfo.DateOfBirth);
                  if (employeeInfo.Sex == "M")
                  {
                      rdMale.Checked = true;
                  }
                  else if (employeeInfo.Sex == "F")
                  {
                      rdFemale.Checked = true;
                  }
                  txtUBankName.Text = employeeInfo.BankName;
                  txtUBankAddress.Text =
employeeInfo.BankAddress;
                  txtUAccountNo.Text = employeeInfo.AccountNo;
```

```
                        btnUpdateRecord.Text = "Update";
                    }
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message, "Error");
            }
        }

        private void btnDelete_Click(object sender, EventArgs e)
        {
            BenefitBL benefitBL = new BenefitBL();
            int currentRow = 0;
            string employeeNo = "";

            try
            {
                if (dgvEmployees != null)
                {
currentRow = dgvEmployees.SelectedCells[0].RowIndex;
                    employeeNo =
dgvEmployees.Rows[currentRow].Cells[0].Value.ToString();
                }
            }
            catch (Exception ex)
            {
                ex.ToString();
            }

            try
            {
                if (employeeNo != null && employeeNo != "")
                {
                    if (benefitBL.DeleteEmployee(employeeNo))
                    {
                        MessageBox.Show("Record deleted
sucessfully");
                        btnSearch_Click(sender, e);
                    }
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message, "Error");
            }
        }

       private void btnCheckQualify_Click(object sender, EventArgs e)
        {
//MessageBox.Show("This action will determine if selected member is
qualified to receive pension or not.", "Check Qualification Status");
            BenefitBL benefitBL = new BenefitBL();
            EmployeeInfo employeeInfo = new EmployeeInfo();

            int currentRow = 0;
            string employeeNo = "";
            int age = 0;
            int creditMonths = 0;
            //int creditMonths2 = 0.0;
            int maxCreditMonth = 120;

            try
            {
                if (dgvEmployees != null)
                {
                    currentRow =
dgvEmployees.SelectedCells[0].RowIndex;
                    employeeNo =
dgvEmployees.Rows[currentRow].Cells[0].Value.ToString();

                    if (employeeNo != "")
                    {
                        employeeInfo =
benefitBL.GetEmployee(employeeNo);

                        if (employeeInfo.DateOfBirth == "")
                        {
                  MessageBox.Show("Missing Employee Date of Birth");
                        }
                        else
                        {
age = DateTime.Now.Year -
DateTime.Parse(employeeInfo.DateOfBirth).Year;
creditMonths = benefitBL.GetTotalCreditMonths(employeeNo);
                             if (age < 60 )
                            {
MessageBox.Show("Employee age = " + age + ", Total credit month(s) =
" + creditMonths + " and is not qualified for pension");
                            }
                            if (creditMonths == 0 )
                            {
MessageBox.Show("Employee age = " + age + ", Total credit month(s) =
" + creditMonths + " and is not qualified for pension");
                            }


                            if (age >= 60 && creditMonths <
maxCreditMonth)                            {
                                if (MessageBox.Show("Employee age = "
+ age + ", Total credit month(s) = " + creditMonths + " and is
qualified for Retirement Grant. \n Do you want to process employee
Retirement grant?", "Grant Qualification", MessageBoxButtons.YesNo,
MessageBoxIcon.Question) == DialogResult.Yes)
                                {
                                    if
(ComputeRetirementPension3(employeeNo))
                                    {
  MessageBox.Show("Retirement Grant has been successfully computed");
                                    }
                                }
                            }

              if (age >= 60 && creditMonths >= maxCreditMonth)
                            {
              if (MessageBox.Show("Employee age = " + age + ", Total
credit month(s) = " + creditMonths + " and is qualified for pension.
\n Do you want to process employee pension and gratuity?", "Pension
Qualification", MessageBoxButtons.YesNo, MessageBoxIcon.Question) ==
DialogResult.Yes)
                                {
                                    if
(ComputeRetirementPension(employeeNo))
                                    {
MessageBox.Show("Retirement Pension and Gratuity has been
successfully computed");
                                    }
                                }
                            }


                        }
                    }
                }
            }
            catch (Exception ex)
            {
                ex.ToString();
                MessageBox.Show(ex.Message, "Error");
            }
        }

        private bool ComputeRetirementPension(string employeeNo)
        {
            BenefitBL benefitBL = new BenefitBL();
            int totalCreditMonths = 0;
            double pensionRate = 30.00 / 100.00;
            double bestFinalAverageAnnual = 0.0;
            double totalPensionDue = 0.0;

            double minPension = 4400.00;
            double maxPension1 = 0.0;
            double maxPension2 = 0.0;
            double maxMaxPension = 0.0;

            double lastContribution =
benefitBL.GetLastContribution(employeeNo);
            double gratuity = 0.0;
            double initalPayment = 0.0;
            double monthlyPayment = 0.0;

            totalCreditMonths =
benefitBL.GetTotalCreditMonths(employeeNo);
            bestFinalAverageAnnual = BestFinalAverage(employeeNo);
            totalPensionDue = pensionRate * bestFinalAverageAnnual;
            monthlyPayment = totalPensionDue/12;

            maxPension1 = (0.65 * (BestFinalAverage(employeeNo)))/12;
            maxPension2 = 4 * 5500.00;

            maxMaxPension = (maxPension1 > maxPension2) ? maxPension1
: maxPension2;

            // Pension lower than min wage
            if (monthlyPayment < minPension)
            {
                monthlyPayment = minPension;
            }

            // Pension Higher than max allowed pension
            if (monthlyPayment > maxMaxPension)
            {
                monthlyPayment = maxMaxPension;
            }

            gratuity = 24 * monthlyPayment;
            initalPayment = gratuity + monthlyPayment;

            return benefitBL.UpdatePensionerInfo(employeeNo,
gratuity, monthlyPayment);
        }
        private bool ComputeRetirementPension2(string employeeNo)
        {
            BenefitBL benefitBL = new BenefitBL();
            int totalCreditMonths = 0;
            double pensionRate = 31.5 / 100.00;
            double bestFinalAverageAnnual = 0.0;
            double totalPensionDue = 0.0;

            double minPension = 4400.00;
            double maxPension1 = 0.0;
            double maxPension2 = 0.0;
            double maxMaxPension = 0.0;

            double lastContribution =
benefitBL.GetLastContribution(employeeNo);
            double gratuity = 0.0;
            double initalPayment = 0.0;
            double monthlyPayment = 0.0;

            totalCreditMonths =
benefitBL.GetTotalCreditMonths(employeeNo);
            bestFinalAverageAnnual = BestFinalAverage(employeeNo);
            totalPensionDue = pensionRate * bestFinalAverageAnnual;
            monthlyPayment = totalPensionDue / 12;

           maxPension1 = (0.65 * (BestFinalAverage(employeeNo))) /
12;
           maxPension2 = 4 * 5500.00;

           maxMaxPension = (maxPension1 > maxPension2) ? maxPension1
: maxPension2;

          // Pension lower than min wage
            if (monthlyPayment < minPension)
            {
                monthlyPayment = minPension;
            }

            // Pension Higher than max allowed pension
            if (monthlyPayment > maxMaxPension)
            {
                monthlyPayment = maxMaxPension;
            }

            gratuity = 24 * monthlyPayment;
            initalPayment = gratuity + monthlyPayment;

            return benefitBL.UpdatePensionerInfo(employeeNo,
gratuity, monthlyPayment);
        }

        private bool ComputeRetirementPension3(string employeeNo)
        {
            BenefitBL benefitBL = new BenefitBL();
            int totalCreditMonths = 0;
            double pensionRate = 48000.00;
            double totalPensionDue = 0.0;
            double lastContribution =
benefitBL.GetAllContribution(employeeNo);
            double gratuity = 0.0;
            double initalPayment = 0.0;
            double monthlyPayment = 0.0;
            double AllContributions = 0.0;


            totalCreditMonths =
benefitBL.GetTotalCreditMonths(employeeNo);
            AllContributions = (0.25 * lastContribution);
            totalPensionDue = ((pensionRate * totalCreditMonths)/12 +
AllContributions)/12;
                //+ AllContributions)/totalCreditMonths;
            monthlyPayment = 0.0; ;
            gratuity = totalPensionDue;
            initalPayment = gratuity + monthlyPayment;

            return benefitBL.UpdatePensionerInfo(employeeNo,
gratuity, monthlyPayment);
        }

        private double BestFinalAverage(string employeeNo)
        {
            BenefitBL benefitBL = new BenefitBL();
            DataSet dsBestContr;
            double total36Months = 0.0;
            double ave36Months = 0.0;
            double aveAnualMonths = 0.0;

            try
            {
                dsBestContr = benefitBL.GetBestMonths(employeeNo);

                if (dsBestContr != null &&
dsBestContr.Tables[0].Rows.Count > 0)
                {
                    if (dsBestContr.Tables[0].Rows.Count >= 36)
                    {
                        for (int i = 0; i < 36; i++)
                        {
                            total36Months = total36Months +
Convert.ToDouble(dsBestContr.Tables[0].Rows[i]["TOTALCON"].ToString()
);
                        }

                        ave36Months = total36Months / 36;
                        aveAnualMonths = ave36Months * 12;
                    }
                    else
                    {
                        for (int i = 0; i <
dsBestContr.Tables[0].Rows.Count; i++)
                        {
                            total36Months = total36Months +
Convert.ToDouble(dsBestContr.Tables[0].Rows[i]["TOTALCON"].ToString()
);
                        }

                        ave36Months = total36Months /
dsBestContr.Tables[0].Rows.Count;
                        aveAnualMonths = ave36Months * 12;
                    }
                }
            }
            catch (Exception ex)
            {
                ex.ToString();
                MessageBox.Show(ex.Message, "Error");
            }

            return aveAnualMonths;
        }
    }
}


using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using Benefit.BLL;
using Benefit.VOL;

namespace NSITFBenefit
{
    public partial class FrmEmployers : Form
    {
        public FrmEmployers()
        {
            InitializeComponent();
        }

        private void btnSearch_Click(object sender, EventArgs e)
        {
            string employerCode = txtEmployerNo.Text;
            string employerName = txtEmployerName.Text;

            ResetForm();
            btnUpdateRecord.Text = "Insert";

            DataSet dsGetItems;

            try
            {

                dsGetItems = new
BenefitBL().GetEmployers(employerCode, employerName);
                if (dsGetItems != null)
                {
                    dgvEmployers.DataSource = dsGetItems.Tables[0];
                }
                else
                {
                    dgvEmployers.DataSource = null;
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message, "Error");
            }
        }


       private void btnUpdateRecord_Click(object sender, EventArgs e)
        {
            if (txtUEmployerNumber.Text != "" &&
txtUEmployerName.Text != "" && btnUpdateRecord.Text.ToLower() ==
"update")
            {
                EmployerInfo employerInfo = new EmployerInfo();
                BenefitBL benefitBL = new BenefitBL();

                employerInfo.EmployerNo = txtUEmployerNumber.Text;
                employerInfo.Name = txtUEmployerName.Text;
                employerInfo.Address = txtUAddress.Text;
                employerInfo.PostalAddress = txtUPostalAddress.Text;
                employerInfo.BranchCode = txtUBranchCode.Text;

                try
                {
                    if (benefitBL.UpdateEmployer(employerInfo))
                    {
            MessageBox.Show("Record had been updated sucessfully");
                        ResetForm();
                        btnUpdateRecord.Text = "Insert";
                    }
                    else
                    {
            MessageBox.Show("Unknown error during update");
                    }
                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message, "Error");
                }
            }
            else if (txtUEmployerNumber.Text != "" &&
txtUEmployerName.Text != "" && btnUpdateRecord.Text.ToLower() ==
"insert")
            {
                EmployerInfo employerInfo = new EmployerInfo();
                BenefitBL benefitBL = new BenefitBL();

                employerInfo.EmployerNo = txtUEmployerNumber.Text;
                employerInfo.Name = txtUEmployerName.Text;
                employerInfo.Address = txtUAddress.Text;
                employerInfo.PostalAddress = txtUPostalAddress.Text;
                employerInfo.BranchCode = txtUBranchCode.Text;

                try
                {
                    if (benefitBL.InsertEmployer(employerInfo))
                    {
```

```
            MessageBox.Show("Record had been inserted sucessfully");
                        ResetForm();
                        btnUpdateRecord.Text = "Insert";
                    }
                    else
                    {
            MessageBox.Show("Unknown error during insert");
                    }
                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message, "Error");
                }
            }
        }

        private void ResetForm()
        {
            txtUEmployerName.Text = "";
            txtUEmployerNumber.Text = "";
            txtUAddress.Text = "";
            txtUPostalAddress.Text = "";
            txtUBranchCode.Text = "";
        }

        private void dgvEmployers_Click(object sender, EventArgs e)
        {
            BenefitBL benefitBL = new BenefitBL();
            EmployerInfo employerInfo;
            int currentRow = 0;
            string employerNo = "";

            try
            {
                if (dgvEmployers != null)
                {
          currentRow = dgvEmployers.SelectedCells[0].RowIndex;
                    employerNo =
dgvEmployers.Rows[currentRow].Cells[0].Value.ToString();
                }
            }
            catch (Exception ex)
            {
                ex.ToString();
            }

            try
            {
                if (employerNo != null && employerNo != "")
                {
                    employerInfo = benefitBL.GetEmployer(employerNo);
          if (employerInfo != null && employerInfo.EmployerNo != "")
                    {
          txtUEmployerNumber.Text = employerInfo.EmployerNo;
            txtUEmployerName.Text = employerInfo.Name;
                 txtUAddress.Text = employerInfo.Address;
           txtUPostalAddress.Text = employerInfo.PostalAddress;
              txtUBranchCode.Text = employerInfo.BranchCode;
             btnUpdateRecord.Text = "Update";
                    }
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message, "Error");
            }
        }

        private void btnDelete_Click(object sender, EventArgs e)
        {
            BenefitBL benefitBL = new BenefitBL();
            int currentRow = 0;
            string employerNo = "";

            try
            {
                if (dgvEmployers != null)
                {
               currentRow = dgvEmployers.SelectedCells[0].RowIndex;
employerNo = dgvEmployers.Rows[currentRow].Cells[0].Value.ToString();
                }
            }
            catch (Exception ex)
            {
                ex.ToString();
            }

            try
            {
                if (employerNo != null && employerNo != "")
                {
                    if (benefitBL.DeleteEmployer(employerNo))
                    {
                    MessageBox.Show("Record deleted sucessfully");
                        btnSearch_Click(sender, e);
                    }
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message, "Error");
            }
        }
    }
}

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using Benefit.BLL;
using Benefit.VOL;

namespace NSITFBenefit
{
    public partial class FrmSchedules : Form
    {
        public FrmSchedules()
        {
            InitializeComponent();
        }

        private void btnSearch_Click(object sender, EventArgs e)
        {
            string employeeNo = txtEmployeeNo.Text;

            ResetForm();
            btnUpdateRecord.Text = "Insert";

            DataSet dsGetItems;

            try
            {

               dsGetItems = new BenefitBL().GetSchedules(employeeNo);
                if (dsGetItems != null)
                {
                    dgvSchedules.DataSource = dsGetItems.Tables[0];
                }
                else
                {
                    dgvSchedules.DataSource = null;
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message, "Error");
            }
        }

        private void btnUpdateRecord_Click(object sender, EventArgs
e)
        {
            if (txtUStaffContr.Text != "" && txtUEmployeeNo.Text !=
"" && btnUpdateRecord.Text.ToLower() == "update")
            {
                ScheduleInfo scheduleInfo = new ScheduleInfo();
                BenefitBL benefitBL = new BenefitBL();

                scheduleInfo.EmployeeNo = txtUEmployeeNo.Text;
                scheduleInfo.EmployerNo = txtUEmployerNo.Text;
                  scheduleInfo.Year = (txtUYear.Text == "") ? 0 :
                  Convert.ToInt32(txtUYear.Text);

                scheduleInfo.StaffContr = (txtUStaffContr.Text == "")
? 0.0: Convert.ToDouble(txtUStaffContr.Text);
                scheduleInfo.CompanyContr = (txtUCompanyContr.Text ==
"")? 0.0: Convert.ToDouble(txtUCompanyContr.Text);
                scheduleInfo.TotalContr = scheduleInfo.StaffContr +
scheduleInfo.CompanyContr;
                scheduleInfo.MonthCode = (txtUMonthCode.Text == "") ?
0 : Convert.ToInt32(txtUMonthCode.Text);

                try
                {
                    if (benefitBL.UpdateSchedule(scheduleInfo))
                    {
             MessageBox.Show("Record had been updated sucessfully");
                        ResetForm();
                        btnUpdateRecord.Text = "Insert";
                    }
                    else
                    {
                     MessageBox.Show("Unknown error during update");
                    }
                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message, "Error");
                }
            }
            else if (txtUStaffContr.Text != "" && txtUEmployeeNo.Text
!= "" && btnUpdateRecord.Text.ToLower() == "insert")
            {
                ScheduleInfo scheduleInfo = new ScheduleInfo();
                BenefitBL benefitBL = new BenefitBL();

                scheduleInfo.EmployeeNo = txtUEmployeeNo.Text;
                scheduleInfo.EmployerNo = txtUEmployerNo.Text;
                scheduleInfo.Year = (txtUYear.Text == "") ? 0 :
Convert.ToInt32(txtUYear.Text);

                scheduleInfo.StaffContr = (txtUStaffContr.Text == "")
? 0.0 : Convert.ToDouble(txtUStaffContr.Text);
                scheduleInfo.CompanyContr = (txtUCompanyContr.Text ==
"") ? 0.0 : Convert.ToDouble(txtUCompanyContr.Text);
                scheduleInfo.TotalContr = scheduleInfo.StaffContr +
scheduleInfo.CompanyContr;
                scheduleInfo.MonthCode = (txtUMonthCode.Text == "") ?
0 : Convert.ToInt32(txtUMonthCode.Text);
                try
                {
                    if (benefitBL.InsertSchedule(scheduleInfo))
                    {
            MessageBox.Show("Record had been inserted sucessfully");
                        ResetForm();
                        btnUpdateRecord.Text = "Insert";
                    }
                    else
                    {
            MessageBox.Show("Unknown error during insert");
                    }
                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message, "Error");
                }
            }
        }

        private void ResetForm()
        {
            txtUEmployeeNo.Text = "";
            txtUEmployerNo.Text = "";
            txtUYear.Text = "";

            txtUStaffContr.Text = "";
            txtUCompanyContr.Text = "";
            txtUTotalContr.Text = "";
            txtUMonthCode.Text = "";
        }
        private void dgvSchedules_Click(object sender, EventArgs e)
        {
            BenefitBL benefitBL = new BenefitBL();
            ScheduleInfo scheduleInfo;
            int currentRow = 0;
            string employeeNo = "";
            string employerNo = "";
            int year = 0;
            int monthCode = 0;

            try
            {
                if (dgvSchedules != null)
                {
      currentRow = dgvSchedules.SelectedCells[0].RowIndex;
employerNo = dgvSchedules.Rows[currentRow].Cells[0].Value.ToString();
employeeNo = dgvSchedules.Rows[currentRow].Cells[1].Value.ToString();
year =
Convert.ToInt32(dgvSchedules.Rows[currentRow].Cells[5].Value.ToString
());
monthCode =
Convert.ToInt32(dgvSchedules.Rows[currentRow].Cells[6].Value.ToString
());
                }
            }
            catch (Exception ex)
            {
                ex.ToString();
            }

            try
            {
                if (employeeNo != null && employeeNo != "")
                {
                    scheduleInfo = benefitBL.GetSchedule(employeeNo,
employerNo, year, monthCode);
                    if (scheduleInfo != null &&
scheduleInfo.EmployeeNo != "")
                    {
                        txtUEmployeeNo.Text =
scheduleInfo.EmployeeNo;
                        txtUEmployerNo.Text =
scheduleInfo.EmployerNo;
                        txtUYear.Text = scheduleInfo.Year.ToString();

                        txtUStaffContr.Text =
scheduleInfo.StaffContr.ToString();
                        txtUCompanyContr.Text =
scheduleInfo.CompanyContr.ToString();
                        txtUTotalContr.Text =
scheduleInfo.TotalContr.ToString();
                        txtUMonthCode.Text =
scheduleInfo.MonthCode.ToString();

                        btnUpdateRecord.Text = "Update";
                    }
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message, "Error");
            }
        }

        private void btnDelete_Click(object sender, EventArgs e)
        {
            BenefitBL benefitBL = new BenefitBL();
            int currentRow = 0;
            string employeeNo = "";
            string employerNo = "";
            int year = 0;
            int monthCode = 0;

            try
            {
                if (dgvSchedules != null)
                {
           currentRow = dgvSchedules.SelectedCells[0].RowIndex;

employerNo = dgvSchedules.Rows[currentRow].Cells[0].Value.ToString();
employeeNo = dgvSchedules.Rows[currentRow].Cells[1].Value.ToString();
                    year =
Convert.ToInt32(dgvSchedules.Rows[currentRow].Cells[2].Value.ToString
());
                    monthCode =
Convert.ToInt32(dgvSchedules.Rows[currentRow].Cells[3].Value.ToString
());
                }
            }
            catch (Exception ex)
            {
                ex.ToString();
            }

            try
            {
                if (employeeNo != null && employeeNo != "")
                {
                    if (benefitBL.DeleteSchedule(employeeNo,
employerNo, year, monthCode))
                    {
                      MessageBox.Show("Record deleted sucessfully");
                        btnSearch_Click(sender, e);
                    }
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message, "Error");
            }
        }
    }
}
```

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using Benefit.BLL;
using Benefit.VOL;

namespace NSITFBenefit
{
    public partial class FrmPensioners : Form
    {
        public FrmPensioners()
        {
            InitializeComponent();
        }

        private void btnSearch_Click(object sender, EventArgs e)
        {
            string employeeNo = txtEmployeeNo.Text;
            string bank = txtBank.Text;
            string surname = txtSurname.Text;
            string firstName = txtFirstName.Text;

            DataSet dsGetItems;

            try
            {

                dsGetItems = new BenefitBL().GetPensioners(bank,
employeeNo, surname, firstName);

                if (dsGetItems != null)
                {
                    dgvPensioners.DataSource = dsGetItems.Tables[0];
                }
                else
                {
                    dgvPensioners.DataSource = null;
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message, "Error");
            }
        }
    }
}
```