SUDAN UNIVERSITY OF SCIENCE & TECHNOLOGY7

FACULTY OF COMPUTER SCIENCE & INFORMATION TECHNOLOGY

QUALITY ASSURANCE OF INTEGRATED SYSTEM MAINTENANCE

OCTOBER 2015

THESIS SUMITTED AS A PARTIAL REQUIREMENTS 0F B.Sc. (HONOR) DEGREE IN SOFTWARE ENGINEERING

بسم الله الرحمن الرحيم

SUDAN UNIVERSITY OF SCIENCE & TECHNOLOGY

FACULTY OF COMPUTER SCIENCE & INFORMATION TECHNOLOGY

QUALITY ASSURANCE OF INTEGRATED SYSTEM MAINTENANCE

OCTOBER 2015

THESIS SUMITTED AS A PARTIAL REQUIREMENTS 0F B.Sc. (HONOR) DEGREE IN SOFTWARE ENGINEERING

PREPARED BY:

MWADA AZHARY ABDALRAHEEM
TAYSEER BABIKER HMED

ESRA MOHAMED OMER

SIGNATURE OF SUPERVISOR: **DATE**

AHMED ABDALAZIZ

OCTOBER2015

………………………..

الآية

قال تعالى ﴿ وَفَوْقَ كُلِّ ذِي عِلْمٍ عَلِيمٌ ﴾

صدق الله العظيم

سورة يوسف الاية رقم[76]

الحمد

اللهم لك الحمد أَكمله ، ولك الثناء أجمله ، ولك القول أبلغه ، ولك العلم أحكمه ، ولك السلطان أقومه ، ولك الجلال أعظمه ، الحمدلله كثيرا مباركا فيه ، الحمدلله الذي لا يرجى الا فضله ، ولا رازق غيره .

اللهم لك الحمد حتى ترضى ، ولك الحمد اذا رضيت ، ولك الحمد بعد الرضا ،

اللهم لك الحمد كما ينبغي لجلال وجهك ، وعظيم سلطانك.

DEDICATION

To those who unconditional support with my studies, I am honored to have you as my parents. Thank you for giving me chance to prove and improve myself through all my walks of life, please do not ever change. I love you, my parents.

To all those who is inspiring me, and being the pillow, role models, cheerleading squad and sounding boards I have needed, my brothers and sisters.

To all those who is believing in me; for allowing me to further my studies, please do not ever doubt my dedication and love for you, my Friends.

To all those who extended their hands to help us,

ACKNOWLEDGEMENT

I would like to express my utmost gratitude and appreciation to God and several people who have engaged in a tremendous and remarkable role in the course of this project.

Thank for supervisor (Ahmed Abdul-Aziz) Who supervised the project, he did not spare his advice, guidance and ideas as well as our university (Sudan University of Science and Technology) who gave me the golden opportunity to do this project on the topic (quality assurance of integrated system maintenance), also thanks to our teacher (Amged Mohamed) and (Omer Alharith) who also helped us in solving a lot of problems in project, I am really thankful to them.

ABSTRACT

The project is control the quality of existing system after changing the requirements of the integrated system which is consists of two systems (sales - stores) by adding and modifying requirements for the integrated system.

Making the unit test for each of them, then integration testing, then testing the whole system taking into account the quality management to modification of the integrated system based on quality model used on the old system, or modify the model based on requirements relating to the integrated system after adjustment.

المستخلص

هذا المشروع عباره ضبط الجوده لنظام موجود بعد تغيير متطلبات النظام المتكامل الذي يتكون من نظامين (مبيعات - مخازن) وذلك بإضافة وتعديل متطلبات النظام المتكامل.

يتم إجراء اختبار وحدة لكل منهما، ثم اختبار التكامل ، ثم اختبار النظام كلل مع مراعاة إدارة الجودة في تعديل النظام المتكامل بناء على نموذج الجودة المستخدمة في النظام القديم، أو تعديل النموذج على أساس الاحتياجات المتعلقة بالنظام المتكامل بعد التعديل.

INDEX OF TERMINOLOGIES

Abbreviation	Terminology
SMF	Software Maintenance Framework
IEEE	Institute of Electrical and Electronics Engineers
ESD	US Air force Electronic System Division
RADC	Rome Air Development Centre
ISO	International Organization for Standardization
DIN	Deutsche Institute for Numbering
SOA	service-oriented architecture
SQA	Society of Quality Assurance
UML	The Unified Modeling Language
SSL	Secure Sockets Layer
IDE	an integrated development environment
API	application program interface
GUI	Graphic user Interface
CGI	Common Gateway Interface
CLI	Command line Interface
XML	Extensible Markup Language
RAD	rapid application development
MR	Modification Request
QA	Quality Assurance
LOC	Lines Of Code
CC	Cyclamates Complexity
PHP	Hypertext Preprocessor

INDEX OF FIGURES

INDEX OF TABLES

INDEX OF CONTENT

CHAPTER ONE

GENERAL FRAMEWORK OF PROJECT

1.1 THE INTRODUCTION OF PROJECT
1.2 PROJECT PROBLEM
1.3 PROJECT OBJECTIVES
1.4 THE SCOPE OF PROJECT
1.5 THE IMPORTANCE OF PROJECT

INTRODUCTION

This chapter consists of five sections, the first introduction about the project, the second describes research problem, the third is about project objectives, and the fourth is about the limits of the project, and the fifth and last is about importance of project.

1.1 INTRODUCTION TO PROJECT

Need to control the quality when maintaining the integrated system which consists of two separate systems, to achieve the goals of individuals and organizations.

To guarantee that the system achieves what is required, there will be a test of the integrated system after maintenance, taking into account the quality management to modify the integrated system.

1.2 RESEARCH PROBLEM

Maintenance of the systems effectively is essential and requires certain skills, this project about to control the quality of integrated system when changing, modifying or adding new requirements to the integrated system which consists of two separates systems.

1.3 OBJECTIVES

The project intend to achieve this objectives

- Adding and modifying requirements for the integrated system.
- Control the quality when maintaining system.
- Improve the system quality.

1.4SCOPE

There is an integrated system consisting of two separate systems require maintenance to perform a specific task, we change the requirements with the addition of new requirements for each separate systems.

Making the unit test for each of them, then integration testing, then test the whole system taking into account the quality management to modification of the integrated system based on quality model used on the old system, or modify the model based on requirements relating to the integrated system after adjustment.

1.5 IMPORTANCE

- The importance of project in that it is based on the change and the addition of the requirements in an integrated system that component of two separate systems and tested without affecting the requirements of each system.
- Application of quality on the integrated system concept in stages of modification and additional requirement until they are conformed to the specifications of the required.
- Management of quality to modify the integrated system based on the quality used in the old system model or modify the model based on the requirements of the integrated system type after modifications.
- The development of applications that have been built and simplify the process of communication with, and the use of sources of information technology already existing.

CHAPTER TWO

THEORETICAL FRAMEWORK AND PREVIOUS STUDIES

2.1 THEORETICAL FRAMEWORK

2.2 PREVIOUS STUDIES

INTRODUCTION:

This chapter deals with the theoretical framework for the system in the first section talking about maintenance and testing as well as quality and in the second section speaks of previous studies similar to this project.

2.1 THEORETICAL FRAMEWOK

2.1.1 MAINTENANCE

MAINTENANCE DEFINITION:

The act of keeping an entity in the existing state of repair, Efficiency, or validity to preserve from failure or decline.[1]

WHY SOFTWARE MAINTENANCE IS NEEDED

- **TO PROVIDE CONTINUITY OF SERVICE**

  Systems need to keep running, systems cannot be allowed just to stop if an error occurs, and System cannot be allowed just to stop if an error occurs. Unexpected failure of software can be life threatening. Many facets of daily life are now managed by computer. There can be severe consequences to system failure such as serious inconvenience or significant financial implications. Maintenance activities aimed at keeping a system operational include bug-fixing, recovering from failure, and accommodating changes in the operating system and hardware.

- **TO SUPPORT MANDATORY UPGRADES**

  This type of change would be necessary because of such things as amendments to government regulations will necessitate modifications in the software used, the need to maintain a competitive edge over rival products will trigger this kind of change.

- **TO SUPPORT USER REQUESTS FOR IMPROVEMENTS**
- **TO FACILITATE FUTURE MAINTENANCE WORK**

  It does not take long to learn that shortcuts at the software development stage are very costly in the long run. It is often financially and commercially justifiable to initiate change solely to make future maintenance easier. [1]

SOFTWARE MAINTENANCE FRAMEWORK

Software maintenance is not an activity carried out in a vacuum. It affects, and interacts with the environment within which it is carried out.

Framework **-** a set of ideas, conditions, or assumptions that determine how something will be approached, perceived, or understood.[1]

Understanding the framework and the relationship between the factors comprising this framework allows prediction of problem areas and the ability to avoid them.

SOFTWARE MAINTENANCE FRAMEWORK COMPONENTS

Software Maintenance Framework (SMF) will be used to discuss some of these factors. The elements of this framework are the user requirements, organizational and operational environments, maintenance process, software product, and the maintenance personnel.

To understand the sources of the software maintenance challenge, you need an understanding of these components, their characteristics and the effect of their interactions**.**

- **USER**

  Refer to individuals who use the system, regardless of their involvement in its development or maintenance.

- **ENVIRONMENT**

  The environments affecting software systems are the operating environment and the organizational environment.

- **OPERATING ENVIRONMENT**

  All software and hardware systems that influence or act upon a software product in any way.

- **HARDWARE INNOVATIONS**

  The hardware platform on which a software system runs may be subject to change during the lifetime of the software.

- **SOFTWARE INNOVATIONS**

  Like hardware, changes in the host software may warrant a corresponding modification in the software product.

- **ORGANIZATIONAL ENVIRONMENT**

  All non-software or non-hardware related environmental factors. [1]

MAINTENANCE PROCESS

The maintenance process itself is a major player in the software maintenance framework. Significant factors are the capturing of change requirements, programming practice, paradigms and error detection. [1]

SOFTWARE PRODUCT

Programs are seldom static and can never be so when they implement large systems that are in continuous use. Lehman compares this phenomenon to the evolution of biological organisms and of social groupings .Remember however, that it is not just the programs themselves but also the accompanying documentation and operating procedures that are subject to such changes. [1]

Aspects of a software product that contribute to the maintenance challenge include:

- Maturity and difficulty of the application domain: The requirements of applications that have been widely used and well understood are less likely to undergo substantial modification on installation than those that are still in their infancy. An aspect that may also affect maintenance is the inherent difficulty of the original problem.
- Quality of the documentation: The lack of up-to-date systems' documentation is one of the major problems that software maintainers face.
- Programs are often modified without a corresponding update of the documents affected. Even in environments where there are automatic documentation support tools, their contents may be inaccurate. Worse still, there may be no documentation at all. Inadequate documentation adversely affects maintenance productivity even for a programmer maintaining his / her own program, and in the vast majority of cases people are maintaining programs written by others.
- Malleability of the programs: The malleable or 'soft' nature of software products makes them more vulnerable to undesirable modification than hardware items.
- Inherent quality: The nature of the evolution of a software product is very closely tied to the nature of its associated programs. Based on the results derived from empirical observations of large industrial software systems. [1]

SOFTWARE CHANGE

Change the act, process, or result of being made different in some particular. [1]

CLASSIFICATION OF CHANGES

- **CORRECTIVE CHANGE**

  Corrective change refers to modification initiated by defects in the software.

  - **Design errors** occur when, for example, changes made to the software are incorrect, incomplete, wrongly communicated or the change request is misunderstood.
  - **Logic errors** result from invalid tests and conclusions, incorrect implementation of design specifications, faulty logic flow or incomplete testing of data.
  - **Coding errors** are caused by incorrect implementation of detailed logic design and incorrect use of the source code logic. Defects are also caused by data processing errors and system performance errors. [1]

- **PERFECTIVE CHANGE**

  This term is used to describe changes undertaken to expand the existing requirements of a system. [1]

- **PREVENTIVE CHANGE**

  Preventive change is undertaken to prevent malfunctions or to improve maintainability of the software. [1]

- **ADAPTIVE CHANGE**

  Adaptive change is a change driven by the need to accommodate modifications in the environment of the software system. [1]

2.1.2 QUALITY:

Researchers differed in unified and comprehensive quality definition, quality from the perspective of one person to another is different in many citizens, but we can say that one of the concepts of quality is that the product meets all the desired requirements.

QUALITY DEFINITIONS

INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS (IEEE) DEFINITION:

The composite characteristics of the software that determine the degree to which the software in use will meet the expectations of the customer.[2]

(PRESSMAN'S DEFINITION) DEFINITION OF QUALITY:

Conformance to explicitly stated functional and performance requirements, explicitly documented development standards, and implicit characteristics that are expected of all professionally developed software.[3]

QUALITY ASSURANCE (QA):

A set of activities designed to evaluate the process by which the products are developed or manufactured. Contrast with quality control.[3]

QUALITY MODELS:

There are many models that can be applied to the system, including:

MCCALL MODEL:

Used in the United States for very large projects in the military, space and public domain. It was developed in 1976-7 by the US Air force Electronic System Division (ESD), the Rome Air Development Centre (RADC) and General Electric (GE) with the aim of improving the quality of software products. One explicit aim was to make quality measurable.

Started with a volume of 55 quality characteristics which have an important influence on quality, and called them "factors", for the purposes of simplification trimmed Mack Cole numbering 11 property are: -

1- Efficiency.
2- Reliability.
3- Usability.
4- Maintainability.
5- Testability.
6- Reuse.
7- Portability.
8- Flexibility.
9- Harmonic.
10- Correction.
11- Safety. [2]

BOËHM MODEL:

Has added a number of new factors to list Mc Cole, even to the number reached 16 , with the passage of time, the quality of the work of others , some models of the redefinition of the factors exist and has added some new factors . As there are a lot of models, including ISO, IEEE, DIN.agv main of these models is to make quality measurable.

QUALITY FACTORS:

USABILITY:

Usability is the cost/effort to learn and handle a product, User staffs were simply required to learn how to operate the system, input data, receive output and generally keep the system running.

ACCURACY:

Accuracy is the extent to which a program which fulfill its specification, it is a difficult factor to pin down because of the lack of standard terminology, easy to use the term interchangeably with other factors like reliability and integrity.

TRANSFERABILITY:

The cost of transferring the product from its hardware or operational environment to another one

MAINTAINABILITY:

The cost of localizing and correcting errors:

a. corrective maintenance
b. maintenance harmonic-adaptive maintenance
c. perfective maintenance

REUSABILITY:

The cost of transferring a module or program to another application, Reusability addresses the concept of writing code so that it can be used more than once.

FLEXIBILITY:

The cost of product modification like being able to change or reconfigure the user interface to suit users' preferences.

TESTABILITY:

The cost of program testing for the purpose of safe guarding that the specific requirements are met.[2]

SPLIT QUALITY FACTORS TO GROUPS:

Quality factors can be divided into two groups:

1. **EXTERNAL FACTORS**:

It is the final quality of the system, which appear to the outside world, and the impact on users' (end user), Examples include: usability, reliability, safety.

2. **INTERNAL FACTORS:**

Which are in the structuring of the system and takes care of the development team and the maintenance team, Examples are: portability, maintainability, reusability. [2]

QUALITY MEASURES:

THE DEFINITION OF QUALITY STANDARDS:

Quantitative measure shows any element has a certain degree of quality property. [From theory]

TYPES OF METRICS:

Metrics are classified by the development cycle (lifecycle) to:

1- PROCESS METRICS:

Process metrics are known as management metrics and used to measure the properties of the process which is used to obtain the software. Process metrics include the cost metrics, efforts metrics, and advancement metrics and reuse metrics.

2-PRODUCTS METRICS:

Product metrics are also known as quality metrics and is used to measure the properties of the software. Product metrics includes product non reliability metrics, functionality metrics, performance metrics, usability metrics, cost metrics, size metrics, complexity metrics and style metrics. Products metrics help in improving the quality of different system component & comparisons between existing systems.[4]

ADVANTAGES OF SOFTWARE METRICS:

- Analysis, comparison and critical study of various programming language with respect to their characteristics.
- In comparing and evaluating capabilities and productivity of people involved in software development.
- In getting an idea about the complexity of the code.
- In providing feedback to software managers about the progress and quality during various phases of software development life cycle.
- In allocation of testing resources for testing the code.
- In comparison and making design tradeoffs between software development and maintenance cost. [4]

2.1.3 SOFTWARE TESTING:

TESTING DEFINITION:

Testing is the examination of a software system in the context of a given specification set. The purpose of testing is to find errors in software systems i.e. to identify ways in which it does not conform to an agreed specification. [1]

Test techniques include, but are not limited to, the process of executing a program or application with the intent of finding software bugs (errors or other defects).It involves the execution of a software component or system component to evaluate one or more properties of interest.

In general, these properties indicate the extent to which the component or system under test, so in this section we want discuss testing of the component of two systems works together as whole to figure out if the system is agreed the specification. [1]

REASON TO TEST THE SOFTWARE:

Testing software to see if it works is much the same. Unless you can comprehensively test every eventuality, every context of every input (which you can't, the possibilities are infinite) the best you can do by testing is to say that it works in this finite set of circumstances, and gives an indication of how it might work in others.[1]

TESTINGTYPES:

TESTING CODE:

Code can be tested at many different levels do individual statements execute according to specification, do procedures provides expected output for given input, does the program as a whole perform in a particular way? Within this are many issues to be borne in mind. For example, it is possible to execute each statement without touching upon certain conditions. However, test cases should try to take account of all possible conditions and combinations of conditions, with special emphasis on boundary conditions and values where behavior is often erroneous.[1]

BLACK BOX AND WHITE BOX TESTING:

- In black box testing, the system acts as a black box - we don't see inside it, we just see what goes in and what comes out. Test cases are derived from the specification, and take no account of the internal operation of the program.
- In white box testing we 'see inside the box' and look at the detail of the code.
- A program giving the correct result to a black box test, is not necessarily executing its code correctly. Similarly, a program statement executing correctly does not mean that it conforms to its specification.[1]

STRUCTURED TESTING:

- An aim of structured testing is to maximize the number of errors found by the test cases used and to avoid redundant test cases. Consider the program that takes two integers as input and outputs their product, 39,600 different combinations could be tested and still miss some of the boundary conditions. Redundant testing is inefficient.

- Testing should be planned by assuming that the program contains errors. It's a very safe assumption to make, yet all too often, testing strategies are built upon the premise that the program works.[1]

INTEGRATION TESTING:

Testing a program starting with tests of its elements and then combining them to test larger elements is known as integration testing. A procedure, function or module may work on its own, but fail to execute correctly when tested together with other elements. Consider a function that takes two numbers and returns the result of subtracting one from the other. The order in which the numbers are supplied to the function is vital to correct operation, but incorrect ordering won't necessarily show up until the function is tested in conjunction with other elements of the program that use it.

Testing program elements requires program stubs to be written. These are small programs that execute the element under test. Program stubs do not appear in the final product, but are in themselves a powerful and reusable testing tool. [1]

REGRESSION TESTING:

Testing finds errors and also checks that amended code has fixed the problem. However, software has a tendency to instability. We have discussed for example the ripple effect that modifications can produce. Thus an amendment that fixes one error may introduce or reintroduce others. Regression testing is the running of tests both to see that the identified bug has been fixed and to check that other errors have not been introduced. [1]

VERIFICATION AND VALIDATION:

The verification and validation of software is a key in building systems that can be trusted. Verification, ensuring accuracy against an agreed set of

requirements and specifications, is largely what this chapter has been about. Validation is external certification that a system can demonstrate a level of compliance, perhaps to a legal requirement such as a specific safety standard.
Carrying out verification and validation activities is not enough. They must be documented. Without adequate documentation, it will not be possible to demonstrate compliance. [1]

AIM OF THE VERIFICATION AND VALIDATION:

- Achieve better systems i.e. systems with improved reliability, performance, quality and cost effectiveness. Ratikin provides guidance on essential techniques in software verification and validation, showing also how to reconcile conflicting demands e.g. of quality versus tight deadlines. The impact on cost and scheduling of verification and validation processes is a legitimate concern, and it is important to get the balance right. However, undue criticism on grounds of cost should not be allowed to shortcut necessary verification and validation work.

- Verification and validation methodologies provide a robust framework for the creation of quality software system, and are more effective when performed independently of the team building or maintaining the system.
- A useful guide, as in many areas, is to look at the guidelines and standards already developed.[1]
- **TESTING PLANS:**

  A test plan can vary from a short informal document to a multi-volume series, depending upon the purpose for which it is intended.

IEEE PLANS:

"A document describing the scope, approach, resources, and schedule of intended testing activities. It identifies test items, the features to be tested, the testing tasks, who will do each task, and any risks requiring contingency planning."[1]

GOOD TEST PLANS FACILITATE TESTING IN MANY WAYS INCLUDING:

- Providing lists of useful test cases identifying such things as boundary conditions and classes of test data. This improves efficiency and means important test cases are less likely to be missed.
- Providing information in what scale of the job is likely to be and what resources will be needed.
- Providing information to identify and prioritize tasks, thus aiding organization of the testing team and identifying roles and responsibilities.[1]

2.2 PREVIOUS STUDIES:

INTRODUCTION TO THE PREVIOUS STUDIES:

Through search about studies on apply of the concepts and techniques that were used to link two independent systems were reached to earlier studies for applying SOA service infrastructure technology.

THE DEFINITION OF PREVIOUS STUDIES:

- **SERVICE BASED STUDENTS INFORMATION MANAGEMENT SYSTEM.**

By Marib Mohamed, Salwa Ibrahim and Ahmed Abdul Aziz, at August 2012.

THE AIM OF THE STUDY:

The research aims to implement an integrated system that manages student at the College of Computer Science and Information Technology Information and enable it to optimal use of resources within the university and to facilitate access to those resources, also the target of this system of training on the use of new technology and its use in solving a problem.

THE RESULT OF THE STUDY:

This previous study has contributed in the development of the student system for college, where reached for applying a system that works to manage student’s information in college through applying technology infrastructure services SOA.

- **QUALITY ASSURANCE OF INTEGRATED SYSTEM.**

By Ibrahim Jafar Mustafa, Ahmed Esam El-Din Ahmed,
Abdurrahman Ali Mohammed Ahmed and
Omar Ahmed Ali at August 2014.

THE AIM OF THE STUDY:

The research aims to take advantage of the concepts of SOA and SQA in building an integrated system is on its way to link systems easy to handle, ensures access to data sources and handled with ease.

THE RESULT OF THE STUDY:

This previous study has contributed in building an integrated system by linking the two independent systems and applied the concept of the quality in an integrated system in all phases of the link until it is matched with the required specifications.

CHAPTER THREE

RESEARCH METHODOLOGY AND TECHNIQUES USED

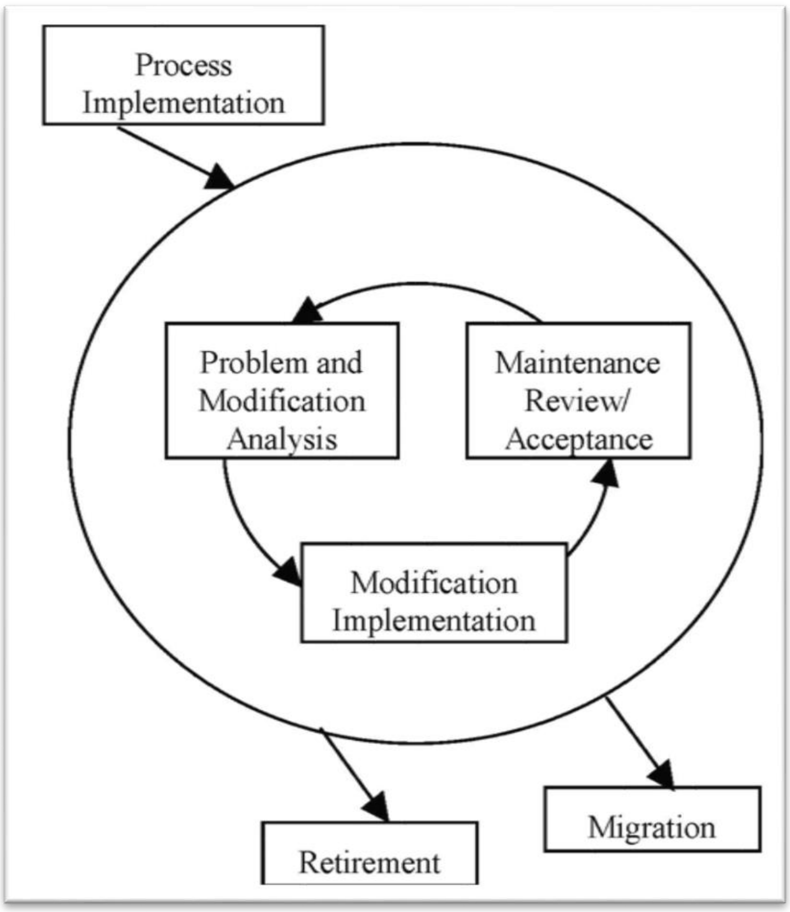
INTRODUCTION:

This chapter contains the research methodology and the tools and techniques used in the system.

3.1 RESEARCH METHODOLOGY:

We have been applied software maintenance standards IEEE 1219 and ISO/IEC 14764 shown in figure (3.1), the Maintenance Process contains the activities and tasks necessary to modify an existing software product while preserving its integrity. The activities which comprise the Maintenance Process are Process Implementation, Problem and Modification Analysis, Modification Implementation, Maintenance Review/Acceptance.

Implementation process establishes the planes and procedures ,Problem and Modification Analysis activated after the software transition and are called iteratively when the need for modification arises, in the modification implementation the maintainer develops and tests the modification of the software product, in Maintenance Review/Acceptance ensures that the modifications to the system are correct and that they were accomplished in accordance with the approved standards using the correct methodology , Migration the maintainer needs to determine the actions needed to accomplish the migration, and then develop and document the steps required to effect the migration , and Retirement once a software product has reached the end of its useful life, it must be retired. An analysis is should be performed to assist in making the decision to retire a software product. [5]


Process Implementation
Problem and Modification Analysis
Maintenance Review/ Acceptance
Modification Implementation
Retirement
Migration

Form (3.1) illustrates the stages of the establishment of the proposed system [5]

3.2 TOOLS AND TECHNIQUES

3.2.1 ENTERPRISE ARCHITECT

Is a technical document that described the system status and applications that are used in the enterprise also described as flows of information from the other within the same enterprise system, one of the tools used for engineering subsidized computer software, is used in the analysis and design of software systems operations, the primary objective of it improve your UML situation institution, depends on determination to language, a visual language described the system status, can cover all activities related to the development of systems of the initial phase of the analysis to the system delivery .its stage provides the basic framework which defines and describes the base required by the institution and the process of translation Strategy of our work in the future, as organization can achieve its goals effectively and efficiently and to achieve its vision.

PROS OF ENTERPRISE ARCHITECT:

- Activity based management and analysis of large, complex systems.
- Manage and monitor complex systems requirements.
- Design and build systems using (UML) language, and build independent models of the system to be analyzed and constructive.
- Deals with many programming languages such as (java),(C),(C++).
- Improve decision-making processes and the ability to adapt to changing demands. [6]

UNIFIED MODELING LANGUAGE (UML):

Is a general-purpose, developmental, modeling language in the field of software engineering that is intended to provide a standard way to visualize the design of a system. [11]

OBJECTIVES OFUML:

- Expresses the relationships between a system and its environment
- Clarifies requirements: Specifying goals leads to asking "why", "how" and "how else".
- Intended to address the current software development issues.
- Include all the concepts that necessary to support a modern iterative process based on building strong architecture to solve user case-driven requirements.
- Connects requirements to design.
- Simple and expressive.[11]

3.2.2 WAMP SERVER:

Wamp server refers to a software stack for the Microsoft windows operating system, created by Romain Bourdon and consisting of the Apache web server, Open SSL for Secure Sockets Layer (SSL) support, My SQL database and Hypertext Preprocessor(PHP) programming language.

Wamp Server includes Ease of handling and use, through the user interface enables database management. [12]

3.2.3 MICROSOFT VISUAL STUDIO:

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs for Microsoft Windows, as well as web sites, web applications and web services. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silver light. It can produce both native code and managed code**.** [12]

3.2.4 MICROSOFT ACCESS:

It is a graphical program operates under the Environment GUI. This program contains (Windows) on a variety of objects that can be used to view and manage information such as tables, forms, reports, queries and units of macro units module and pages and access to data. [12]

3.2.5 PHP:

PHP is a server-side scripting language designed for web development but also used as a general-purpose programming language, PHP code can be simply mixed with HTML code, or it can be used in combination with various tinplating engines and web frameworks. PHP code is usually processed by a PHP interpreter, which is usually implemented as a web server's native module or a Common Gateway Interface (CGI) executable. After the PHP code is interpreted and executed, the web server sends the resulting output to its client, usually in the form of a part of the generated web page; for example, PHP code can generate a web page's HTML code, an image, or some other data. PHP has also evolved to include a command-line interface (CLI) capability and can be used in standalone graphical applications. [12]

3.2.6 XML:

Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format which is both human-readable and machine-readable. It is defined by the W3C's XML 1.0 Specification and by several other related specifications, all of which are free open standards.

The design goals of XML emphasize simplicity, generality and usability across the Internet. It is a textual data format with strong support via Unicode for different human languages. Although the design of XML focuses on documents, it is widely used for the representation of arbitrary data structures such as those used in web services.

Several schema systems exist to aid in the definition of XML-based languages, while many application programming interfaces (APIs) have been developed to aid the processing of XML data. [12]

3.2.7 VISUAL BASIC

Visual Basic is a legacy third-generation event-driven programming language and integrated development environment (IDE) from Microsoft for its COM programming model first released in 1991. Microsoft intended Visual Basic to be relatively easy to learn and use, it was derived from BASIC and enables the rapid application development (RAD) of graphical user interface (GUI) applications, access to databases using Data Access Objects, Remote Data Objects, or ActiveX Data Objects, and creation of ActiveX controls and objects.

Programmer can create an application using the components provided by the Visual Basic program itself. Over time the community of programmers developed third party components, Programs written in Visual Basic can also use the Windows API, which requires external function declarations.

Since VB defines default attributes and actions for the components, a programmer can develop a simple program without writing much code. Programs built with earlier versions suffered performance problems, but faster computers and native code compilation has made this less of an issue. [12]

CHAPTER FOUR

REQUIREMENTS ANALYSIS AND DESIGN

4.1 REQUIREMENT IDENTIFICATION, CLASSIFICATION, PRIORITIZATION

4.2 REQUIREMENTS ANALYSIS

4.3 SYSTEM DESIGN SCHEMA

INTRODUCTION:

This chapter contains problem identification, classification, prioritization, analysis and design of the integration system.

4.1REQUIREMENT IDENTIFICATION, CLASSIFICATION, PRIORITIZATION

4.1.1 SALES SYSTEM

- **SHOW THE DATE OF PURCHASE TO THE CUSTOMER:**

  This requirement is considered as MR (1), the customer needs to see product name, date of purchase, quantity and total price, when total price reach to a certain number the system will make discount for this customer.
  The type of maintenance is perfective to improve the performance.
  As preliminary estimate, the modification size of this requirement is small and doesn’t take more effort and time, and regard as low priority because when a customer buys some item the bill is already shown but for a certain day.

- **HIDING FINISHED ITEMS FROM SHOW:**

  This requirement is considered as MR (2), the items that have zero quantity must not appear to the customer, the type of maintenance that correct the discovered faults is corrective and have a medium priority.
  The modification is simple so the size of modification is small.

- **MESSAGE BETWEEN ADMIN AND CUSTOMER:**

  MR (3) is perfective maintenance because it is improve performance of the system and enhance its reliability, and consider the contact between admin and customer is important so it has a medium priority.

  As preliminary estimate of modification that takes an effort, so it has a big modification size.

- **MAKE REPORTS TO ADMIN:**

  MR(4), show the customer date of purchase, customer info, all product even the finished one, and the balance so the type of maintenance is perfective, it require an effort for modifying, it has high priority because the admin must know all about the system information.

- **CREATE BALANCE:**

  MR (5), allow only the admin to add balance by entering serial numbers , charging the balance in the previous system was done by adding the balance directly to the data base and it was traditional approach which mean the type of maintenance is corrective, the modification has big size.

4.1.2 WEB SERVICE:

- **UPDATE QUANTITY OF PRODUCTS:**

  MR (6), when a customer buys some item the store must modify the quantity this item according to this, its regard as corrective maintenance.

  Updating the quantity of all items in the stores system is necessary which make the priority high.

- **MAKE ALERT WHEN ITEM QUANTITY REACH TO A CERTAIN NUMBER:**

  MR (7), when some item is about to finish from the store its better for the admin to pay attention so it’s perfective maintenance and has medium priority, this modification is done by adding some function.

- **VALIDATION FOR QUANTITY AND PRODUCT CODE:**

  MR (8), in case customer require item that doesn’t exist or more than available quantity the system must aware him.

  It’s obvious that type of maintenance is corrective, it’s essentially then it has high priority and takes effort to modify.

4.2 REQUIREMENTS ANALYSIS:

In this section we will make feasibility analysis and detailed analysis for the whole validated MR. [6]

- **SHOW THE DATE OF PURCHASE TO THE CUSTOMER:**
  - Feasibility Report:

    This modification doesn’t impact in other requirements but effected by others, doesn’t allow unauthorized users to see other date of purchase which improve safety and security and also useful for a customer to see his info, the report review must be clear.

  - Detailed Analysis Report:

    To make this modification we need getting to know sales code and it’s involves validation from the customer, according to that, appearing his information from the database.

    This requirement agreed with existing security features, hence it doesn’t decrease system security and safety, and make unit testing to insure its meeting our expectations.

- **HIDING FINISHED ITEMS FROM SHOW:**
  - Feasibility Report:

    This requirement affect in show functions only, as alternate solution it is possible to show items with zero quantity and when customer choose it, the system will aware him, it is decrease the process for user and system, and improve system performance.

  - Detailed Analysis Report:

    This modification require knowledge with sales and stores code, that's involves disappear of the items having zero quantity.

    Although it has nothing to do with security issues but the security remain well. We will make unit and integration testing to vitrify the integration of the two systems.

- **MESSAGE BETWEEN ADMIN AND CUSTOMER:**
  - Feasibility Report:

    Independent requirement, and it's improve the efficiency and the reliability of the system, administer information on website helps customer to contact with him, as alternate solution, it easy to contact with admin while it's not allowed for unauthorized users to access messages.
  - Detailed Analysis Report:

    To do this modification requires know sales code, to identify suitable position for message, it involves adding new table in database; it was compatible with existing security feature and doesn't influence on system security.

    Make unit testing and whole system testing to insure the connection between them.

- **MAKE REPORTS TO ADMIN:**
  - Feasibility Report:

    Adman’s reports make him more aware of system components and updates which decrease the probability of the system’s unreliability, this reports is not accessed except by admin which improve system safety.

    Reports review presented in tabular form in an orderly manner which make easy to the admin to keep track of the system.
  - Detailed Analysis Report:

    There must be cognizance of sales and stores code, to makes this modification which include retrieving data from database, then making unit and integration testing.

- **CREATE BALANCE:**
  - Feasibility Report:

    The influence of charging balance by the admin is give the customer the ability to add balance to his account, letting the admin do this process improve system security and his interactive with the system.
  - Detailed Analysis Report:

    The modification include adding new interface to the system which featuring the safety and doesn’t effect on security, unit testing has been used as test strategy.

- **UPDATE QUANTITY OF PRODUCTS:**
  - Feasibility Report:

    This procedure impact on making alert and interface of products show, by this requirement the users will know the original quantity of products, so it’s important to update the quantity to avoid conflicts.

- Detailed Analysis Report:

  To make this modification must be Familiar with sales code, web service code and store's database, add new function to web service code as element of modification which doesn't effect on existing safety and security features, need to do unit, integration, and whole system testing.

- **MAKE ALERT WHEN ITEM QUANTITY REACH TO A CERTAIN NUMBER:**
  - Feasibility Report:

    This requirement affected by update quantity and doesn't impacts in another requirements, instead making list of the items which about to finish in case doing this in stores or web service.
    This alert is done only for the admin, which increase the interactive between the system and admin and decrease the effort of figuring out the finished items.
  - Detailed Analysis Report:

    Its need to know the web service and sales code and stores database, this modification adding new function in the web service and interface to view the finished items.

    It's also co incise with existence security and safety issues, and need to do unit, integration, and whole system testing.

- **VALIDATION FOR QUANTITY AND PRODUCT CODE:**
  - Feasibility Report:

    This modification influence in all requirements that associate with purchase operation, in addition to that in the validation process the interface of system seems clear to the customer who improve efficiency and performance.

- Detailed Analysis Report:

  The database of stores, web service, and sales code need to be known by the developer.

  The modification involve adding new function and interface to review the result , Also unit, integration, and whole system testing need to be done.

4.3 SYSTEM DESIGN SCHEMA

This section contains UML and the schema that require in it.

4.3.1 USE CASE DIAGRAM:

Presents the activities carried out by the customer shown in Figure 4.1 through its use of resources also presents the use cases for each part of the system components.

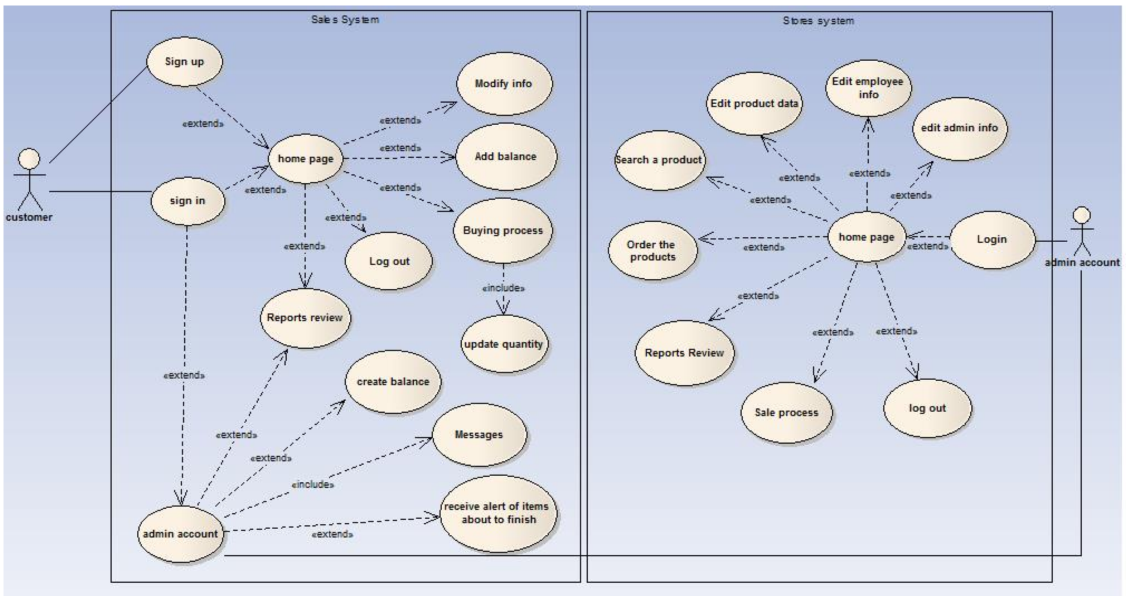


Figure 4.1 show use case diagram

4.3.2 ACTIVITY DIAGRAM:

Present the system functionality as in real world figure 4.2

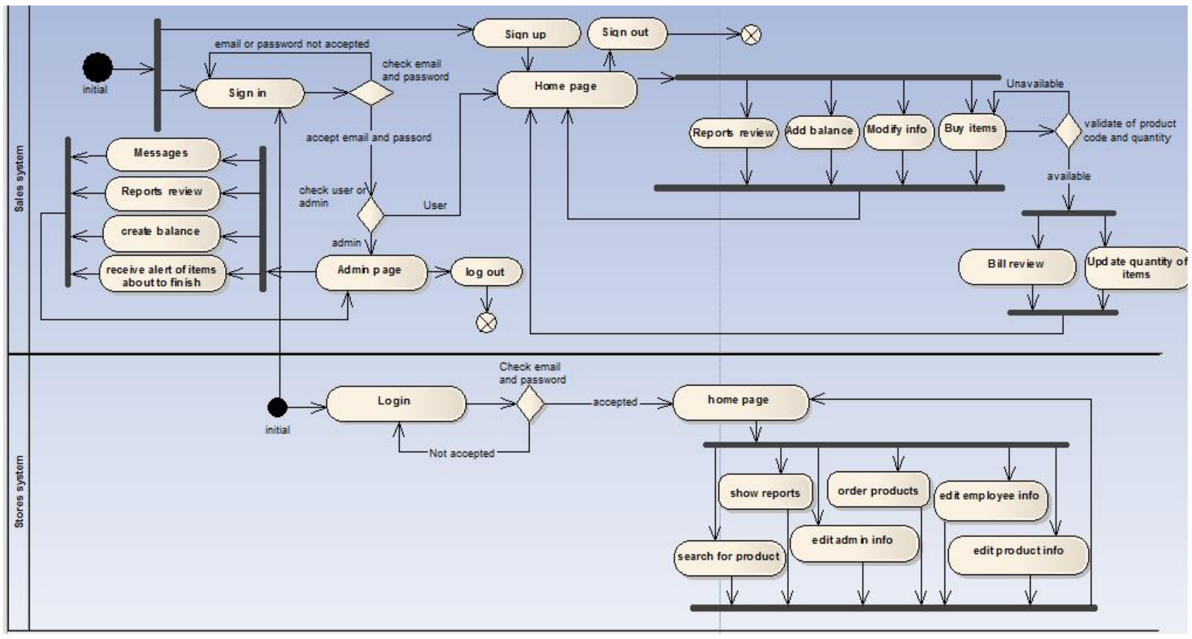


Figure 4.2 show activity diagram

4.3.3 CLASS DIAGRAM:

Show the Categories and important functions in the system figure 4.3

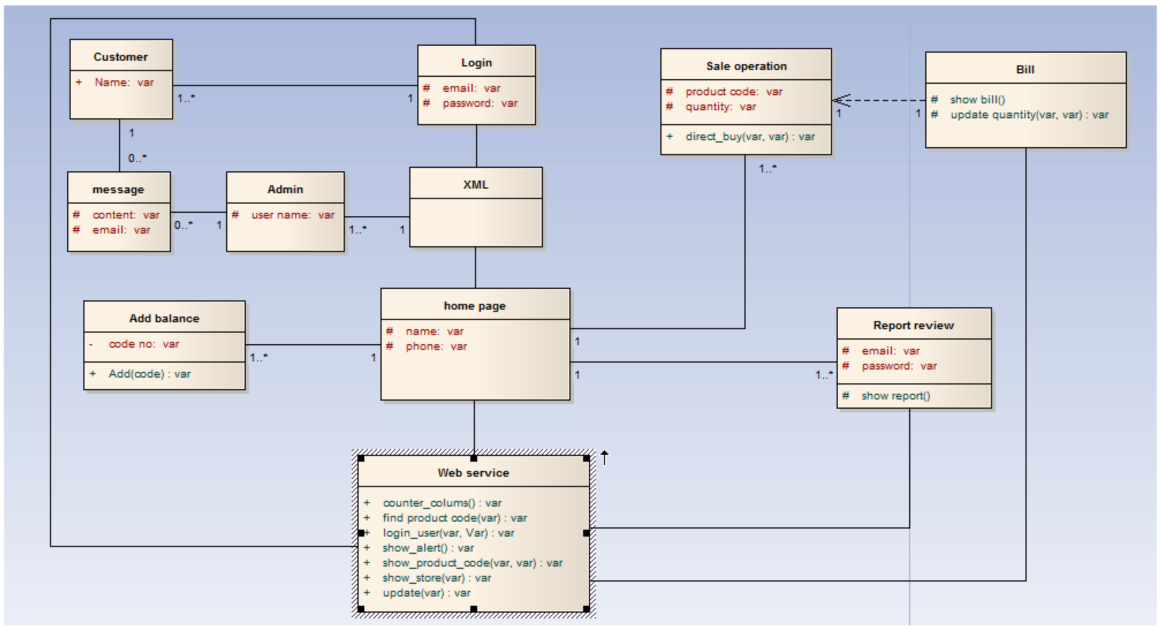


Figure 4.3 show class diagram

4.3.4 COMPONENT DIAGRAM:

Describe software components and their dependencies to each other figure 4.4.

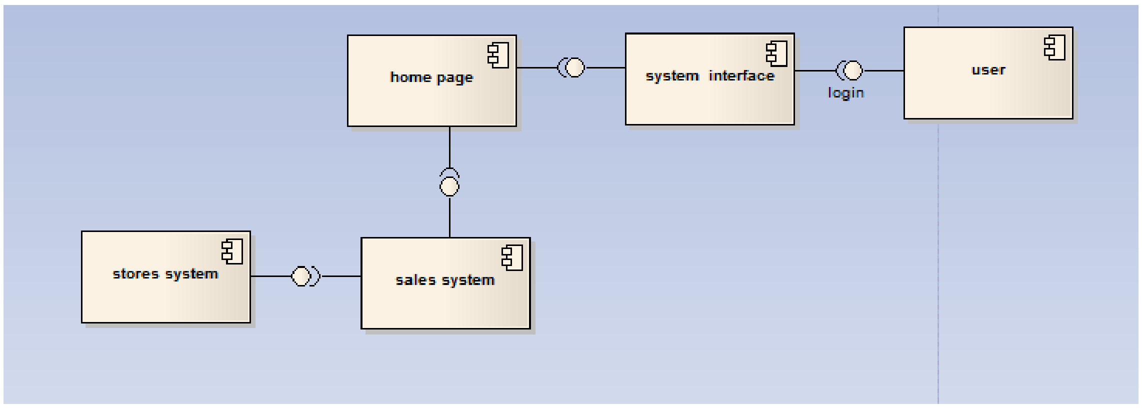


Figure 1.4 show component diagram

4.3.5 DEPLOYMENT DIAGRAM:

Models the run-time architecture of a system figure 4.5

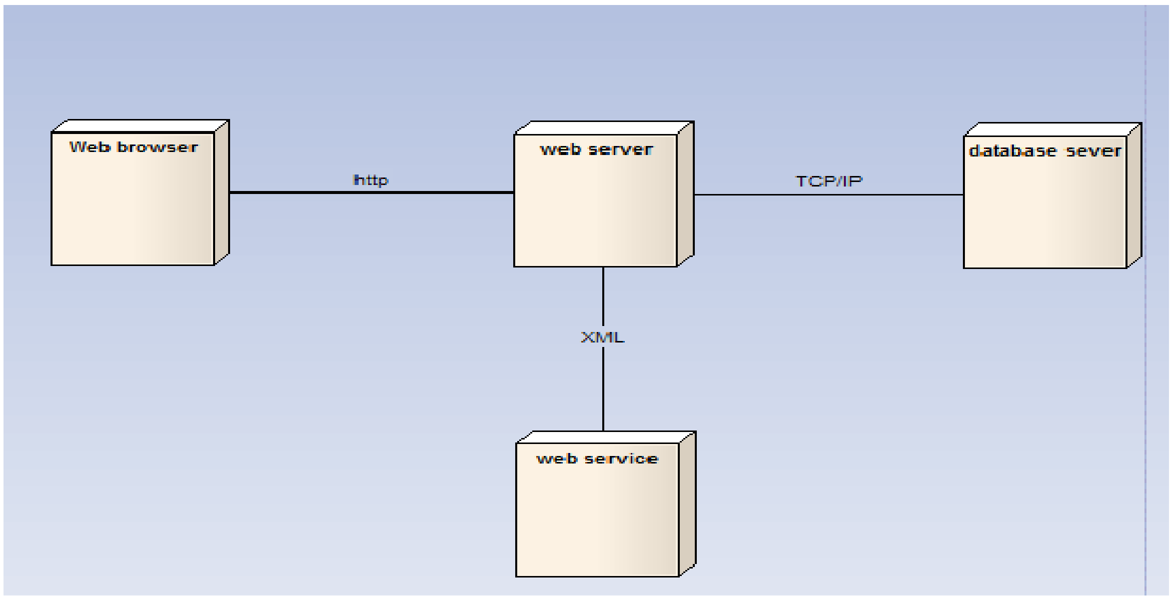


Figure 4.5 show deployment diagram

CHAPTER FIVE

THE SYSTEM IMPLEMENTATION

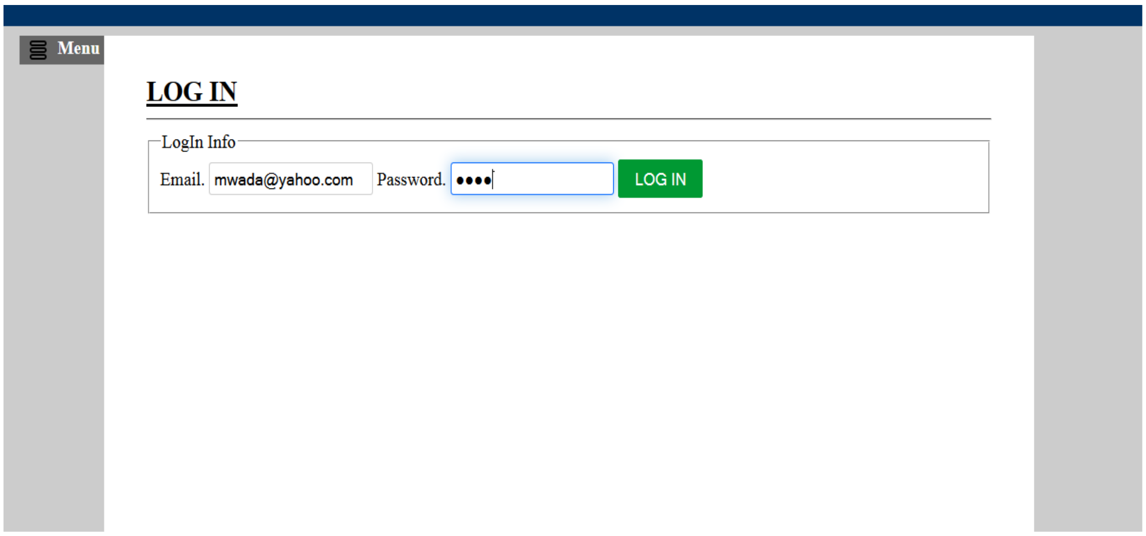
INTRODUCTION

This chapter consists of two sections which contains of system interfaces implementation after maintenance process.

5.1 INTERFACES OF SALES SYSTEM

5.1.1 LOGIN

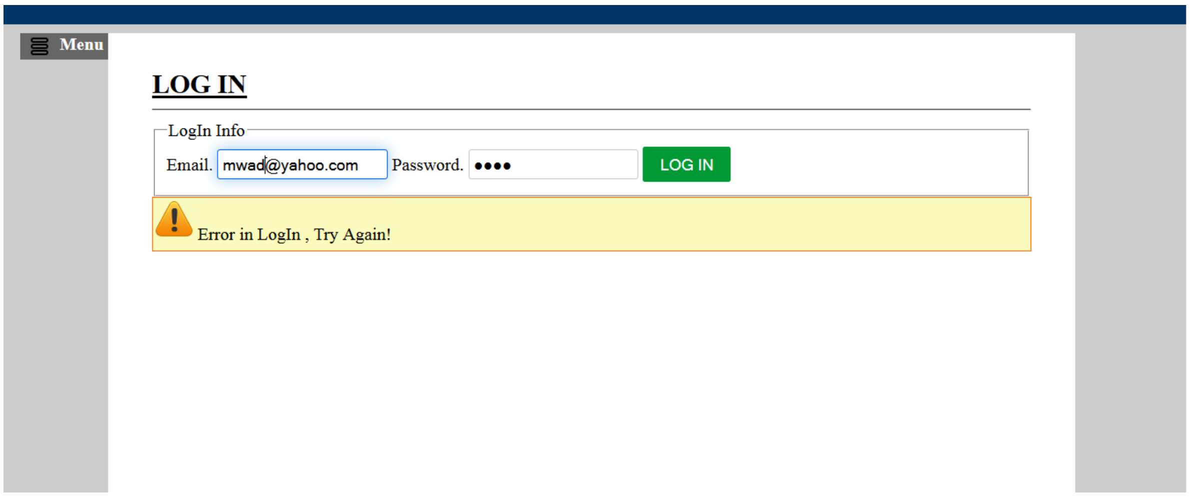
It allows the customer access to products which exists in stores system by logging into the web page after you enter your user name and password and request services with ease as shown in Figure 5.1



mwada@yahoo.com
••••

Figure 5.1 illustrates login

The system check the user name and password then it is allow to access the system, if the user is authorized, if not the system show message to user as shown in Figure 5.2



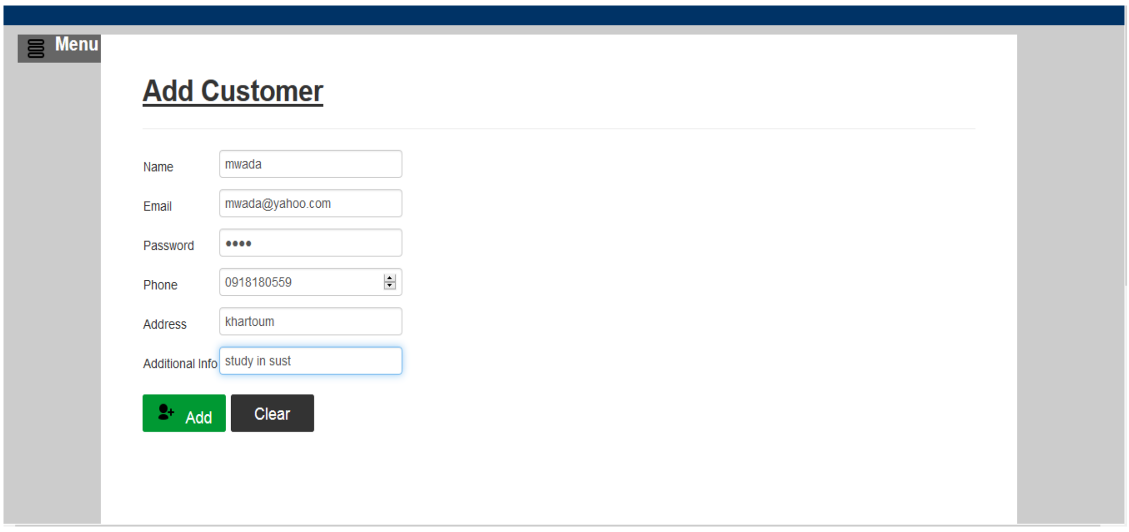
Email. mwad@yahoo.com Password. ●●●● LOG IN

Error in LogIn , Try Again!

Figure 5.2 illustrates unauthorized use

5.1.2 SIGN UP

The customer can create an account by choosing "Sign Up" then entering the personal data as shown in Figure 5.3

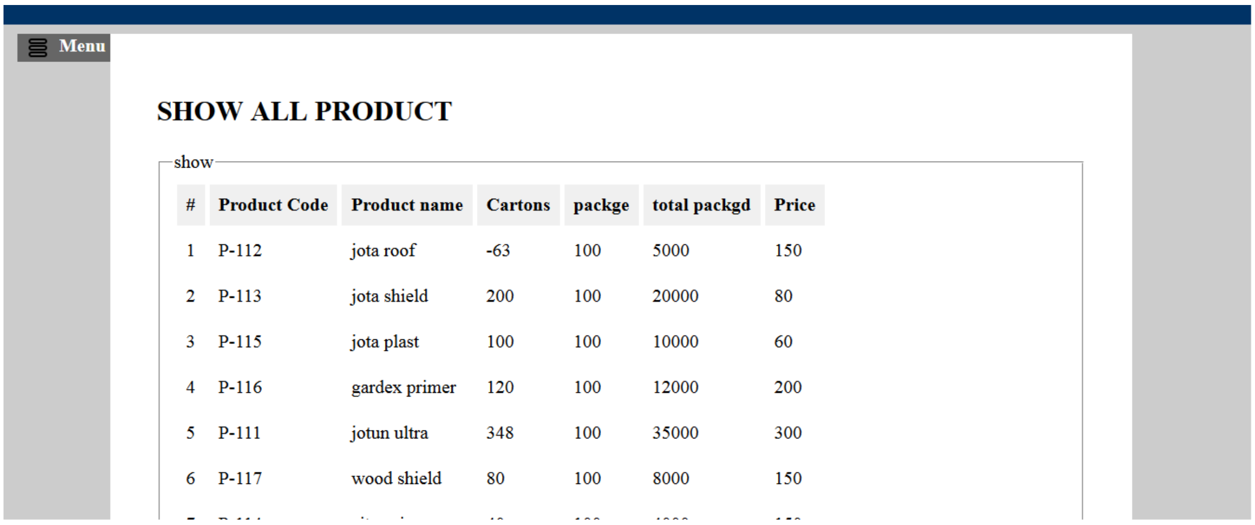


Name	mwada
Email	mwada@yahoo.com
Password	●●●●
Phone	0918180559
Address	khartoum
Additional Info	study in sust

Figure 5.3 illustrates Sign Up

5.1.3 SHOW PRODUCTS

View all the products in the stores as shown in the figure 5. 4



SHOW ALL PRODUCT

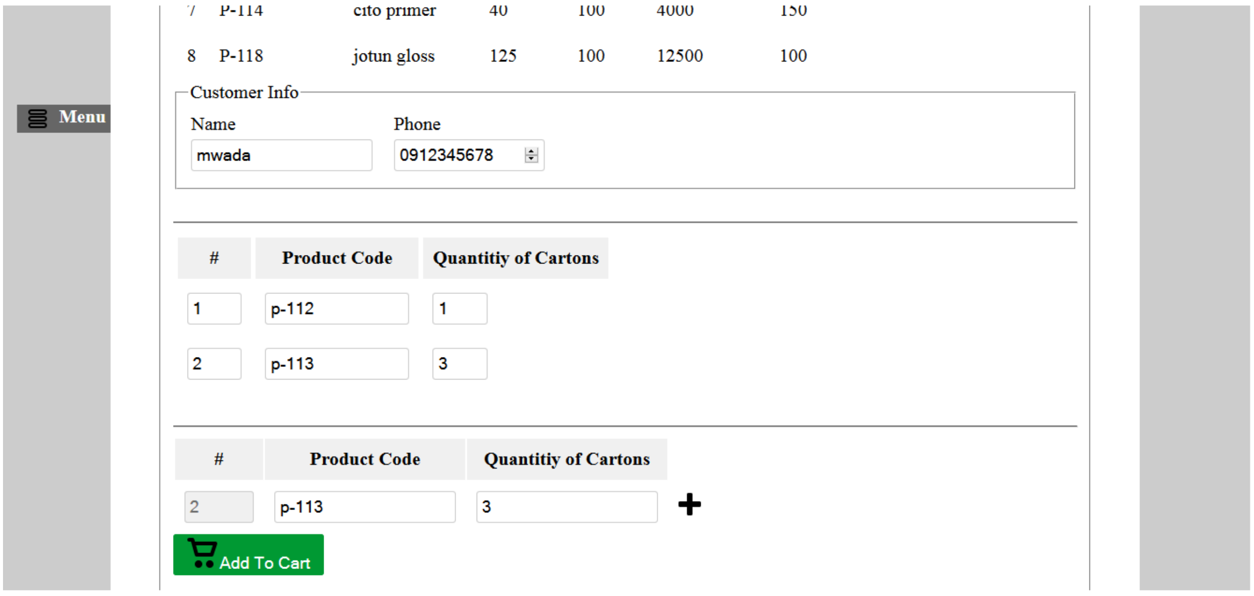
show

#	Product Code	Product name	Cartons	packge	total packgd	Price
1	P-112	jota roof	-63	100	5000	150
2	P-113	jota shield	200	100	20000	80
3	P-115	jota plast	100	100	10000	60
4	P-116	gardex primer	120	100	12000	200
5	P-111	jotun ultra	348	100	35000	300
6	P-117	wood shield	80	100	8000	150

Figure 5.4 illustrates show products

5.1.4 DIRECT SALE

The purchase of products by entering the product number and quantity required as shown in Figure 5.5



7	P-114	cito primer	40	100	4000	150
8	P-118	jotun gloss	125	100	12500	100

Customer Info

Name: mwada

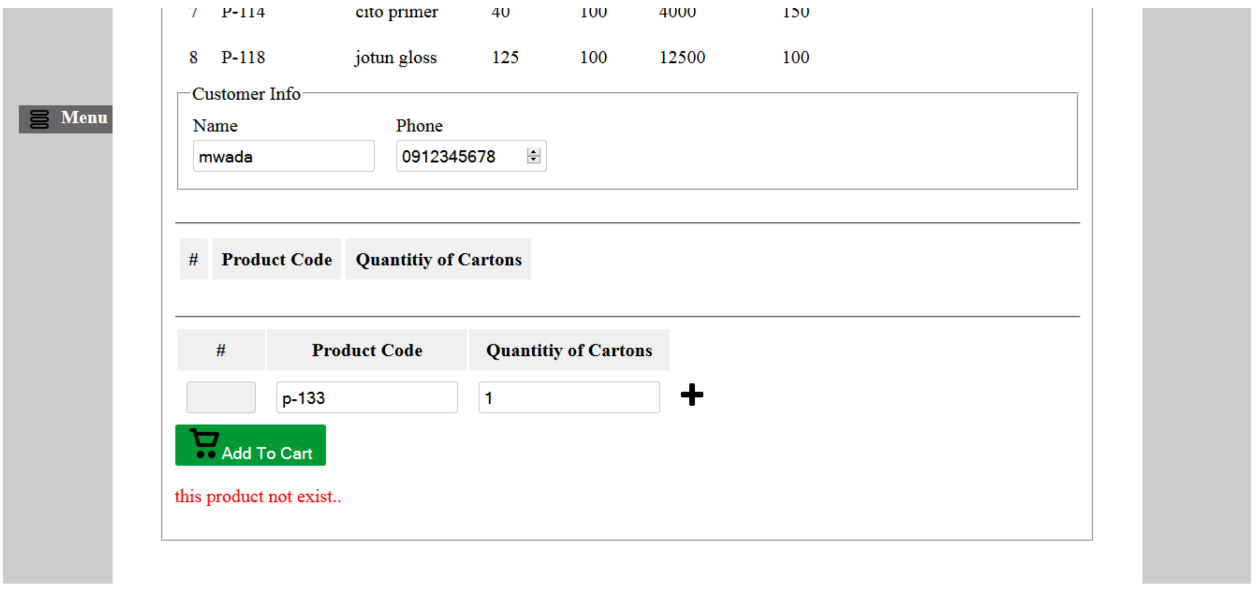
Phone: 0912345678

#	Product Code	Quantitiy of Cartons
1	p-112	1
2	p-113	3

#	Product Code	Quantitiy of Cartons
2	p-113	3

Figure 5.5 illustrates Sale operation

The system check the product code and quantity required as shown in Figure 5.6



7	P-114	cito primer	40	100	4000	150
8	P-118	jotun gloss	125	100	12500	100

Customer Info

Name: mwada
Phone: 0912345678

#	Product Code	Quantitiy of Cartons

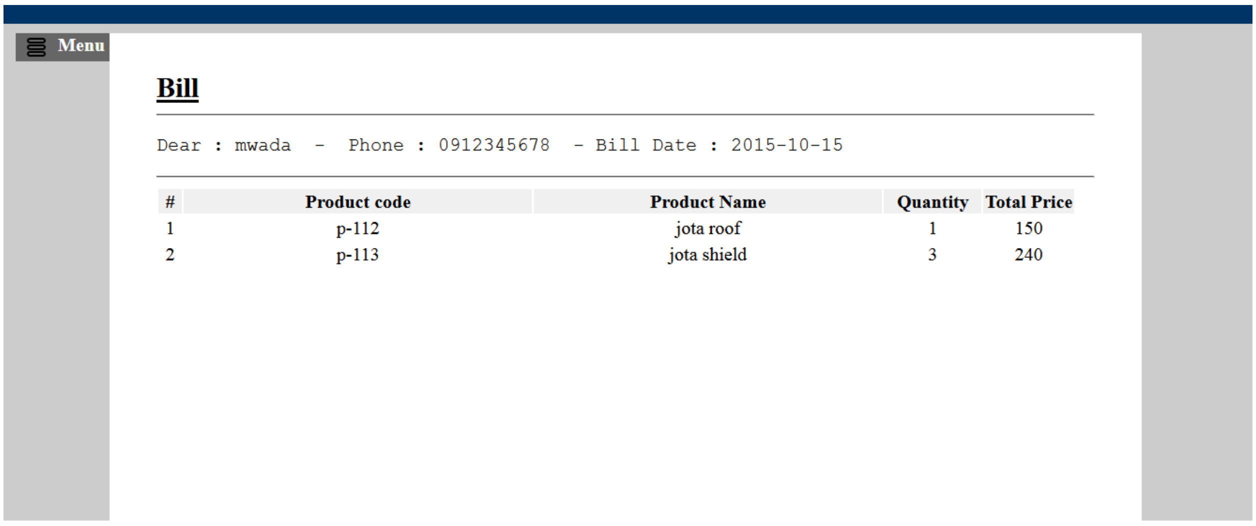
#	Product Code	Quantitiy of Cartons
	p-133	1

this product not exist..

Figure 5.6 illustrates invalid product code

5.1.5BILL

Shown in Figure5.7



Bill

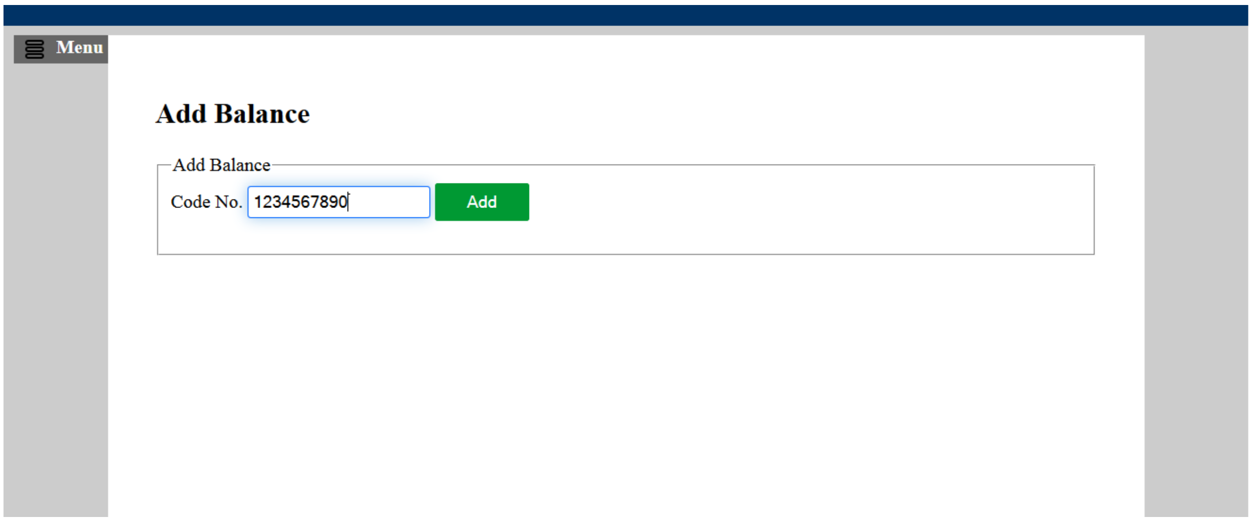
Dear : mwada - Phone : 0912345678 - Bill Date : 2015-10-15

#	Product code	Product Name	Quantity	Total Price
1	p-112	jota roof	1	150
2	p-113	jota shield	3	240

Figure 5.7 illustrates the Bill

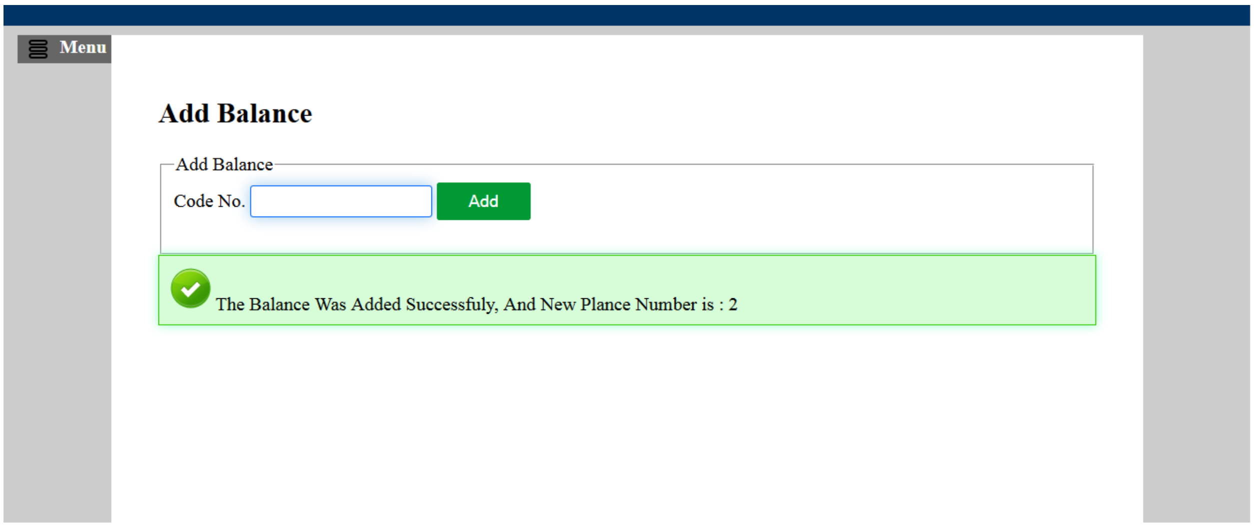
5.1.6 ADD BALANCE

The customer can add money to your account by entering the serial numbers shown in Figure 5.8 and 5.9



Code No. 1234567890

Figure 5.8 illustrates Entering the code Number

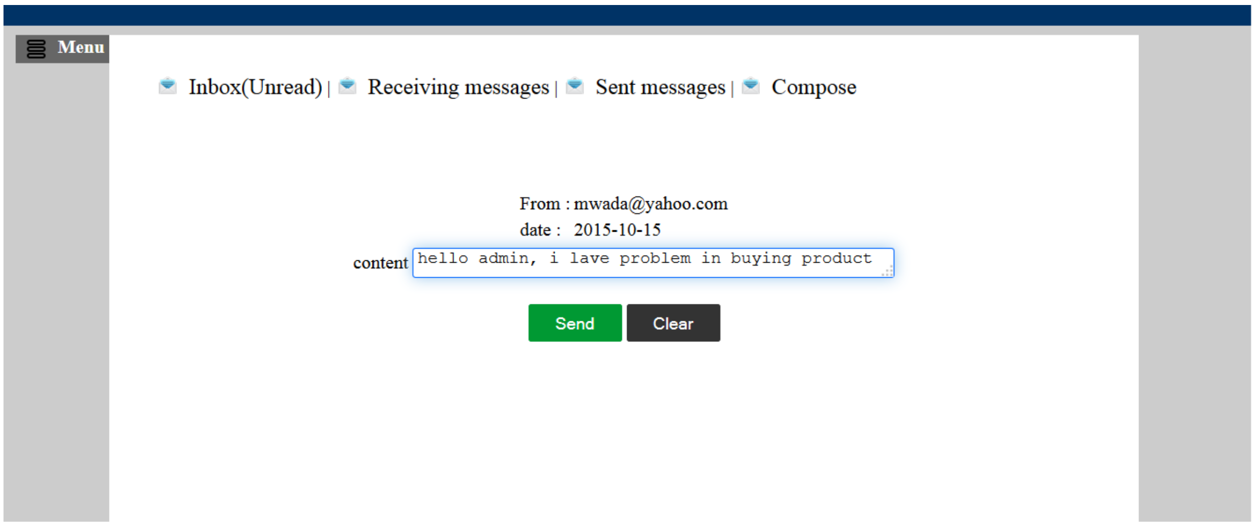


The Balance Was Added Successfuly, And New Plance Number is : 2

Figure 5.9 illustrates Respond

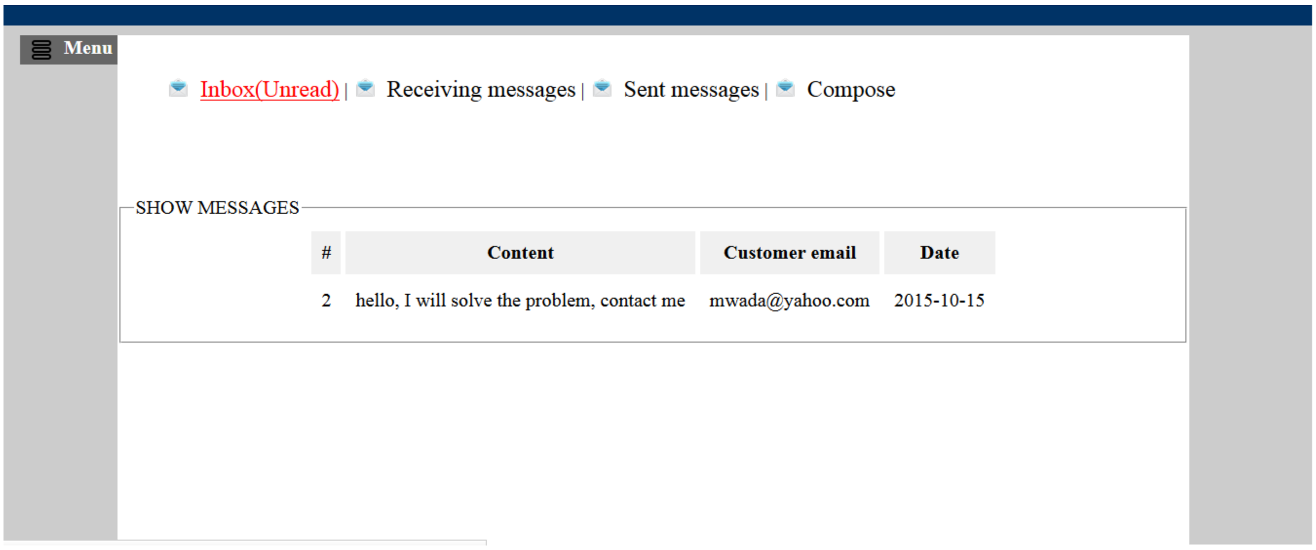
5.1.7 ADMIN MESSAGING

The customer can contact with admin by sending message to as shown in Figure 5.10 and 5.11


Menu
Inbox(Unread) | Receiving messages | Sent messages | Compose
Send
Clear

From : mwada@yahoo.com
date : 2015-10-15
content hello admin, i lave problem in buying product

5.10 illustrates the contact with Admin


Menu
Inbox(Unread) | Receiving messages | Sent messages | Compose

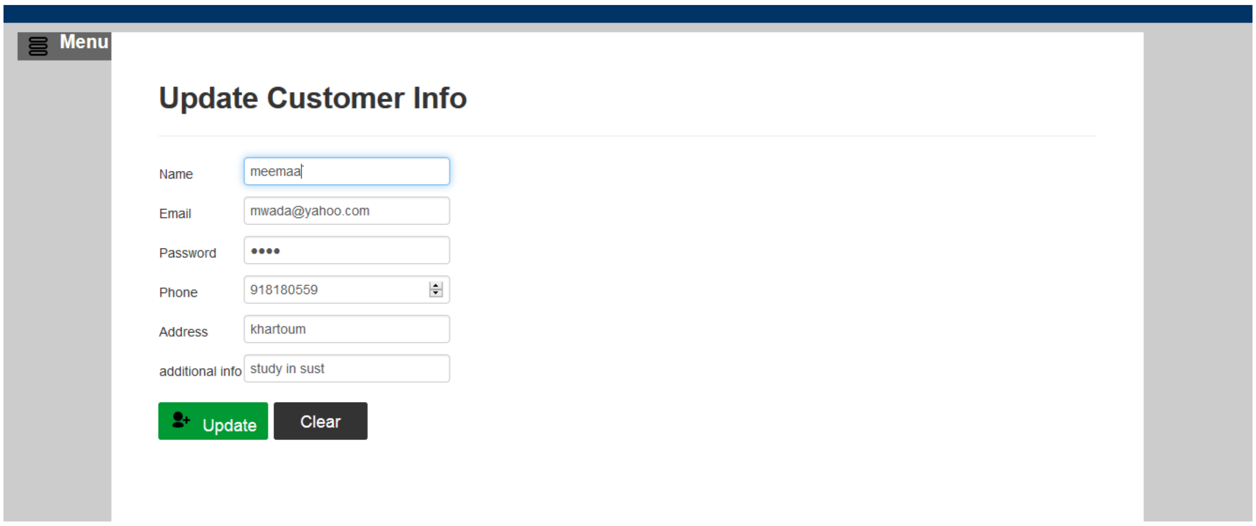
SHOW MESSAGES

#	Content	Customer email	Date
2	hello, I will solve the problem, contact me	mwada@yahoo.com	2015-10-15

5.11 illustrates receive message from admin

5.1.8 MODIFY CUSTOMER

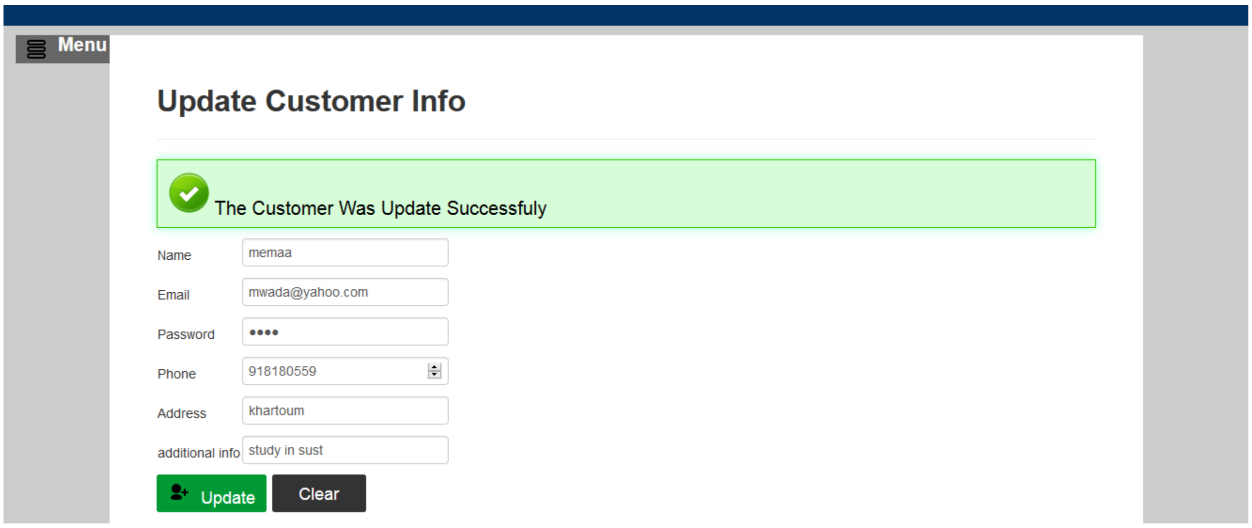
The customer can modify the personnel information as shown in Figure 5.12 and 5.13



Update Customer Info

Name	meemaa
Email	mwada@yahoo.com
Password	••••
Phone	918180559
Address	khartoum
additional info	study in sust

5.12 illustrates modifying customer info



Update Customer Info

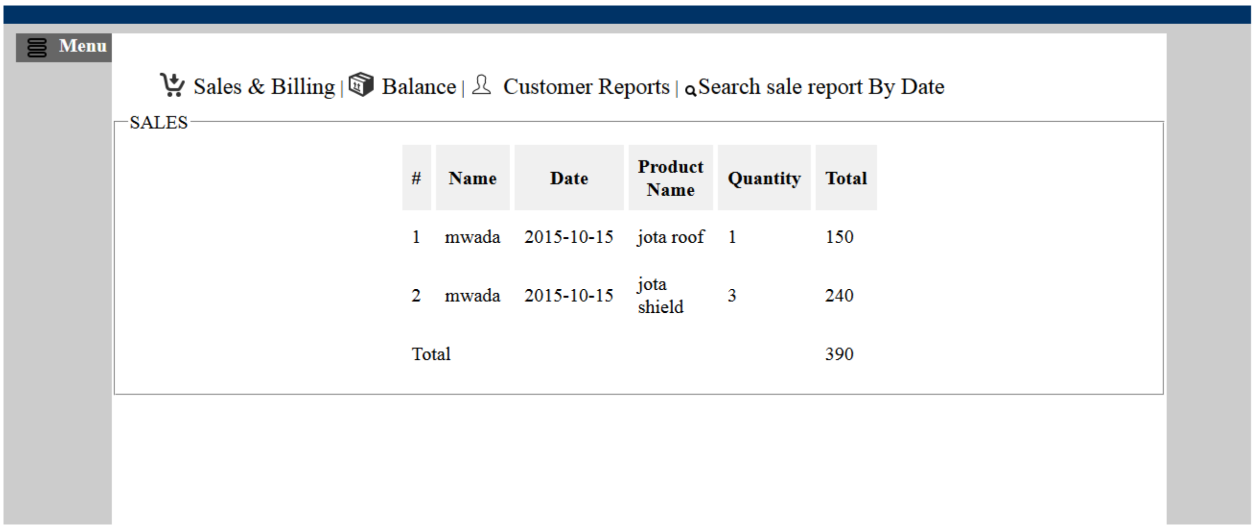
The Customer Was Update Successfuly

Name	memaa
Email	mwada@yahoo.com
Password	••••
Phone	918180559
Address	khartoum
additional info	study in sust

5.13 illustrates modifying response

5.1.9 CUSTOMER REPORTS

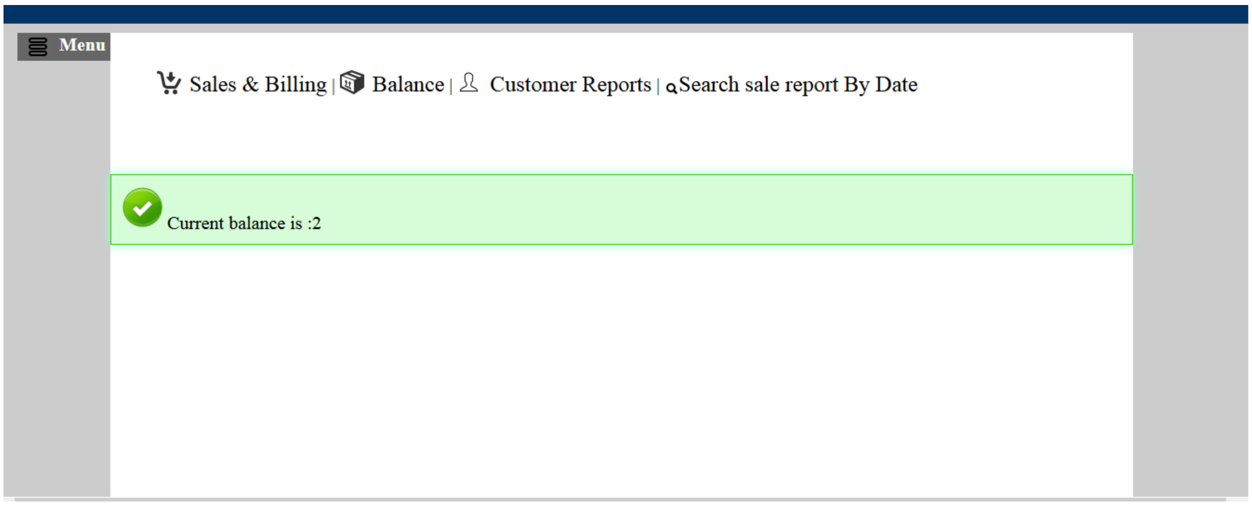
The system show the sale operation reports as shown in Figure 5.14


Menu
Sales & Billing | Balance | Customer Reports | Search sale report By Date
SALES

#	Name	Date	Product Name	Quantity	Total
1	mwada	2015-10-15	jota roof	1	150
2	mwada	2015-10-15	jota shield	3	240
Total					390

5.14 illustrates Sale report

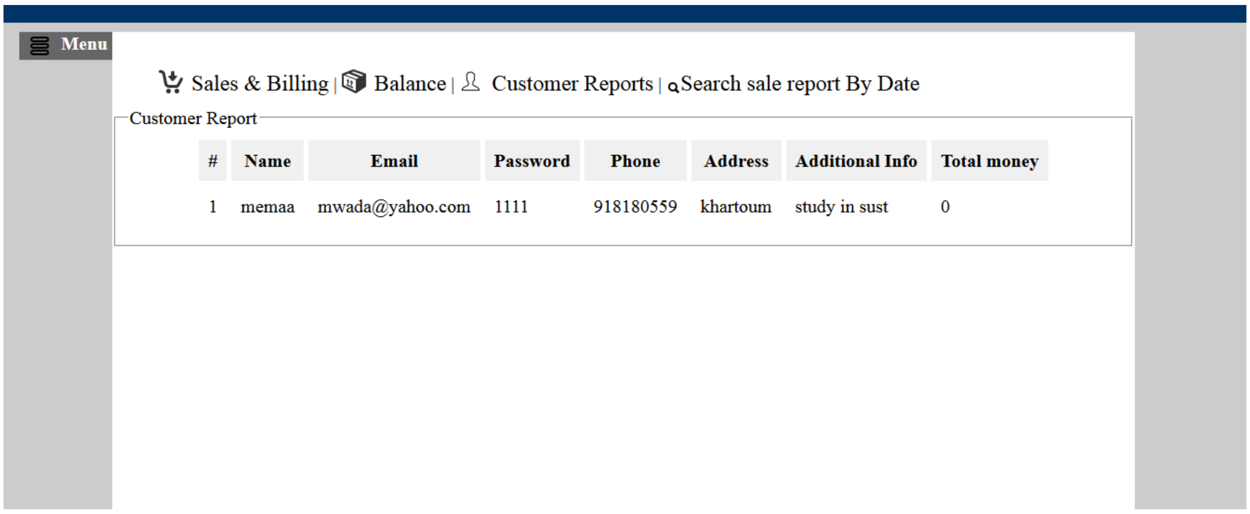
The system show the balance reports as shown in Figure 5.15


Menu
Sales & Billing | Balance | Customer Reports | Search sale report By Date

Current balance is :2

5.15 illustrates balance report

The system show the customers reports as shown in Figure 5.16

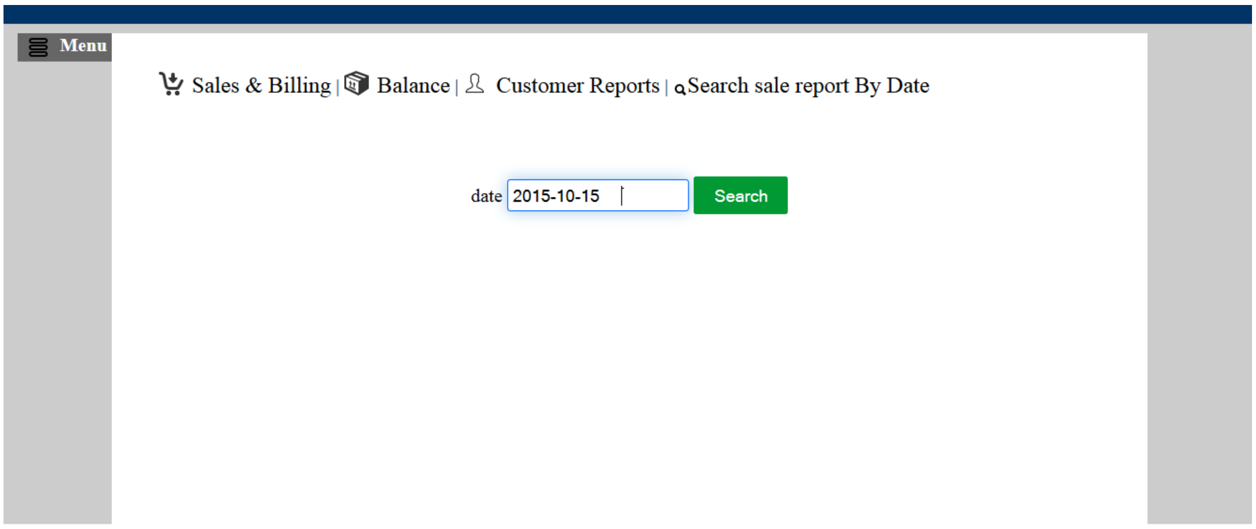

Menu
Sales & Billing | Balance | Customer Reports | Search sale report By Date

Customer Report

#	Name	Email	Password	Phone	Address	Additional Info	Total money
1	memaa	mwada@yahoo.com	1111	918180559	khartoum	study in sust	0

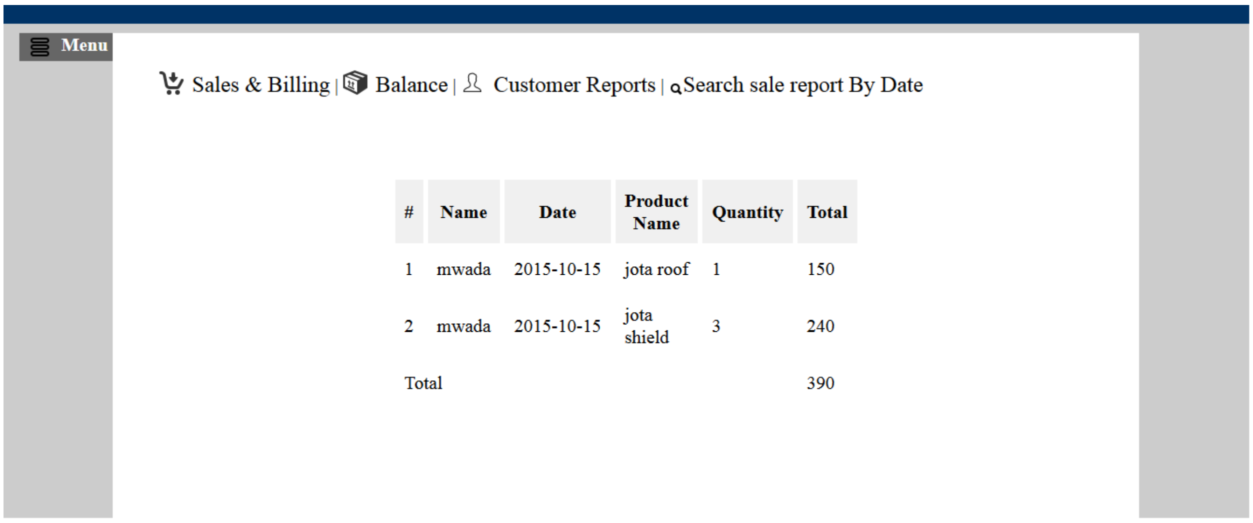
5.16 illustrates customers report

The system show search sales reports by date as shown in Figure 5.17 and 5.18


Menu
Sales & Billing | Balance | Customer Reports | Search sale report By Date
Search

date 2015-10-15

5.17 illustrates entering the date

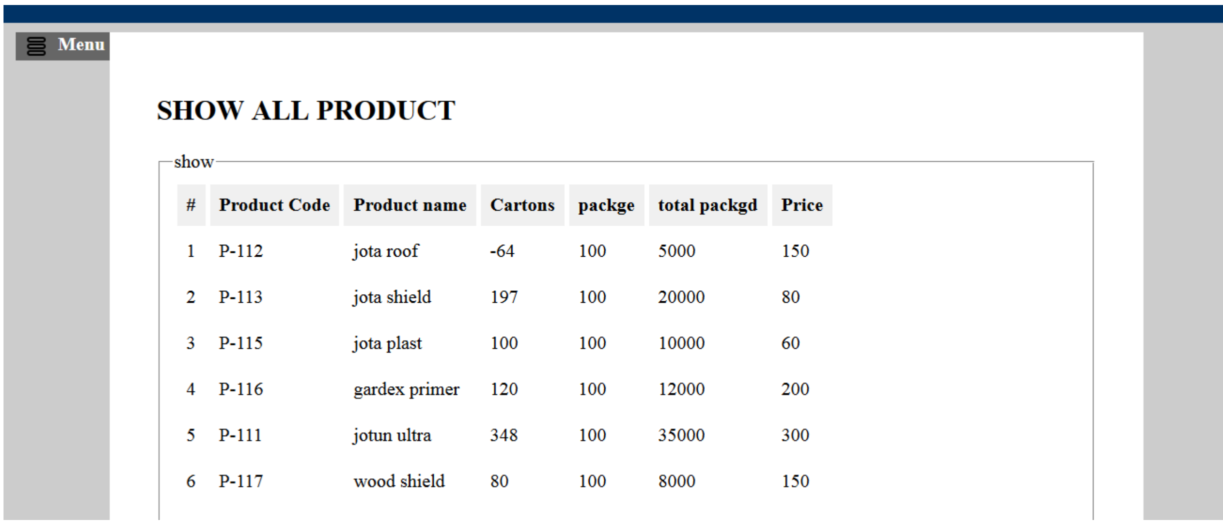

Menu
Sales & Billing | Balance | Customer Reports | Search sale report By Date

#	Name	Date	Product Name	Quantity	Total
1	mwada	2015-10-15	jota roof	1	150
2	mwada	2015-10-15	jota shield	3	240
Total					390

5.18 illustrates sale reports

5.1.10 SHOW PRODUCTS

View all the products in the stores as shown in the figure 5. 19



SHOW ALL PRODUCT

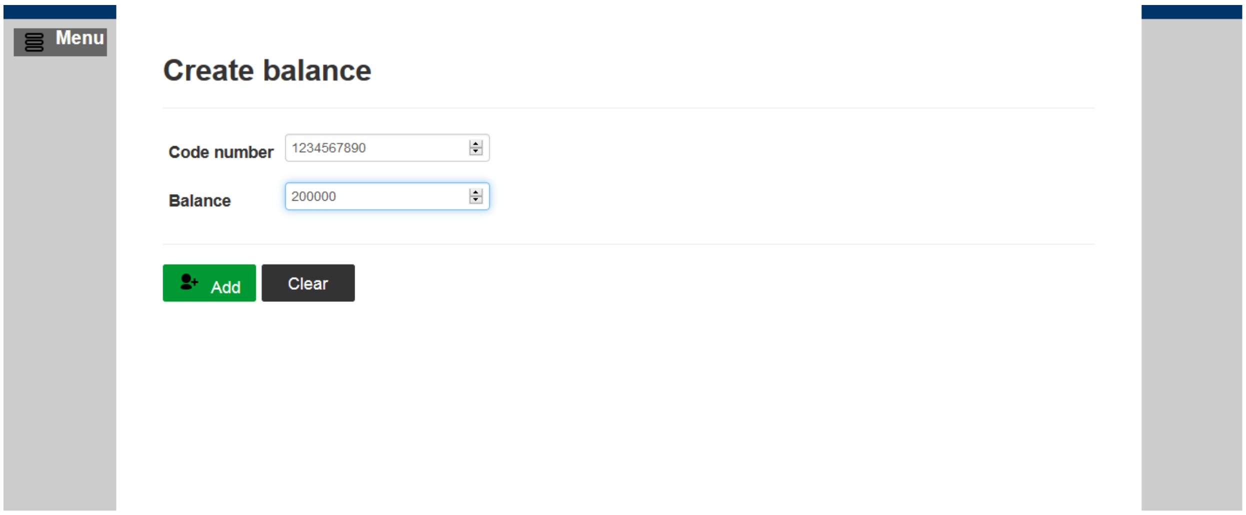
show

#	Product Code	Product name	Cartons	packge	total packgd	Price
1	P-112	jota roof	-64	100	5000	150
2	P-113	jota shield	197	100	20000	80
3	P-115	jota plast	100	100	10000	60
4	P-116	gardex primer	120	100	12000	200
5	P-111	jotun ultra	348	100	35000	300
6	P-117	wood shield	80	100	8000	150

5.19 illustrates show product in admin account

5.1.11 CREATE BALANCE

Enter the code number and balance as shown in the figure 5. 20



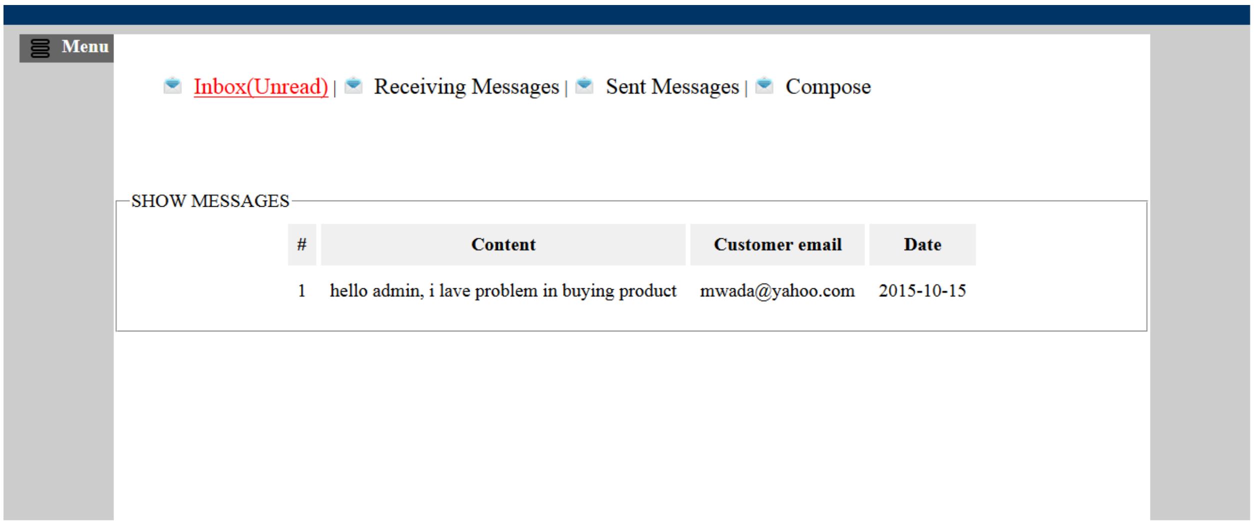
Create balance

Code number	1234567890
Balance	200000

5.20 illustrates entering code number and balance

5.1.11 MESSAGING THE CUSTOMER

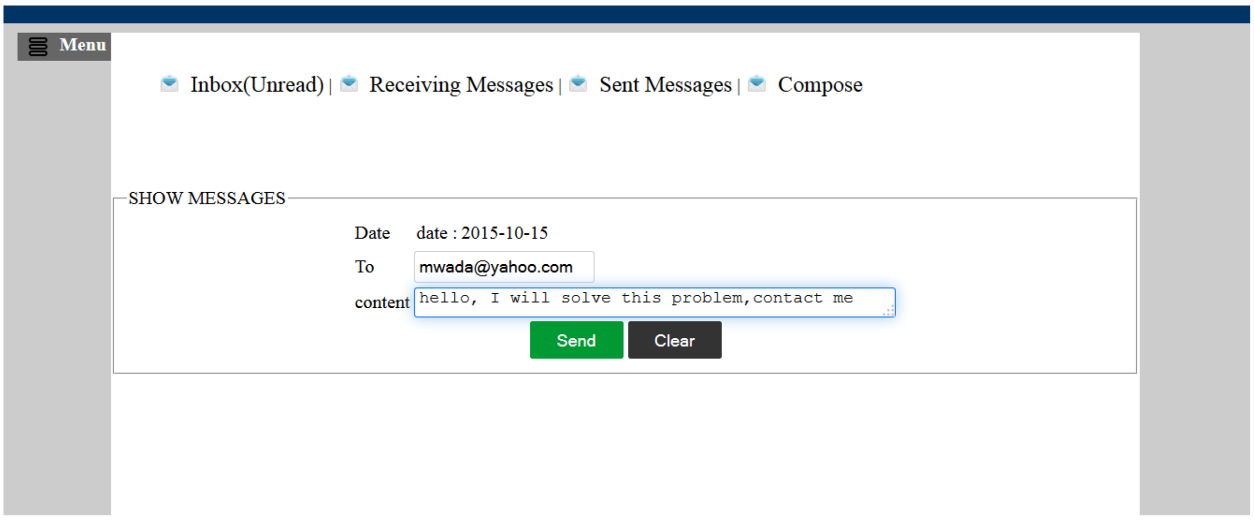
Admin contacting with customer as shown in the figure 5.21 and 5.22


Menu
Inbox(Unread) | Receiving Messages | Sent Messages | Compose

SHOW MESSAGES

#	Content	Customer email	Date
1	hello admin, i lave problem in buying product	mwada@yahoo.com	2015-10-15

5.21 illustrates Admin receiving messages


Menu
Inbox(Unread) | Receiving Messages | Sent Messages | Compose
SHOW MESSAGES
Send
Clear

Date date : 2015-10-15

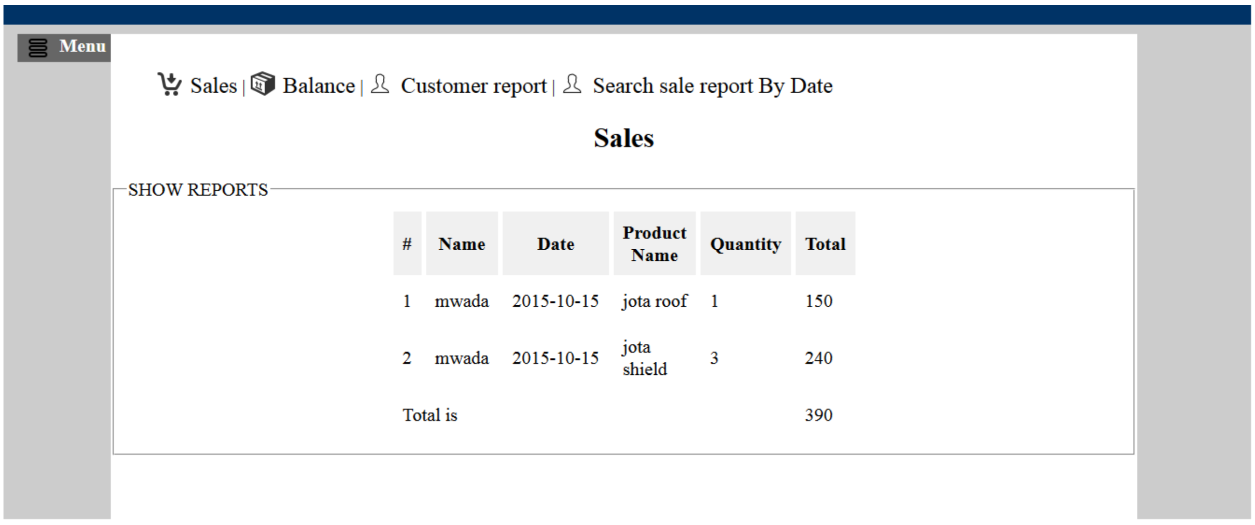
To mwada@yahoo.com

content hello, I will solve this problem,contact me

5.22 illustrates Admin replay message

5.1.12 ADMIN REPORTS

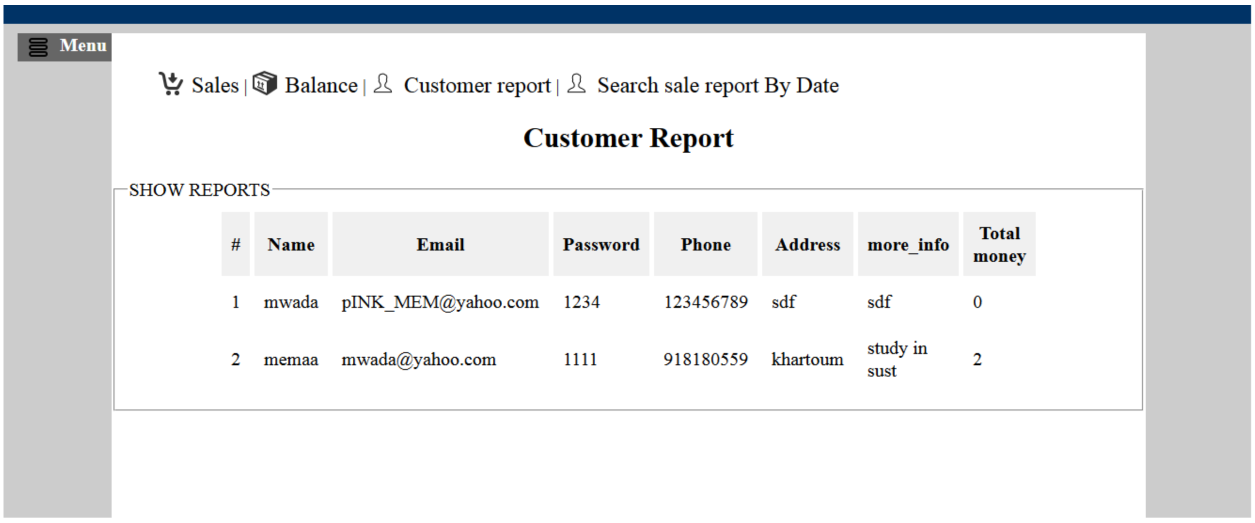
The system show the sale reports to admin as shown in Figure 5.23


Menu
Sales | Balance | Customer report | Search sale report By Date
Sales
SHOW REPORTS

#	Name	Date	Product Name	Quantity	Total
1	mwada	2015-10-15	jota roof	1	150
2	mwada	2015-10-15	jota shield	3	240
Total is					390

5.23 illustrates sale reports to admin

The system show the customer reports to admin as shown in Figure 5.24


Menu
Sales | Balance | Customer report | Search sale report By Date

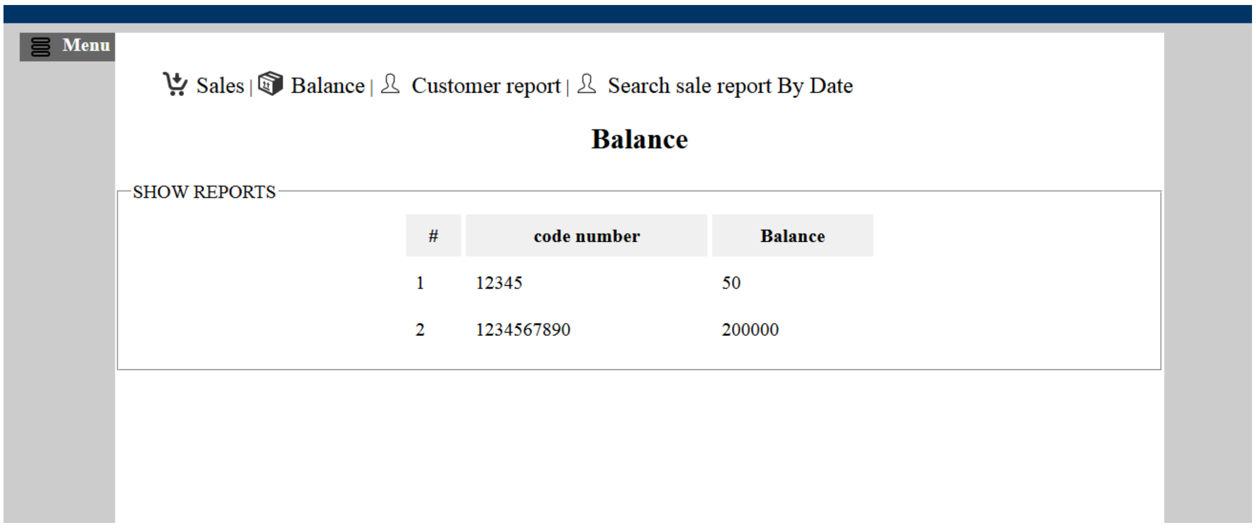
Customer Report

SHOW REPORTS

#	Name	Email	Password	Phone	Address	more_info	Total money
1	mwada	pINK_MEM@yahoo.com	1234	123456789	sdf	sdf	0
2	memaa	mwada@yahoo.com	1111	918180559	khartoum	study in sust	2

5.24 illustrates customer reports to admin

The system show the balance reports to admin as shown in Figure 5.25


Menu
Sales | Balance | Customer report | Search sale report By Date

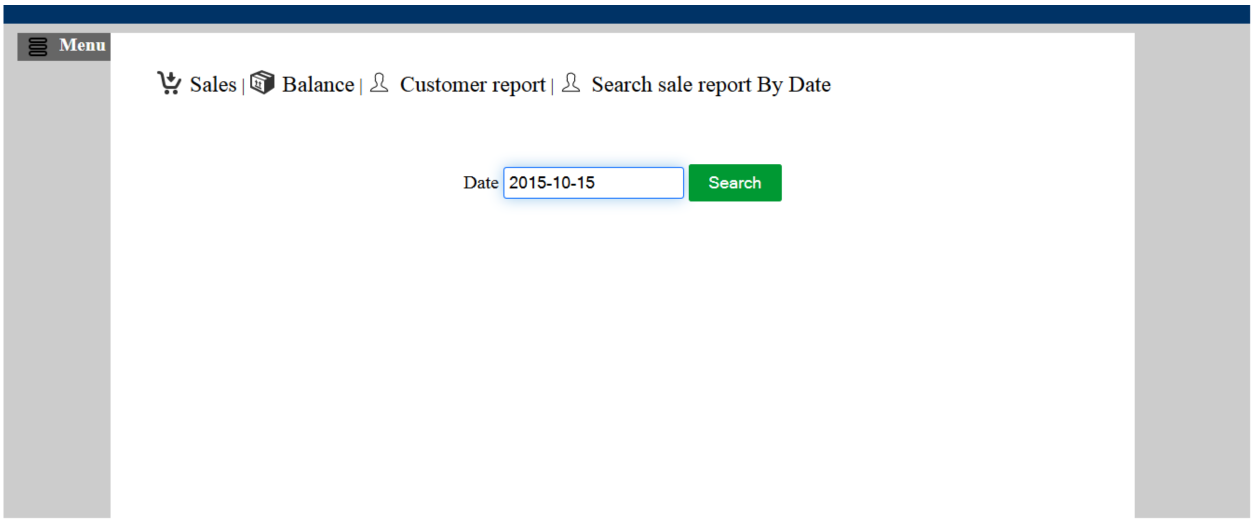
Balance

SHOW REPORTS

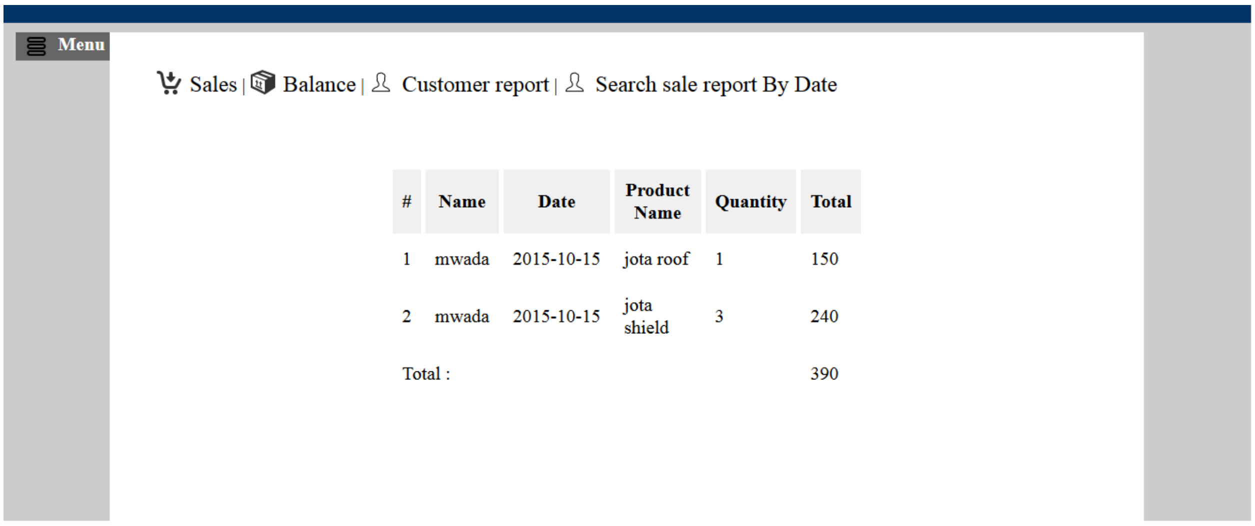
#	code number	Balance
1	12345	50
2	1234567890	200000

5.25 illustrates balance reports to admin

The system show the sale reports to admin by search as shown in Figure 5.26 and 5.27


Menu
Sales | Balance | Customer report | Search sale report By Date
Date
2015-10-15
Search

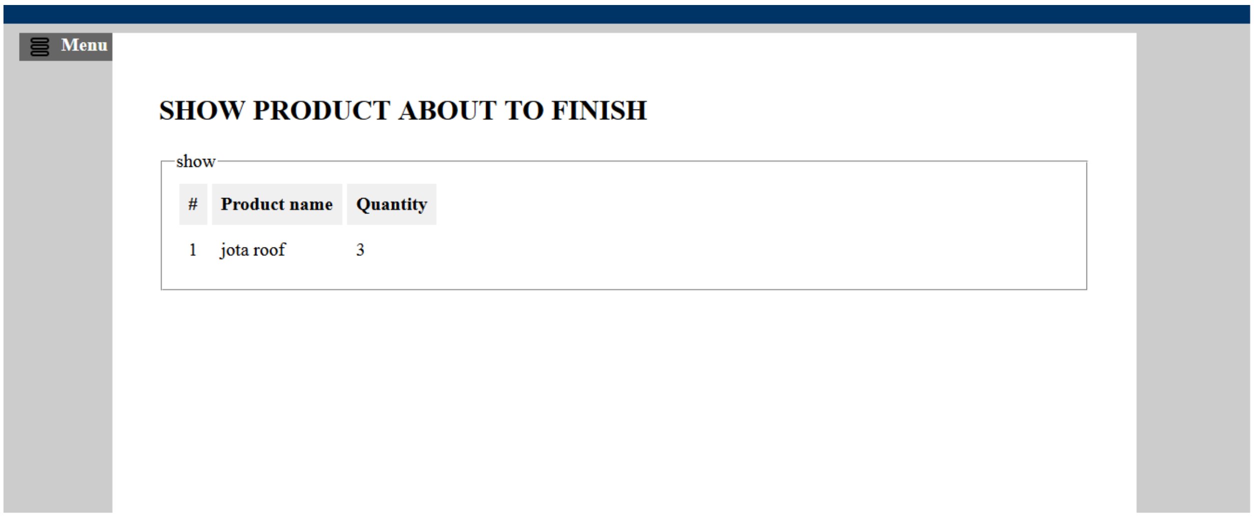
5.26 illustrates entering the date


Menu
Sales | Balance | Customer report | Search sale report By Date

#	Name	Date	Product Name	Quantity	Total
1	mwada	2015-10-15	jota roof	1	150
2	mwada	2015-10-15	jota shield	3	240
Total :					390

5.27 illustrates sale reports

The system show the product about to finishing for admin as shown in Figure 5.28



SHOW PRODUCT ABOUT TO FINISH

show

#	Product name	Quantity
1	jota roof	3

5.28 illustrates products about to finishing for admin

5.2 INTERFACES OF STORES SYSTEM

Login to the stores system by entering a user name and password as illustrated in Figure 5.29

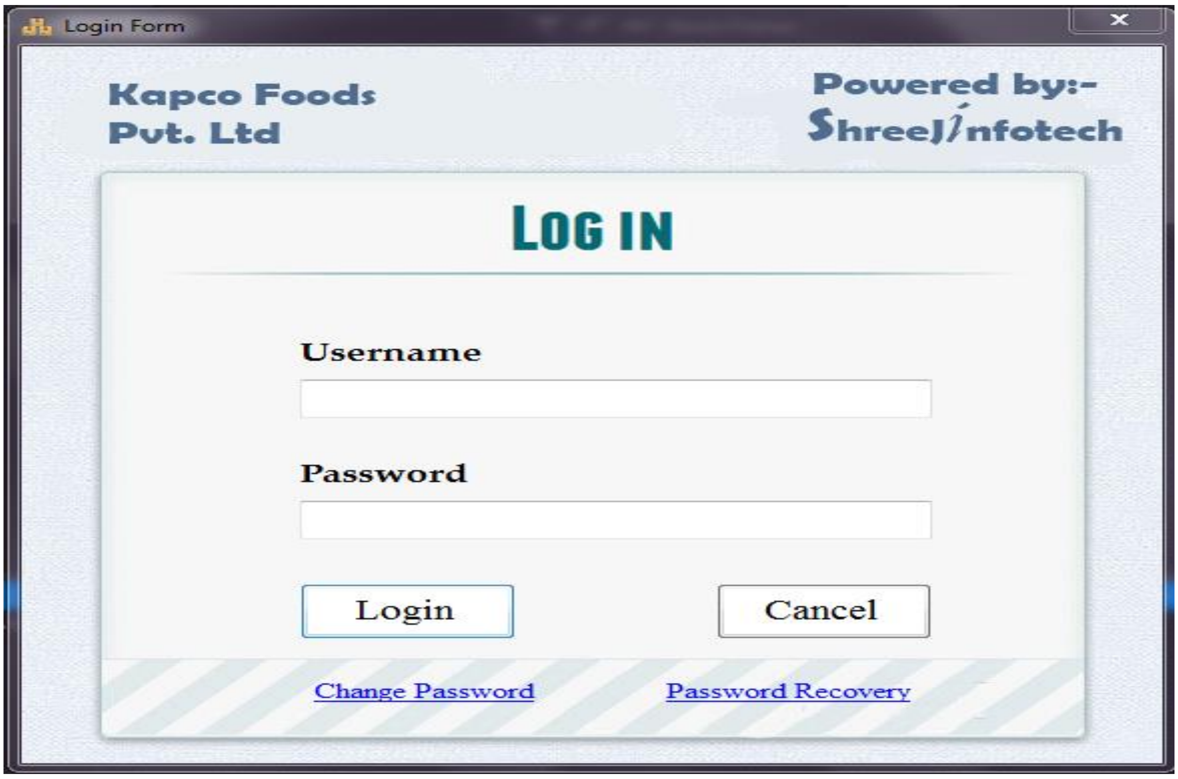
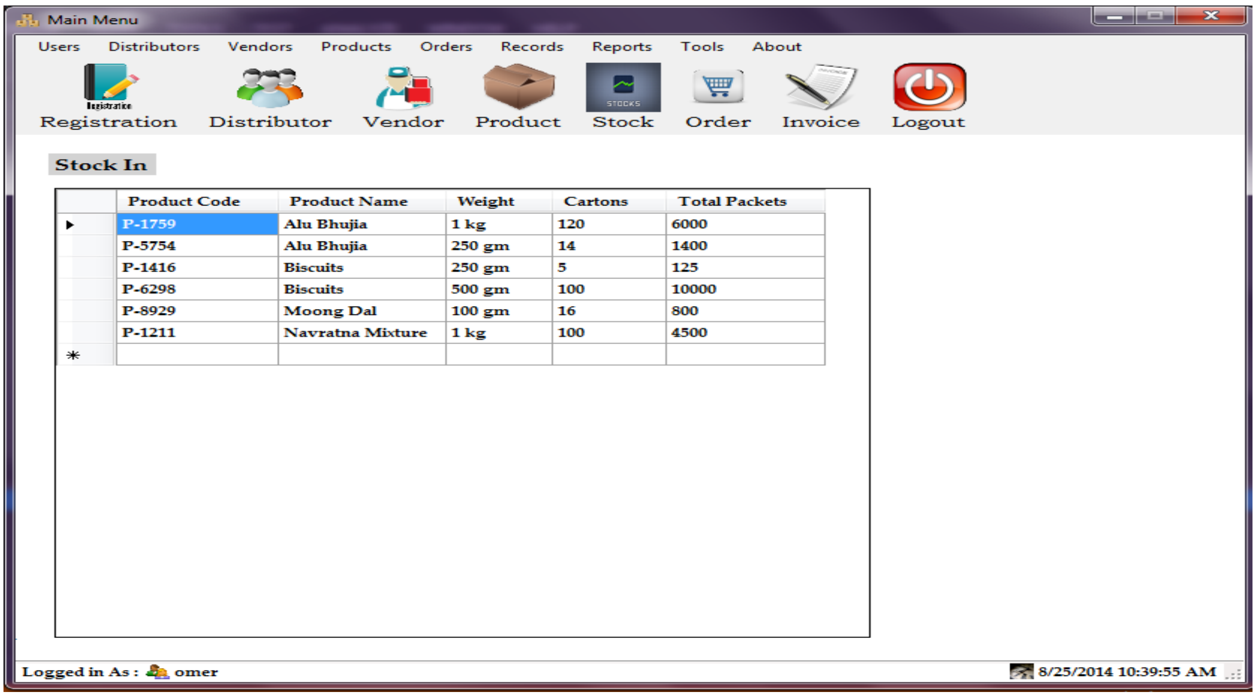


Figure 5.29

Display all the products stored in stock are illustrated in figure 5.30



Stock In

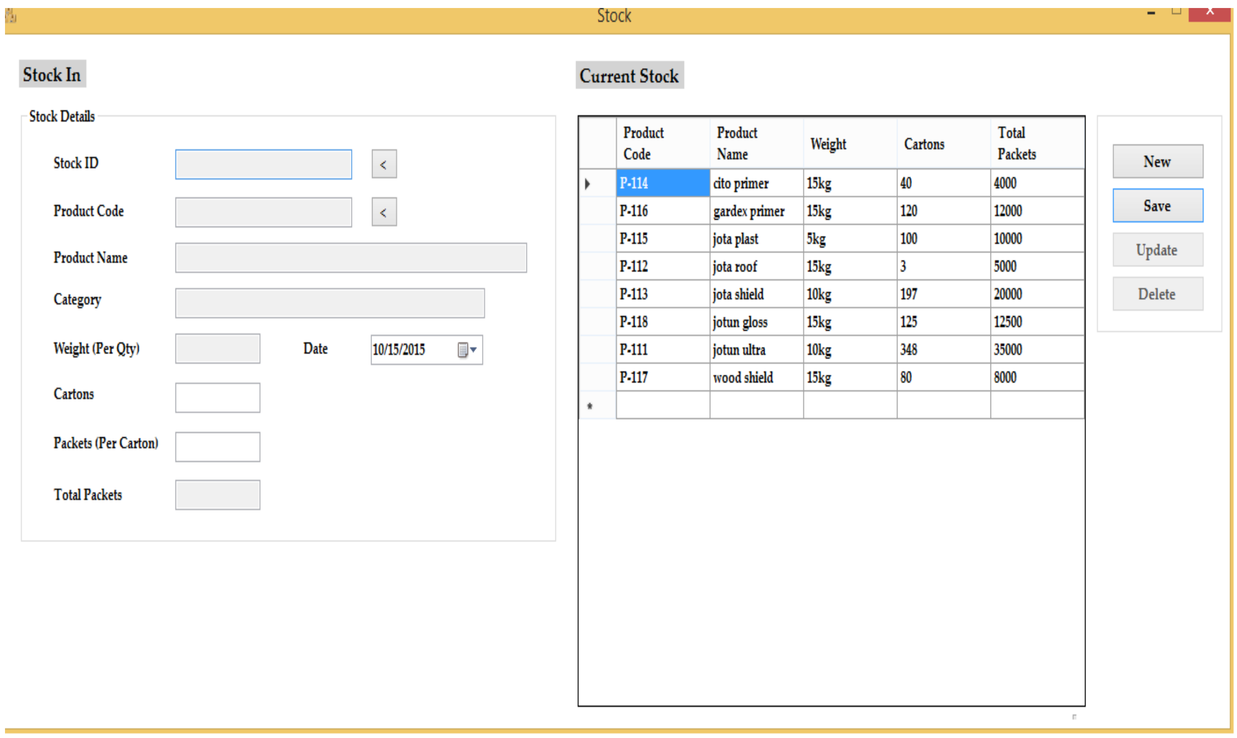
Product Code	Product Name	Weight	Cartons	Total Packets
P-1759	Alu Bhujia	1 kg	120	6000
P-5754	Alu Bhujia	250 gm	14	1400
P-1416	Biscuits	250 gm	5	125
P-6298	Biscuits	500 gm	100	10000
P-8929	Moong Dal	100 gm	16	800
P-1211	Navratna Mixture	1 kg	100	4500

Logged in As : omer

8/25/2014 10:39:55 AM

Figure 5.30

Add, delete or edit the data in stock, as illustrated in Figure 5.31



Stock In

Stock Details

Stock ID	
Product Code	
Product Name	
Category	
Weight (Per Qty)	
Date	10/15/2015
Cartons	
Packets (Per Carton)	
Total Packets	

Current Stock

Product Code	Product Name	Weight	Cartons	Total Packets
P-114	cito primer	15kg	40	4000
P-116	gardex primer	15kg	120	12000
P-115	jota plast	5kg	100	10000
P-112	jota roof	15kg	3	5000
P-113	jota shield	10kg	197	20000
P-118	jotun gloss	15kg	125	12500
P-111	jotun ultra	10kg	348	35000
P-117	wood shield	15kg	80	8000

Figure 5.31

CHAPTER SIX

SYSTEM QUALITY

INTRODUCTION

this section addresses the quality factors, quality standards applied to the system, and the system testing.

6.1QUALITY FACTORS:

This section contain the quality factor that we applied , we are also taking into account quality management to modify the integrated system based on the model of quality used in the old system and modify the quality model based on the type of requirement relating to the integrated system after modifying .

6.1.1 EFFICIENCY:

To make this system an efficient, make send values individually from the database rather than send (string) complete.

6.1.2 MAINTENANCE:

The application of the standards in the code writing and the names of the function variables were significant and meaning.
As well as annotations that is facilitate the understanding of the work of maintenance teams.

6.1.3 INTEROPERABILITY AND PORTABILITY:

So the system is suitable for all environments, (xml) which have compatibility with most other languages is used.

6.1.4 TESTABILITY:

The adaptation of the code to allow them to apply the tests were writing functions allow the introduction of test cases and results output which facilitates the testing process.[10]

THE QUALITY FACTORS ADDED AFTER THE MAINTENANCE:

6.1.5 SECURITY:

The integrity of the system is preserved by ensuring that only authorized personnel have access to the system and only authorized changes are implemented, MR(3) and MR(4)achieve this factor.[9]

6.1.6 USABILITY:

System interface is designed in which user can interact with it, and the goal of this interaction is the easy of recognize and understood, and it also aids the user's decision for making the process.

Generally the interface design to user in MR (8) provide the suitable input to achieve the desired output, also MR (3) is correspond to this factor.

6.2 QUALITY FACTOR METRICS:

There are many quality metrics that must apply to the systems, but here we applied some of the quality metrics that measure our quality factors

Such as complexity, and the ability to understand and the maintenance, as well as measure of size, to make sure that the maintenance process achieve the quality model before modifying as well enhanced quality factors.

6.2.1 LINES OF CODE (LOC):

- **PHYSICAL LINES OF CODE**
  The lines of code calculate the total number of executable lines, including the comment line.[8]

6.2.1.1 SALES CODE:

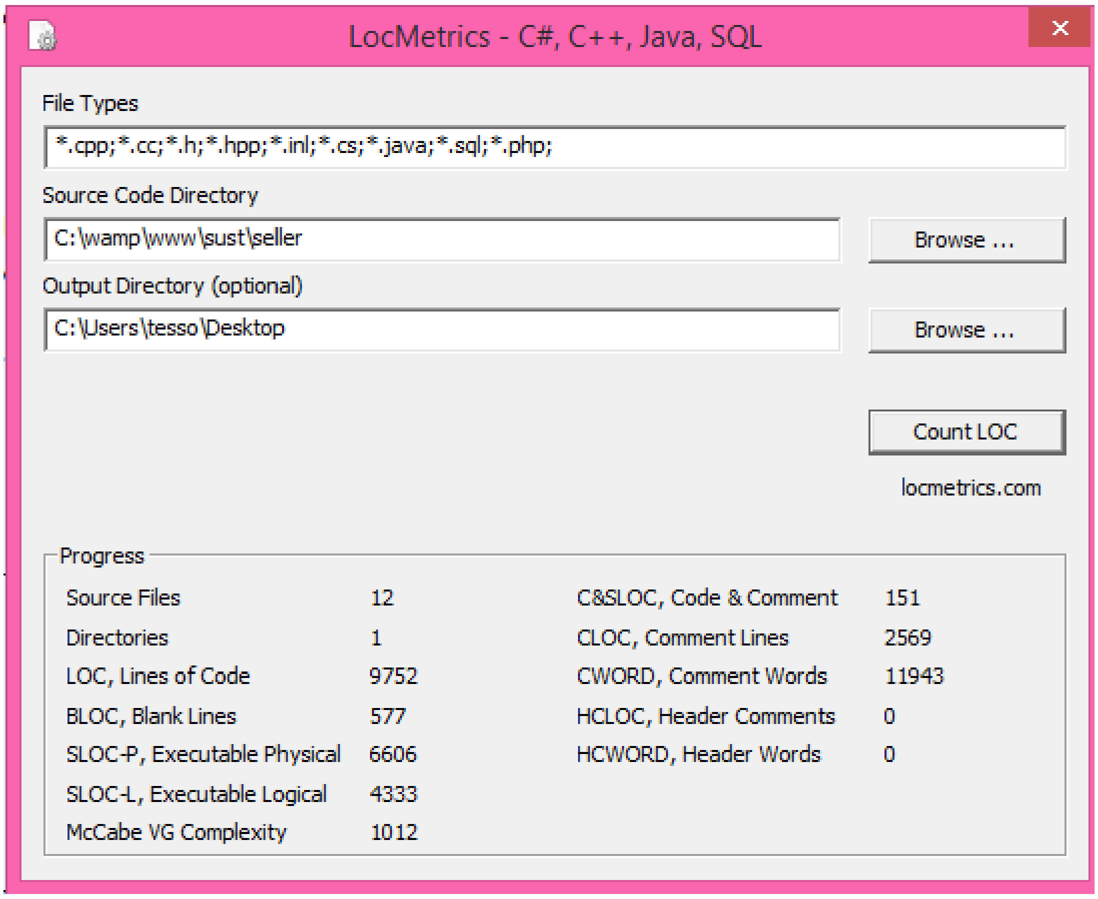
Table (6.1) shows the sales line of code

Class Name	**Lines Of Code**
Admin_report	300
Admin_message	320
Admin_account	216
Reports	266
Login	100
Customer_message	283
Customer_account	180
Create_balance	120
Bill	140
Add_balance	112

- **LOGICAL LINES OF CODE**

  Logical Lines of Code is the number of programming language statements also called Effective Lines of Code. Logical lines of code measured by tools called locMetrics it is a software metrics used to measure the size of the program by counting the number of executable lines in the text of the program's source code.

  Figure show shows sales code locMetrics



File Types: *.cpp;*.cc;*.h;*.hpp;*.inl;*.cs;*.java;*.sql;*.php;

Source Code Directory: C:\wamp\www\sust\seller

Output Directory (optional): C:\Users\tesso\Desktop

Progress

Source Files	12	C&SLOC, Code & Comment	151
Directories	1	CLOC, Comment Lines	2569
LOC, Lines of Code	9752	CWORD, Comment Words	11943
BLOC, Blank Lines	577	HCLOC, Header Comments	0
SLOC-P, Executable Physical	6606	HCWORD, Header Words	0
SLOC-L, Executable Logical	4333		
McCabe VG Complexity	1012		

Figure 6.1 locMetrics

6.2.1.2 WEB SERVER CODE:

Table (6.2) show the Web server line of code

Number of method	**Method Name**	**Lines of method**
1	counter_colums	20
2	show_productcode	26
3	find_productcode	15
4	Update	15
5	login_user	24
6	show_stock	34
7	Show_alert	20

6.2.2 COMMENTPERCENTAGE:

The number of lines that have comments divided on the number of lines that is free. Whenever the percentage increase in this measure, it is increases the permeability of understanding as well as maintainability.
Comment Percentage= number of comments /number of lines. [8]

6.2.2.1 VISUAL BASIC CODE PERCENTAGE:

- Number of comments=20
- The comment Percentage=10%

6.2.2.2 PHP CODE PERCENTAGE:

Table (6.3) present applying of CP on PHP code

Item Name	Number of lines	Number of comments	Comment percentage
Customer_account	216	5	2.3%
Bill	140	12	8.3%
Login	100	5	5%
Reports	266	8	3%
Create balance	120	10	8.3%
Admin_acount	160	12	7.5%
Admin_report	300	20	6.6%
Admin_message	230	18	7.8%
Customer_message	230	18	7.8%

6.2.3 FAN-IN FAN-OUT COMPLEXITY:

Functions with a large Fan-out are more expensive to maintain, Function with high Fan-in mean that the function is implementing a number of functionalities.

DIFINITIONS

FAN IN:

Fan in for certain function is the number of functions that call this function.

FAN OUT:

Fan out for certain function is number of functions, this function calls. [13]

Formula

Complexity = Length$\times$ $(\mathbf{Fan_{in} x\ Fan_{out}})^2$. [13]

Table (6.4) applying fan in fan out for visual basic code:

Function name	**Fan in**	**Fan out**	**Fan-In Fan-Out Complexity**
Counter columns	2	2	$(2*2)^2*20=320$
Show product code	1	1	$(1*1)^2*26=26$
Find product code	1	1	$(1*1)^2*15=15$
Change quant	1	2	$(1*2)^2*15=60$
Login user	1	1	$(1*1)^2*24=24$
Show stock	2	2	$(2*2)^2*34=544$
Show alert	1	1	$(1*1)^2*20=20$
Find product quantity	1	1	$(1*1)^2*15=15$

6.3 CYCLOMATIC COMPLEXITY (CC):

Used to evaluate complexity of the algorithm in a function, which is calculation of the required number of test case stoutest fully function.

The formula is: **“connections - nodes + 2”.**

If the value is small, this is better, because it means reducing testing and increase understanding. [8]

6.3.1 CYCLOMATIC COMPLEXITY FOR PHP CODE

This figure present graph of show_alert method

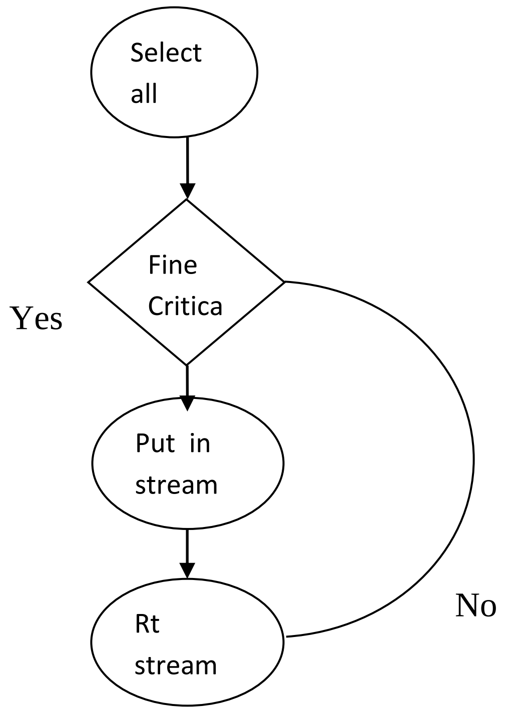


Figure 6.2 show alert method

This figure present the graph of change _quantity method

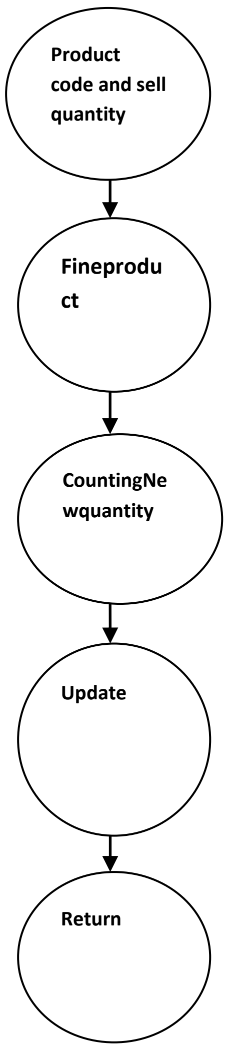


Figure 6.3 change quantity method

This figure shows the graph of find product code method

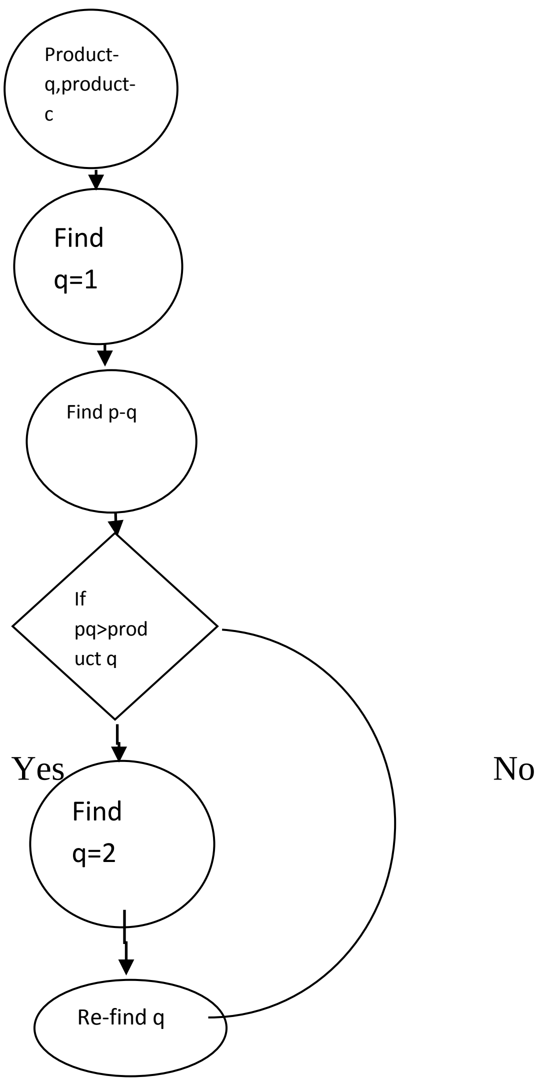


Figure 6.4 find product code method

This figure show the graph of find product quantity method

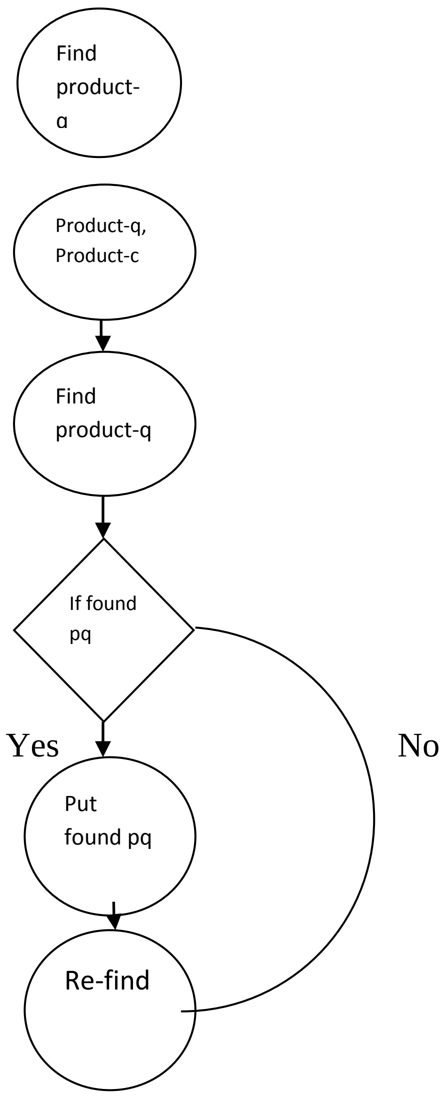


Figure 6.5 find product quantity method

Table (6.5) illustrates the implementation of (CC) measure on visual basic code

Method	**Number**	**Number of node**	**Complexity**
show alert()	4	4	4-4+2=2
Change qunt()	4	5	4-5+2=1
Find product Code()	5	6	5-6+2=1
Find product Quantity()	4	5	4-4+2=2

6.2.4 LENGTH OF IDENTIFIERS

This is a measure of the average length of identifiers (names of variables and functions) which is present in the program, by count the number of characters variables or functions divided by the number of variables or functions, If the length of identifiers is long then it indicates that the identifier name is meaningful and meaningful identifiers enhance understandability of the code.

Formula:

Average length of identifier = total number of characters/number of identifiers.

Average length of functions identifier = 68/6 = 11.1

Average length of variables identifier = 110/21 = 5.2

6.2.5 DEPTH OF CONDITIONAL NESTING

This measure represents how deep the nested if-statements in a program, by count the number of nested if-statements levels divided by number of nested if-statements, if the nested if-statements are very deep then program becomes hard to understand and it becomes error-prone.

Formula:

Average depth of conditional nesting = sum of depth levels / number of if-condition

Average = 24/ 16 = 1.5

.

6.2.5 SECURITY METRICS:

A set of security requirements (SR) identified from risk management is used to describe the security goals of the software systems. These security requirements are the basis for measuring security.

Goal-Question-Metrics (GQM) involves a set of goals that generate questions that define those goals as completely as possible by quantifying them. Specifying measures answers the questions to identify whether the goal can be achieved or not.

Metrics are given (on basis of full, average, weak compliance) a value to the question.

Goal= Ensure Identification, authentication and authorization User, organization (SR).

TABLE 6.6 METRICS FOR IDENTIFICATION, AUTHENTICATION AND AUTHORIZATION REQUIREMENTS

Question (Q)=SR.Q	**Metrics(M)=SR.M**
How are the users identified?	Subjective evaluation by how all level user uniquely identified for using the system
How are the users authenticated?	Name, password
How is authentication session managed?	No limited time
No of fail attempt to lock user account?	No such fail login attempt
How grant user authorization?	role based
How many resource and level of use grant for single user?	only level of use defined
How many layers of authentication / authorization check?	All layers

Scores from metrics for every question are:

SR.Q1= 1 (Full compliance: because every customer has unique id and user name).
SR.Q2= 1 (Full compliance: because every customer has unique name and password that protect the system from unauthorized users).
SR.Q3= 0 (weak compliance: because the sessions have no limited time).
SR.Q4= 0 (weak compliance: because the system has no certain number for login attempt).
SR.Q5= 1 (Full compliance: role based (authorized customer or admin)).
SR.Q6=0.5 (average compliance, only level of use defined).
SR.Q7= 1 (Full compliance, because the authentication/ authorization check per usage).

Total score =100 * (1+1+0+0+1+0.5+1)/7= 64%.

So identification, authentication and authorization requirements can achieve 64% from the system. [14]

6.2.6 USABILITY METRICS:

Internal usability metrics are used for predicting the extent to which the software in question can be understood, learned, and operated, attractive and compliant with usability regulations and guidelines. [7]

Table (6.7) show the usability metrics. [7]

Criteria	**Matrices Description**	**Matrices**	**Measured value**
understandability	Function understandability	understandability = A/B A=number of output data items which user successfully get. B= number of output data items available from the interface.	14/15 =0.9
Learnability	Easy to learn	Average time taken to learn to use a function correctly.	5m
Operability	Input undo-ability	Operability= A/B A= number of input errors which the successfully correct. B=number of attempts to correct input errors.	3/15=0.2
Attractiveness	Interface appearance customizability	Attractiveness=A/B A=number of turns which user failed to select input /output expression B=number of turns which user tried to select input /output expression	½=0.5

6.2.8 FUNCTION POINT

Function Point Analysis is a structured technique for solving system's problem. It is a method to divide systems into smaller components, so they can be better understood and analyzed, its unit measure for software.

The function point method measures the software size on the basis of well-defined functional characteristics of the software system. Its represent:

1-Data that is entering a system (external input) like logical transaction output or system feeds.

2-Data that is leaving the system (external output) such as online displays and report.

3-Data stored within the system like user defined data.

4- External interfaces like interfaces of other systems.

BENEFIT OF FUNCTION POINT ANALYSIS

-Gives the precise size of the software function**.**

-Can be counted by different people at different times, and get the same scales of errors.

- It helps to connect the sizing information to the user or client.

- Used to determine whether a tool, a language, an environment, is more productive when compared with others. [4]

FUNCTION POINT EXAMPLES

- Application development with function point such as considering the cost needed to develop application in some system.

- Comparison of tow systems.

- Implementing of new functionalities with function point such as knowing how count the cost needed to introduce new functions in certain system.

CHAPTER SEVEN

RESULTS AND RECOMMENDATIONS

INTRODUCTION:

This section discusses the most important results that we have reached after implementation of the system**.**

7.1 RESULTS:

The model was tested based on the existing quality factors, and the results (quality metrics, testing) met the quality standards after modifying and development of the system**.**

7.2 RECOMMENDATIONS:

- Adding another feature for the system security.
- Apply the function point analysis on the system.
- Apply more quality metrics.
- Make the system applicable in multiple platform (mobile).
- Increase the payment ways.
- Create Positioning Service (GPS) to locate the nearest branch of the customer.
- It will be great if the system is adopted by huge company.

7.3 CONCLUSION:

We have successfully accomplished this research, which helps in the quality control of the integrated system after the maintenance process and applying the test strategy, and the quality metrics for the whole system, taking into account that the quality of the system met our expectation.

A LIST OREFERENCES AND SOURCES

BOOKS

[1] Grubb, Penny, and Armstrong A. Takang. *Software maintenance: concepts and practice*. World Scientific, 2003.

[2] Ronan Fitzpatrick- Software Quality: Definitions and Strategic Issues - 1996

[3] Daniel Galin -Software Quality Assurance (From theory to implementation) - first publish 2004

[4] A Study of Software Metrics-

1 Assistant Professor, JIET Jind. gurujangra@gmail.com

2 Professor, Dept. of CSE, Ch. Devi Lal University Sirsa

3 Professor, Dept. of CSE, Ch. Devi Lal University Sirsa - January 2011

[5] ISO/IEC 14764 IEEE Std 14764- Software Engineering -Software Life Cycle Processes Maintenance-2006- IEEE 2006.

[6] IEEE Standard for Software Maintenance- Software Engineering Standards Committee of the IEEE Computer Society- Approved 25 June 1998.

[7] ISO/IEC 9126-3-Software engineering –Product quality – Part 3: Internal metrics – 2002.

[8] Mozamil jah -Software Metrics-Usability and Evaluation of Software Quality -university west-department of technology, mathematics and computer science, S-46186 Trollhattan, Swedden - June 2008.

[9] -Brian Chess - Metrics That Matter: Quantifying Software Security Risk - Fortify Software.

[10] Ronan Fitzpatrick- Software Quality: Definitions and Strategic Issues- Staffordshire University, School of Computing Report- April 1996.

WEBSITES

[11]UML-wikipedia-https://en.wikipedia.org/wiki/Unified_Modeling_Language-

Time 8:45pm **Date** 3-8-2015.

[12] The free encyclopedia - en.wikipedia.org.-

Time 12:03pm **Date** 15-8-2015.

[13] http://www.aivosto.com/project/help/pm-sf.html

Time 4:56pm **Date** 24-10-2015.

APPENDIX I:

This appendix shows test cases used to test the system to make sure the system to objective and anticipation.

LOGIN METHOD TESTING:

Test cases	**Input**	**Expected result**	**Actual result**	**Test result**
Test case(1)	mwada@yahoo.com,1111	Admin page	Admin page	Accept
Test case(2)	Teso1573@gmail.com,1234	Customer page	Customer page	Accept
Test case(3)	mwada@yahoo.com,1112	Caption error message	Caption error message	Accept
Test case(4)	Teso1573@gmail.com,1233	Caption error message	Caption error message	Accept
Test case(5)	mwda@yahoo.com ,1111	Caption error message	Caption error message	Accept
Test case(6)	Teso1577@gmail.com,1234	Caption error message	Caption error message	Accept

TESTPARTICIPATION IN THESYSTEM FUNCTION:

Test cases	**Input**	**Expected result**	**Actual result**	**Test result**
Test case(1)	Esra ,099387377, esra@gmail.com,omdor,female	Subscribed	Subscribed	Accept
Test case(2)	Esra ,22hhhh,esra@gmail.com ,omdor,female	Captionerrormessage	Captionerrormessage	Accept
Test case(3)	Esra ,099387377, esra.com,omdor,female	Captionerrormessage	Captionerrormessage	Accept
Test case(4)	Esra ,099387377, esra@gmail.com,female	Captionerrormessage	Captionerrormessage	Accept
Test case(5)	1111 ,099387377,esra@gmail.com, omdor,female	Captionerrormessage	Captionerrormessage	Accept

DIRECT SELL FUNCTION TESTING:

Test cases	Input	Expected result	Actual result	Test result
Test case(1)	Esra ,099387377,p-1211,7	Added to the list of purchases	Added to the list of purchases	Accept
Test case(2)	p-1211,7	Caption error message	Caption error message	Accept
Test case(3)	099387377,p-1211,7	Caption error message	Caption error message	Accept
Test case(4)	Esra ,099387377,p-1211,150	Caption error message	Caption error message	Accept
Test case(5)	Esra ,099387377,1211,7	Caption error message	Caption error message	Accept
Test case(6)	Esra ,0998377,1211,7	Caption error message	Caption error message	Accept
Test case(7)	Esra ,099387377,p1211,100	Caption error message	Caption error message	Accept
Test case(8)	Esra ,099387377,p-1211	Caption error message	Caption error message	Accept

ADD BALANCE FUNCTION TESTING

Test cases	Input	Expected result	Actual result	Test result
Test case(1)	11223341	Adding balance to the customer account	Adding balance to the customer account	Accept
Test case(2)	678338	Caption error message	Caption error message	Accept
Test case(3)	empty	Caption error message	Caption error message	Accept

MODIFY CUSTOMER FUNCTION TESTING:

Test cases	**Input**	**Expected result**	**Actual result**	**Test result**
Test case(1)	Esra ,0927953022,esra@gmail.com,omdor,female	Customer modified	Customer modified	Accept
Test case(2)	Esra ,22hhhh,esra@gmail.com,omdor,female	Caption error message	Caption error message	Accept
Test case(3)	Esra , 0927953022,esra.com,omdor,female	Caption error message	Caption error message	Accept
Test case(4)	Esra , 0927953022,esra@gmail.com,female	Caption error message	Caption error message	Accept
Test case(5)	1111 , 0927953022,esra@gmail.com,omdor,female	Caption error message	Caption error message	Accept

CUSTOMER REPORT FUNCTION TESTIN

Test cases	Input	Expected result	Actual result	Test result
Test case(1)	Teso1573@gmail.com,1234	Sales report	Sales report	Accept
Test case(2)	Teso1573@gmail.com,1234	Balance report	Balance report	Accept
Test case(3)	Teso1573@gmail.com,1234	Billing report	Billing report	Accept
Test case(4)	Teso1573@gmail.com,1234	Customers report	Customers report	Accept
Test case(5)	Teso1573@gmail.com ,1234, 2-9-2015	Search by date reports	Search by date reports	Accept
Test case(5)	Teso1573@gmail.com ,1234, third2015	Caption error message	Caption error message	Accept

ADMIN REPORTS FUNCTION TESTING:

Test cases	**Input**	**Expected result**	**Actual result**	**Test result**
Test case(1)	mwada,1111	Report about user information	Report about user information	Accept
Test case(2)	mwada,1111	Report about the daily bill	Report about the daily bill	Accept
Test case(3)	mwada,1111	Report about all sell operation	Report about all sell operation	Accept
Test case(4)	mwada,1111,1-3-2015	Report about all sell operation according to the date	Report about all sell operation according to the date	Accept
Test case(5)	mwada,1111,third2015	Caption error message	Caption error message	Accept
Test case(6)	mwada,1111,empty date	Caption error message	Caption error message	Accept

CREATE BALANCE FUNCTION TESTING

Test cases	**Input**	**Expected result**	**Actual result**	**Test result**
Test case(1)	mwada,1111,223373699	Balance created successfully	Balance created successfully	Accept
Test case(2)	mwada,1111,22337jhhg	Caption error message	Caption error message	Accept
Test case(3)	mwada,1111,empty	Caption error message	Caption error message	Accept