**INVENTORY AND INQUIRY MANAGEMENT SYSTEM**

*A PROJECT REPORT*

*Submitted by*

**GUPTA SANKIT V. (10BCE011)**

**SHAH RUSHABH D. (10BCER038)**

*In fulfillment for the award of the degree*

*Of*

*BACHELOR OF ENGINEERING*

*In*

*COMPUTER ENGINEERING*



**INDUS INSTITUTE OF TECHNOLOGY AND ENGINEERING, AHMEDABAD**

**Gujarat Technological University, Ahmedabad**

2013-14.

**PROJECT REPORT**

ON

Inventory and inquiry management system

AT



In the partial fulfillment of the requirement

For the degree of

Bachelor of Computer Engineering

**Prepared by**

|  |
| --- |
| SANKIT GUPTA  (10BCER011)  RUSHABH SHAH  (10BCER038) |
|  |

**Under guidance of**

|  |  |
| --- | --- |
| **External Guide** | **Internal Guide** |
| Mr. Devang Shah  (Gift Touch) | Prof. Zalak Vyas  (IITE) |

**Submitted to**

|  |
| --- |
| INDUS Institute of Technology & ENGINEERING, AHMEDABAD, |
| Gujarat Technological University |

2013- 2014

CANDIDATE’S DECLARATION

We, Sankit Gupta & Rushabh Shah declare that the final year report entitled “**Inventory and Inquiry Management System**” is our own work conducted under the supervision of the guide **Prof. Zalak Vyas**.

We further declare that to the best of our knowledge, the report for B.E. final year does not contain part of the work which has been submitted for the award of B.E. Degree either in this or any other university without proper citation.

Candidate’s Signature Candidate’s Signature

Mr. Sankit Gupta Mr. Rushabh Shah

Guide : Prof. Zalak Vyas

Department of Computer Engineering,

Indus Institute of Technology and Engineering – Ahmedabad.

State: Gujarat

## INDUS INSTITUTE OF TECHNOLOGY AND ENGINEERING

**COMPUTER ENGINEERING**

**2011-12**

### logo

### CERTIFICATE

### Date:

### This is to certify that the dissertation entitled “Inventory and Inquiry Management System” has been carried out by Mr. Sankit Gupta and Mr. Rushabh Shah under my guidance in fulfillment of degree of Bachelor of Engineering in COMPUTER ENGINEERING (Final Year) of Gujarat Technological University, Ahmedabad during the academic year 2013-14

Prof. Zalak Vyas

Computer Engineering Department,

I.I.T.E.,Ahmedabad

Mr. Devang Shah

Gift Touch

Prof. Bhoomi Trivedi

Head of Department

Computer Engineering

I.I.T.E., Ahmedabad, Gujarat

Acknowledgement

We express our sincere thanks to **Prof. ZALAK VYAS**, our internal guide for the project, who helped us in understanding the project and during the implementation of the same. Without her sincere efforts, the making and understanding of various UML diagrams as well as implementation of algorithms would not have been easy. She helped us to clear the basics of object oriented approach, thus made it easier. Her suggestions really helped us to think on a broad perspective and gave us the motivation to do our best. We are extremely thankful to her for providing us her invaluable guidance and expertise throughout the project.

We take this opportunity to express our immense gratitude to **Mr. DEVANG SHAH** (Director, Gift Touch) who always motivated us to do something extraordinary. He provided us with a very friendly environment in his organization. His time particularity and tactics of what and how to learn has helped us a lot. We would like to thank him for providing us all necessary resources which greatly helped us in completing our project in time. By his uncompromising demand for quality and his insistence for meeting the deadlines, we were able to give our best.

We are also thankful to the entire **Gift Touch** family for their co-operation and support. We thank them for providing us such a warm atmosphere to make our project training memorable.

- Sankit Gupta

- Rushabh Shah

Abstract

This report explains the various functionalities that can be included in a management system. The fundamental goal of such a management system is to efficiently fetch the requirements of the stakeholders in it. Developing a user friendly management system, eliminate the traditional paper bound business model and hence lead the organization towards path of growth.

This management system helps the management to limit the users eligible to access the organization’s details that may turn out to be most important in this competitive business world. Also users can be provided different access levels so that they can perform their predefined tasks without indulging themselves into other problems.

Whereas this helps the management team, it also helps the user/employees to perform tasks easily with user friendly graphical user interface. Various tasks like adding, updating or deleting an inquiry, product, category, supplier, client, purchase-order, invoice, maintain the organizational expenses, etc. can be performed easily with this kind of management system.

Hence with this all functionalities, this kind of a management system reduces the chances of human errors in statistical data and accurate data availability of products can be obtained using it. Thus, management system not only eliminates any chance of error but also provide an efficient solution for it.

****

**Company Profile**

introduction

Promotional merchandise, promotional items, promotional products, promotional gifts, or advertising gifts, sometimes nicknamed swag or schwag, are articles of merchandise (often branded with a [logo](http://en.wikipedia.org/wiki/Logo)) used in marketing and communication programs. They are given away to promote a company, [corporate image](http://en.wikipedia.org/wiki/Corporate_image), [brand](http://en.wikipedia.org/wiki/Brand), or event. These items are usually imprinted with a company's name, logo or slogan, and given away at [trade shows](http://en.wikipedia.org/wiki/Trade_show), [conferences](http://en.wikipedia.org/wiki/Business_conference), and as part of [guerrilla marketing](http://en.wikipedia.org/wiki/Guerrilla_marketing) campaigns.

Distributors have the ability to source & supply tens of thousands of products from across the globe. Even with the advent and growth of the Internet this [supply chain](http://en.wikipedia.org/wiki/Supply_chain) has not changed, for a few reasons:

Promotional products by definition are custom printed with a logo, company name or message usually in specific [PMS](http://en.wikipedia.org/wiki/Pantone#Original_Pantone_Color_Matching_System) colors. Distributors help end-users gather artwork in the correct format and in some cases, distributors might create artwork for end-users. Distributors then interface with manufacturers, printers or suppliers, forwarding artwork in the correct format and correct size for the job. Since good distributors are well aware of several manufacturers' capabilities, they can save an end-user time and money searching for a printer or manufacturer who can produce and ship the end-user's products on time, on specification and in the required quantities.

Many distributors operate on the internet and/or in person. Many suppliers wish not to invest in the staffing to service end-users' needs, which is the purpose of merchandise distributor companies.

COMAPANY OVERVIEW

* GIFT TOUCH is a promotional merchandising and marketing services with the ultimate goal to enhance clients’ brand.  By understanding clients; marketing strategy, objectives, and target audience…the people at GIFT TOUCH turns the impossible into a unique reality.
* GIFT TOUCH serves as a one stop shop for R&R programs, brand merchandising, corporate gifting. GIFT TOUCH has a client portfolio that consists of top companies across sectors such as Pharma, BFSI, Telecom, IT , Manufacturing ,FMCG ,Automobiles.
* GIFT TOUCH is authorized channel partner for leading brands like Luminarc, Delsey , 3M , Timex , Oster , Godrej , Swiss Military , Eveready and many more . GIFT TOUCH was founded in 1991 by Mr. Devang Shah in Ahmedabad. Over the years, GIFT TOUCH has become a leading promotional market leader. Today, GIFT TOUCH works on a reseller platform, (besides direct marketing) with dealers in major cities across India.
* GIFT TOUCH has been awarded with ‘*Championship of Innovation*’ award by 3M, the company which is itself known for innovation. GIFT TOUCH has been awarded with ‘*Outstanding Performance* ‘ award by P&G.

WHAT COMPANY DO

* Promotional marketing is about more than just putting your logo on cool stuff. Custom promotional products are an integral part of any comprehensive marketing campaign.
* Showcasing your logo on a promotional product that captures the essence of your business can help you build a strong relationship with your audience. It’s a chance to make a tangible connection with your target audience.
* At GIFT TOUCH , we provide our clients with inputs as per their budget, concept so that it synergizes with their brand ultimately leading to their brand’s recognition.
* Today , to satisfy the customer requirements , we at GIFT TOUCH emphasize on the following key areas
  + - Quality
    - Concept
    - Packaging
    - Service

**Table Of Content**

**Title Page No**

**DECLARATION i**

**COMPANY PROFILE ii**

**COLLEGE CERTIFICATE vi**

**ACKNOWLEDGEMENT vii**

**ABSTRACT viii**

**LIST OF FIGURES ix**

**LIST OF TABELS x**

**CHAPTER 1 INTRODUCTION………………………………….. 1**

1.1 Project Summary……………………………………. 2

1.2 Project Purpose……………………………………… 3 1.3 Project Scope………………………………………… 4

1.4 Objectives……………………………………………. 6

1.4.1 Main Objectives……………………………... 6

1.4.2 Secondary Objectives……………………….. 7

1.5 Technology and Literature Overview……………...... 8

1.6 Synopsis……………………………………………... 12

**CHAPTER 2 LITURATURE ERP………………………….......... 13**

* 1. Introduction of ERP………………………………… 14
  2. Why ERP?……………..……………………………. 15
  3. Characteristics of ERP……………………………… 16
  4. Comparison………………………………………… 17
     1. Advantages………..………………………… 17
     2. Benefits…………..…………………………. 18
     3. Disadvantages………………………………... 18
  5. References…………………………………………… 19

**CHAPTER 3 PROJECT MANAGEMENT………………………. 20**

* 1. Project Planning Objectives………………………… 21
     1. Software Scope…………………………….... 21
     2. Resource…………………………………….. 21
        1. Human Resource…………………..... 22
        2. Reusable Software Resources……..... 22
        3. Environment Resource…………….. .. 22
     3. Project Development Approach…………….. 22

3.1.3.1 Agile Software Development………. 23

3.1.3.2 Agile Methods………………………. 24

3.1.3.3 Extreme Programming……………… 25

3.1.3.4 Scrum……………………………….. 25

3.1.3.5 Feature Driven Development……..... 27

* 1. Risk Management……………………………….…… 29
     1. Risk Identification……………………….…… 29
        1. Risk Identification artifacts……..…… 29
     2. Risk Projection……………………………….. 30
     3. Risk Planning………………………………… 30
     4. Risk Refinement……………………………… 31
     5. Tracking And Control……………………….. 31
  2. Project Scheduling………………………………....... 33
     1. Basic Principal……………………………….. 33
     2. Compartmentalization ………………….…… 33
     3. Project Organization……………………........ 34
     4. TimeLine Chart………………………………. 34
        1. Time Allocation……………………... 34
        2. Task Sets…………………………....... 34

**CHAPTER 4 SYSTEM REQUIREMENTS………………………… 37**

* 1. User Characteristics………………………………….. 38
  2. Technology…………………………………………... 39
     1. WAMP Server……….……………... 39
     2. JQuery………………………………… 40
     3. MySQL……………………………….. 41
  3. Functional Requirement………………………………. 44
     1. Inquiry System………………………………. 44
     2. Inventory System…………………….………. 45
  4. Non Functional Requirement………………………… 47
  5. Hardware and Software Requirement……………….. 49
  6. Assumption and Dependencies……………………… 50

**CHAPTER 5 DETAIL DISCRIPTION…………………..……….. 51**

* 1. User/Employee Module……………..………………….. 52

5.2Administrator Module…………………………………… 53

5.3Data Structures………..…………………………………. 54

**CHAPTER 6 SYSTEM DESIGN…………………..……………… 60**

* 1. Sequence Diagram……………………………................ 61

6.2Activity Diagram………..……………………………….. 63

6.3Class-DFD Diagram………………………………………. 65

6.4Use-Case Diagram………………….…………………… 68

6.5Site Map……………….…………..……………………… 71

6.6Website Screen-Shots………………………….…………. 74

* + 1. Login Page………………………………….. 74
    2. Home Page………………………………….. 75

6.6.3 Product Menu…………………………….. 76

6.6.4 Inquiry Menu……………………………… 78

6.6.5 Purchase Order Menu…………………… 79

6.6.6 User Manager……….……………………. 80

6.6.7 Notifications...........................................…... 81

6.6.8 Miscellaneous………………………………... 83

**CHAPTER 7 IMPLEMENTATION…………………..………….. 86**

* 1. Implementation Environment…………………………… 87

7.2Security Feature………………………………………….. 88

7.3Coding Standards……………………….………………... 89

**CHAPTER 8 TESTING……..…………………………………….. 94**

* 1. Testing Strategy……………………………..………….. 95

8.2Testing Methods…………………………………..….…. 96

**CHAPTER 9 CONCLUSION…………….………………………. 101**

1. Conclusion………………………………………….. 102
2. Bibliography……………………………………… 103

List Of Figures

|  |  |  |
| --- | --- | --- |
| **Figure No** | **Title** | **Page No.** |
| Figure 1.1 | Synopsis | 12 |
| Figure 3.1 | Project Resource Pyramid | 21 |
| Figure 3.2 | Prototyping Model | 28 |
| Figure 3.3 | Work breakdown structure | 29 |
| Figure 3.7 | Project Organization | 34 |
| Figure 4.1 | MySQL Working | 43 |
| Figure 4.2 | Inquiry use-case | 45 |
| Figure 4.3 | Inventory use-case | 46 |
| Figure 4.4 | Warehouse use-case | 46 |
| Figure 6.(1-12) | UML Diagrams | 61 |
| Figure 6.(18-40) | Screenshots | 74 |

List of Tables

|  |  |  |
| --- | --- | --- |
| **Table No** | **Title** | **Page No.** |
| Table 3.3 | Risk Type | 29 |
| Table 3.4 | Risk Category | 30 |
| Table 3.5 | Risk Planning | 31 |
| Table 3.6 | Tracing and Control Risk Type | 32 |
| Table 3.8 | Time Line Chart | 36 |
| Table 5.x | Database Tables | 54 |
| Table 6.(13-17) | Sitemap | 71 |
| Table 6.41 | Master Database | 85 |
| Table 7.1 | Comments | 89 |
| Table 7.2 | Documentation Field | 93 |

**Chapter 1**

**Introduction**

* **PROJECT SUMMARY**
* **PURPOSE**
* **SCOPE**
* **OBJECTIVE**
* **TECHNOLOGY AND LITERATURE REVIEW**
* **SYNOPSIS**

1.1 PROJECT SUMMARY

The main goal of any management system is to integrate data and processes from all areas of an organization and unify it for easy access and flow of work. Management system eliminates the traditional paper based business model with efficient tool eliminating the risk of loss of data due to human errors. Integration is an extremely important part to such management system. They usually accomplish integration by creating one single database that employs multiple software subsystems providing different areas of an organization with various business functions.

**FEATURES:**

* Management system helps in streamlining ones Business Process and Workflows
* It helps one in better customer support and services
* One can gain real time data from various departments allowing quick and powerful decision making
* Reduces paper work, duplication of entries and Manual entries.
* Management system can help manage sales and purchases in an organization. Hence, leading to control over organization’s financial accounts.
* It helps in efficient working of supply-chain management through synchronizing the product’s availability with inventory and requirements.
* It also helps in managing ones customer relationships, supplier contacts as well as customization of products.

1.2 PURPOSE

The purpose of developing this management system is to address the basic requirements of any organization i.e. inventory, inquiry, sales, purchase and above all the customers. Before this the concerned organization was based on the paper-based business model which frequently led to discrepancies due to manual work. Introducing a management system in such an organization would eliminate such human errors and lead to profit-making for the organization.

Due to the increasing competition to sustain in business world, any organization would need to efficiently manage its clients and handle their orders or inquires smoothly. Also it needs to have the accurate stock report held in the warehouses. In case of availability of the product it must be open for sale and sales of product should also be blocked that don’t appear in available stocks. Hence managing over all departments becomes a subject of prime importance with this management system.

We have tried to address such concerns in this project by continuously brainstorming and fetching requirements from the stakeholders.

* + 1. **MAIN PURPOSE**

1. To handle inquiries done by the potential clients and managing them in case these inquiries turn to projects/orders.
2. To update the stock report by adding or deducting product quantity whenever products move in or out of the warehouse, respectively.
3. To generate purchase orders whenever product is to be purchased from suppliers and correspondingly, generating invoices whenever products are sold to clients.
4. Customization of categories and relative products.

1.3 PROJECT SCOPE

In order to develop a reasonable project plan, it is required to functionally bind characteristics of the software so that by following the software engineering principles. The project management activity is the determination of software scopes. By considering the following objectives of software scopes, we derived the software scope statements.

* Software must be unambiguous and understandable at management and technical levels.
* There must be quantitative and collective data products.

Software scope is defined by Activity context, information objectives, function and performance.

* + 1. **CONTEXT**

Inventory and Inquiry Management System is a PHP based Enterprising Resource Planning software. It is a greatly simplified to handle multiple inquiries, inventory, sales and purchases. This software system will be a central management system for an organization indulged in corporate gifting segment. This system is designed to maximize the organization’s productivity by providing tools to assist the user in automating the various functionalities of the departments, which would otherwise have to be performed manually. By maximizing the organization’s work efficiency the system will meet the organizational needs while remaining easy to understand and use. Employing the software in Gift Touch environment so that it can entertain as many user queries and increase work efficiency.

* + 1. **INFORMATION OBJECTIVES**

More specifically, this system is designed to allow an organization to manage and communicate with a group of clients and suppliers to entertain inquiries and purchase product as per the stock conditions. The software will facilitate communication between different departments of the organization, hence leading to productivity. Preformatted inquiry forms are provided to generate unique customer inquiry. Such inquiries are then entertained and invoices are generated at final stage. Also if stock of product vanishes rom the warehouse, purchase orders can be generated from the system.

**1.3.3 FUNCTION & PERFORMANCE**

The system first allows the employee to register an inquiry made by some potential client regarding some category of product. Now the inquiry is generated and samples are sent to the client if such product exists. If the inquiry converts into order or project, the quantity of the product I reviewed in the stock report, unavailability of which leads to generation of purchase order for such product. If the product is in stock, required customization is done on the product and sent to the client. Attached to this is an invoice generated along with this delivery process. During all this transactions the products’ quantity is updated every time it moves in and out of inventory.

1.4 OBJECTIVES

**1.4.1 MAIN OBJECTIVES**

1. **Inventory :** Inventory that sits too long carries floor plan costs that eat into profit. Yet, when you don’t have enough of your fast-moving inventory in stock, you either lose the sale or are forced to buy the product at a higher price to fulfill the order. The right ERP system helps you optimize inventory levels with more accurate demand forecasting, comprehensive real-time status and efficient replenishment processes.
2. **Purchase :** Keeping up with the vendors to obtain order status, latest pricing, and so forth can get cumbersome and time-consuming, increasing risk of not having the right amount of product in stock at the right time and at the right cost. When properly configured, an ERP system streamlines the entire purchasing process equipping your staff to more efficiently manage vendor relationships, keep track of orders, and stay on top of myriad pricing agreements to ensure proper discounts so that you can preserve as much profit as possible per transaction.
3. **Inquiry :** A form will be provided wherein the inquiry details(name of company, contact person, product, product budget, theme, etc.) will be entered. On sending this form a query will be generated and it will also be updated in the inquiry database. The database will have an option for the status of the inquiry(if the inquiry is catered, or if samples are given, or if the samples are selected, etc.).

**1.4.2 SECONDARY OBJECTIVES**

* User level access. Admin can manage users whereas employee can just perform basic tasks.
* Multiple products can be ordered using a single purchase order.
* Amount of each product as per its quantity can be calculated.
* Amount of multiple products in a single purchase order or invoice can be summed to get final amount.
* Corresponding tax value can directly be included in the final amount to give total.
* Clients and suppliers can be easily managed. Their working status helps the organization take the decision of further dealing with them.

1.5. TECHNOLOGY AND LITRATURE REVIEW

**1.5.1 JAVASCRIPT**

JavaScript (JS) is a [dynamic](http://en.wikipedia.org/wiki/Dynamic_programming_language) computer [programming language](http://en.wikipedia.org/wiki/Programming_language). It is most commonly used as part of [web browsers](http://en.wikipedia.org/wiki/Web_browser), whose implementations allow [client-side scripts](http://en.wikipedia.org/wiki/Client-side_scripting) to [interact with the user](http://en.wikipedia.org/wiki/User_interface), control the browser, communicate [asynchronously](http://en.wikipedia.org/wiki/Ajax_(programming)), and alter the [document content](http://en.wikipedia.org/wiki/Document_Object_Model) that is displayed. It is also being used in server-side programming, game development and the creation of desktop and mobile applications.

The application of JavaScript in use outside of web pages—for example, in [PDF](http://en.wikipedia.org/wiki/Portable_Document_Format) documents, [site-specific browsers](http://en.wikipedia.org/wiki/Site-specific_browser), and [desktop widgets](http://en.wikipedia.org/wiki/Desktop_widget)—is also significant. Newer and faster JavaScript [VMs](http://en.wikipedia.org/wiki/Virtual_machine) and platforms built upon them (notably [Node.js](http://en.wikipedia.org/wiki/Node.js)) have also increased the popularity of JavaScript for server-side [web applications](http://en.wikipedia.org/wiki/Web_application). On the client side, JavaScript was traditionally implemented as an [interpreted](http://en.wikipedia.org/wiki/Interpreter_(computing)) language but [just-in-time](http://en.wikipedia.org/wiki/Just-in-time_compilation) compilation is now performed by recent (post-2012) browsers.

### 1.5.1.1 Imperative and structured

JavaScript supports much of the [structured programming](http://en.wikipedia.org/wiki/Structured_programming) syntax from [C](http://en.wikipedia.org/wiki/C_(computer_language)) (e.g., if statements, while loops, switch statements, etc.). One partial exception is [scoping](http://en.wikipedia.org/wiki/Scope_(computer_science)): C-style [block scoping](http://en.wikipedia.org/wiki/Block_scoping) is not supported. Instead, JavaScript has [function scoping](http://en.wikipedia.org/wiki/Function_scoping) (although, block scoping using the let keyword was added in JavaScript 1.7). Like C, JavaScript makes a distinction between [expressions](http://en.wikipedia.org/wiki/Expression_(programming)) and [statements](http://en.wikipedia.org/wiki/Statement_(programming)). One syntactic difference from C is [automatic semicolon insertion](http://en.wikipedia.org/wiki/Defensive_semicolon), which allows the semicolons that would normally terminate statements to be omitted.

### 1.5.1.2 Dynamic

**Dynamic typing**

As in most [scripting languages](http://en.wikipedia.org/wiki/Scripting_language), [types](http://en.wikipedia.org/wiki/Type_system) are associated with [values](http://en.wikipedia.org/wiki/Value_(computer_science)), not with [variables](http://en.wikipedia.org/wiki/Variable_(programming)). For example, a variable x could be bound to a number, then later rebound to a [string](http://en.wikipedia.org/wiki/String_(computer_science)). JavaScript supports various ways to test the type of an object, including [duck typing](http://en.wikipedia.org/wiki/Duck_typing).

**Object-based**

JavaScript is almost entirely [object-based](http://en.wikipedia.org/wiki/Object-based). JavaScript [objects](http://en.wikipedia.org/wiki/Object_(computer_science)) are [associative arrays](http://en.wikipedia.org/wiki/Associative_array), augmented with prototypes (see below). Object property names are string keys. They support two equivalent syntaxes: dot notation (obj.x = 10) and bracket notation (obj['x'] = 10). Properties and their values can be added, changed, or deleted at run-time. Most properties of an object (and those on its prototype inheritance chain) can be enumerated using a for...in loop. JavaScript has a small number of built-in objects such as Function and Date.

**Run-time evaluation**

JavaScript includes an [eval](http://en.wikipedia.org/wiki/Eval" \o "Eval) function that can execute statements provided as strings at run-time.

DHTML allows scripting languages to change variables in a web page's definition language, which in turn affects the look and function of otherwise "static" HTML page content, after the page has been fully loaded and during the viewing process. Thus the dynamic characteristic of DHTML is the way it functions while a page is viewed, not in its ability to generate a unique page with each page load.

**1.5.2 MySQL**

MySQL is a relational database management system (RDBMS) that runs as a server providing multi-user access to a number of databases. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements

Free-software-open source projects that require a full-featured database management system often use MySQL. For commercial use, several paid editions are available, and offer additional functionality. Applications which use MySQL databases include: TYPO3, Joomla, WordPress, phpBB, Drupal and other software built on the LAMP software stack.

MySQL is a popular choice of database for use in web applications, and is a central component of the widely used LAMP web application software stack- LAMP is an acronym for “Linux, Apache, MySQL, and Pearl/PHP/Python”.

**Features**

* Cross-platform support
* Triggers and Cursors
* Updatable Views
* SSL supports and authentication required
* Independent storage engines
* Query caching
* ACID compliance using transaction capable storage engines
* Supports stunnel 4 connections for access to MySQL

1.5.3 PHP

PHP is a [server-side scripting](http://en.wikipedia.org/wiki/Server-side_scripting) language designed for [web development](http://en.wikipedia.org/wiki/Web_development) but also used as a [general-purpose programming language](http://en.wikipedia.org/wiki/General-purpose_programming_language). PHP code is [interpreted](http://en.wikipedia.org/wiki/Interpreter_(computing)) by a web server with a PHP processor module, which generates the resulting web page: PHP commands can be embedded directly into an [HTML](http://en.wikipedia.org/wiki/HTML) source document rather than calling an external file to process data. It has also evolved to include a [command-line interface](http://en.wikipedia.org/wiki/Command-line_interface) capability and can be used in [standalone](http://en.wikipedia.org/wiki/Computer_software) [graphical applications](http://en.wikipedia.org/wiki/Graphical_user_interface). PHP is [free software](http://en.wikipedia.org/wiki/Free_software) released under the [PHP License](http://en.wikipedia.org/wiki/PHP_License). PHP can be deployed on most web servers and also as a standalone [shell](http://en.wikipedia.org/wiki/Shell_(computing)) on almost every [operating system](http://en.wikipedia.org/wiki/Operating_system) and [platform](http://en.wikipedia.org/wiki/Computing_platform), free of charge.

1.6 SYNOPSIS

|  |  |
| --- | --- |
| Project title | Inventory and Inquiry Management System |
| **Project description** | Designing an enterprise management system – inquiry and inventory for Gift Touch, Ahmedabad. |
| **Objective** | Design a management system for Gift Touch which can integrate inventories and inquiries generated, for efficient handling. |
| **Purpose** | To get accurate product availability and status of various orders or projects under development. |
| **Name of company** | Gift Touch |
| **Anticipated outcome** | Management system that collects inquiries from the user, maintains stock report, manages clients, suppliers and users too. |
| **Time frame** | Friday & Saturday |
| **Time Duration** | Approximately 9 Months |
| **Software specification** | Windows, NuSphere, MySQL, WAMP Server |
| **Division of responsibility** | Sankit designed logics, analyzed functionalities and developed codes for various them whereas Rushabh handled the detailed requirement fetching from the organization, database handling and data analysis. |
| **Address of Company** | 206, SHAPATH-II,  Near Madhur Hotel,  Opposite Rajpath Club,  S. G. Highway,  Ahmedabad-15. |

Fig 1.1 : Synopsis

**Chapter 2**

**Literature ERP**

* **INTRODUCTION OF ERP**
* **WHY ERP?**
* **COMPARISON TO SPECIAL-PURPOSE APPLICATIONS**
* **ADVANTAGES**
* **DISADVANTAGES**
* **BENEFITS**
* **REFERENCES**

2.1 Introduction of ERP

**Definition**: **Enterprise resource planning (ERP)** is business management software—usually a suite of integrated applications—that a company can use to store and manage data from every stage of business, including:

* Product planning, cost and development
* Manufacturing
* Marketing and sales
* Inventory management
* Shipping and payment

ERP provides an integrated real-time view of core business processes, using common databases maintained by a database management system. ERP systems track business resources—cash, raw materials, production capacity—and the status of business commitments: orders, purchase orders, and payroll. The applications that make up the system share data across the various departments (manufacturing, purchasing, sales, accounting, etc.) that entered the data. ERP facilitates information flow between all business functions, and manages connections to outside stakeholders.

Organizations consider the ERP system a vital organizational tool because it integrates varied organizational systems and facilitates error-free transactions and production. However, ERP system development is different from traditional systems development. ERP systems run on a variety of [computer hardware](http://en.wikipedia.org/wiki/Computer_hardware) and [network](http://en.wikipedia.org/wiki/Computer_network) configurations, typically using a databases an information repository.

2.2 Why ERP?

An organization that has no ERP will be running on many kinds of software that do not allow interaction. Customization also may be difficult it in some cases. This will negatively affect the optimized functioning of organization's business activities.

The organization will be facing hardship in many areas of its functions. The engineering design of the software will be needed in order to improve the product, and to follow the client's behavior and choices since the first contact is quite important. Administration of the different receipts interdependence will be very complex, such as invoices regarding materials purchases, general expenditures or salaries.

All of these things change when an ERP system is implemented. Information flows constantly and allows you to follow a client's processes at any moment, no matter which part of the process they are going through. Purchases and expenditures are registered in a centralized database which allows you to have close control over these activities. In this regard ERP helps you to prevent possible abuse.

A powerfully integrated ERP system enables interactions of marketing, sales, quality control, products processes, supply lines, stocks and many other areas and it can be in a single database. This will eliminate the occasional loss of and retyping errors. It integrates all departments and functions across a company in a single computer system that is able to serve all those different department's particular needs.

An ERP system also automates business processes by placing them into a useful format that is standardized and common for the whole organization. Moreover it could even be used between their suppliers and customers.

2.3 Characteristics

ERP (Enterprise Resource Planning) systems typically include the following characteristics:

* An integrated system that operates in (or near) real time without relying on periodic updates.
* A common database that supports all applications
* A consistent look and feel across modules
* Installation of the system without elaborate application/data integration by the Information Technology (IT) department, provided the implementation is not done in small steps.

 2.4 Comparison to special-purpose application

### 2.4.1 Advantages

The fundamental advantage of ERP is that integrating myriad businesses processes saves time and expense. Management can make decisions faster and with fewer errors. Data becomes visible across the organization. Tasks that benefit from this integration include.

* Sales forecasting, which allows inventory optimization.
* Chronological history of every transaction through relevant data compilation in every area of operation.
* Order tracking, from acceptance through fulfillment
* Revenue tracking, from [invoice](http://en.wikipedia.org/wiki/Invoice) through cash receipt
* Matching [purchase orders](http://en.wikipedia.org/wiki/Purchase_order) (what was ordered), inventory receipts (what arrived), and [costing](http://en.wikipedia.org/wiki/Cost) (what the vendor invoiced)

ERP systems centralize business data, which:

* Eliminates the need to synchronize changes between multiple systems—consolidation of finance, marketing, sales, human resource, and manufacturing applications
* Brings legitimacy and transparency to each bit of statistical data
* Facilitates standard [product naming/coding](http://en.wikipedia.org/wiki/Product_naming_convention)
* Provides a comprehensive enterprise view (no "islands of information"), making real–time information available to management anywhere, any time to make proper decisions
* Protects sensitive data by consolidating multiple security systems into a single structure.

### 2.4.2 Benefits

* ERP can greatly improve quality and efficiency of the business. By keeping a company's internal business process running smoothly, ERP can lead to better outputs that benefit the company, such as customer service and manufacturing.
* ERP supports upper level management, providing critical decision making information. This decision support lets upper management make managerial choices that enhance the business.
* ERP creates a more agile company that better adapts to change. ERP makes a company more flexible and less rigidly structured so organization components operate more cohesively, enhancing the business—internally and externally
* ERP can improve data security. A common control system, such as the kind offered by ERP systems, allows organizations the ability to more easily ensure key company data is not compromised.
* ERP provides increased opportunities for collaboration. Data takes many forms in the modern enterprise. Documents, files, forms, audio and video, emails. Often, each data medium has its own mechanism for allowing collaboration. ERP provides a collaborative platform that lets employees spend more time collaborating on content rather than mastering the learning curve of communicating in various formats across distributed systems.

### 2.4.3 Disadvantages

* Customization is problematic.
* Re-engineering business processes to fit the ERP system may damage competitiveness or divert focus from other critical activities.
* ERP can cost more than less integrated or less comprehensive solutions.
* High ERP switching costs can increase the ERP vendor's negotiating power, which can increase support, maintenance, and upgrade expenses.
* Overcoming resistance to sharing sensitive information between departments can divert management attention.
* Integration of truly independent businesses can create unnecessary dependencies.
* Extensive training requirements take resources from daily operations.
* Due to ERP's architecture (OLTP, On-Line Transaction Processing) ERP systems are not well suited for production planning and supply chain management (SCM).
* Harmonization of ERP systems can be a mammoth task (especially for big companies) and requires a lot of time, planning, and money.

2.5 references

1. <http://en.wikipedia.org/wiki/Enterprise_resource_planning#Integration>
2. <http://www.eresourceerp.com/Why-is-ERP-important-to-a-company.html>
3. <http://en.wikipedia.org/wiki/List_of_ERP_software_packages>

**Chapter 3**

**Project Management**

* **PROJECT PLANNING**
* **RISK MANAGEMENT**
* **PROJECT SCHEDULING**

3.1 PROJECT PLANNING

The objective of software project planning is to provide a framework that enables the manager to make reasonable estimates, cost, and scheduling. These estimates are made within a limited time frame at the beginning of a software project and should be updated regularly as the project progresses.

The project is developed at Gift Touch, S. G. Highway, Ahmedabad in time duration of 15rd August, 2013 to 15th April, 2014. During the project development period we have submitted report to the internal guide on alternate Saturday review and inspection. Also we have to hold meetings with our external guide as well as the project owner on regular basis in the Company.

There are many issues in the software project planning as follows

**3.1.1 SOFTWARE SCOPE**

The project being a live project and simulation concept implementation being complex, the project planning is done in a way to get the maximum of resources an on time completion of the project.

Function and performance allocated to software during system engineering should be assessed to establish a project scope that is unambiguous and understandable at the management and technical levels.

**3.1.2 RESOURCES**

People

Reusable software

Components

**Hardware and Software tool**

Fig 3.1 : Resources Planning

The second software planning task is to estimation of the resources required to accomplish the software development effort. The above figure illustrates development resources as a pyramid. The development environment – hardware and software tools-sits at the foundation of the resources pyramid and provide the infrastructure to support the development effort. At a higher level, we encounter reusable software components-software building blocks that can dramatically reduce development costs and accelerate delivery. At the top of the pyramid the primary resource-people.

* + - 1. **HUMAN RESOURCES**

The human resources required are

1. Database, network administrator.
2. Project Guides.
   * + 1. **REUSABLE SOFTWARE RESOURCE**

Reusability is possible as and when require in this application. We can update it when new company register or company modified its data. Reusable software reduces design, coding and testing cost by amortizing effort over several designs .All other software components are building as per the need

**3.1.2.3 ENVIRONMENT RESOURCE**

The environment that supports the software project, often called the software engineering includes software and hardware.

**3.1.3 PROJECT DEVELOPMENT APPROACH**

As we came to know that traditional software development approaches are more mechanistic which concentrate more on Processes, tools, contracts and plans. In contrast to traditional methods, agile methods keep emphasis on interaction, working software, embracing change at any moment of the project, customer relationships.

The method can be agile if it is:

* Incremental
* Cooperative
* Straightforward
* Adaptive
* “Agile view is more people centric rather than plan-centric.” Agile methods are not defined by a small set of principles, practices and techniques. It creates a strategic capability which has capability of responding to change, capability to balance the structure and flexibility, capability of innovation and creations through development team and uncertainty.
* Other different Agile Software development models such as
* XP (Extreme programming)
* Scrum
* FDD (Feature driven development)
* DSDM (Dynamic systems development method)
  + - 1. **AGILE SOFTWARE DEVELOPMENT**

Agility refers to the quality of being agile. Internet software industry and Mobile and wireless application development industry are looking for a very good approach of software development. Conventional software development methods have completely closed the requirements process before analysis and design process. As this approach is not always feasible and compatible with all others projects. In contrast to the conventional approaches, Agile methods allow developers to make late changes in the requirement specification document.

The focus of the agile software development as given by “Agile Software Development Manifesto” is presented in the following:

* Individuals and interactions over processes and tools
* Working software over comprehensive documentation.
* Customer collaboration over contract negotiation
* Responding to change over following a plan
* There is vital importance of communication between the individual who are in development team, since development centers are located at different places. The necessity of interaction between Individuals over different tools and different versions and processes is very vital.
* The only objective of software development team is to continuously deliver the working software for the customers. New releases must be produced for frequent intervals. The developers try to keep the code simple, straight forward and technically as advanced as possible and will try to lessen the documentation.
* The relationship between developers and the stakeholders is most important as the pace and the size of the project grows. The cooperation and negotiation between clients and the developers is the key for the relationship. Agile methods are using in maintaining good relationship with clients.
* The development team should be well-informed and authorized to consider the possible adjustments and enhancements emerging during the development process.

**3.1.3.2 AGILE METHODS**

Agile methods are designed to produce the first delivery in weeks, to achieve and early win and rapid feedback. These methods invent simple answers so that change can be less. These also improve design issues and quality as they are based on iteratively incremental method.

* What makes a method an Agile? When the process are:
* Incremental: Small releases with rapid iterations
* Cooperative: Customer and developer relationships
* Straight: The method which is easy to learn and modify with documentation
* Adaptive: Able to embrace changes instantly
* Different Agile Software Development methods
* Extreme programming
* Scrum
* Feature driven development
  + - 1. **EXTREME PROGRAMMING**

The life cycle of Extreme programming consists of five phases

* Exploration
* Planning
* Iteration on releases
* Product ionizing
* Maintenance
* Death

Extreme programming is a light weight software methodology for a small to medium sized teams developing software in the rapidly changing requirements situation. XP matches the behavior of successful programmers in the wild

* Tests
* Refactoring
* Evolutionary delivery
* Incremental planning
* Less overhead
  + - 1. **SCURM**

The term ‘SCRUM’ originally derives from a strategy in the game of rugby where it denotes “getting an out of play ball back into the game” with teamwork. Scrum concentrates on how the team members should function in order to produce the system flexibly in a constantly changing environment. Scrum is extremely simple model used by different software companies from long time, which works with existing engineering practices and is scalable and work with common sense which is to say it is very easy.

Scrum process includes three phases

* Pre-game
* Development
* Post-game

1. Pre-game phase includes two sub-phases

* Planning and Architecture design
* *Planning* phase includes the development of the required system. A Backlog list is created, which contains all the requirements that are known at that moment. Every iteration the Back log list is updated by scrum team to gain commitment for the next iteration.
* Architecture phase: In this phase an abstract view of the model is designed by viewing Backlog list.

2. The Development phase:

This phase takes care of the different variable like time frame, quality, requirements, recourses, technologies and tools. The system is developed in Sprints. Sprints are the iterative cycles where functionality is developed or enhanced to produce new increments. Each Sprint includes the traditional phases of software engineering.

* Requirement
* Analysis
* Design
* Evolution and delivery

3. The Post-game phase close to release.

* + Roles in Scrum
  + SCRUM master
  + Product owner
  + Scrum Team
  + Customer
  + User
  + Management
    - * 1. **FEATURE DRIVEN DEVELOPMENT**

FDD is agile approach for software development systems. It provides enough structure for large items, emphasis on quality and working software, delivers frequent, tangible results.

Phases of FDD

* Develop an overall model
* Build a features list
* Plan by feature
* Design by feature
* Build by feature

To solve actual problems in industry settings, software engineer or a team of engineers must incorporate development strategy that encompasses the process, methods and tools layers and generic phases. This strategy is often referred to as process model or an engineering paradigm. A process model for software engineering is chosen based on the nature of the project and application, the methods and tools to be used and the controls and deliverables that are required.

We have followed Prototype Model Approach to develop Software Project Survey Application.

Customer evaluation

Design

Build prototype

Refine requirements incorporating customer suggestions

Design

Requirement gathering

Coding

Test

Maintain

Accepted by customer

Figure 3.2: Prototyping Model

3.2 RISK MANAGEMENT

Software is a difficult undertaking. Lots of things can go wrong and frankly, many often do. It’s for this reason that being prepared understanding the risks and taking proactive measure to avoid or manage them is a key element of good software project management. Recognizing what can go wrong is the first step called ‘Risk Identification’. Next each risk is analyzed to determine the likelihood that it will occur and the damage that it will do if it does occur. Once this information is established, risks are ranked, by probability and impact. Finally a plan is developed to manage those risks with high probability.

**General Risks:**

The general risks that can affect the development of software are as follows:

* **Lack of resources**: The resources which are needed for the development of this project are not available during project.
* **Time Duration:** We have limited time period so it takes well analyzed time chart to implement correctly and completely.
* **Lack of information**: The project is based on Image processing and satellite data products. Lake of this information and knowledge can also consume much of time.
  + 1. **RISK IDENTIFICATION**

Risk identification is a systematic attempt to specify threats to the project plan. By identifying known and predictable risks, I take first step towards avoiding them when possible and controlling them when necessary.

**3.2.1.1 RISK IDENTIFICATION ARTIFACTS**

I considered the following types of risk to identify the risk in proper manners. The next table shows the type of risks.

|  |  |
| --- | --- |
| **Risk Type** | **Description** |
| Project Risks | This type of risk can threaten the project plan. That is, if project risks become real, it is likely that project schedule and personal requirement problems and their impact on a software project.  Example : Requirement Change, Specification delay |
| Technical Risks | This type of risk can threaten the quality and timeliness of software to be produced. If a technical risk becomes a reality then implementation may become impossible or difficult.  Example : Technology, Hardware unavailability,  Heap Space goes out of range |
| Business Risks | This type of risk can threaten the viability of the software to be built. Business risk often the project or the product.  Example : Size under estimated |

Table 3.3 : Risk Type

* + 1. **RISK PROJECTION**

Risk projection ,also called risk estimation, attempts to rate each risk in two ways-the likelihood or probability that risk is real and consequences of the problems associated with the risk should it occur. The following table shows the artifacts used in the risk projections.

|  |  |
| --- | --- |
| **Risk category** | **Description** |
| Catastrophic | Risk can cause the whole system to fail. |
| Critical | There may be significant degradation of the system may result. |
| Minor | The risk can be easily recovered from the system failure. |
| Negligible | The risk can be negligible and shall not affect in the performance in the system at all. |

Table 3.4 : Risk Category

**3.2.3 RISK PLANNING**

Risk planning process is considered when each of the key risk has been identified. Risk reduction Strategy is used as abatement procedure. This involves planning ways to contain the damage due to a risk.

|  |  |
| --- | --- |
| **Risk** | **Strategy** |
| Requirement Changes | Derive traceability information to access requirements, change impact and maximize information hiding. |
| Power Failure | To reduce the risk, UPS facility  Is used for backup storage. |
| Schedule risk | To reduce this risk, we are going to complete our project according to our schedule. |
| Performance | Investigate Database which can effectively process. |

Table 3.5 : Risk Planning

**3.2.4 RISK REFINEMENT**

A project can get affected by a large variety of risks. In order to be able to systematically identify the important risk which might affect the project it is necessary to categorize risks into different classes which we will discuss in next section. During Risk analysis process, each identified risk is considered in turn and judgment is made about the probability and the seriousness of the risk.

* + 1. **TRACKING AND CONTROL**

Weekly status reports were sent to the Teachers and each module discusses in the meetings. Information about work done and the further working report is exchanged during the weekly meetings to make sure is on schedule to meet all our milestones.

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk ID#** | **Risk Type** | **Impact** | **Affected component** |
| 1 | Software components which should be reused contain defects limit their functionality | Moderate | Serious |
| 2 | Change of requirements which require proposal of major design rework. | High  Moderate | Serious  Serious |
| 3 | Scheduling slippage:  The time required for the development of software is underestimated, so schedule  slippage will occur | High | Serious |
| 4 | The size of software is underestimated | High | Tolerable |
| 5 | Power Failure | High | Tolerable |
| 6 | Inexperienced team member | Medium | Tolerable |

Table 3.6 : Tracking and Control of Risk Type

3.3 PROJECT SCHEDULING

* Project scheduling is one of the main key aspects of any project. Any project must be schedule before developing it.
* When project developer works on scheduled project it is more advantageous for him/her to compare to unscheduled project. It gives us timeline for finishing the particular activity. Scheduling gives us idea about project length, its cost, its normal duration of completion and we can also find out the shortest way to complete the project with less overall cost of project.
* Project schedule describes dependency between activities. The estimated time required to reach each milestones and allocation of people to activities.

**3.3.1 BASIC PRINCIPAL**

“Software project scheduling is an activity that distributes estimated effort across the planned project duration by allocating the effort to specific software engineering tasks.”

**Proper scheduling requires:**

* All tasks appear in network and dependent on some of other.
* Effort and timing are intelligently allocated to each task.
* Interdependencies between tasks are properly indicated.
* Resources are allocated for the work to be done.
  + 1. **COMPARTMENTALIZATION**

My software project is compartmentalized into the following tasks,

* Designing the GUI and establishing the sequence of the pages or control of pages.
* Coding.
* Validations.
* Testing.
  + 1. **PROJECT ORGANIZATION**

This describes the way in which the development team is organized the people involved and their roles in team. Here Project Organization Chart is shown.

Project Leader

(Mr. Devang Shah)

Project Guide

(Prof. Zalak Vyas)

Project Developer

(Sankit Gupta)

(Rushabh Shah)

Figure 3.7 : Project Organization

**3.3.4 TIMELINE CHART**

**3.3.4.1 TIME ALLOCATION**

Each task to be scheduled must be allocated some number of work units.

**3.3.4.2 TASK SETS**

We have selected the “prototype Process Model” so that there are six different work tasks to work together. A tasks set is a collection of software engineering work tasks, milestones, and deliverables that must be accomplished to complete a particular project. Tasks set are designed to accommodate different types of projects and different degree of Rigore.

Most software organizations encounter the following projects.

* Concept Development.
* New Application development.
* Application enhancement.
* Application maintenance.
* Re-engineering project.

###### Refinement of major tasks

The above tasks are again refinement as follows:

* Analysis of required system

1. Study of required flow
2. Study of methodology

* Defining goals and objectives

**1.**  Preparing the goal

**2.**  Defining the flow of the project

* Finding out resources required

**1.** Identifying the resources

**2.** Making arrangement for getting the resources

* Coding and Testing.

**1.** Designing the tables and creating them.

**2**. Coding for the Forms.

**3.** Validation of the forms.

**4.** Testing

* Documentation

**1.** Divide document in small parts

**2**. Documentation of each part

**3.** Integration of all parts

**4.** Review of project documentation.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Task** | **Start** | **Finish** | **Days** | **Comments** |
| 1 | Study of overall system | 1st Aug. 2013 | 21st Aug. 2013 | 21 | - |
| 2 | Requirement Analysis | 22nd Aug. 2013 | 30th Aug. 2013 | 10 | - |
| 3 | System Diagrams | 31st Aug. 2013 | 15th Sep. 2014 | 16 | - |
| 4 | System Designing | 16th Sep 2013 | 30th Oct 2013 | 15 | - |
| 5 | Coding | 15th Oct 2013 | 25th March 2014 | 5 Months | - |
| 6 | Testing | 25th March 2014 | 9th April 2014 | 15 | - |
| 7 | Documentation | 11th April 2014 | 20th April 2014 | 10 | - |

Table 3.8 : Time Line Chart

**Chapter 4**

**System Requirement**

* **USER CHARACTERISTICS**
* **TECHNOLOGY**
* **FUNCTIONAL REQUIREMENT**
* **NON FUNCTIONAL REQUIREMENT**
* **SOFTWARE AND HARDWARE SCOPE**
* **ASSUMPTION AND DEPENDENCIES**

4.1 USER CHARACTERISTICS

Analyzing user characteristics is an important aspect of any project. It allows us to clearly define and focus on who the end users are for the project. Also, it allows checking the progress of the project to ensure that we are still developing the system for the end users. The user must have following characteristics

* User must have basic knowledge of Computers.
* User should understand the use of all modules.
* User should be aware about WAMP server.
* User can easily interact with the proposed system.
* User must know the technical terms used in the company for performing different tasks.
* User should be also being aware about the running process of the system.
* User must be familiar with Windows Operating System.
  1. TECHNOLGY
     1. **WAMP Server**

The acronym WAMP refers to a set of free ([open source](http://www.webopedia.com/TERM/O/open_source.html))[applications](http://www.webopedia.com/TERM/A/application.html), combined with Microsoft Windows, which are commonly used in [Web server](http://www.webopedia.com/TERM/W/Web_server.htm) environments. The WAMP stack provides developers with the four key elements of a Web server:  an [operating system](http://www.webopedia.com/TERM/O/operating_system.htm), [database](http://www.webopedia.com/TERM/D/database.html), Web server and Web scripting software. The combined usage of these programs is called a server stack. In this stack, [Microsoft Windows](http://www.webopedia.com/TERM/M/Microsoft_Windows.html) is the operating system (OS), [Apache](http://www.webopedia.com/TERM/A/Apache_Web_server.html) is the Web server, [MySQL](http://www.webopedia.com/TERM/M/MySQL.html) handles the database components, while [PHP](http://www.webopedia.com/TERM/P/PHP.html), [Python](http://www.webopedia.com/TERM/P/Python.htm), or [PERL](http://www.webopedia.com/TERM/P/Perl.html) represents the dynamic scripting languages.

As a reminder, Apache is the most popular open source web server, MySQL is the most popular open-source database, used by huge number of websites around the world (among them, you will find Google, YouTube, Wikipedia, Facebook, Flickr, etc) and PHP is a widely used general-purpose server-side scripting language designed to produce dynamic web pages.

By combining these components into a single installation package, **WampServer** allows users **to set up a server locally on their Windows machine** to create dynamic web applications with Apache, PHP and the MySQL database in the same development conditions as on the production server.

One of the great benefits of **WampServer** is that it allows you to develop, upgrade components, perform any web development task and carefully test everything offline first, which reduces the risks of creating problems on the live server.

* + - 1. **BENEFITS OF WAMP SERVER**

Wamp server is a collection of programs you can use to turn your regular desktop PC to a fully compatible web server with HTTP, PHP, MySQL, PHPMyAdmin, SQLBuddy applications. The advantage is that it is easy configurable with the built-in tools. Also it is structured in the way that you have everything you need at a click distance. The configuration screens provide extra information how settings should look like. Another thing would be that the packed applications are configured to be compatible between them, all being automatic configured, you just unpack your script, database, application, run the installer (if it has any).

* + 1. **JQUERY**

JQuery is a library of JavaScript Functions. JQuery is a lightweight "write less, do more" JavaScript library. The JQuery library contains the following features:

* HTML element selections
* HTML element manipulation
* CSS manipulation
* HTML event functions
* JavaScript Effects and animations
* HTML DOM traversal and modification

It is an open source JavaScript library that simplifies the interaction between HTML and JavaScript.

* + - 1. **ADVANTAGES**
* Ease of use  
  this is pretty much the main advantage of using JQuery, it is a lot more easy to use compared to standard JavaScript and other JavaScript libraries. Apart from simple syntax, it also requires much less lines of code to achieve the same feature in comparison.
* Large library  
  JQuery enables you to perform hordes of functions in comparison to other JavaScript libraries.
* Strong opensource community. (Several jQuery plugins available)  
  JQuery, while relatively new, has a following that religiously devote their time to develop and enhance the functionality of JQuery. Thus there are hundreds of prewritten plugins available for download to instantly speed up your development process.
* Great documentation and tutorials  
  The JQuery website has a comprehensive documentation and tutorials to get even an absolute beginner in programming to get the ball rolling with this library.
* Ajax support  
  JQuery lets you develop Ajax templates with ease, Ajax enables a sleeker interface where actions can be performed on pages without requiring the entire page to be reloaded. A pretty but simple example of this can be seen above the fold on
  + - 1. **DISADVANTAGES**
* Functionality maybe limited  
  While JQuery has an impressive library in terms of quantity, depending on how much customization you require on your website, functionality maybe limited thus using raw java script maybe inevitable in some cases.
* JQuery javascript file required  
  The JQuery java script file is required to run JQuery commands, while the size of this file is relatively small (25-100KB depending on server), it is still a strain on the client computer and maybe your web server as well if you intend to host the JQuery script on your own web server.
  + 1. **MY SQL**

**MySQL** is a relational database management system (RDBMS) that runs as a server providing multi-user access to a number of databases.

MySQL is a popular choice of database for use in web applications, and is a central component of the widely used LAMP web application software stack—LAMP is an acronym for "[Linux](http://en.wikipedia.org/wiki/Linux), [Apache](http://en.wikipedia.org/wiki/Apache_HTTP_Server), MySQL, and Perl/PHP/Python/JAVA".

MySQL is primarily an RDBMS and therefore ships with no GUI tools to administer MySQL databases or manage data contained within. Users may use the included command-line tools, or download MySQL frontends from various parties that have developed desktop software and web applications to manage MySQL databases, build database structure, and work with data records.

The MySQL Query Browser is a graphical tool provided by MySQL for creating, executing, and optimizing queries in a graphical environment. While all queries executed in the MySQL Query Browser can also be run from the command-line using the MySQL utility, the MySQL Query Browser allows for the querying and editing of data in a more intuitive, graphical manner.

## *WHAT IS MYSQL AND WHY ARE WE USING IT?*

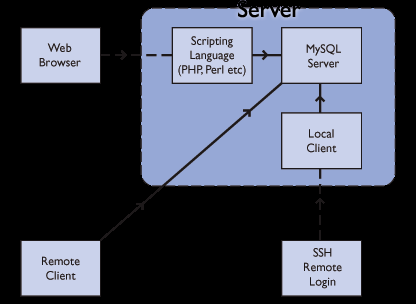
MySQL is a powerful Relational Database Management System (RDBMS) which we will use to learn the basic principles of database and data manipulation using Structured Query Language (SQL) statements. SQL is a database language that is used to retrieve, insert, delete and update stored data. This is achieved by constructing conditional statements that conform to a specific syntax (i.e. the strict order required of elements for a statement to work).

Although it is assumed that most people reading this know what a database and SQL are (if not necessarily how to use them), there follows a little recap that it does no harm

* + - 1. **HOW DOES MYSQL WORKS?**

MySQL is a database server program and as such is installed on one machine, but can 'serve' the database to a variety of locations.

The MySQL Server is installed on a Server and can be accessed directly via various client interfaces, which send SQL statements to the server and then display the results to a user. Some of these are:

* A Local Client - a program on the same machine as the server. An example of this is the command line MySQL client software we will be using in the rest of the MySQL workshops (although there are other programs including graphical interfaces).
* A Scripting Language - can pass SQL queries to the server and display the result.
* A Remote Client - a programmed on a different machine that can connect to the server and run SQL statements.
* Remote Login - You may be able to connect to the Server Machine to run one of its local clients.
* Web Browser - you can use a web browser and scripts that someone has written (we're going to use this method for the rest of the workshop).

.

Fig 4.1 : MYSQL Working

4.3 FUNCTIONAL REQUIREMENT

The various functional requirements are included in the software. This requirements are as follows :

* + 1. **Inquiry System:**
       1. **Introduction**

This subsection relates to requirements for the inquiry section of ERP.

**4.3.1.2 Inputs**

Input from the user client will be required to know what the client expects as a product. For this an online from will be generated where he/she can provide input.

* + - 1. **Processing**

The query fired by the client will generate a query into organization database which will then be handled manually by public relation department.

**4.3.14 Outputs**

The PRD will then send related product details to the client through e-mail and contact him/her personally if required.

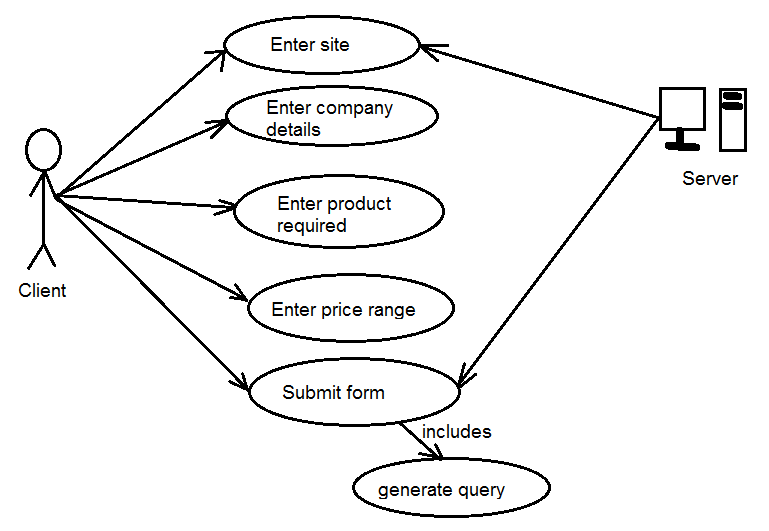


Fig 4.2 : Inquiry System Use Case

* + 1. **Inventory System:**

**4.3.2.1 Introduction**

This subsection relates to inventory status report.

**4.3.2.2 Inputs**

This service is availed to few authenticate people of the organization. It gives the inventory report to such users. As an input it needs valid user id and password.

**4.3.2.3 Processing**

The inventory data is processed in central database which are provided as per user wants, whether based on product or on warehouse.

**4.3.2.4 Outputs**

The central database will then provide the numeric details of the amount of stocks held in different warehouses at that instance.

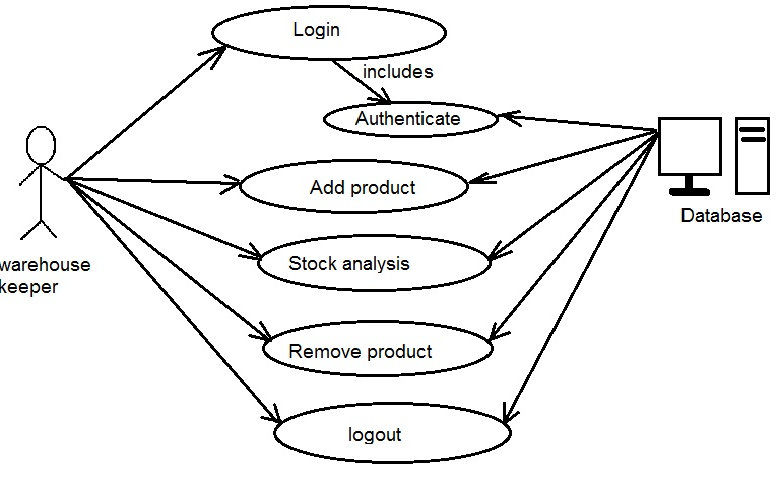


Fig 4.3 : Inventory System Use Case

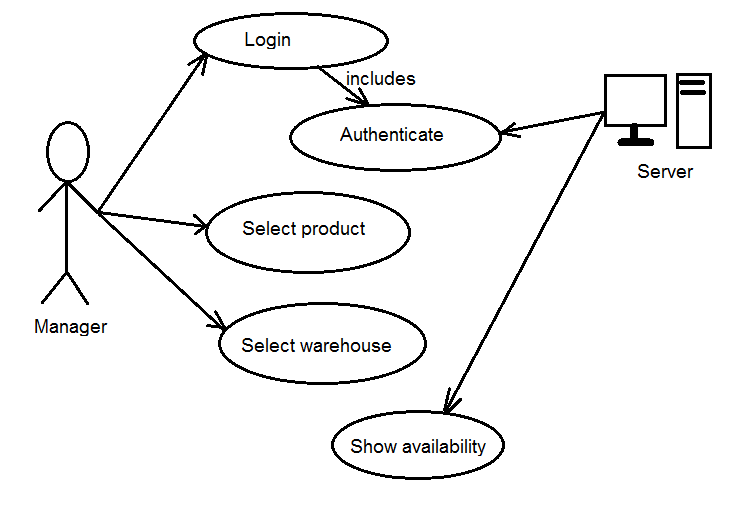


Fig 4.4 : Warehouse System Use-Case

4.4 NON FUNCTIONAL REQUIREMENT

There are lots of features that we have included in the software. But some of them were described over here.

* **Easy to maintain records:**

All the records are stored in the database. The company can create new records or can delete or update existing records from the database as per the requirement.

* **Graphical User Interface**:

Easy to understand. The tasks can be performed by any employee/user very easily, due to efficient graphical user interface.

* **Performance:**

Performance is the most important aspect of any software. It is very important to know the maximum time that can be consumed during the execution of the query fired in worst situations.

* **Reliability:**

The reliability of the software is as important as performance of the software. The software developed should be reliable so as to increase productivity of the organization. It must provide accurate data about the information fetched.

* **Availability:**

The software must be available anywhere anytime. The software to be developed will be available online so as to check status reports of the warehouses as and when required.

### Security:

The security of any online firm is of prime importance. It is essential to provide security to it so no other person than those validated by the firm can access the stock reports.

### Maintainability:

Technology changes every day. Hence, it is necessary for any firm to keep it updated with new technology to compete with outer world. Hence, it is necessary to regularly update and maintain this software.

### Portability:

In case the firm needs to run this software at any other of its branch office, the software must be portable so that it can run on different bases.

4.5 SOFWARE AND HARDWARE SPECIFICATION

Hardware and Software Requirements are used to describe the minimum hardware and software requirements to run the Software. These requirements are described below.

**4.5.1 HARDWARE REQUIREMENT**

* 1.6 GHZ CPU
* 256 MB of RAM
* Monitor
* Keyboard/Mouse

**4.5.2 SOFTWARE REQUIREMENT**

* Windows Operating System
* WAMP Server
* Any Web Browser

4.6 ASSUMPTION AND DEPENDENCIES

* The system completely depends on the functioning capabilities of the central database that will store all information regarding the stocks, proper functioning of this database is of prime importance for this software.
* The access to the central database depends completely on the availability of internet access through a device. Without proper network connection access to data is not possible.
* The software also depends on simultaneous manual updates of stocks so that the database can provide accurate stock holdings of the warehouses. Hence, warehouse keeper is an important dependency factor for efficient usage of this software.

**Chapter 5**

**Detail Description**

* **USER MODULE**
* **ADMINISTRATOR MODULE**
* **Data Structures**

5.1 USER/EMPLOYEE MODULE

* + Login
  + Registration/Account Details
    - Name
    - Email
    - Contact
    - Password
  + Actions
    - Inquiry Feed
    - Invoice generation
    - Purchase order generation
    - Customer Management
    - Supplier Management
    - Expenses Management
    - Product customization
    - Category Managing
  + Logout

5.2 ADMINISTRATOR (ADMIN) MODULE

* + Manage Admin Account
* Profile information
* Other detail information with logo
  + Login
  + Registration/Account Details
    - Name
    - Email
    - Contact
    - Password
  + Actions
    - Inquiry Feed
    - Invoice generation
    - Purchase order generation
    - Customer Management
    - Supplier Management
    - Expenses Management
    - Product customization
    - Category Managing
* User/Employee Management
  + Add user/employee
  + Edit access levels
  + Update catalog
  + Block Users/Employees
  + Logout

5.3 DATA STRUCTURES

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Collation** | **Attributes** | **Null** | **Default** | **Extra** |
| **Customer\_id** | Int(20) |  |  | No | None | AUTO\_INCREMENT |
| **Customer\_company** | Varchar(100) | Latin1\_swedish\_ci |  | No | None |  |
| **Customer\_name** | Varchar(50) | Latin1\_swedish\_ci |  | No | None |  |
| **Tin\_no** | Int(10) |  |  | No | None |  |
| **Customer\_address** | Varchar(100) | Latin1\_swedish\_ci |  | No | None |  |
| **Customer\_contact** | Varchar(20) | Latin1\_swedish\_ci |  | No | None |  |
| **Customer\_email** | Varchar(50) | Latin1\_swedish\_ci |  | No | None |  |
| **Customer\_status** | Tinyint(2) |  |  | No | None |  |
| **Customer\_createdon** | TIMESTAMP |  |  | No | CURRENT\_TIMESTAMP |  |

Table 5.1 : customer\_master

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **collation** | **Attributes** | **Null** | **Default** | **Extra** |
| **Exp\_id** | Int(11) |  |  | no | None | AUTO\_INCREMENT |
| **Exp\_title** | Varchar(30) | Latin1\_swedish\_ci |  | No | None |  |
| **Exp\_description** | Varchar(100) | Latin1\_swedish\_ci |  | No | None |  |
| **Exp\_date** | Date |  |  | No | None |  |
| **Exp\_amount** | Double(10,2) |  |  | No | None |  |
| **Exp\_createdon** | TIMESTAMP |  | On update | No | CURRENT\_TIMESTAMP | On update CURRENT\_TIMESTAMP |

Table 5.2 : expense\_manager

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Collation** | **Attributes** | **Null** | **Default** | **Extra** |
| **Inventory\_id** | Int(10) |  |  | No | None | AUTO\_INCREMENT |
| **Product** | Int(10) |  |  | No | None |  |
| **Quantity** | Int(10) |  |  | No | None |  |
| **Rate** | Double(10,2) |  |  | No | None |  |
| **Created\_on** | TIMESTAMP |  |  | No | CURRENT\_TIMESTAMP |  |

Table 5.3 : inventory

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Collation** | **Attributes** | **Null** | **Default** | **Extra** |
| **Invoice\_desc\_id** | Int(10) |  |  | No | None | AUTO\_INCREMENT |
| **Invoice\_id** | Int(10) |  |  | No | None |  |
| **Product\_id** | Int(10) |  |  | No | None |  |
| **Quantity** | Int(10) |  |  | No | none |  |
| **Rate** | Double(10,2) |  |  | No | None |  |
| **Total** | Double(10,2) |  |  | No | None |  |
| **Created\_on** | TIMESTAMP |  | On update | No | CURRENT\_TIMESTAMP | On update CURRENT\_TIMESTAMP |

Table 5.4 : invoice\_description

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Collation** | **Attributes** | **Null** | **Default** | **Extra** |
| **Invoice\_id** | Int(10) |  |  | No | None | AUTO\_INCREMENT |
| **Customer** | Int(10) |  |  | No | None |  |
| **Date** | Date |  |  | No | None |  |
| **Amount** | Double(10,2) |  |  | No | None |  |
| **Tax** | Double(5,2) |  |  | No | None |  |
| **Total** | Double(10,2) |  |  | No | None |  |
| **Status** | Tinyint(2) |  |  | No | None |  |
| **Created\_on** | TIMESTAMP |  | ON UPDATE | No | CURRENT\_TIMESTAMP |  |

Table 5.5 : invoice\_master

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Collation** | **Attributes** | **Null** | **Default** | **Extras** |
| **Pod\_id** | Int(10) |  |  | NO | None | AUTO\_INCREMENT |
| **Purchaseorder\_id** | Int(10) |  |  | NO | None |  |
| **Product\_id** | Int(10) |  |  | NO | None |  |
| **Product\_quantity** | Int(10) |  |  | NO | None |  |
| **Product\_rate** | Double(10,2) |  |  | NO | None |  |
| **Total** | Double(10,2) |  |  | NO | None |  |
| **Status** | Tinyint(2) | Latin1\_swedish\_ci |  | NO | None |  |
| **Created\_on** | TIMESTAMP |  |  | NO | CURRENT\_TIMESTAMP |  |

Table 5.6 : purchaseorder\_description

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Collation** | **Attributes** | **Null** | **Default** | **Extras** |
| **Purchaseorder\_id** | Int(10) |  |  | NO | None | AUTO\_INCREMENT |
| **Purchaseorder\_date** | Date |  |  | NO | None |  |
| **Supplier\_id** | Int(10) |  |  | NO | None |  |
| **Amount** | Double(10,2) |  |  | NO | None |  |
| **Tax** | Double(5,2) |  |  | NO | None |  |
| **Total** | Double(10,2) |  |  | NO | None |  |
| **Status** | Tinyint(2) |  |  | NO | None |  |
| **Created\_on** | TIMESTAMP |  |  | NO | CURRENT\_TIMESTAMP |  |

Table 5.7 : purchaseorder\_master

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Collation** | **Attributes** | **Null** | **Default** | **Extras** |
| **Product\_id** | Int(10) |  |  | NO | None | AUTO\_INCREMENT |
| **Product\_name** | Varchar(50) | Latin1\_swedish\_ci |  | NO | None |  |
| **Category\_name** | Int(10) | Latin1\_swedish\_ci |  | NO | None |  |
| **Product\_image** | Varchar(100) | Latin1\_swedish\_ci |  | NO | None |  |
| **Product\_desc** | Text | Latin1\_swedish\_ci |  | NO | None |  |
| **Product\_quant** | Int(10) |  |  | NO | None |  |
| **Rate** | Double(10,2) |  |  | NO | None |  |
| **Status** | Tinyint(2) |  |  | NO | None |  |
| **Created\_on** | TIMESTAMP |  | ON UPDATE | NO | CURRENT\_TIMESTAMP |  |

Table 5.8 : product\_master

Table 5.9 : project\_manager

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | | **Collation** | | **Attributes** | | **Null** | | **Default** | | **Extras** | |
| **P\_id** | Int(10) | |  | |  | | No | | NONE | | AUTO\_INCREMENT | |
| **Cname** | Varchar(100) | | Latin1\_swedish\_ci | |  | | No | | NONE | |  | |
| **Caddress** | Text | | Latin1\_swedish\_ci | |  | | No | | NONE | |  | |
| **Cperson** | Varchar(50) | | Latin1\_swedish\_ci | |  | | No | | NONE | |  | |
| **Cno** | Int(15) | |  | |  | | No | | NONE | |  | |
| **Email** | Varchar(100) | | Latin1\_swedish\_ci | |  | | No | | NONE | |  | |
| **Pcat** | Int(10) | | Latin1\_swedish\_ci | |  | | No | | NONE | |  | |
| **Preq** | Text | | Latin1\_swedish\_ci | |  | | No | | NONE | |  | |
| **Quant** | Int(10) | |  | |  | | No | | NONE | |  | |
| **Budget** | Int(10) | |  | |  | | No | | NONE | |  | |
| **Inq\_date** | Date | |  | |  | | No | | NONE | |  | |
| **Status** | TINYINT(2) | | Latin1\_swedish\_ci | |  | | No | | NONE | |  | |
| **Field** | | **Type** | | **Collation** | | **Attributes** | | **Null** | | **Default** | | **Extras** |
| **Registration\_id** | | Int(10) | |  | |  | | NO | | None | | AUTO\_INCREMENT |
| **Name** | | Varchar(50) | | Latin1\_swedish\_ci | |  | | No | | None | |  |
| **Email** | | Varchar(100) | | Latin1\_swedish\_ci | |  | | No | | None | |  |
| **Pswd** | | Varchar(15) | | Latin1\_swedish\_ci | |  | | No | | None | |  |
| **Contact** | | Int(10) | |  | |  | | No | | None | |  |
| **Status** | | Tinyint(2) | |  | |  | | No | | None | |  |
| **Created\_on** | | TIMESTAMP | |  | |  | | No | | CURRENT\_TIMESTAMP | |  |

Table 5.10 : registration

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Collation** | **Attributes** | **Null** | **Default** | **Extras** |
| **Supplier\_id** | Int(10) |  |  | No | None | AUTO\_INCREMENT |
| **Supplier\_firm** | Varchar(100) | Latin1\_swedish\_ci |  | No | None |  |
| **Supplier\_name** | Varchar(100) | Latin1\_swedish\_ci |  | No | None |  |
| **Supplier\_address** | Text | Latin1\_swedish\_ci |  | No | None |  |
| **Supplier\_contact** | Int(11) |  |  | No | None |  |
| **Supplier\_email** | Varcgar(50) | Latin1\_swedish\_ci |  | No | None |  |
| **Supplier\_desc** | Text | Latin1\_swedish\_ci |  | No | None |  |
| **Status** | Tinyint(2) |  |  | No | None |  |
| **Created\_on** | TIMESTAMP |  | On update | No | CURRENT\_TIMESTAMP |  |

Table 5.11 : supplier\_master

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Collation** | **Attributes** | **Null** | **Default** | **Extras** |
| **User\_id** | Int(11) |  |  | NO | None | AUTO\_INCREMENT |
| **User\_name** | Varchar(50) | Latin\_swedish\_ci |  | NO | None |  |
| **User\_pswd** | Varchar(20) | Latin\_swedish\_ci |  | NO | None |  |
| **User\_email** | Varchar(50) | Latin\_swedish\_ci |  | NO | None |  |
| **User\_status** | Tinyint(2) |  |  | NO | None |  |
| **Access\_level** | Tinyint(2) |  |  | NO | None |  |
| **Created\_on** | TIMESTAMP |  |  | NO | CURRENT\_TIMESTAMP |  |

Table 5.12 : user\_master

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Collation** | **Attributes** | **Null** | **Default** | **Extras** |
| **Category\_id** | Int(10) |  |  | NO | None | AUTO\_INCREMENT |
| **Category\_name** | Varchar(50) | Latin\_swedish\_ci |  | NO | None |  |
| **Created\_on** | TIMESTAMP |  |  | NO | CURRENT\_TIMESTAMP | On update CURRENT\_TIMESTAMP |

Table 5.13 : category\_master

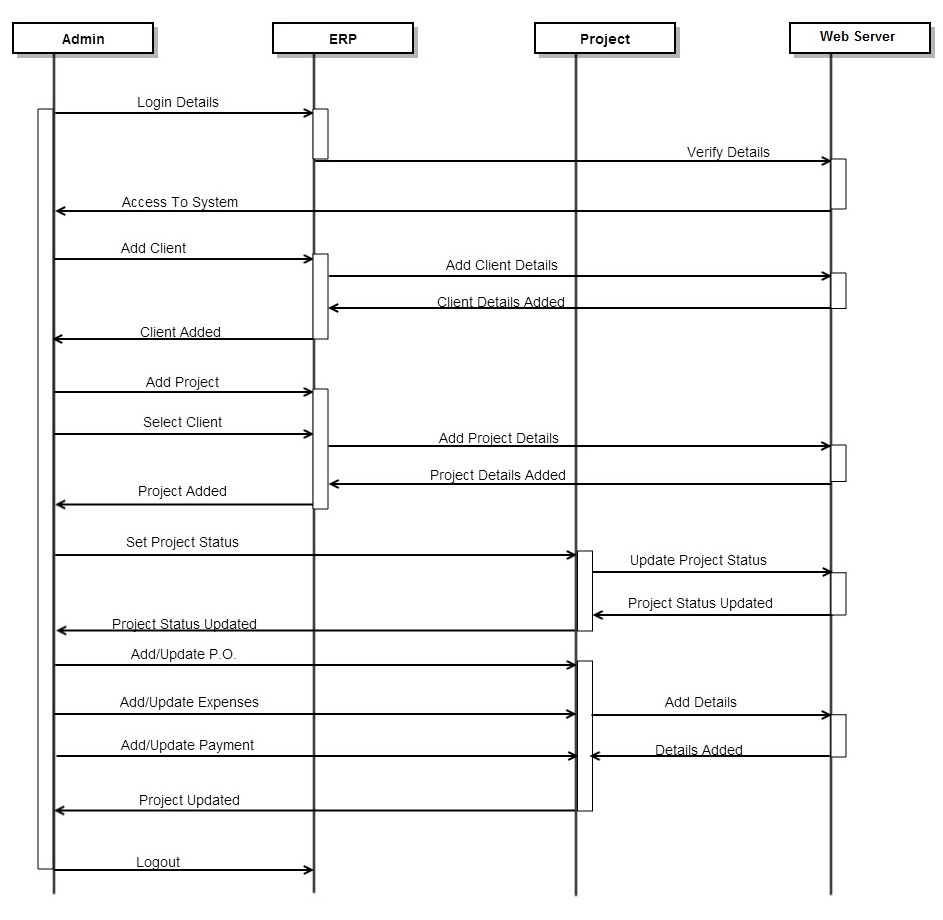
**Chapter 6**

**System Design**

* **SEQUENCE DIAGRAM**
* **ACTIVITY DIAGRAM**
* **CLASS DIAGRAM**
* **DFD DIAGRAM**
* **USE CASE DIAGRAM**

6.1 SEQUENCE DIAGRAMS

The sequence diagram is used primarily to show the interactions between objects in the sequential order that those interactions occur. Much like the class diagram, developers typically think sequence diagrams were meant exclusively for them.

**Sequence Diagram for project management : **

**Fig 6.1**

**Sequence Diagram for purchase order :**

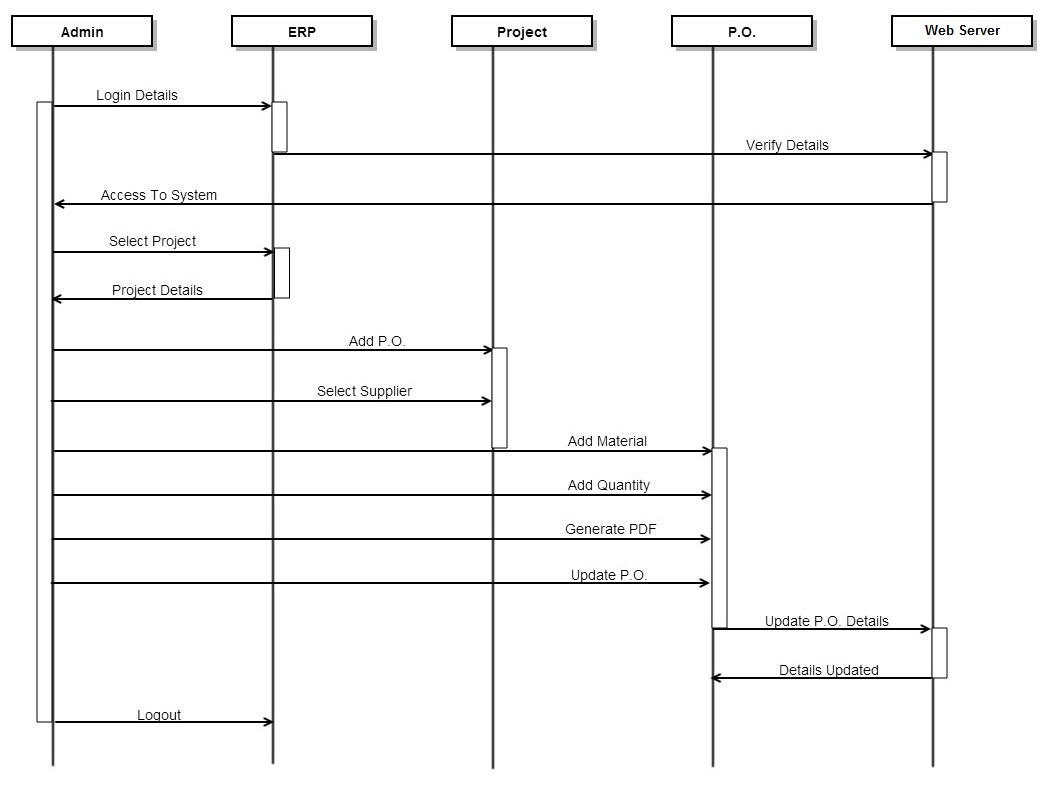
****

Fig : 6.2

6.2 ACTIVITY DIAGRAMS

**Activity Diagram for Project Management:**

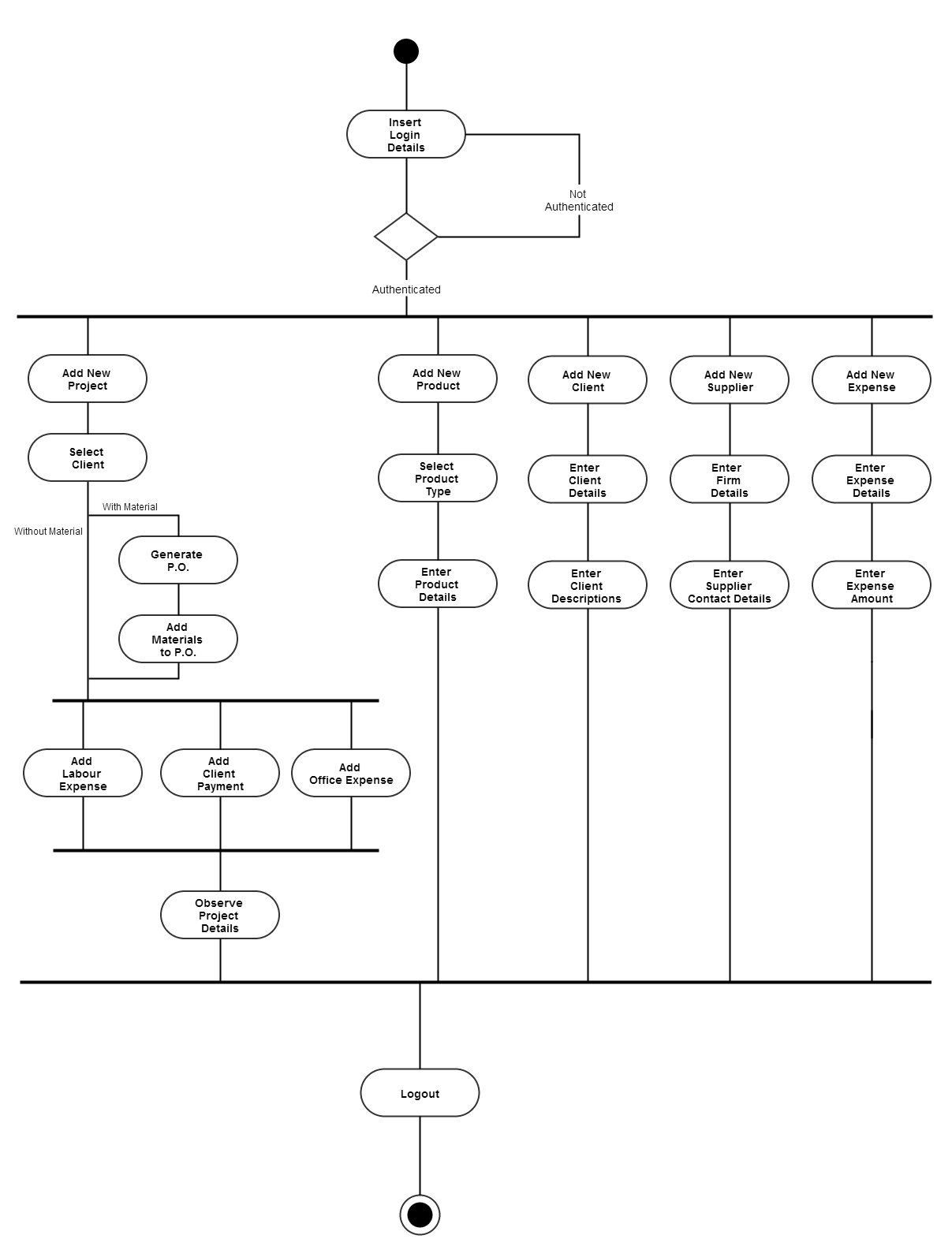
****

Fig : 6.3

**Activity Diagram for Client Management:**

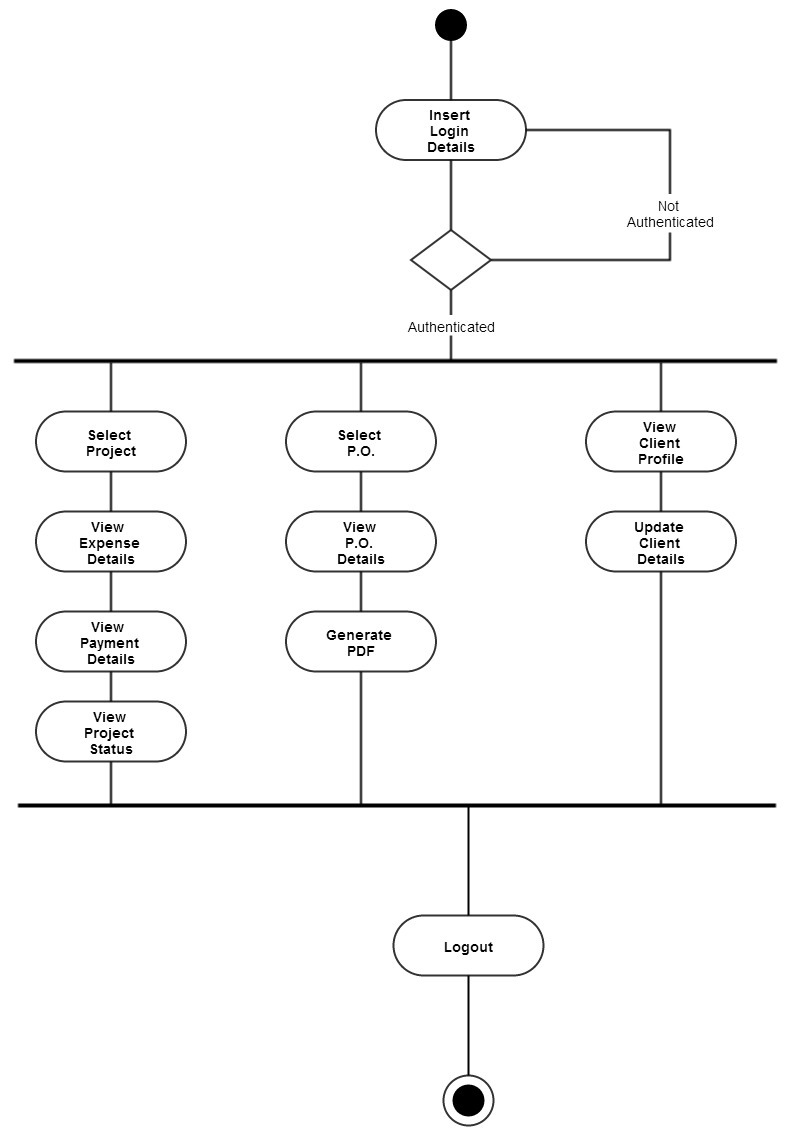
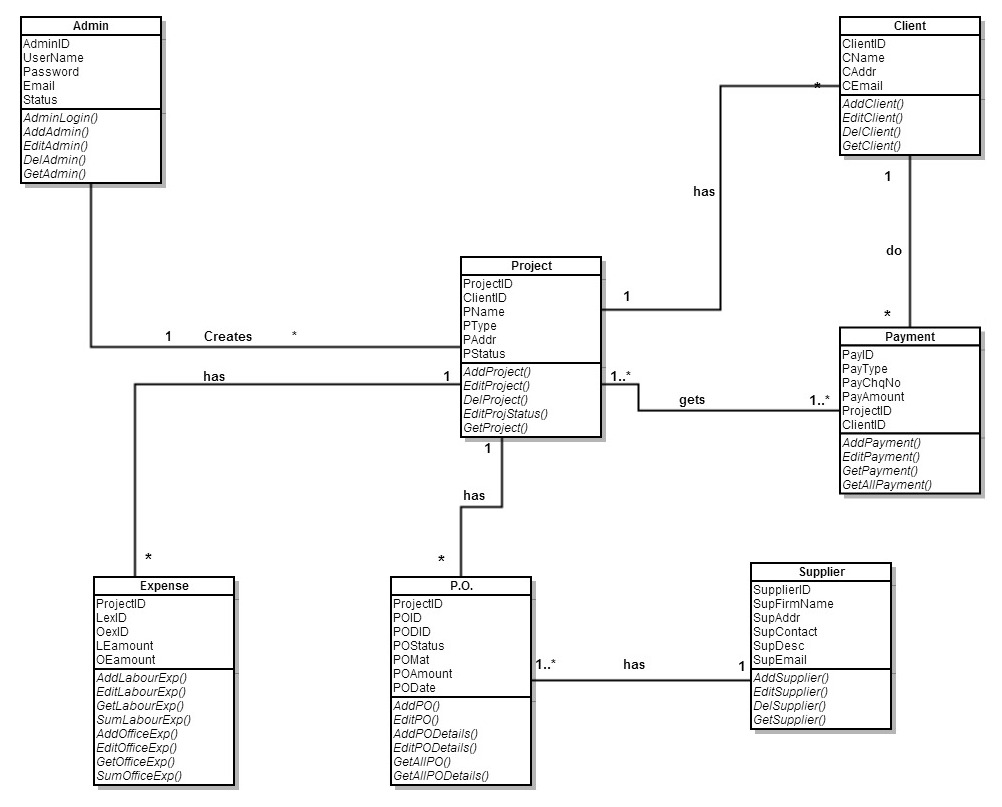
****

Fig : 6.4

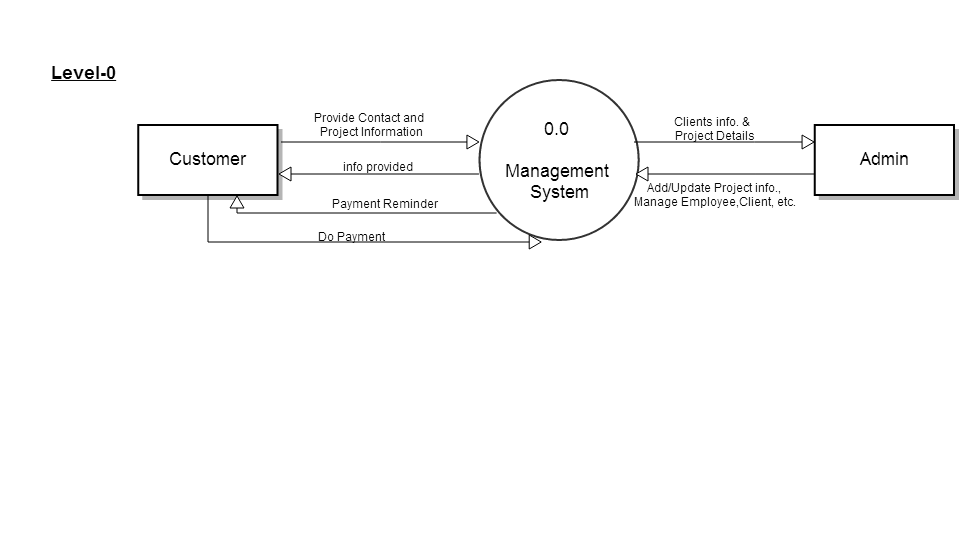
6.3 CLASS-DFD DIAGRAMS

**Class Diagram:**

****

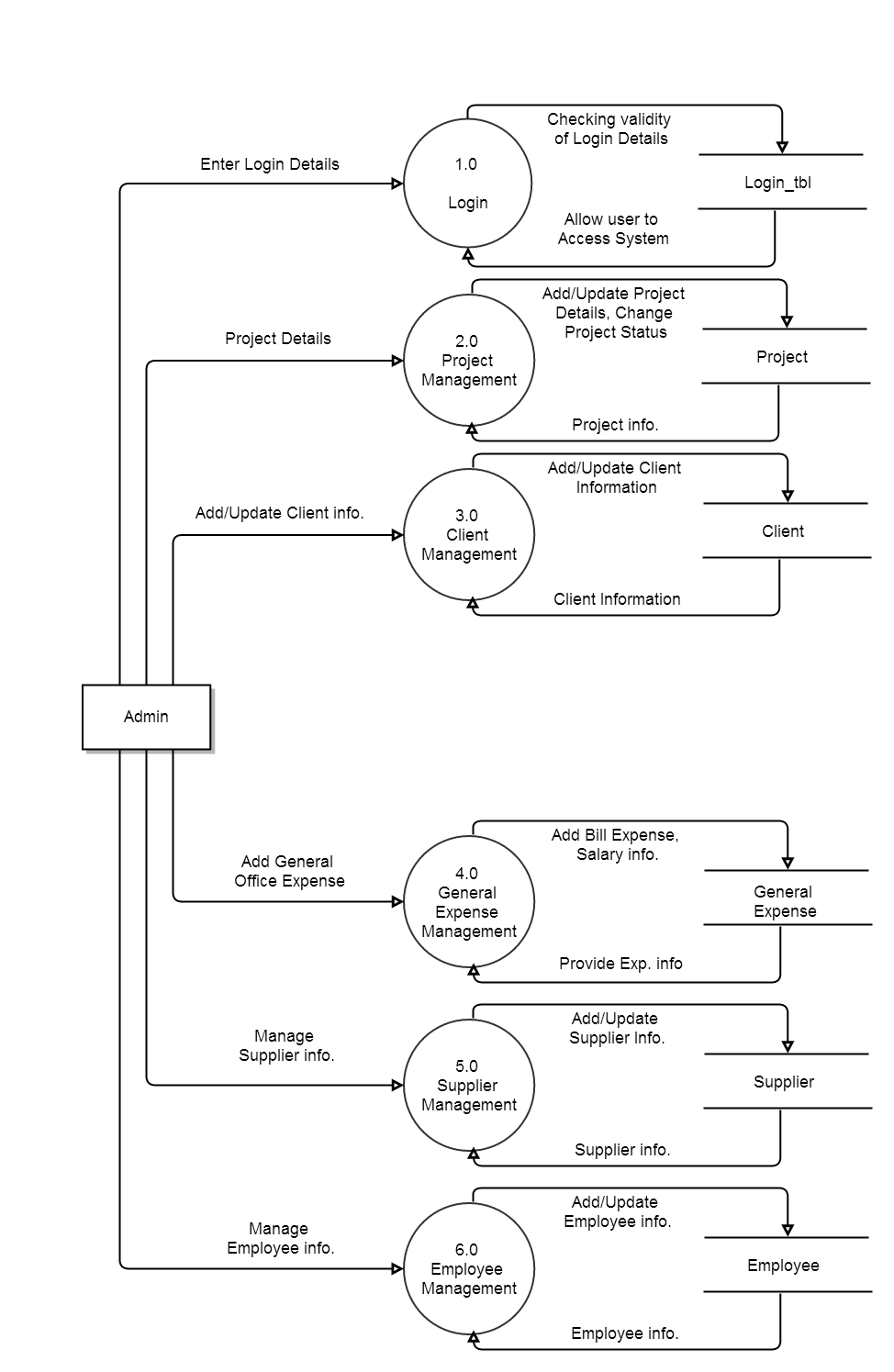
**Fig 6.5**

**DFD Level-0 :**

****

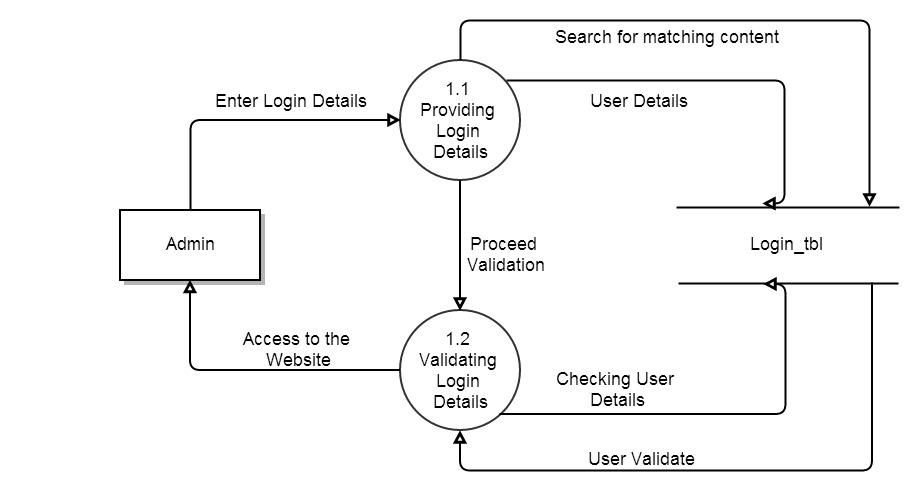
**Fig : 6.6**

**DFD Level-1:**

****

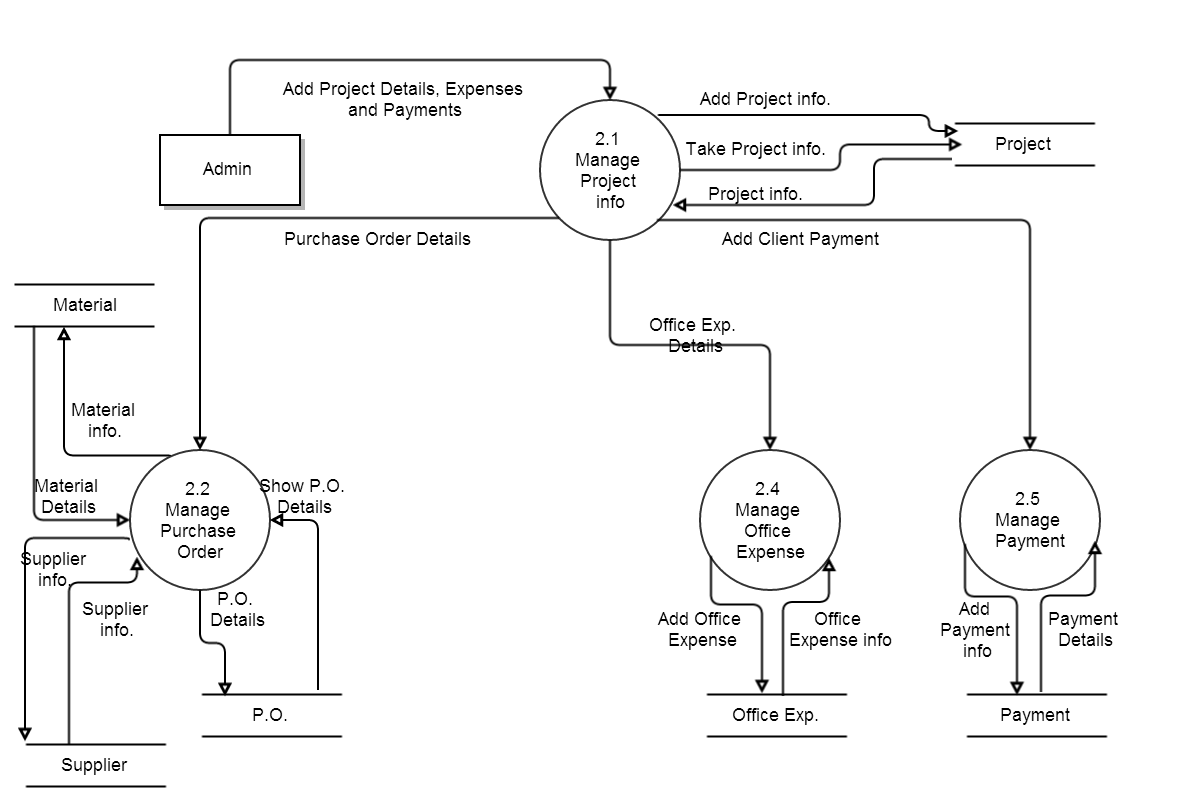
**Fig : 6.7**

**DFD Level-2 Login:**

****

**Fig : 6.8**

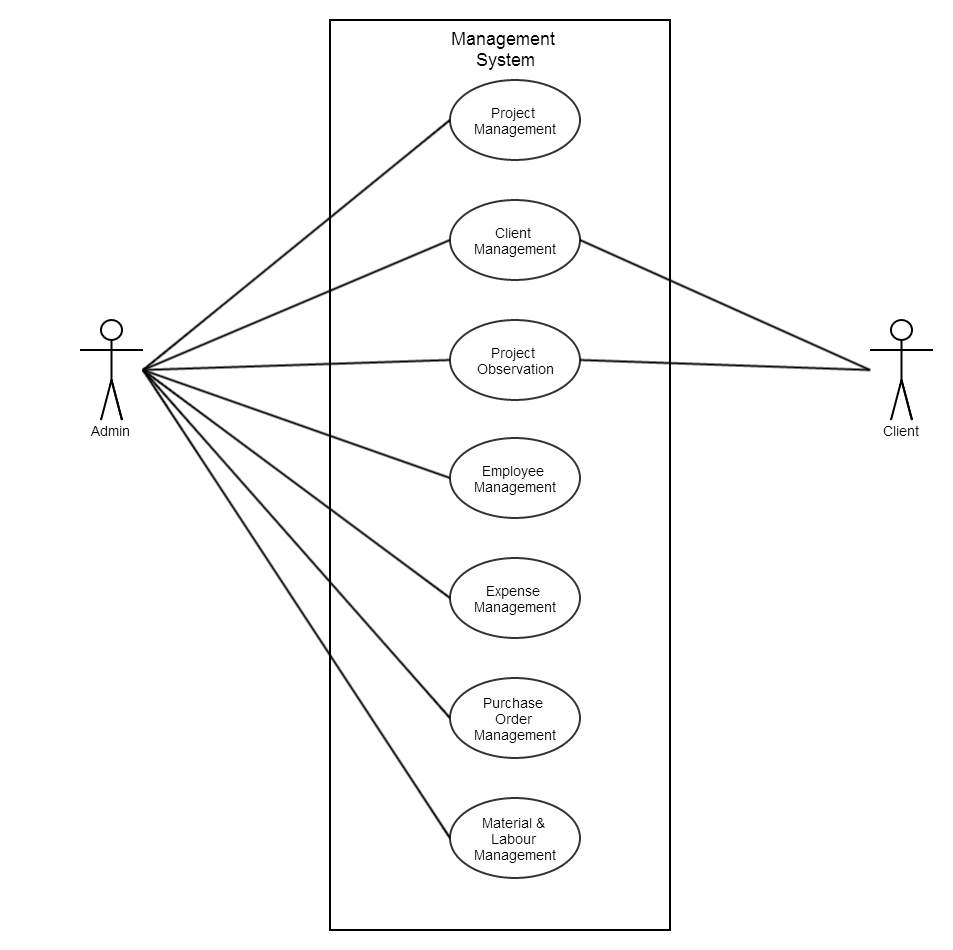
**DFD Level-2 Project Management:**

****

**Fig : 6.9**

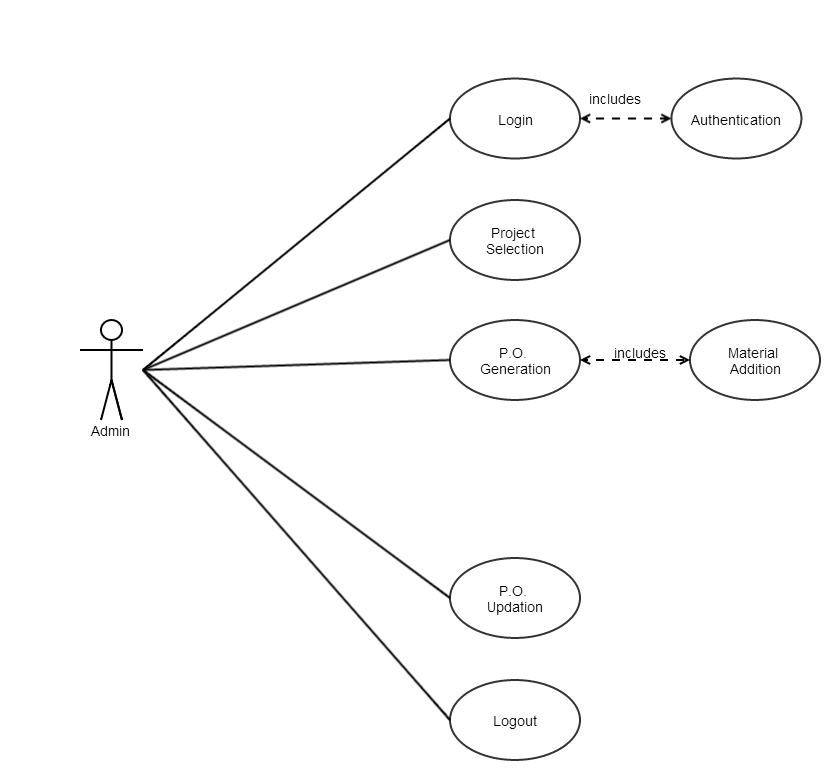
6.4 USE-CASE DIAGRAMS

**Use-case Diagram:**

****

**Fig : 6.10**

**Use-case Purchase Order:**

****

**Fig : 6.11**

**Use-case Project:**

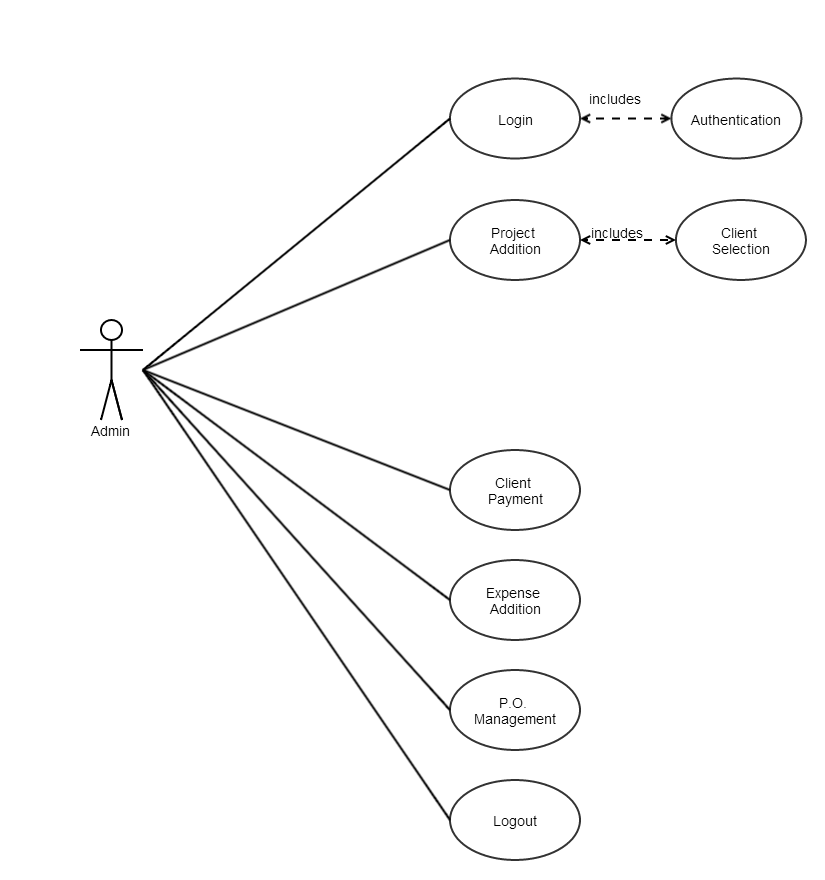
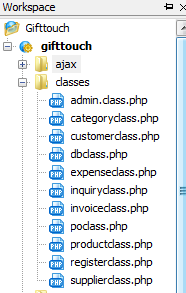
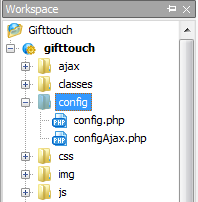
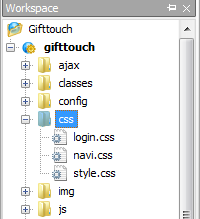
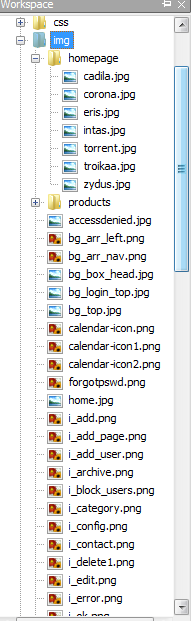
****

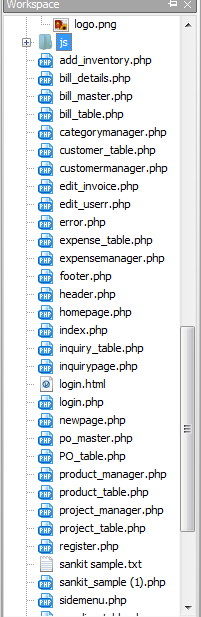
Fig : 6.126.5 SITE MAP

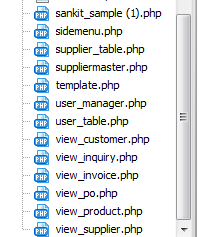
** Fig : 6.13 Class Files**

** Fig : 6.14 Configuration Files**

** Fig :6.15 CSS Files**

** Fig : 6.16 Image Files**

****

** Fig : 6.17 PHP Files**

**6.6 SCREENSHOTS**

**6.6.1 Login Page**

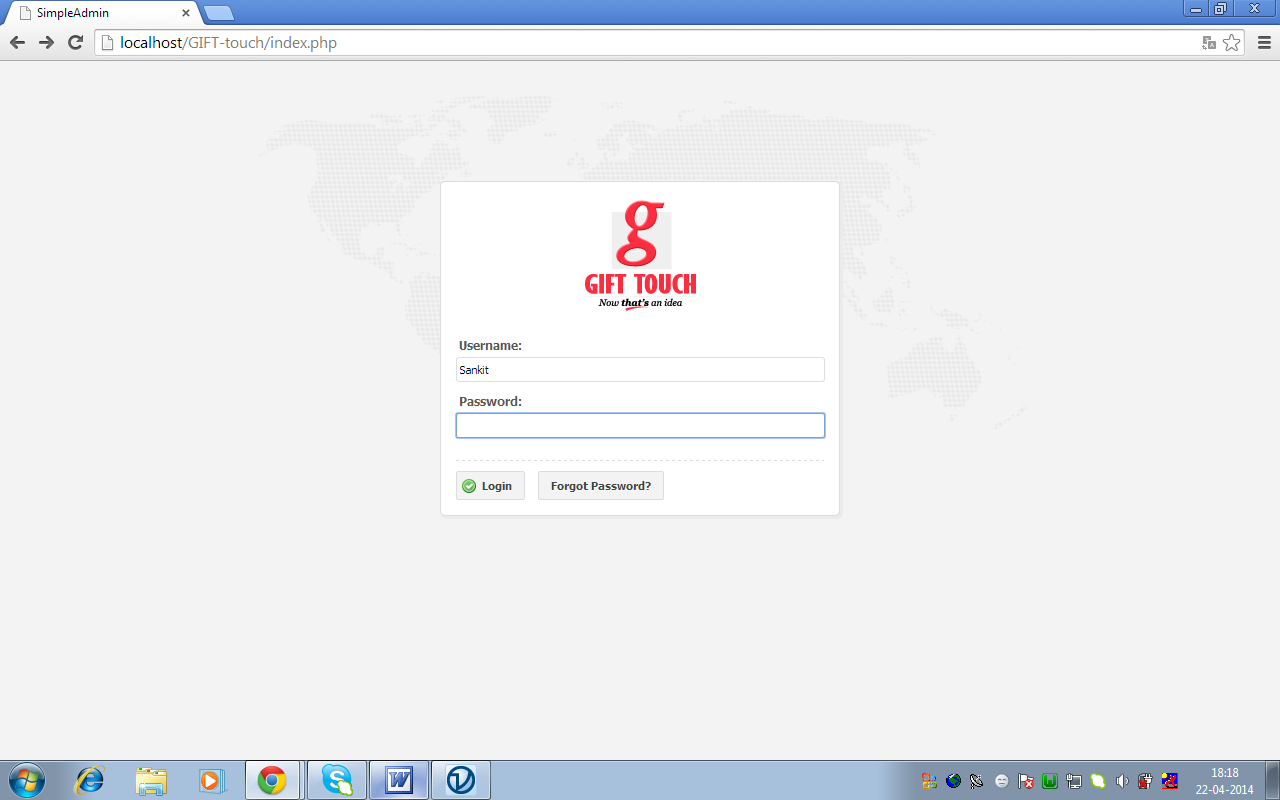


Fig 6.18 : Login Page

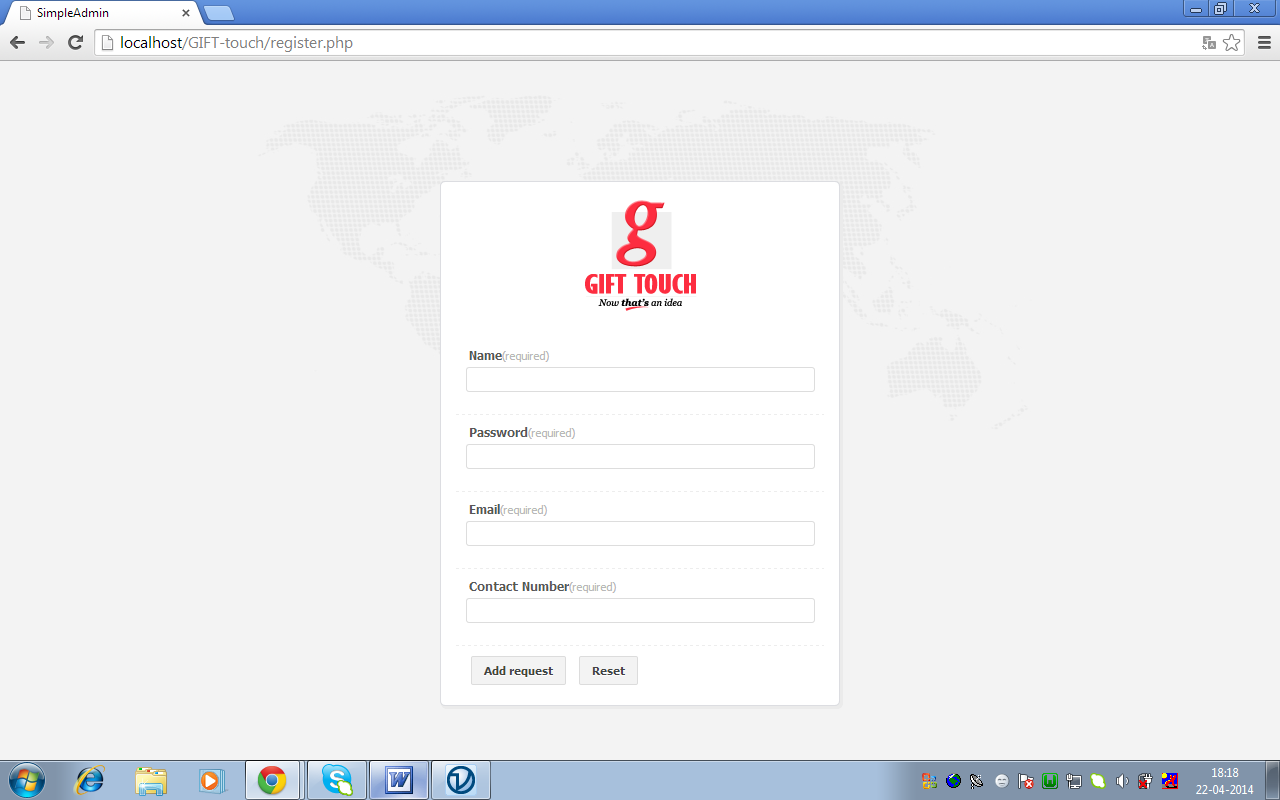


Fig 6.19 : Forgot Password/Registration Page

**6.6.2 Home Page**

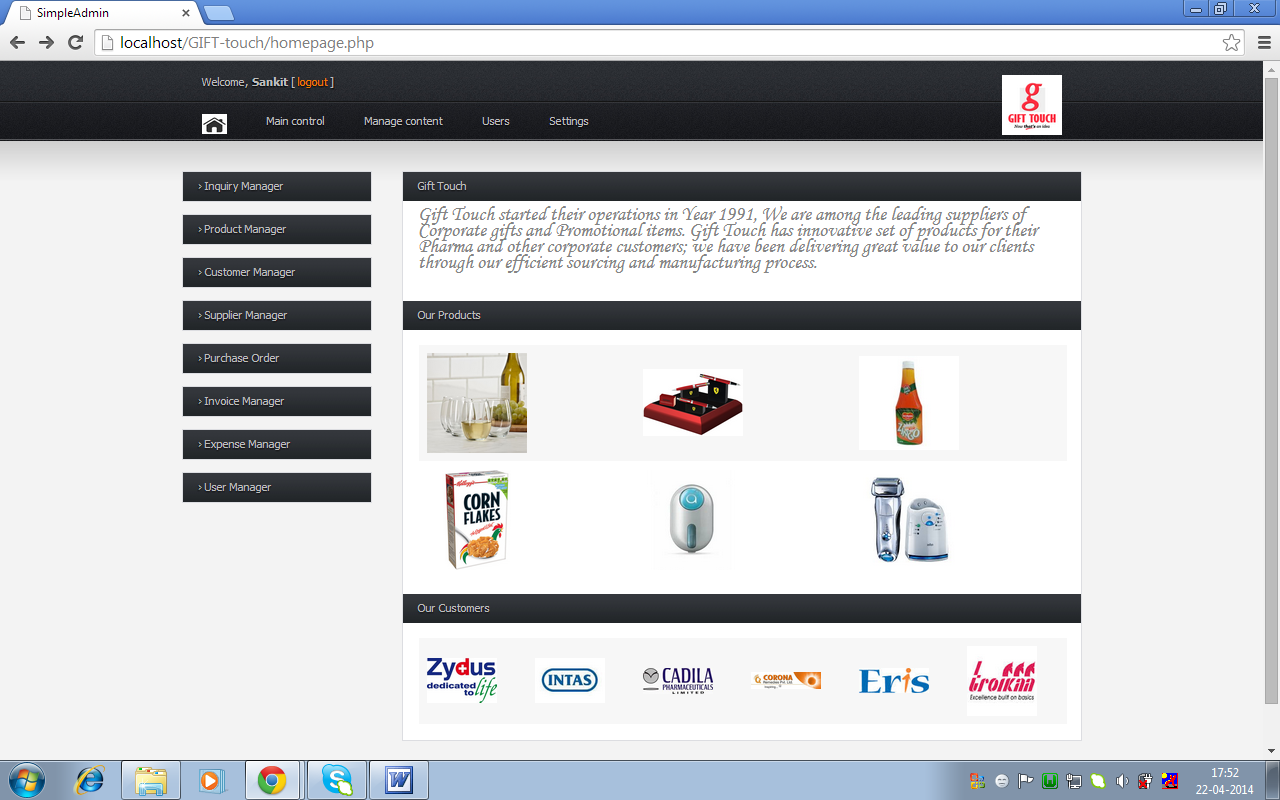


Fig 6.20 : Home Page(Admin)

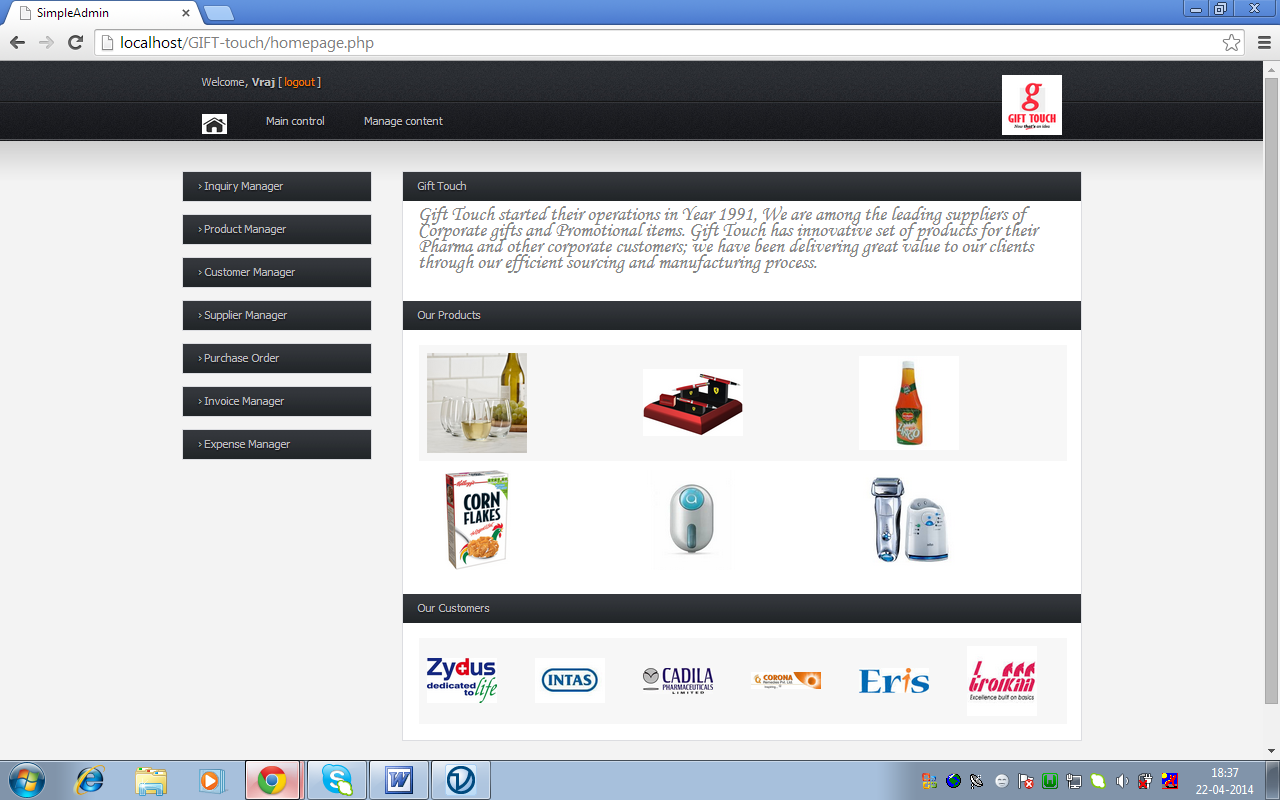


Fig 6.21 : Home Page(User)(Controlled Access)

**6.6.3 Product Menu**

Fig 6.22 : Show Product List

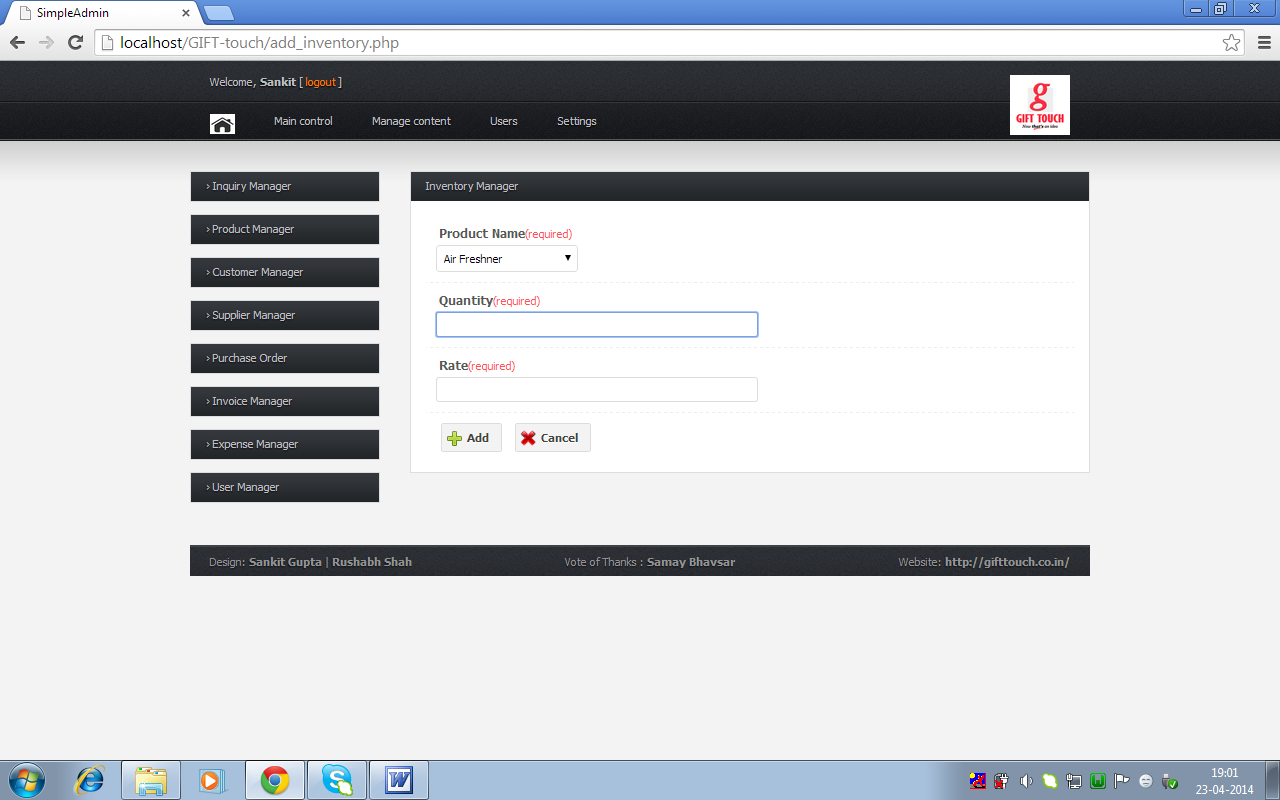


Fig 6.23 : Add to Inventory

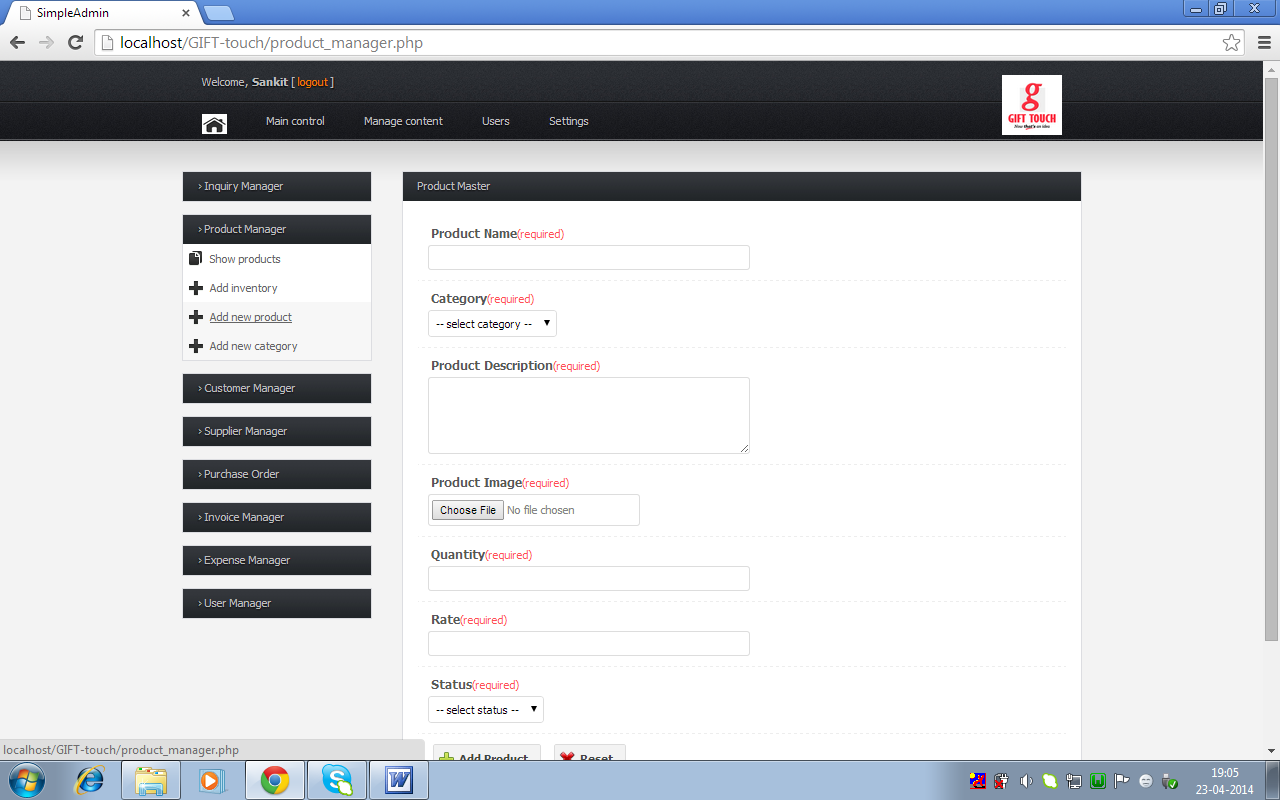


Fig 6.24 : Add Product

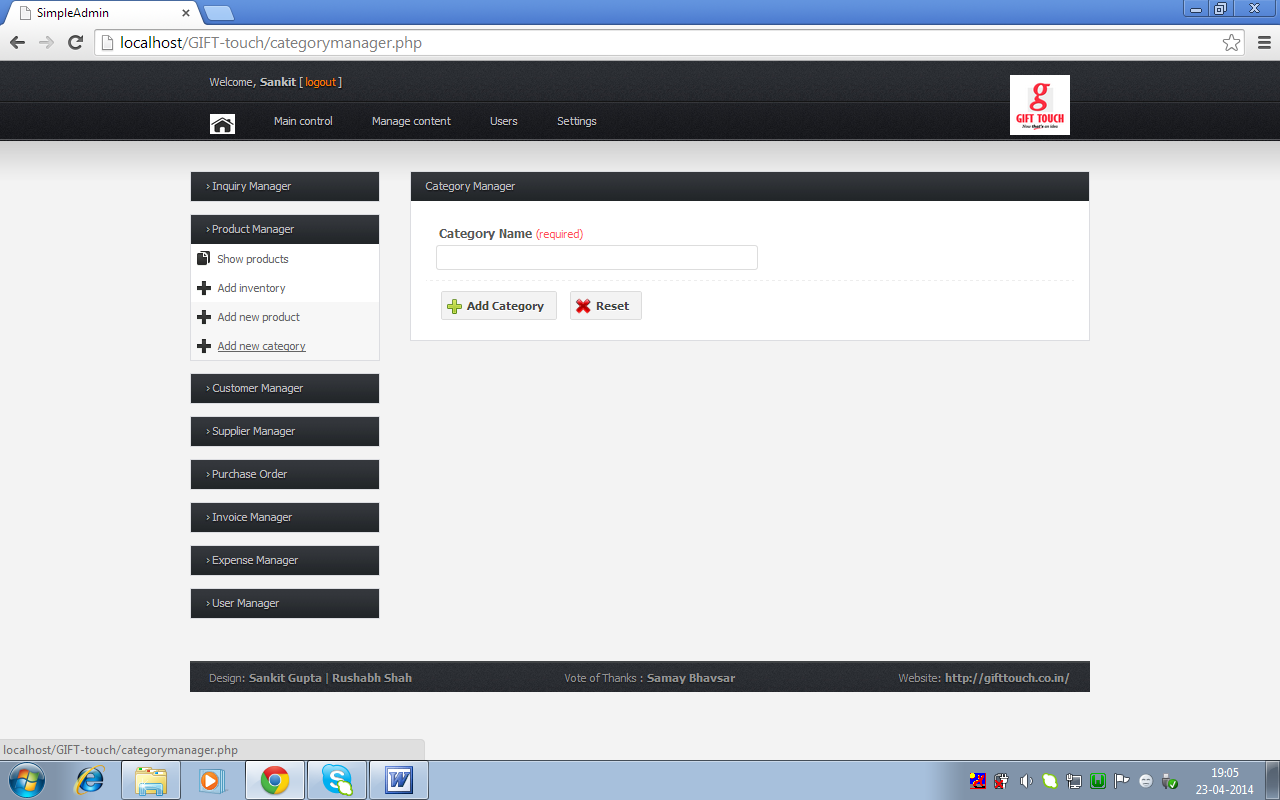


Fig 6.25 : Add Category

**6.6.4 Inquiry Menu**

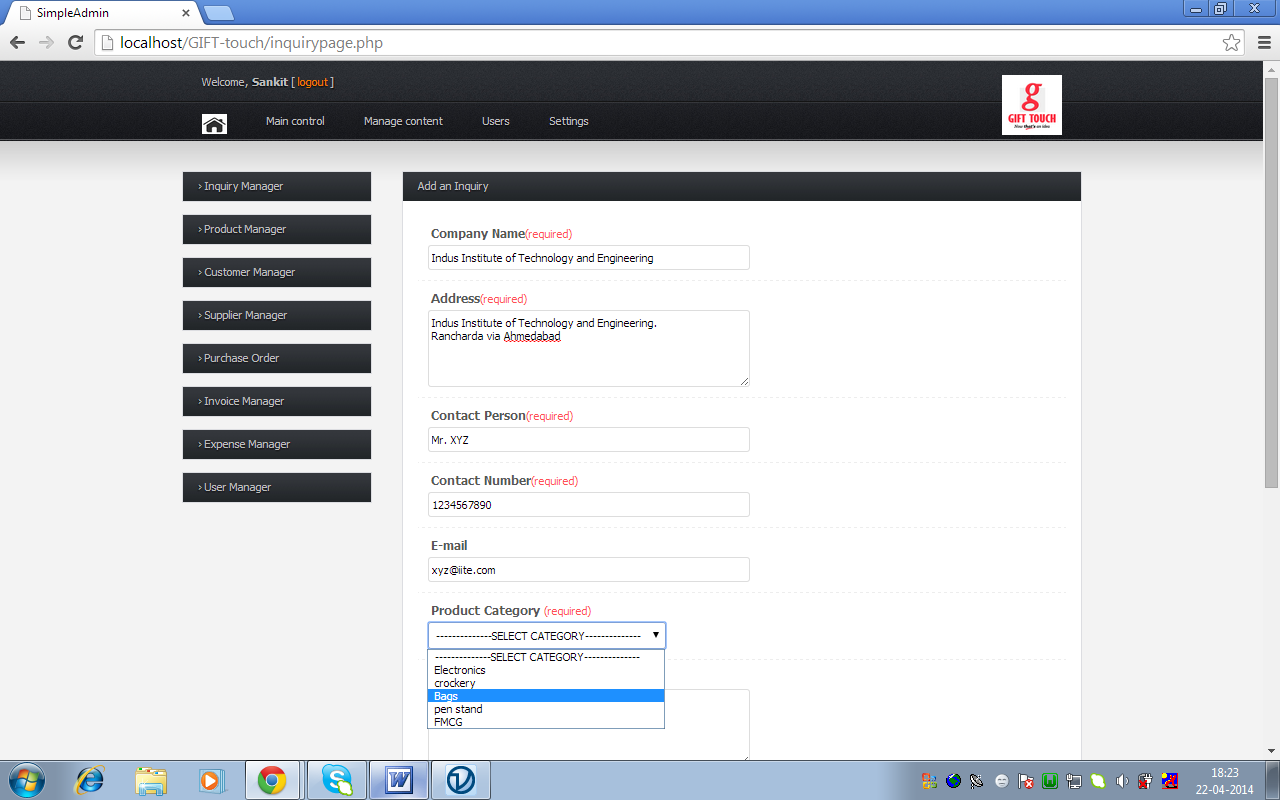


Fig 6.26 : Inquiry Page

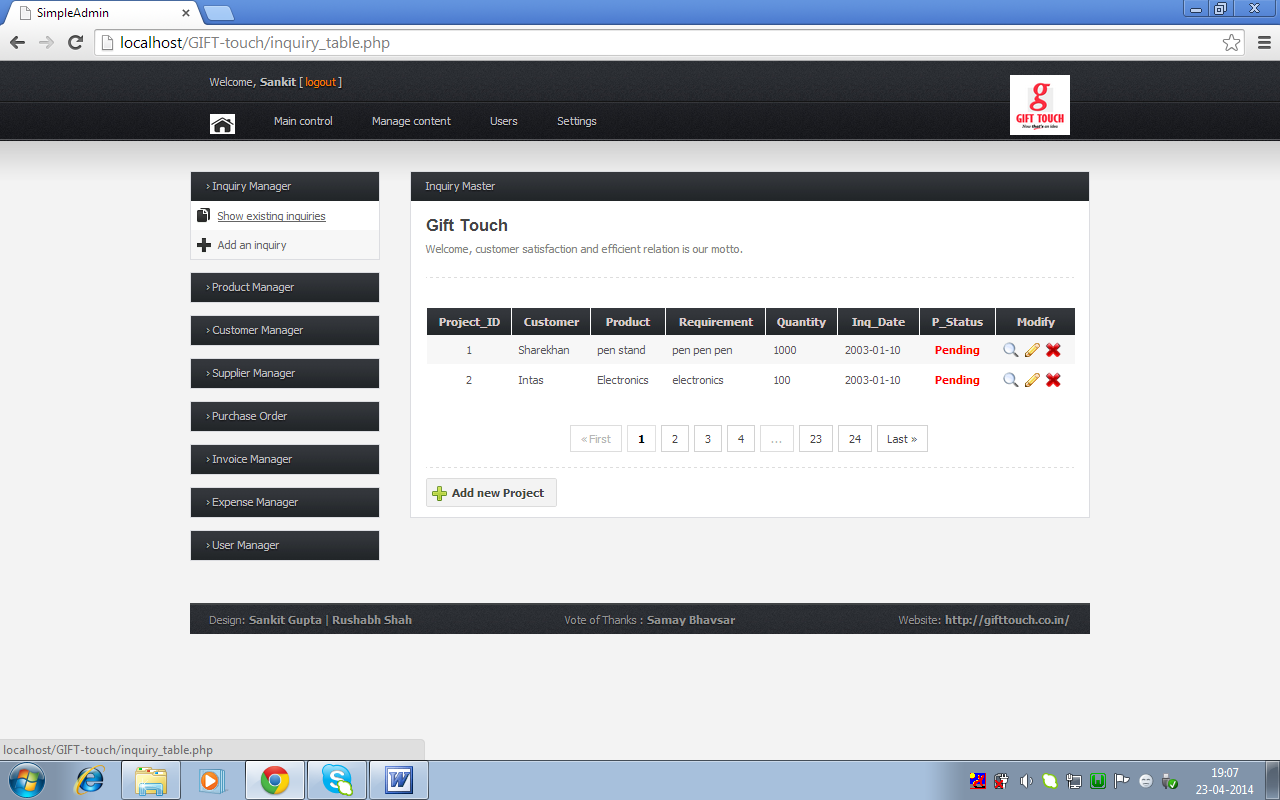


Fig 6.27 : Show Inquiry

**6.6.5 Purchase Order Menu**

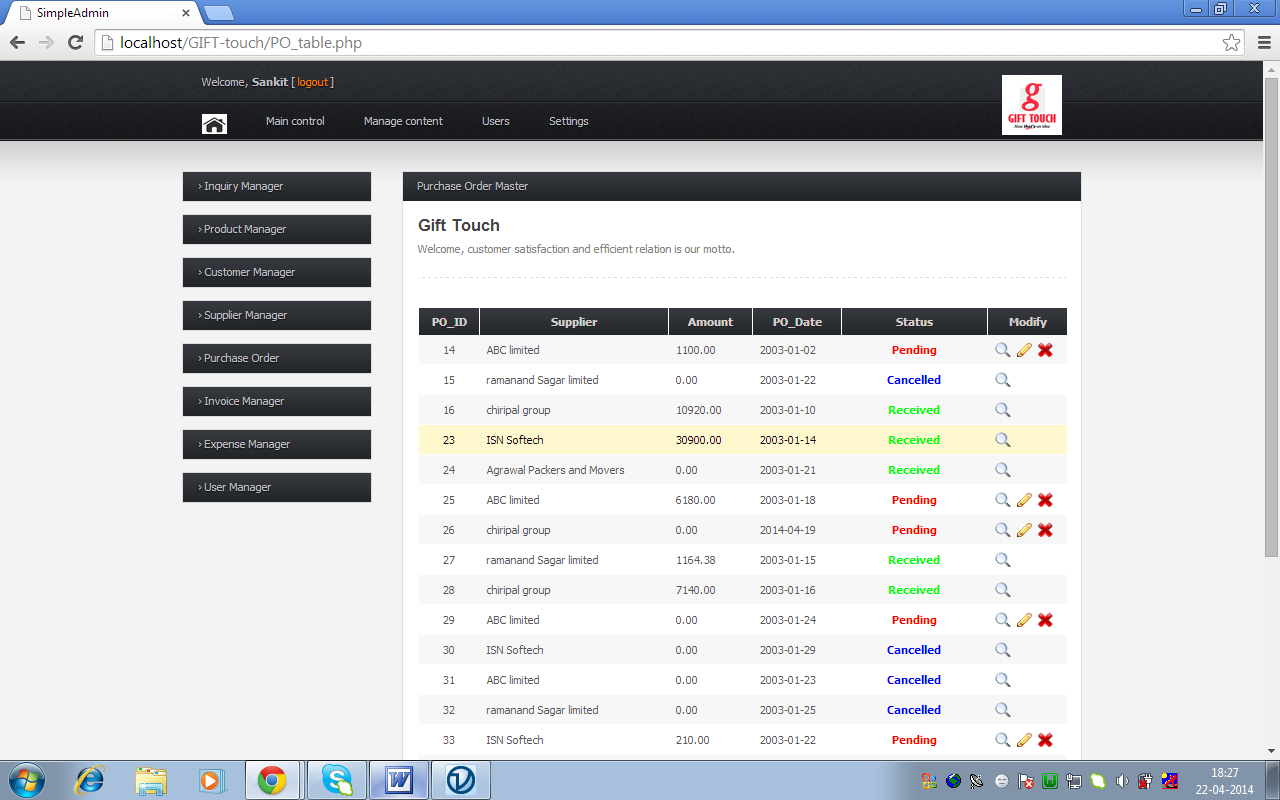


Fig 6.28 : Purchase Order Table

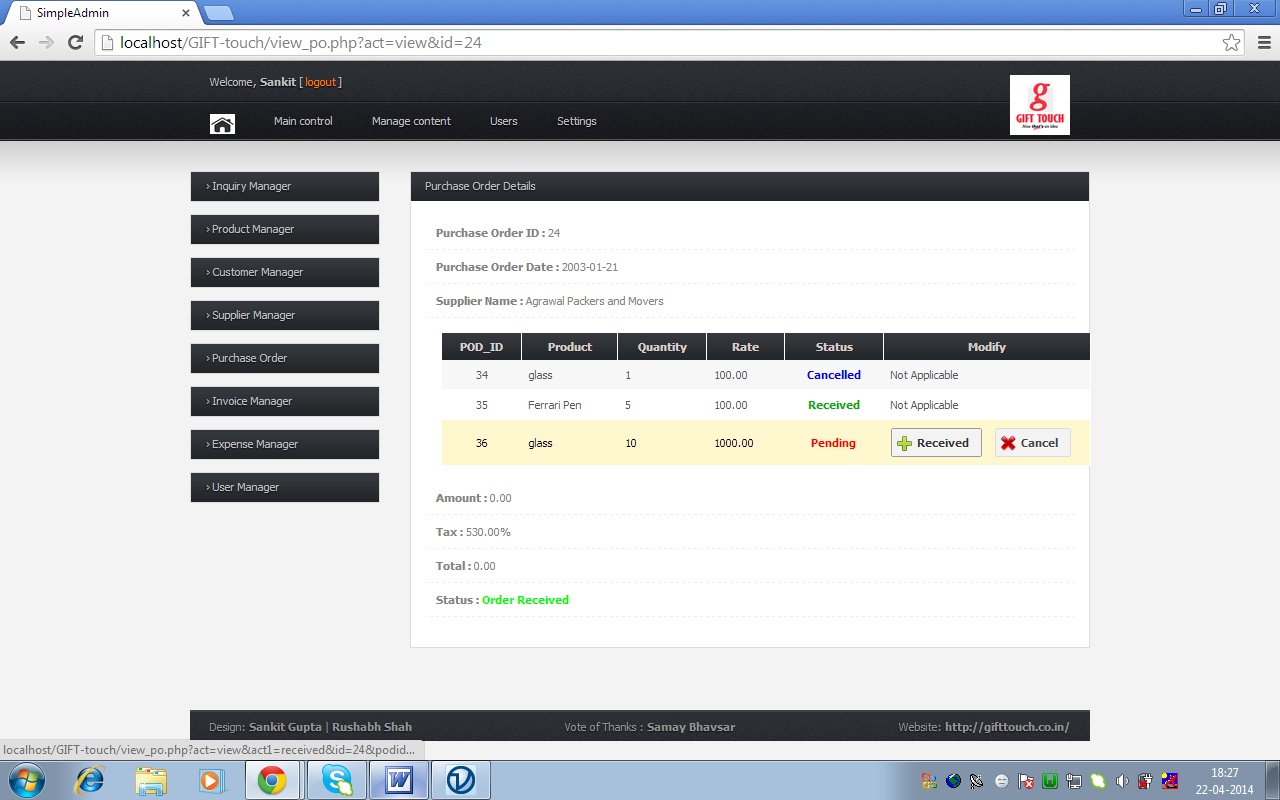


Fig 6.29 : Purchase Order View

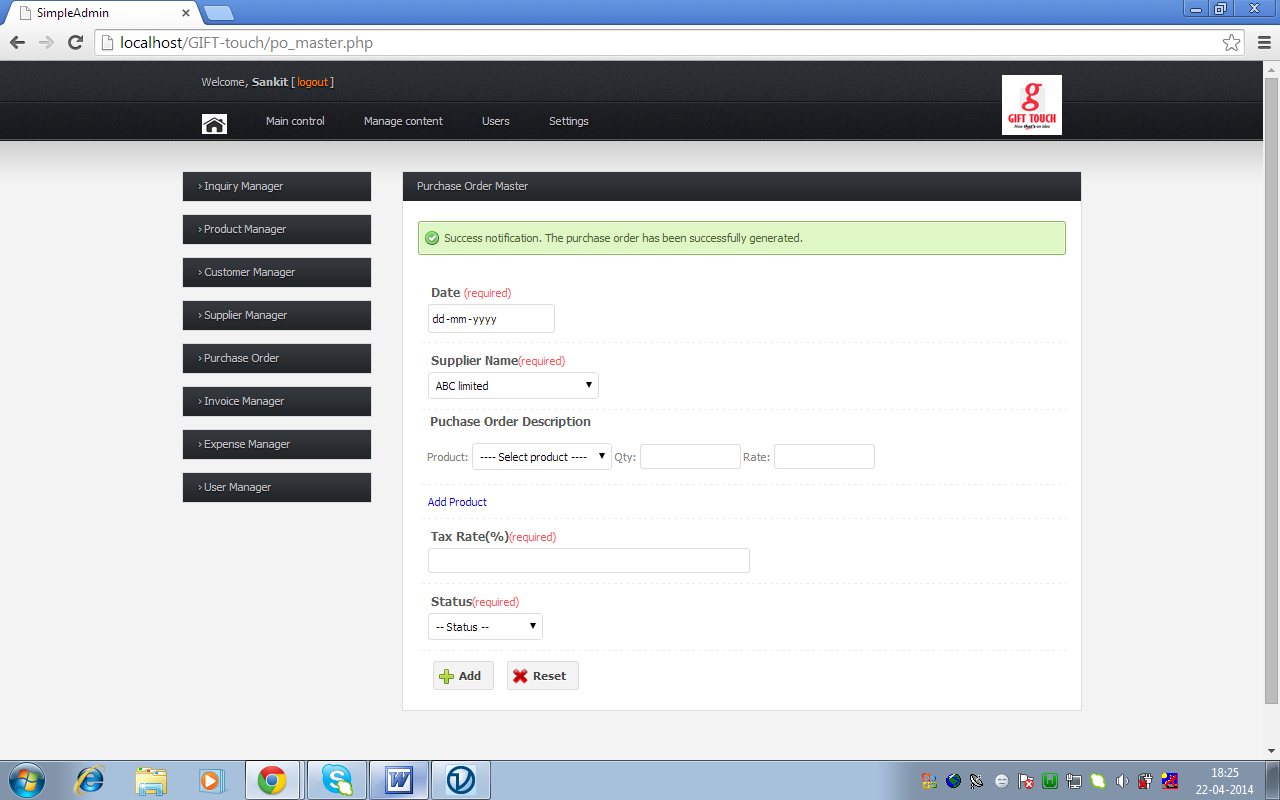


Fig 6.30 : Purchase Order Generation

**6.6.6 User Manager**

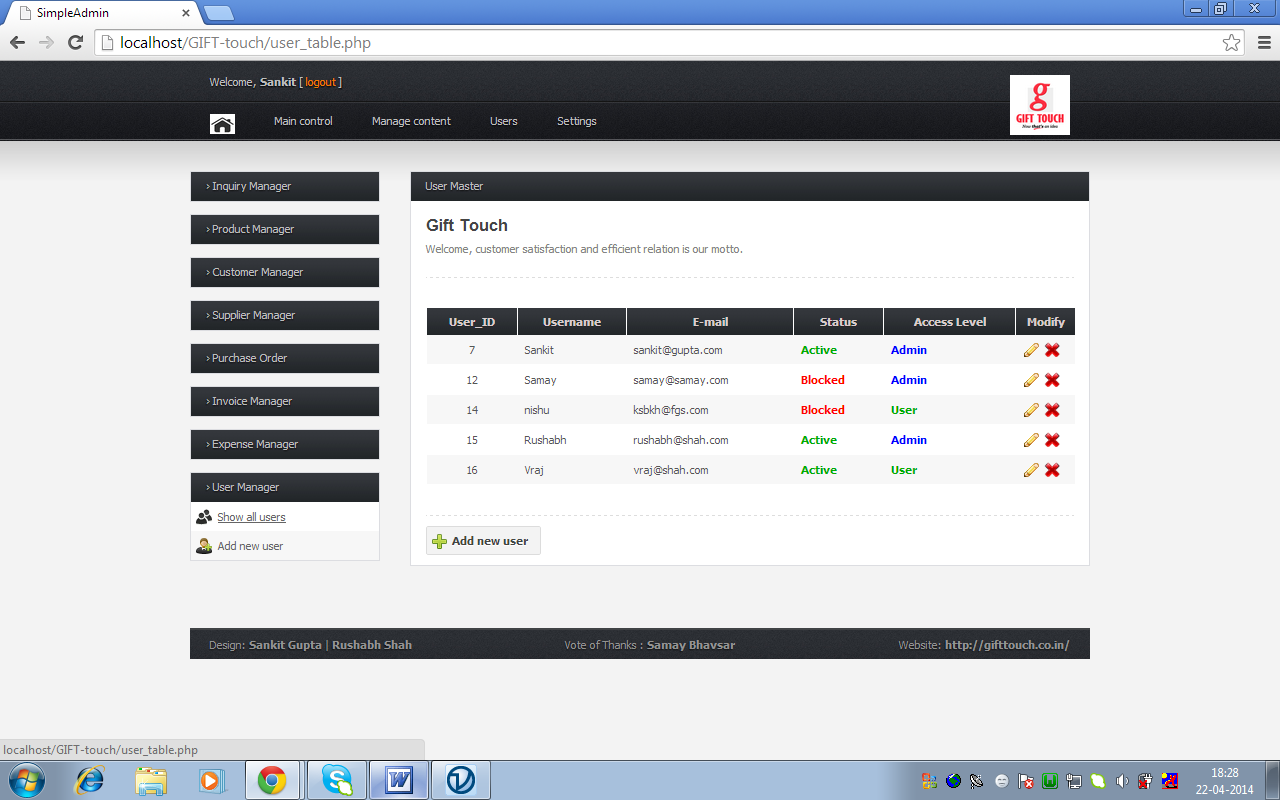


Fig 6.31 : User Manager

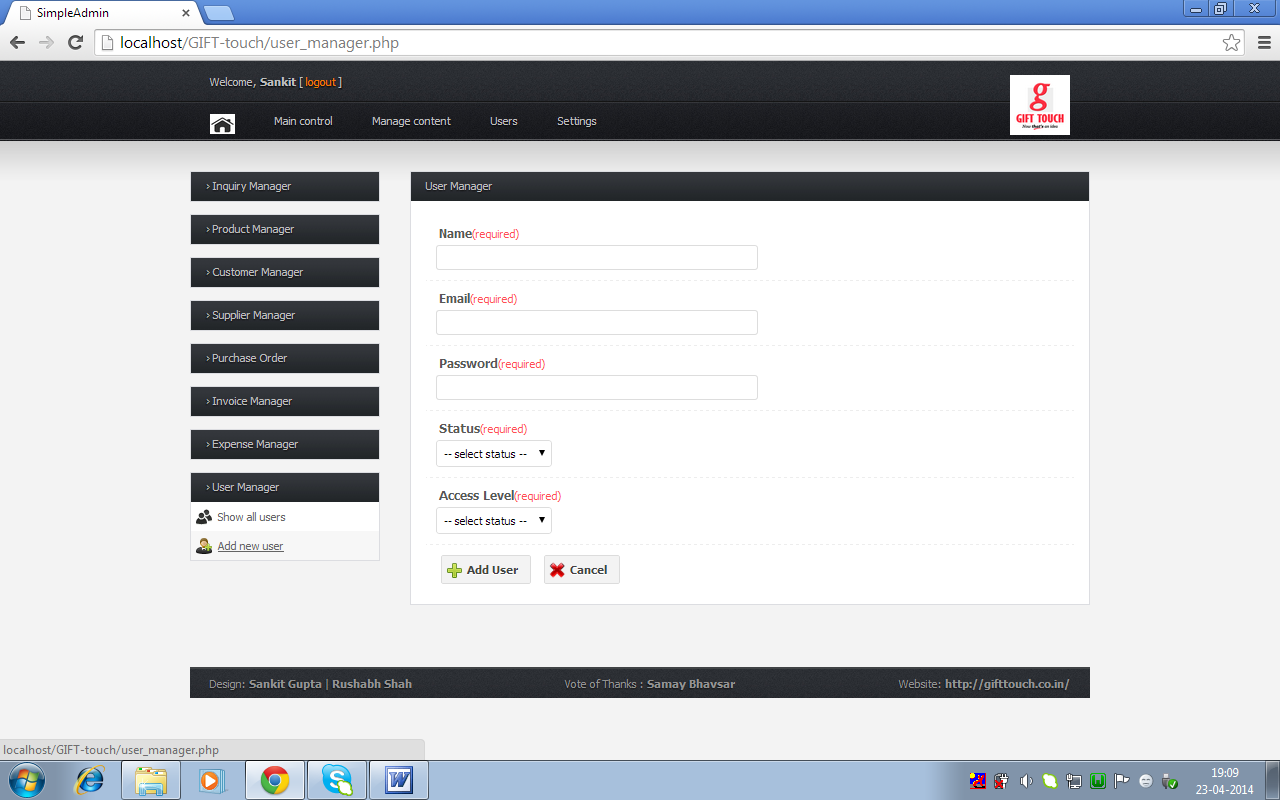


Fig 6.32 : Add User

**6.6.7 Notifications**

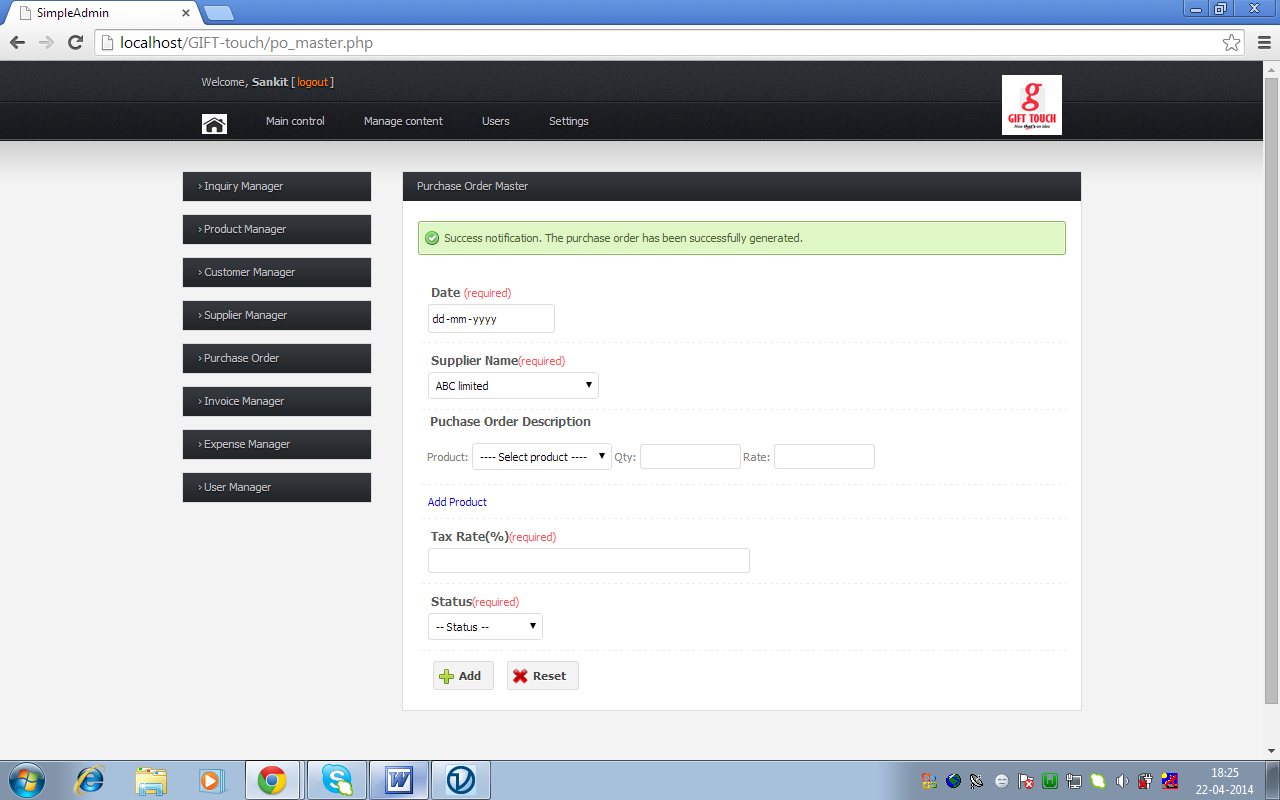


Fig 6.33 : Success Notification

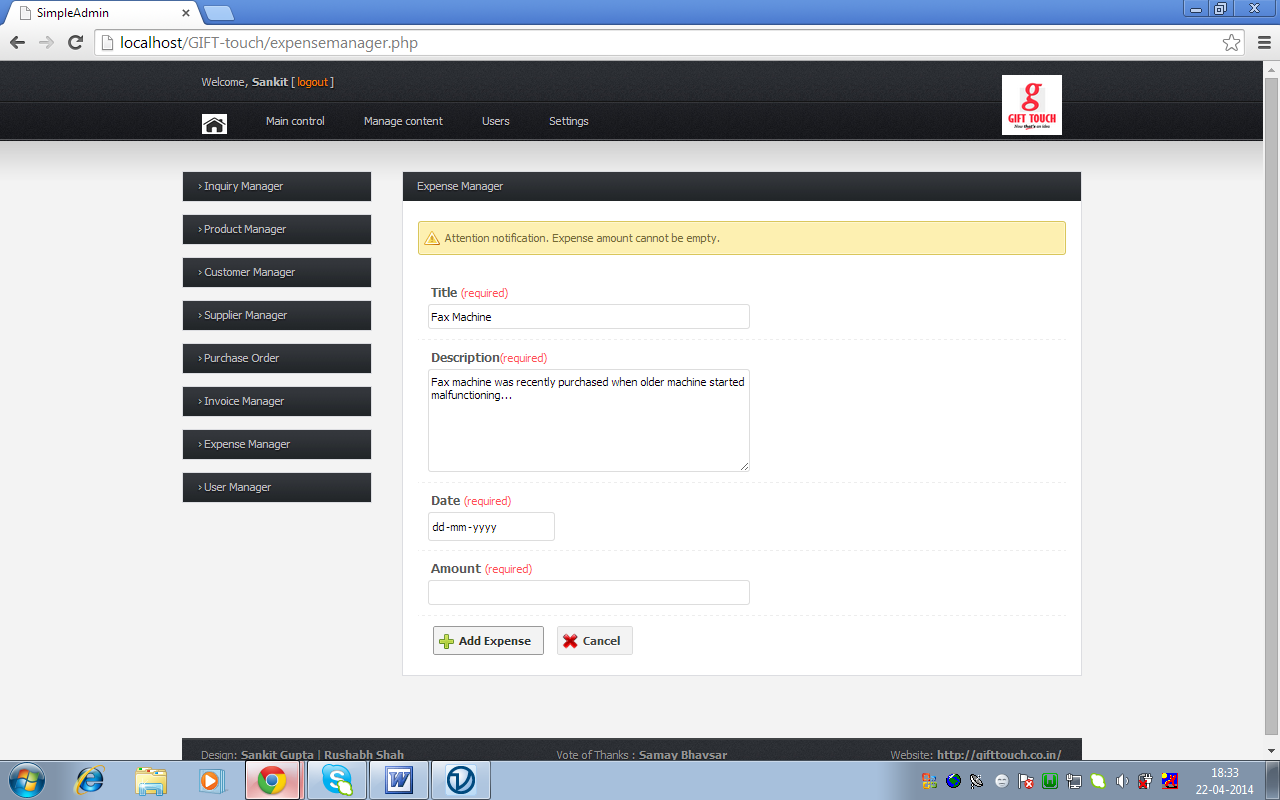


Fig 6.34 : Essential Field Error



Fig 6.35 : Validation Error

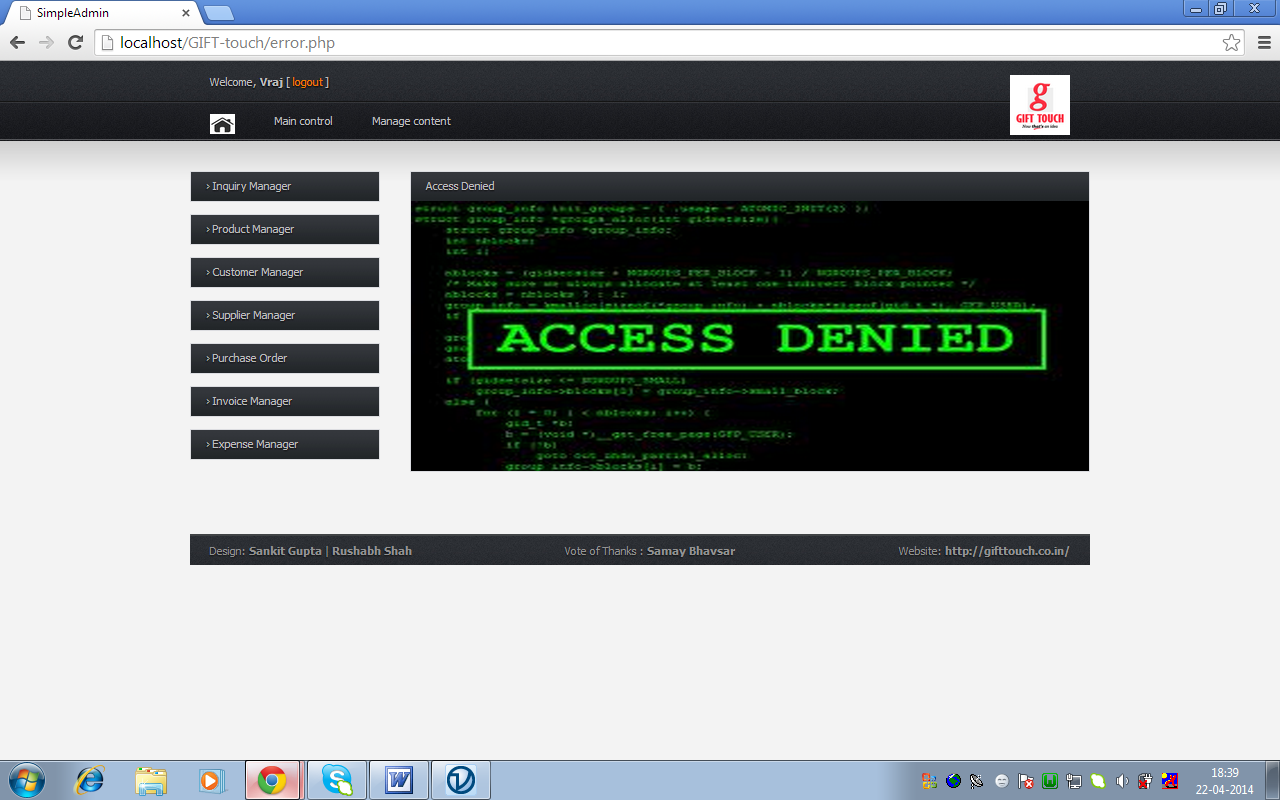


Fig 6.36: Access Level Error

**6.6.8 Miscellaneous**

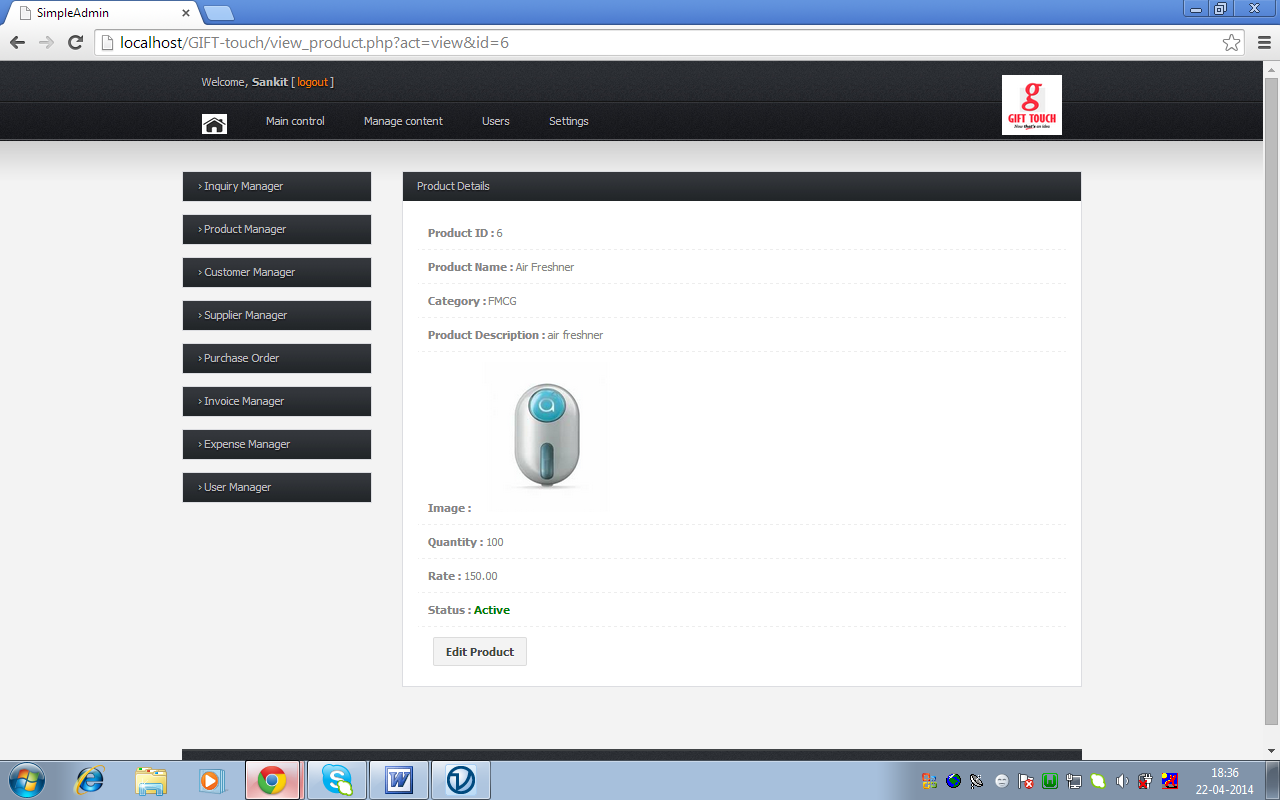


Fig 6.37 : Show/View Page

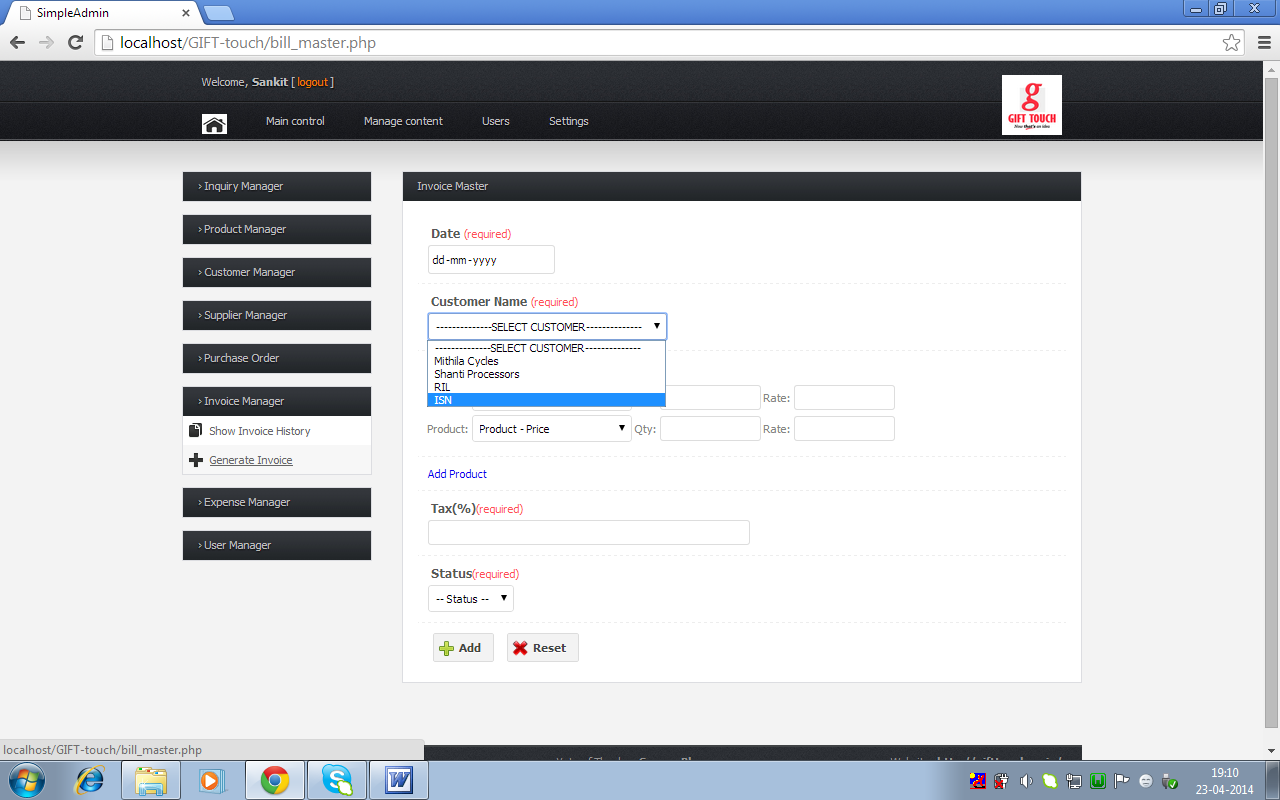


Fig 6.38 : Invoice generation

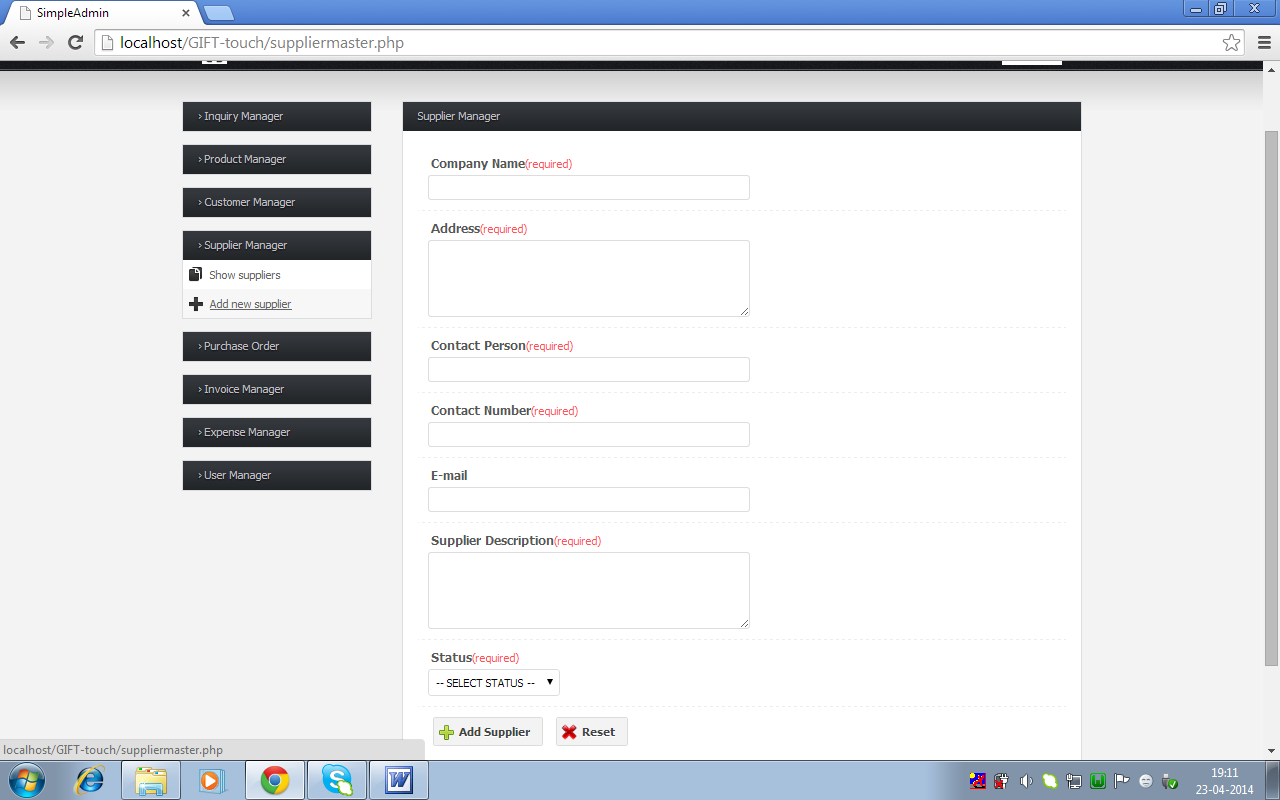


Fig 6.39 : Supplier Manager

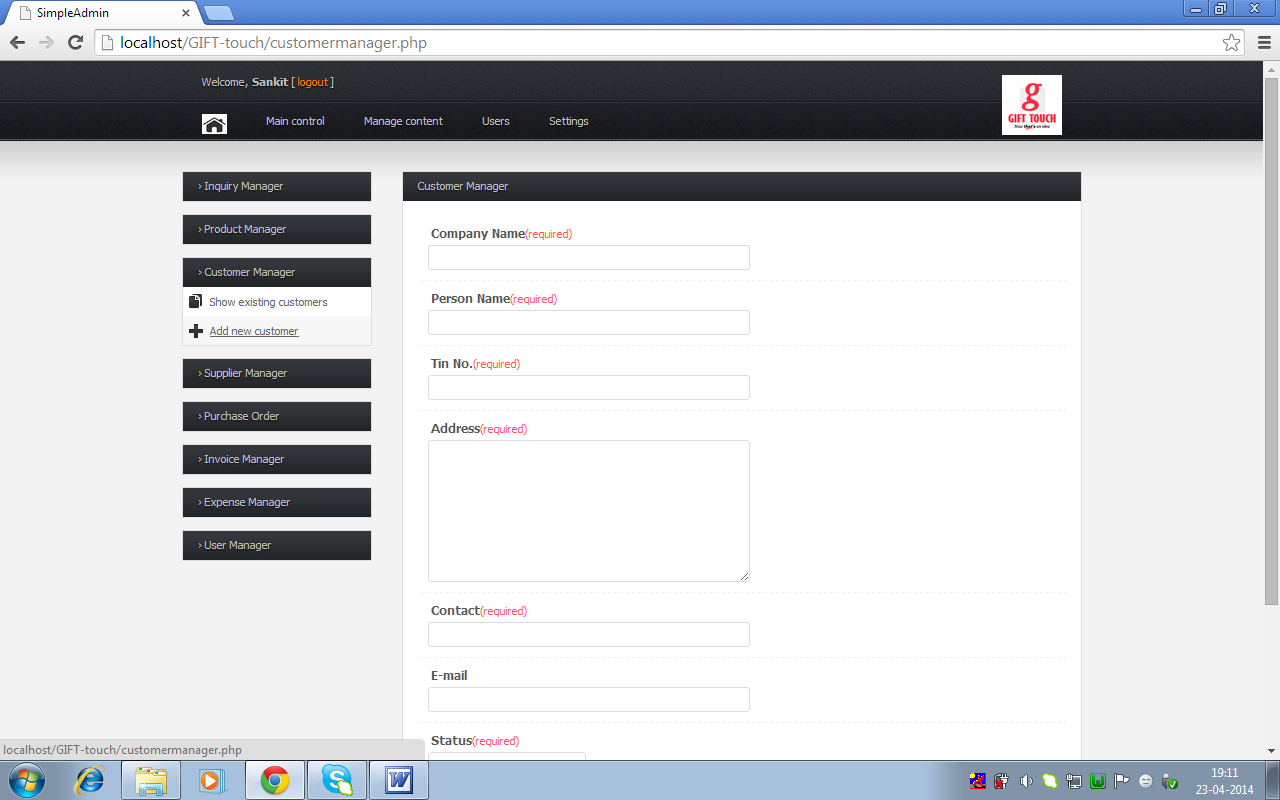


Fig 6.40 : Customer Manager

| **Action** | | | | | | | | [**Type**](http://localhost/phpmyadmin/db_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&pos=0&sort=type&sort_order=ASC) | [**Collation**](http://localhost/phpmyadmin/db_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&pos=0&sort=collation&sort_order=ASC) | | [**Overhead**](http://localhost/phpmyadmin/db_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&pos=0&sort=overhead&sort_order=DESC) |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **category\_master** | [Browse](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=category_master&pos=0) | [Structure](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=category_master) | | [Search](http://localhost/phpmyadmin/tbl_select.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=category_master) | [Insert](http://localhost/phpmyadmin/tbl_change.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=category_master) | | [Empty](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=category_master&sql_query=TRUNCATE+%60category_master%60&zero_rows=Table+category_master+has+been+emptied) | 5 | InnoDB | | [32.0 KiB](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=category_master#showusage) | - |
| **customer\_master** | [Browse](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=customer_master&pos=0) | [Structure](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=customer_master) | | [Search](http://localhost/phpmyadmin/tbl_select.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=customer_master) | [Insert](http://localhost/phpmyadmin/tbl_change.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=customer_master) | | [Empty](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=customer_master&sql_query=TRUNCATE+%60customer_master%60&zero_rows=Table+customer_master+has+been+emptied) | 4 | InnoDB | | [32.0 KiB](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=customer_master#showusage) | - |
| **expense\_manager** | [Browse](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=expense_manager&pos=0) | [Structure](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=expense_manager) | | [Search](http://localhost/phpmyadmin/tbl_select.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=expense_manager) | [Insert](http://localhost/phpmyadmin/tbl_change.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=expense_manager) | | [Empty](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=expense_manager&sql_query=TRUNCATE+%60expense_manager%60&zero_rows=Table+expense_manager+has+been+emptied) | 2 | InnoDB | | [16.0 KiB](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=expense_manager#showusage) | - |
| **inventory** | [Browse](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=inventory&pos=0) | [Structure](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=inventory) | | [Search](http://localhost/phpmyadmin/tbl_select.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=inventory) | [Insert](http://localhost/phpmyadmin/tbl_change.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=inventory) | | [Empty](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=inventory&sql_query=TRUNCATE+%60inventory%60&zero_rows=Table+inventory+has+been+emptied) | 2 | InnoDB | | [16.0 KiB](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=inventory#showusage) | - |
| **invoice\_description** | [Browse](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=invoice_description&pos=0) | [Structure](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=invoice_description) | | [Search](http://localhost/phpmyadmin/tbl_select.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=invoice_description) | [Insert](http://localhost/phpmyadmin/tbl_change.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=invoice_description) | | [Empty](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=invoice_description&sql_query=TRUNCATE+%60invoice_description%60&zero_rows=Table+invoice_description+has+been+emptied) | 12 | InnoDB | | [16.0 KiB](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=invoice_description#showusage) | - |
| **invoice\_master** | [Browse](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=invoice_master&pos=0) | [Structure](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=invoice_master) | | [Search](http://localhost/phpmyadmin/tbl_select.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=invoice_master) | [Insert](http://localhost/phpmyadmin/tbl_change.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=invoice_master) | | [Empty](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=invoice_master&sql_query=TRUNCATE+%60invoice_master%60&zero_rows=Table+invoice_master+has+been+emptied) | 7 | InnoDB | | [16.0 KiB](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=invoice_master#showusage) | - |
| **payorder\_description** | [Browse](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=payorder_description&pos=0) | [Structure](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=payorder_description) | | [Search](http://localhost/phpmyadmin/tbl_select.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=payorder_description) | [Insert](http://localhost/phpmyadmin/tbl_change.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=payorder_description) | | [Empty](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=payorder_description&sql_query=TRUNCATE+%60payorder_description%60&zero_rows=Table+payorder_description+has+been+emptied) | 36 | InnoDB | | [16.0 KiB](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=payorder_description#showusage) | - |
| **payorder\_master** | [Browse](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=payorder_master&pos=0) | [Structure](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=payorder_master) | | [Search](http://localhost/phpmyadmin/tbl_select.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=payorder_master) | [Insert](http://localhost/phpmyadmin/tbl_change.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=payorder_master) | | [Empty](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=payorder_master&sql_query=TRUNCATE+%60payorder_master%60&zero_rows=Table+payorder_master+has+been+emptied) | 16 | InnoDB | | [16.0 KiB](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=payorder_master#showusage) | - |
| **product\_master** | [Browse](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=product_master&pos=0) | [Structure](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=product_master) | | [Search](http://localhost/phpmyadmin/tbl_select.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=product_master) | [Insert](http://localhost/phpmyadmin/tbl_change.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=product_master) | | [Empty](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=product_master&sql_query=TRUNCATE+%60product_master%60&zero_rows=Table+product_master+has+been+emptied) | 6 | InnoDB | | [32.0 KiB](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=product_master#showusage) | - |
| **project\_manager** | [Browse](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=project_manager&pos=0) | [Structure](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=project_manager) | | [Search](http://localhost/phpmyadmin/tbl_select.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=project_manager) | [Insert](http://localhost/phpmyadmin/tbl_change.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=project_manager) | | [Empty](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=project_manager&sql_query=TRUNCATE+%60project_manager%60&zero_rows=Table+project_manager+has+been+emptied) | 2 | InnoDB | | [16.0 KiB](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=project_manager#showusage) | - |
| **registration** | Browse | [Structure](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=registration) | | Search | [Insert](http://localhost/phpmyadmin/tbl_change.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=registration) | | Empty | 0 | InnoDB | | [16.0 KiB](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=registration#showusage) | - |
| **supplier\_master** | [Browse](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=supplier_master&pos=0) | [Structure](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=supplier_master) | | [Search](http://localhost/phpmyadmin/tbl_select.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=supplier_master) | [Insert](http://localhost/phpmyadmin/tbl_change.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=supplier_master) | | [Empty](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=supplier_master&sql_query=TRUNCATE+%60supplier_master%60&zero_rows=Table+supplier_master+has+been+emptied) | 5 | InnoDB | | [16.0 KiB](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=supplier_master#showusage) | - |
| **user\_master** | [Browse](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=user_master&pos=0) | [Structure](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=user_master) | | [Search](http://localhost/phpmyadmin/tbl_select.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=user_master) | [Insert](http://localhost/phpmyadmin/tbl_change.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=user_master) | | [Empty](http://localhost/phpmyadmin/sql.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=user_master&sql_query=TRUNCATE+%60user_master%60&zero_rows=Table+user_master+has+been+emptied) | 5 | InnoDB | | [16.0 KiB](http://localhost/phpmyadmin/tbl_structure.php?db=gift_touch_db&token=68874fabe8f7eeec6bc844303bafd40a&goto=db_structure.php&table=user_master#showusage) | - |
| **13 table(s)** | **102** | | **InnoDB** | | | **latin1\_swedish\_ci** | | | |

Fig 6.41 : Database Tables

**Chapter 7**

**Implementation Planning**

* **IMPLEMENTATION ENVIRONMENT**
* **SECURITY FEATURES**
* **CODING STANDARDS**

**7.1 IMPLEMENTATION ENVIRONMENT**

Challenges identified for successful design and implementation of this project are dominated by: complexity, reliability/availability, transparent data access while respecting security, a high experiment data rate and data volume, scientific exploitation from multiple sites. All the systems have a GUI interface to interact with the user.

For fetching the data from database, user should know the user name and password of the system in which database is available. It is also require that system is connected through LAN.

**7.1.1 MULTI USER**

This Software is a multi-user Application. This must be uploading on the server and start through internet on which you want to work. Multiple instances of the Application can be run at a time.

**7.1.2 GUI BASED**

This Software consists of GUI that is user friendly but the background process consists of much center of attraction. That is written in such a way that manipulation of multiple images can be easily done.

**7.2 SECURITY FEATURES**

As security aspect is taking more importance in the development of any application, this project was also developed using the security concerns in the mind. Today we find many programs being hacked and important data are being stolen without the knowledge of the user (administrator). So as a general feature the security and secure coding practice has become the necessity for any project.

This process was developed in a highly secure environment where importance of data and its confidentiality was very crucial. The information that this software processes had to be protected so that any malicious software cannot crack the system and use it for a wrong purpose.

So, during implementation, a very special security algorithm MD5 technique was used to store the user passwords in encrypted form. This technique generates an encrypted text for the passwords or other confidential data items and while same things are to be compared with input, the same algorithm is applied on input and encrypted text is compared with stored encrypted data. Hence decoding of such data is eliminated, leading to a safe transaction of data items.

This project has been implemented using this security coding model and all the important steps for the implementation have been taken for that. So it is sure that the project will not have any security lapse or anyone cannot breach the security of this project easily.

**7.3 CODING STANDARDS**

A consistent naming pattern is one of the most important elements of predictability and discoverability in a managed class library. Widespread use and understanding of these naming guidelines should eliminate unclear code and make it easier for developers to understand shared code.

* + 1. **Naming Convention**

Use full English descriptors that accurately describe the variable/field/class/interface

For example, use names like firstName, grandTotal, or CorporateCustomer.

Use terminology applicable to the domain .If the users of the system refer to their clients as Customer, then use the term Customer for the class, not client.

Use mixed case to make names readable, Use abbreviations sparingly, but if you do so then use then intelligently and document it

For example, to use a short form for the word “number”, choose one of nbr, no or num.

Avoid long names (<15 characters is a good tradeoff)

Avoid names that are similar or differ only in case

* + 1. **Documentation**

Comments should add to the clarity of code.

Avoid decoration, i.e., do not use banner-like comments

|  |  |  |
| --- | --- | --- |
| **Comment Type** | **Usage** | **Example** |
| **Documentation**  Starts with /\*\* and ends with \*/ | Used before declarations of interfaces, classes, member functions, and fields to document them. | /\*\*  \* Customer – a person or  \* organization  \*/ |
| **C style**  Starts with /\* and ends with \*/ | Used to document out lines of code that are no longer applicable. It is helpful in debugging. | /\*  This code was commented out by Ashish Sarin  \*/ |
| **Single line**  Starts with // and go until the end of the line | Used internally within member functions to document business logic, sections of code, and declarations of temporary variables. | **//** If the amount is greater  // than 10 multiply by 100 |

Table 7.1 : Comments

**7.3.3 Techniques for Writing Clean Code:**

**Document the code:** Already discussed above

**Paragraph/Indent the code:** Any code between the { and } should be properly indented

**Paragraph and punctuate multi-line statements**

**Example:**

**protected** **void** doGet(HttpServletRequest request, HttpServletResponse response)

**Use white space:** A few blank lines or spaces can help make the code more readable. Single blank lines to separate logical groups of code, such as control structures

Two blank lines to separate member function definitions

**Specify the order of Operations:** Use extra parenthesis to increase the readability of the code using AND and OR comparisons. This facilitates in identifying the exact order of operations in the code

**Write short, single command lines** Code should do one operation per line So only one statement should be there per line

* + 1. **Standards for Fields (Attributes / Properties)**

**7.3.4.1 Naming Fields**

Use a Full English Descriptor for Field Names

Fields that is collections, such as arrays or vectors, should be given names that are plural to indicate that they represent multiple values.

**Examples**

firstName orderItems

If the name of the field begins with an acronym then the acronym should be completely in lower case

**Example**

sqlDatabase

**7.3.4.2 Naming Components**

Use full English descriptor postfixed by the widget type. This makes it easy for a developer to identify the purpose of the components as well as its type.

**Example**

okButton customerList fileMenu newFileMenuItem

**7.3.4.3 Naming Constants**

In Java, constants, values that do not change, are typically implemented as static final fields of classes. The convention is to use full English words, all in upper case, with underscores between the words

**Example**

MINIMUM\_BALANCE MAX\_VALUE

**7.3.4.4 Field Visibility**

Fields should not be declared public for reasons of encapsulation. All fields should be declared private and accessor methods should be used to access / modify the field value. This results in less coupling between classes as the protected / public / package access of field can result in direct access of the field from other classes

**7.3.4.5 Documenting a Field**

**Document all applicable invariants:** Invariants of a field are the conditions that are always true about it. By documenting the restrictions on the values of a field one can understand important business rules, making it easier to understand how the code works / how the code is supposed to work

**Visibility decisions:** If a field is declared anything but private then it should be documented why it has not been declared private.

|  |  |  |
| --- | --- | --- |
| **Identifier Type** | **Rules for Naming** | **Examples** |
| **Packages** | The prefix of a unique package name is always written in all-lowercase ASCII letters and should be one of the top-level domain names, currently com, edu, gov, mil, net, org, or one of the English two-letter codes identifying countries as specified in ISO Standard 3166, 1981.  Subsequent components of the package name vary according to an organization's own internal naming conventions. Such conventions might specify that certain directory name components be division, department, project, machine, or login names. | com.sun.eng  com.apple.quicktime.v2  edu.cmu.cs.bovik.cheese |
| **Classes** | Class names should be nouns, in mixed case with the first letter of each internal word capitalized. Try to keep your class names simple and descriptive. Use whole words-avoid acronyms and abbreviations (unless the abbreviation is much more widely used than the long form, such as URL or HTML). | class Raster; class ImageSprite; |
| **Interfaces** | Interface names should be capitalized like class names. | interface RasterDelegate; interface Storing; |
| **Methods** | Methods should be verbs, in mixed case with the first letter lowercase, with the first letter of each internal word capitalized. | run(); runFast(); getBackground(); |
| **Constants** | The names of variables declared class constants and of ANSI constants should be all uppercase with words separated by underscores ("\_"). (ANSI constants should be avoided, for ease of debugging.) | static final int MIN\_WIDTH = 4;  static final int MAX\_WIDTH = 999;  static final int GET\_THE\_CPU = 1; |
| **Variables** | Variable names should be short yet meaningful. The choice of a variable name should be mnemonic- that is, designed to indicate to the casual observer the intent of its use. One-character variable names should be avoided except for temporary "throwaway" variables. Common names for temporary variables are i, j, k, m, and n for integers; c, d, and e for characters.  Except for variables, all instance, class, and class constants are in mixed case with a lowercase first letter. Internal words start with capital letters. Variable names should not start with underscore \_ or dollar sign $ characters, even though [[1]](#footnote-1)both are allowed. | int i;  char c;  float myWidth; |

Table 7.2 : Important Documentation Field

**Chapter 8**

**Testing**

* **TESTING STRATEGY**
* **TESTING METHODS**

**8.1 TESTING STRATEGY**

Testing is very important phase for the development of a project. For any project you cannot say that it is working properly without having proper testing done with it. Testing of a project will assure you and your client that your project is having certain quality with it and will not hold you in the middle of the sea.

That is why this project was also having test as a phase to ensure no mistake remained while development of this project. Every component of this project was thoroughly tested and was supplied to the customer only after it passed all the test cases.

For this project, two types of tests were followed:

**8.1.1 UNIT TEST**

Here each module was tested thoroughly using white box testing method. White box testing means the cases did sneak peak in the code snippets and check whether the result coming out of the case is correct or not. Also proper error checking is done or not. This kind of test was done along with the development of the module. Each module has its long list of test cases and must pass all of them.

For the display module along with the white box test, we had black box test also. As this module was the one with which a user interacts more, so this type of testing must be done for this module. Also both – white box and black box testing was also needed to be done; and to make the testing more optimal, we did white box for unit testing and black box at integration testing.

**8.1.2 INTEGRATION TEST**

This module was done after all the modules were developed and tested thoroughly. After the development of all modules, each module was attached to the main module and then tested again for the proper working of it. Here at the integration level, we did the black box testing. That will give a new dimension for the testing purpose. After the testing was done and all the errors were completed, next module was assigned for the integration process.

For all kinds of tests, if any error found in during the test, then the process was to correct that error first, then do all the test from the scratch and then go to next test case.

**8.2 TESTING METHODS**

There is a plethora of testing methods and testing techniques, serving multiple purposes in different life cycle phases. Classified by purpose, software testing can be divided into: correctness testing, performance testing, reliability testing and security testing. Classified by life-cycle phase, software testing can be classified into the following categories: requirements phase testing, design phase testing, program phase testing, evaluating test results, installation phase testing, acceptance testing and maintenance testing. By scope, software testing can be categorized as follows: unit testing, component testing, integration testing, and system test.

#### 8.2.1 CORECTNESS TESTING

Correctness is the minimum requirement of software, the essential purpose of testing. Correctness testing will need some type of oracle, to tell the right behavior from the wrong one. The tester may or may not know the inside details of the software module under test, e.g. control flow, data flow, etc. Therefore, either a white-box point of view or black-box point of view can be taken in testing software. We must note that the black-box and white-box ideas are not limited in correctness testing only.

* **BLACK BOX TESTING**

The black-box approach is a testing method in which test data are derived from the specified functional requirements without regard to the final program structure. It is also termed data-driven, input/output driven, or requirements-based testing. Because only the functionality of the software module is of concern, black-box testing also mainly refers to functional testing -- a testing method emphasized on executing the functions and examination of their input and output data. The tester treats the software under test as a black box -- only the inputs, outputs and specification are visible, and the functionality is determined by observing the outputs to corresponding inputs. In testing, various inputs are exercised and the outputs are compared against specification to validate the correctness. All test cases are derived from the specification. No implementation details of the code are considered.

It is obvious that the more we have covered in the input space, the more problems we will find and therefore we will be more confident about the quality of the software. Ideally we would be tempted to exhaustively test the input space. But as stated above, exhaustively testing the combinations of valid inputs will be impossible for most of the programs, let alone considering invalid inputs, timing, sequence, and resource variables. Combinatorial explosion is the major roadblock in functional testing. To make things worse, we can never be sure whether the specification is either correct or complete. Due to limitations of the language used in the specifications (usually natural language), ambiguity is often inevitable. Even if we use some type of formal or restricted language, we may still fail to write down all the possible cases in the specification. Sometimes, the specification itself becomes an intractable problem: it is not possible to specify precisely every situation that can be encountered using limited words. And people can seldom specify clearly what they want -- they usually can tell whether a prototype is, or is not, what they want after they have been finished. Specification problems contribute approximately 30 percent of all bugs in software.

The research in black-box testing mainly focuses on how to maximize the effectiveness of testing with minimum cost, usually the number of test cases. It is not possible to exhaust the input space, but it is possible to exhaustively test a subset of the input space. Partitioning is one of the common techniques. If we have partitioned the input space and assume all the input values in a partition is equivalent, then we only need to test one representative value in each partition to sufficiently cover the whole input space. Domain testing partitions the input domain into regions, and considers the input values in each domain an equivalent class Good partitioning requires knowledge of the software structure. A good testing plan will not only contain black-box testing, but also white-box approaches, and combinations of the two.

* **WHITE BOX TESTING**

Contrary to black-box testing, software is viewed as a white-box, or glass-box in white-box testing, as the structure and flow of the software under test are visible to the tester. Testing plans are made according to the details of the software implementation, such as programming language, logic, and styles. Test cases are derived from the program structure. White-box testing is also called glass-box testing, logic-driven testing or design-based testing.

There are many techniques available in white-box testing, because the problem of intractability is eased by specific knowledge and attention on the structure of the software under test. The intention of exhausting some aspect of the software is still strong in white-box testing, and some degree of exhaustion can be achieved, such as executing each line of code at least once (statement coverage), traverse every branch statements (branch coverage), or cover all the possible combinations of true and false condition predicates (Multiple condition coverage).

Control-flow testing, loop testing, and data-flow testing, all maps the corresponding flow structure of the software into a directed graph. Test cases are carefully selected based on the criterion that all the nodes or paths are covered or traversed at least once. By doing so we may discover unnecessary "dead" code -- code that is of no use, or never get executed at all, which can not be discovered by functional testing.

In mutation testing, the original program code is perturbed and many mutated programs are created, each contains one fault. Each faulty version of the program is called a mutant. Test data are selected based on the effectiveness of failing the mutants. The more mutants a test case can kill, the better the test case is considered. The problem with mutation testing is that it is too computationally expensive to use. The boundary between black-box approach and white-box approach is not clear-cut. Many testing strategies mentioned above, may not be safely classified into black-box testing or white-box testing. It is also true for transaction-flow testing, syntax testing, finite-state testing, and many other testing strategies not discussed in this text. One reason is that all the above techniques will need some knowledge of the specification of the software under test. Another reason is that the idea of specification itself is broad -- it may contain any requirement including the structure, programming language, and programming style as part of the specification content.

We may be reluctant to consider random testing as a testing technique. The test case selection is simple and straightforward: they are randomly chosen. Study in indicates that random testing is more cost effective for many programs. Some very subtle errors can be discovered with low cost. And it is also not inferior in coverage than other carefully designed testing techniques. One can also obtain reliability estimate using random testing results based on operational profiles. Effectively combining random testing with other testing techniques may yield more powerful and cost-effective testing strategies.

#### 8.2.2 PERFORMANCE TESTING

Not all software systems have specifications on performance explicitly. But every system will have implicit performance requirements. The software should not take infinite time or infinite resource to execute. "Performance bugs" sometimes are used to refer to those design problems in software that cause the system performance to degrade.

Performance has always been a great concern and a driving force of computer evolution. Performance evaluation of a software system usually includes: resource usage, throughput. Typical resources that need to be considered include CPU cycles, disk space, disk access operations, and memory usage. The goal of performance testing can be performance bottleneck identification, performance comparison and evaluation, etc. The typical method of doing performance testing is using a benchmark -- a program, workload or trace designed to be representative of the typical system usage.

#### 8.2.3 RELAIBILITY TESTING

Software reliability refers to the probability of failure-free operation of a system. It is related to many aspects of software, including the testing process. Directly estimating software reliability by quantifying its related factors can be difficult. Testing is an effective sampling method to measure software reliability. Guided by the operational profile, software testing (usually black-box testing) can be used to obtain failure data, and an estimation model can be further used to analyze the data to estimate the present reliability and predict future reliability. Therefore, based on the estimation, the developers can decide whether to release the software, and the users can decide whether to adopt and use the software. Risk of using software can also be assessed based on reliability information advocates that the primary goal of testing should be to measure the dependability of tested software.

There is agreement on the intuitive meaning of dependable software: it does not fail in unexpected or catastrophic ways. Robustness testing and stress testing are variances of reliability testing based on this simple criterion.

The robustness of a software component is the degree to which it can function correctly in the presence of exceptional inputs or stressful environmental conditions. Robustness testing differs with correctness testing in the sense that the functional correctness of the software is not of concern. It only watches for robustness problems such as machine crashes, process hangs or abnormal termination. Stress testing, or load testing, is often used to test the whole system rather than the software alone. In such tests the software or system are exercised with or beyond the specified limits. Typical stress includes resource exhaustion, bursts of activities, and sustained high loads.

**Chapter 9**

**Conclusion**

**9.1 CONCLUSION**

In this growing world of opportunities every organization needs to have an edge against other competitors with its efficient relations with its customers as well as other stakeholders. To get such an edge the organization needs to take technological benefits and should try to keep itself updated in this growing market for its sustainability.

Developing a management system helps bridge such gap and this not only synchronizes various organizational functions or departments but also helps in better customer satisfaction. This management system will help the organization eliminate the human errors and will provide the accurate data which will help the organization in dealing efficiently with its suppliers, customers or other stakeholders. Also such management system prevents loss of data with passage of time, as all the data is stored in central database.

Hence, more and more organizations should take such an approach and shift from traditional paper bound business model to electronic management systems for better future aspects.

**9.2 BIBLIOGRAPHY**

1. <http://en.wikipedia.org/wiki/Enterprise_resource_planning#Integration>
2. <http://www.eresourceerp.com/Why-is-ERP-important-to-a-company.html>
3. <http://en.wikipedia.org/wiki/List_of_ERP_software_packages>
4. <http://en.wikipedia.org/wiki/Promotional_merchandise>
5. <http://www.cliftonwebdesign.co.uk/archive/wampserver-set-apache-server-windows/>
6. <http://www.webopedia.com/TERM/W/WAMP.html>
7. <http://en.wikipedia.org/wiki/PHP>
8. <http://en.wikipedia.org/wiki/JavaScript>
9. <http://en.wikipedia.org/wiki/MySQL>

1. [↑](#footnote-ref-1)