

UNIVERSITY OF MUMBAI

G. R. Patil College Of Arts, Science & Commerce

(Affiliated to University Of Mumbai)

2010 – 2011

Project Report On

Super Market

Project work submitted in partial fulfillment of the requirements for the award of the degree of

B.Sc.CS (Computer Science)

SUBMITTED BY: JAGRUTI D. PATIL.

PROJECT GUIDE: MRS. SUNANDA MULGUND,

MRS. SANDHYA PANDEY

G. R. Patil College Of Arts, Science & Commerce

(Affiliated to University Of Mumbai)

2010 – 2011

CERTIFICATE

This is certify that this project entitled VIJAYA SUPER MATKER is a Bonafied work done by JAGRUTI DIGAMBER PATIL in partial fulfillment of the requirements for the B.SC. Computer Science degree during the academic year 2010-2011 and is the original work of the candidate.

PROJECT GUIDE : MRS. SUNANDA MULGUND,

MRS. SANDHYA PANDEY.

INTERNAL EXAMINAR EXTERNAL EXAMINAR

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

H.O.D PRINCIPAL

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_

Place: \_\_\_\_\_\_\_\_\_\_\_

Date:\_\_\_\_\_\_\_\_\_\_\_\_

ACKNOWLEDGEMENT

I wish to thank all those skillful persons who helped me to make this project possible.

First and Foremost, I would like to thank all the Staff members of Subhiksha who helped me in getting all the requisite information for the project.

The technical review was done by MRS. SUNANDA MULGUND,

MRS. SANDHYA PANDEY [Project Guide who helped us and guided us throughout my project]. Every madam has a part in this successful co-operation of my project. However, Mrs. Seema Madam has helped me a lot in critical situations whereas Mrs. Kiran Madam has played an important role in the preliminary stages.

And, finally I wish to thank to all people who helped me directly or indirectly.

PREFACE

Over the last few years, computer already had a considerable impact on many aspects of our society. Medicine, Government, Banking, Educational Institutions, Engineering, Railway and Airline Reservations – these are only some of the fields in which computers are already playing a highly significant role. Over the next few years, there would be a great increase in the application of computer and would have a wide use of computers in our day-to-day life.

Computer science has existed as an independent discipline dealing with the systematic processing of information for more than three decades. In the years to come, the database system will become increasingly widespread and increasingly important. The number of systems installed or under development is growing at a considerable speed.

This project is also a part of this trend and is very easy and flexible to use. It is totally GUI based.

INDEX

|  |  |  |
| --- | --- | --- |
| **Sr. No** | **Title** | **Page No.** |
| **I.** | **Preliminary Investigation** | **8** |
| **1.1** | Organization Overview | 9 |
| **1.2** | Description of system | 10 |
| **1.3** | Limitation of present system | 11 |
| **1.4** | Proposed System | 12 |
| **1.5** | Feasibility Study | 14 |
| **1.6** | Stack Holder | 19 |
| **1.7** | Gantt Chart | 23 |
| **II.** | **System Analysis** | **25** |
| **2.1** | Fact-finding Techniques | 26 |
| **2.2** | Prototypes | 30 |
| **2.3** | Event Table | 32 |
| **2.4** | Use case Diagram | 34 |
| **2.5** | Entity Relationship Diagram | 37 |
| **2.6** | Activity Diagram | 41 |
| **2.7** | Class Diagram | 43 |
| **2.8** | Sequence Diagram | 45 |
| **2.9** | State Diagram | 48 |
| **III.** | **System Design** | **50** |
| **3.1** | Converting ERD into tables | 51 |
| **3.2** | Package Diagram | 53 |
| **3.3** | Program Flow Chart | 55 |
| **3.4** | System Flow Chart | 56 |
| **3.5** | Structure Chart | 57 |
| **IV.** | **System Coding** | **58** |
| **4.1** | Menu Tree | 59 |
| **4.2** | List Of Tables with attributes and constraints | 60 |
| **4.3** | Program Description with Naming Conventions | 63 |
| **4.4** | Screen Layout | 66 |
| **4.5** | Report Layouts | 78 |
| **4.6** | Validations | 81 |
| **4.7** | Test Data | 86 |
| **V.** | Source Code | 89 |
| **VI.** | System Implementation | 107 |
| **VII.** | References and Bibliography | 111 |

*Preliminary Investigation*

Organizational Overview

The present system works manually. The system maintains information about all the Customers who takes the Product from company & Dealers who supply Product to shop. Also system maintains information of Stock of Products,Staff members.Most of the information is stored in registers.

When any enquiry is done,then every time they have to refer registers which is very Time Consuming. The information of all the Customers in the company is maintained viz. name,address,phone no. The Profile number provided to each product during selling & Purchasing time.

Reports regarding to the Customers,Products,Dealers,Stock & Staff of Company are given to the Management Daily and Monthly.

Description Of System

By using the automated proposed system over time will not consume and also mistakes, errors are get decreases. The proposed system is more benifitable than existing present system.

Limitations of Present System

Although the present system is working properly, it has many limitations. They are as follows: -

♣ Most of the work is done manually.

♣ Data is stored in registers, which is very tedious and inefficient.

♣ As the records of students go on increasing, it becomes very difficult to maintain the said.

♣ Searching data from registers is time consuming.

♣ Modification and updation of information manually will require more time as all the related information present in other registers also need to be updated.

♣ Even though, some part of the system is computerized i.e. they are stored in MS-Access databases, the updations done in any one database does not reflect in other databases immediately.

♣ Also, relying on registers is also inefficient. The durability of registers would be less as the pages may get torn as time passes.

Proposed system

This project for the Silvertech Computer institute will prove very useful since it is very much reliable and GUI based.

♣ The front end of the system is in java & back end is SQL Server 2005

♣ Whenever the customer does an enquiry, an enquiry form will be show to them.

♣ The Product details like name,id,quantity,price etc. are stored in Product master.

♣ Whenever, any modifications or deletions are done in the records, they will be immediately reflected in the remaining related information.

♣ Customer Details & Staff Details like name,address,contact no are also maintained.

♣ Whenever, the student goes for a break, his record will be maintained with the current module of the course going on & the modules completed.

♣ There is also a facility for storing and calculating the Bills.

♣ At last, all the reports necessary for the entire system are generated. These reports will be provided in a user-friendly manner.

Feasibility Study

The next step in analysis is to verify the feasibility of the proposed system. All project are feasible given unlimited resource and time are scare project should be optimal in three consumption of resources. This places a constant on approval any project.

Feasibility Study is important in project as it gives idea about which types of hardware, software ,technology &

other resources can be used by the system to all the people related to project like requirement gathering people

,analysts ,designers and also end users.

Feasibility as applied to our system pertain the following area:-

1. Economic Feasibility

2. Operational Feasibility

3. Technical Feasibility

1. **Economic Feasibility:-**

Higher level of automation most often requires more funds. Hence based on the hardware and software specification a desirable alternative costs and benefits to see if the investment made in creating / developing a new system is costlier or more beneficial.

Financial benefits must equal or exceed the costs. To assure this one must estimate the

following:

1. If the Organization has adequate cash flow for funding the development
2. The cost to conduct a full system investigation.
3. The cost of hardware and software for the class of application being considered.
4. The benefits in the form of reduced costs or fewer costly errors.
5. The cost if nothing changes (i.e. the proposed system is not developed) for a project to be judged feasible, it must pass all these tests.
6. If any one of these issues appears infeasible the decision must be reconsidered.

My system is economically feasible as the costs involved in implementing the automated system is in lieu with the cash flow of the shop. Thus the system developed will be beneficial to the users.

However there will be an additional cost for implementing the

technology on which the system is to be implemented.

**2. Operational Feasibility:-**

The operational feasibility is obtained by consulting the system user whether it satisfies the user’s requirements.

A system with an easy interface will always help the user to use

the system.

The new system has completely user friendly interface. It has been designed to be pretty intuitive, so that even an inexperienced person can easily handle the system.

Business functions are reengineered to achieve broader scope and higher level of automation.

Manual processes too are modified. Every company has its own culture

and new system should fit the company culture.

The issues to be taken into concern are**:**

* Corporate Culture.
* Level of computer competency.
* Loss of control on employee by staff/management.
* Change of job responsibility.
* Loss of employment due to increased automation.
* The nature and level of user involvement in the development and implementation of system.
* Revisal of old, longstanding work procedures.
* It is usually a practice to include people trained in organizational

behavior to assist in managing these changes.

As the system will be automated there is a sense of insecurity among the employees working for an event. As the system provides a user friendly simple GUI it is operationally feasible to implement it in the Hotel.

This automation will save time as well as reduce errors.

**3. Technical Feasibility:-**

**About Java :**

* The Java programming language is robust and versatile, enabling developers to:
* Write software on one platform and run it on another.
* Its no wonder the Java platform attracts so many new developers.
* The Java programming language is a high-level language that can be characterized by all of the following buzzwords simple, distributed, architecture neutral, robust, secure, dynamic, portable, High Performance.
* The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms. .

The necessary software required for the development of system is

* Java
* SQL Server 2005

There is requirement for the mentioned software and an expertise for handling system.

Thus in the presence of required hardware, software the proposed system is technically feasible.

Hardware and Software Requirement-

**Hardware Requirements** :-

Pentium4 (P4) or higher version

1 GB RAM or more.

500MB free space in Hard Disk.

**Software Requirements:-**

Back end: - SQL Server 2005.

Front End: - java

Operating System - Windows XP or higher version.

Stake holders

Your primary source of information for system requirements is the various stake holders of the new system. The stake holders of are all people who and interest in the successful implementation of the system.

Generally we categorized stake holders into one of three groups:

1. The user, those who actually use the system on the daily basis.
2. The customer, those who pay for and own the system operators in the computing environment of the organization.

The various kind stake holders who have an interest in a new system. During analysis, the analyst also needs to consider the technical staff as well. One of the most important first steps in determining g system requirements is to identify these various system stake holders. In the past, problems have arisen with new system because only some of the stake holders where included in the project and the system was built exclusively for them. One of an analysts first task is to identify every type of stake holder category are available to the project as business expert.

Users as stake holders:

User roles-that is type of system users should be identified in two dimensions: Horizontally and vertically.

By horizontally we mean that the analyst must look for information flow across business department or functions.

For example:-

A new inventory system may affect receiving garage, sales and manufacturing. So individuals from each of these departments must describe their requirements. The sales department may need to determine when and how to update inventory quantities or to commit inventory at the time of the same, but before it is shipped. Manufacturing may need certain information from the inventory system to assist in scheduling production. So, remembering to include the horizontal dimension in the definition of requirements will ensure that May different departments, even those that may appear unrelated to the new system are included.

By vertical dimension, we mean the information needs of clerical staff of middle management and of senior executives. Each of these stake holders has different information request for the system that must be included in the design. The following section described the characteristics and information needs of the various users on the vertical dimensions. These same characteristics also applied to each development across the horizontal language.

Business User:-

Business users are the people who use system to perform day to day operation of an organization like clerk of showroom. Business uses are providing information about daily operation of sales of car and system must support them.

Information User:-

An information user is a person who provides information to customer. This person might be an operational user or someone else. In some cases a business might want to make information directly available to customer information user may not be permission on sales transition of products.

Management User:-

Managers are responsible for seeing that the company it’s performing for its daily procedure efficiently and effectively

\* What kind of reports must the system produce?

\* What kind of sales performance?

\* What kind of product information and specification must the system keep?

Executive Users:-

The tap executive of an organization interested in strategic issue, as well as the daily issues just described. The typically want information from a system so that they can compare overall improvements in resource utilization.

External User:-

More and more quantity allow external user to have direct access to the system customer may access the system directly through internet with help of website of company. The external user not allows checking transaction. These users more difficult to identify and access because they are not a regular member of the organization.

GANTT CHART

Gantt charts are a project-planning tool that can be used to represent the timing of tasks required to complete a project.

Because Gantt charts are simple to understand and easy to construct, they are used by most project managers for all but the most complex projects.

In a Gantt chart, each task takes up one row. Dates run along the top in increments of days, weeks or months, depending on the total length of the project.

The expected time for each task is represented by a horizontal bar whose left end marks the expected beginning of the task and whose right end marks the expected completion date. Tasks may run sequentially, in parallel or overlapping.

As the project progresses, the chart is updated by filling in the bars to a length proportional to the fraction of work that has been accomplished on the task. This way, one can get a quick reading of project progress by drawing a vertical line through the chart at the current date.

Completed tasks lie to the left of the line and are completely filled in. Current tasks cross the line and are behind schedule if their filled-in section is to the left of the line and ahead of schedule if the filled-in section stops to the right of the line. Future tasks lie completely to the right of the line.

In constructing a Gantt chart, keep the tasks to a manageable number (no more than 15 or 20) so that the chart fits on a single page. More complex projects may require subordinate charts which detail the timing of all the subtasks which make up one of the main tasks.

For team projects, it often helps to have an additional column containing numbers or initials, which identify that on the team is responsible for the task.

Often the project has important events, which you would like to appear on the project timeline, but which are not tasks. For example, you may wish to highlight when a prototype is complete or the date of a design review.

We planned our project using according to the Gantt chart as shown as follows

SYSTEM ANALYSIS

Fact finding techniques

After obtaining the background knowledge, the information on the existing system, its inputs, outputs, costs and other important requirement and features have to be collected and analyzed. The following tools were used for knowing more about the system and gathering more information for developing a new system.

For developing a system for an organization one need to acquire necessary information regarding the system. Information plays vital role in an organization. So collection of information before designing a system solves many problems. Following are the fact finding techniques used.

1. Documentation
2. On Site Observation
3. Interviews
4. Questionnaires

It is called information gathering.

Documentation:-

This involves review of written documents, literature, procedures, forms, manuals govt. publications etc . The preliminary drawback is search time. Sometimes it may be difficult to get certain reports and documentation or may be outdated or expensive and may not be updated. Close look at printed forms are necessary since printed forms are widely used for capturing and providing information. The objective of this is to understand how forms are used.

The following questions are useful in using this methods for fact finding

1. Who uses forms? How important they are to user?
2. Does the forms include all necessary information ?
3. What items should be added or deleted?
4. How many departments receive existing forms? Why?
5. How easy and readable it is to follow?
6. What are the other used of the form?
7. How the forming helps user in decision making?

On-Site Observation:-

The major objective of On site observation is to get close to the “real”

System being studied. This is the most difficult fact finding method. In this the role of the analysis is of observer or listener. He should avoid advice, passing moral judgments on show hostility towards one person and unique friendliness towards other.

The following questions can serve as guideline for on site observation.

i) what kind of system is it? What it is?

ii) What its function and other contribution to organization?

iii) Who are the peoples involved in the system?

iv) What is the history of system?

v) what is the assurance that observation presses is not seriously affecting the system or behavior being observed.

Interview:-

The interview is face to face interpersonal meeting for information gathering information. It is a widely used tool. It provides greater flexibility for gathering information.

In on site observation, we cannot learn people’s perceptions, feeling and motivation etc. Much of the information can be acquiring by direct questions. But there is information of more difficult nature that user staff may be redundant to give directly. For E.g. information on company policies, or specification with senior etc. If asked directly interviewer may give invalid information. But if properly handled correct information can be successfully obtained with interviews and questionnaires. More freely given information is more valid.

Interview offers better opportunity than the questionnaires to evaluate validity of information gathered. Interviewing is an art and requires experience.

Following are the guidelines for successful interview:-

1. Set the stage for interview and open it.
2. Create friendly atmosphere and put the interviewer at ease.
3. The location, time and order of interviewing should assurance privacy and minimum interruption.
4. Phrase questions clearly.
5. Be a good listener, avoid arguments and records responses.
6. Evaluate the outcome of the interview.

Questionnaires

The questionnaire is self-administered tools that is more economical and require less skill to administrate than interview.

In examiners a large number of respondents. At the same provides standardizes wording and instruction places less pressures on subjects for immediate response.

Following are the procedures for the questionnaires construction which consist of six steps.

1. Decide the data to be collected i.e. define the problem to be investigated.
2. Decide what type of questionnaires should be used :-fill in the blanks type or multiple type questions.
3. Outline the topics for the questionnaire and then write the questions.
4. Edit the questionnaires for technical defect or basis that reflect personal value.
5. Pretest (try out) the questionnaires to see how well it works.
6. Do a final editing to ensure that the questionnaires are ready for administration.

Prototype

Agile software development

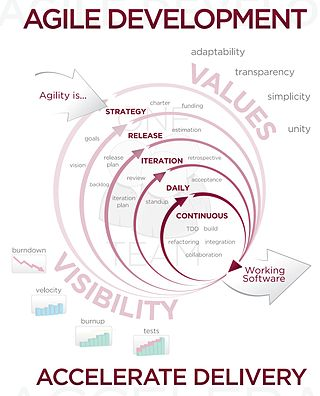
Agile software development is a group of [software development methodologies](http://en.wikipedia.org/wiki/Software_development_methodologies) based on [iterative and incremental development](http://en.wikipedia.org/wiki/Iterative_and_incremental_development), where requirements and solutions evolve through collaboration between [self-organization](http://en.wikipedia.org/wiki/Self-organization#Self-organization_in_agile_software_development), [cross-functional teams](http://en.wikipedia.org/wiki/Cross-functional_team).

There are many specific agile development methods. Most promote development, teamwork, collaboration, and process adaptability throughout the life-cycle of the project.

Agile methods break tasks into small increments with minimal planning, and do not directly involve long-term planning. Iterations are short time frames ([timeboxes](http://en.wikipedia.org/wiki/Timeboxing)) that typically last from one to four weeks. Each iteration involves a team working through a full software development cycle including planning, [requirements analysis](http://en.wikipedia.org/wiki/Requirements_analysis), [design](http://en.wikipedia.org/wiki/Software_Design), [coding](http://en.wikipedia.org/wiki/Computer_programming), [unit testing](http://en.wikipedia.org/wiki/Unit_test), and [acceptance testing](http://en.wikipedia.org/wiki/Acceptance_test) when a working product is demonstrated to stakeholders. This minimizes overall risk and allows the project to adapt to changes quickly. Stakeholders produce documentation as required. An iteration may not add enough functionality to warrant a market release, but the goal is to have an available release (with minimal [bugs](http://en.wikipedia.org/wiki/Software_bug)) at the end of each iteration.[[8]](http://en.wikipedia.org/wiki/Agile_software_development#cite_note-embracing_change-7) Multiple iterations may be required to release a product or new features.

Team composition in an agile project is usually cross-functional and self-organizing without consideration for any existing corporate hierarchy or the corporate roles of team members. Team members normally take responsibility for tasks that deliver the functionality an iteration requires. They decide individually how to meet an iteration's requirements.

Agile methods emphasize face-to-face communication over written documents when the team is all in the same location. Most agile teams work in a single open office (called a [bullpen](http://en.wiktionary.org/wiki/bullpen)), which facilitates such communication. Team size is typically small (5-9 people) to simplify team communication and team collaboration. Larger development efforts may be delivered by multiple teams working toward a common goal or on different parts of an effort. This may require a co-ordination of priorities across teams. When a team works in different locations, they maintain daily contact through [videoconferencing](http://en.wikipedia.org/wiki/Videoconferencing), voice, e-mail, etc.



Event Table

• Event table: A Table that lists events in rows and key pieces of information about each

event in columns.

• Event: An Event occurs at a specific time and place, can be described and should be remembered by the system.

• Trigger: An occurrence that tells the system that a event has occurred, either arrival of data needing processing or of a point in time.

• Source: An external agent or an actor that supplies data to the system.

• Activity: Behavior that the system performs when an event occurs.

• Response: An output produced by the system that goes to the destination.

• Destination: An external agent or an actor that receives data from the system.

Description:

• Events are the transactions.

• Events are the cause of transactions.

• Events will generate transactions.

• While developing the list of events the analyst should note the additional information about the events for later use.

• The list events with trigger, source, activity, response(s), and destination for each event can be placed in

the event table to keep track of them for later use.

• Event table is a convenient way to record about the requirements for information

**customer**

**Payment accepted**

**Payment**

**given**

**staff**

**After products received**

**Bill generation**

**Customer**

**Product received**

**Order placed**

**Staff**

**Request of product**

**Issue of products**

**Customer**

**Product info. received**

**Issuing product info.**

**staff**

**Enquiry of products**

**Issue of product info.**

**Staff**

**Payment accepted**

**Payment given**

**vendor**

**After products received**

**Bill generation**

**Staff**

**Products received**

**Products are purchased**

**staff**

**Order given of product**

**Purchase of product**

**staff**

**Quotations are accepted**

**Quotations**

**Are given**

**vendor**

**Quotation**

**given**

**Quotation**

**generation**

**Destination**

**Response**

**Activity**

**Source**

**Trigger**

**Event**

Use Case Diagram

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by

a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.

The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

A use case is a set of scenarios that describing an interaction between a user and a system. A use case diagram

displays the relationship among actors and use cases. The two main components of a use case diagram are use

cases and actors.



An actor is represents a user or another system that will interact with the system you are modeling. A use case

is an external view of the system that represents some action the user might perform in order to complete a task.

When to Use: Use Cases Diagrams

Use cases are used in almost every project. They are helpful in exposing requirements and planning the project.

During the initial stage of a project most use cases should be defined, but as the project continues more might

become visible.











Entity-Relationship Diagram

Data models are tools used in diagrams to describe the data requirements and assumptions in the system from a top down perspective.

An entity relationship diagram is a graphical\cal representation of an organization’s data storage requirements.

Entity relationship diagrams are abstractions of the real world which simplify the problem to be solved while retaining its essential features. Entity relationship diagrams are used to identify the data that must be captured, stored and retrieved in order to support the business activities performed by the organization; and identify the data required to derive and report on the performance measures that an organization should be monitoring.

There are three basic elements in ER models:

Entities: Entities are the “things” about which we seek information.

Attributes: Attributes are the data we collect about the entities.

Relationships: Relationships provide the structure needed to draw

information from multiple entities.

Developing an ERD requires an understanding of the system and its components.

ERD brings out issues:

i) Ambiguities

ii) Entities and their relationships

iii) What data needs to be stored?

iv) The degree of a relationship

|  |  |  |
| --- | --- | --- |
| Symbols | Symbol Name | Meaning |
|  | Entity | An object that exist and is  distinguishable from all  other objects. |
|  | Weak Entity | An entity set with no  sufficient attributes to  form a primary key. |
|  | Relationship | An association among  Several entities. |
|  | Identifying Relationship | A relationship associating  weak entity set with  identifying entity set |
|  | Attribute | A specification that  defines the property of an  Object, element of file. |
|  | Primary Key | A candidate key that has  been selected as the  identifier for an entity  Type. Primary key values  May be null. It is also  Called as identifier. |
|  | Multi-valued Attribute | An attribute that can have  more than one value for  each entity instance. |
|  | Discriminating Attribute | A set of attributes that  allows distinguishing  among the entities |

**E-R Diagram**

**Product**

**Staff**

**Customer**

**Vendor**

**Pid**

**1**

**M**

**Purchase**

**pid**

**Sale**

**pid**

**M**

**1**

**M**

**1**

Activity Diagram

Activity diagrams describe the workflow behavior of a system. Activity diagrams are similar to state diagrams because activities are the state of doing something.

The diagrams describe the state of activities by showing the

sequence of activities performed. Activity diagrams can show activities that are conditional or parallel.

When to Use: Activity Diagrams

Activity diagrams should be used in conjunction with other modeling techniques such as interaction diagrams and state diagrams.

The main reason to use activity diagrams is to model the workflow behind the system being designed.

Activity Diagrams are also useful for: analyzing a use case by describing what actions need to take place and when they should occur; describing a complicated sequential algorithm; and modeling applications

with parallel processes.

However, activity diagrams should not take the place of interaction diagrams and state diagrams.

Activity diagrams do not give detail about how objects behave or how objects collaborate

Below are figures:

The given below is figure for the activity diagram of the system



Class diagram

The class diagram is the main building block in object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. The classes in a class diagram represent both the main objects and or interactions in the application and the objects to be programmed. In the class diagram these classes are

Represented with boxes which contain three parts:

• A class with three sections.

• The upper part holds the name of the class

• The middle part contains the attributes of the class.

•The bottom part gives the methods or operations the class can take or undertake.

In the system design of a system, a number of classes are identified and grouped together in a class diagram which helps to determine the statically relations between those objects. With detailed modeling, the classes of the conceptual design are often split in a number of subclasses.

In order to further describe the behavior of systems, these class diagrams can be complemented by state diagram or UML state machine. Also instead of class diagrams Object role modeling can be used if you just want to model the classes and their relationships.



Sequence diagram

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order.

It is a construct of a Message Sequence Chart.

Sequence diagrams are sometimes called Event-trace diagrams, event scenarios, and timing 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.

However, an organization's business staff can find sequence diagrams useful to communicate how the business currently works by showing how various business objects interact.

Besides documenting an organization's current affairs, a business-level sequence diagram can be used as a requirements document to communicate requirements for a future system implementation.

During the requirements phase of a project, analysts can take use cases to the next level by providing a more formal level of refinement.

When that occurs, use cases are often refined into one or more sequence diagrams.

An organization's technical staff can find sequence diagrams useful in documenting how a future system should

behave.

During the design phase, architects and developers can use the diagram to force out the system's object interactions, thus fleshing out overall system design.

One of the primary uses of sequence diagrams is in the transition from requirements expressed as use cases to the next and more formal level of refinement.

Use cases are often refined into one or more sequence diagrams.

In addition to their use in designing new systems, sequence diagrams can be used to document how objects in an existing (call it "legacy") system currently interact.

This documentation is very useful when transitioning a

system to another person or organization.

**Sequence Diagram**



**SYSTEM**

login

look for stock

stock info

provide vendor(name or id)

vendor info

* Product := id , name add configuration := quantity ,rate, date, vendor name ,id

Loop for all products

vendor payment form payment made receipt to vendor

new customer(name ,id)

**Go to maintain customer**

**information**

being order

* Product := id , name
* Customer:=id, name

add order := quantity ,rate, date, vendor name ,id

Loop for all items

Accept order

Billing form

Provide (bill no,cheque no and bank name)

Bill to customer

**State diagram**

State diagrams are used to give an abstract description of the behavior of a system. This behavior is analyzed and represented in series of events that could occur in one or more possible states. Hereby "each diagram usually represents objects of a single class and tracks the different states of its objects through the system".

State diagrams can be used to graphically represent finite state machines. This was introduced by Taylor Booth in his 1967 book "Sequential Machines and Automata Theory". Another possible representation is the State transition table.

Bill Payment

generation

Confirm transaction

Full

All

Products

Adding products

Idle

SYSTEM DESIGN

Converting ERD to Tables

Customer Master

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| id | name | address | gender | phone | mobile |
|  |  |  |  |  |  |
| email |
|  |

Staff Master

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| id | name | address | gender | Dep | Age |
|  |  |  |  |  |  |
| Doj | Phone | Mobile | email |
|  |  |  |  |

Product Master

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| id | name | quantity | rate | date |
|  |  |  |  |  |

Vendor Master

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| id | name | address | gender | phone | mobile |
|  |  |  |  |  |  |

Total

|  |  |  |
| --- | --- | --- |
| cid | Date | total |
|  |  |  |

Sale Transaction

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Orderno | Custno | Custname | Custadd | Custphone | Prono |
|  |  |  |  |  |  |
| Proname | Quantity | Rate | Date |
|  |  |  |  |

Purchase Transaction

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Invoice | Id | Name | Add | Phone | Proname |
|  |  |  |  |  |  |
| Prono | Quantity | Rate | Date |
|  |  |  |  |

Bill

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Receiptno | Custid | Total | Date | Mop | Chequeno |
|  |  |  |  |  |  |
| bankname |
|  |

Package Diagram

In addition to the standard UML Dependency relationship, there are two special types of dependencies defined between packages:

• Package import

• Package merge

A package import is "a relationship between an importing namespace and a package, indicating that the importing namespace adds the names of the members of the package to its own namespace."

By default, an unlabeled dependency between two packages is interpreted as a package import relationship.

A package merge is "a directed relationship between two packages that indicates that the contents of the two packages are to be combined. It is very similar to Generalization in the sense that the source element conceptually adds the characteristics of the target element to its own characteristics resulting in an element that combines the characteristics of both".

Vendor

Master

DB

Staff

Master

DB

Product

Master

DB

Customer

Master

DB

Enter

Vendor

Enter

Staff

Enter

Product

Enter

Customer

Vendor

Details

Staff

Details

Product

Details

Customer

Details

View Layer

Domain layer

Data Access Layer

Program Flow Chart

**Read Vendor Info,Stock Info, Customer Info,Sales & Purchase Info**

**Place Customer Order for Product**

**If Product available**

**Yes**

**No**

**Issue Of Product**

**Request for Product to Vendor**

**Bill Generation**

**If Payment is made**

**Check Stock Availability**

**Report Generation**

**Payments**

**No**

**Yes**

**Delivery of the Product for Sale**

**STOP**

**START**

System Flow Chart

**Customer**

**master**

**Vendor**

**master**

**Product**

**master**

**Stock**

**master**

**Master**

**Quatation**

**generation**

**purchase**

**stock**

**sales**

**Bill generation**

**Product**

**Master report**

**Stock**

**Master report**

**Quatation**

**report**

**Purchase**

**report**

**Stock**

**report**

**Sales**

**report**

**Bill generation**

Structure Chart

**Super Market Management System**

**Master**

**Stock**

**Quotation**

**generation**

**Sales and**

**purchase**

**Bill generation**

**Report generation**

**Customer**

**master**

**Vendor**

**master**

**Stock**

**master**

**Product**

**master**

**sales**

**purchase**

**Daily**

**report**

**Monthly**

**report**

**Valid**

**product**

**Valid**

**product**

**Staff**

**Master**

**Masters**

**reports**

SYSTEM CODING

Menu Tree

**MASTER**

**Customer Master**

**Vendor Master**

**Product Master**

**Stock Master**

**TRANSACTION**

**Purchase Transaction**

**Sale transaction**

**REPORTS**

**Daily Report**

**Monthly Report**

**Masters Reports**

**Sale report**

**Sale report**

**purchase report**

**Purchase report**

**Staff Master**

**Sales Bill**

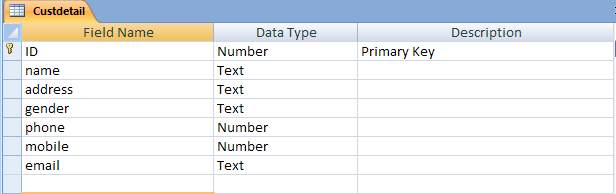
**TERMINATION**

**Log Off**

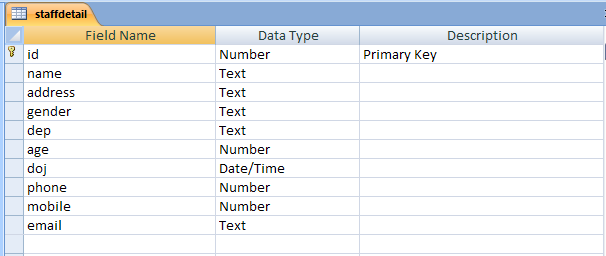
**Exit**

List Of Tables with Attributes and constraint

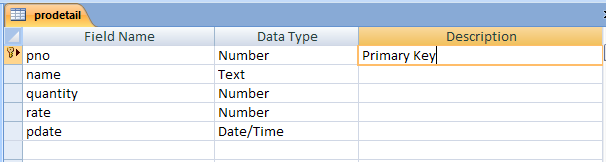
Customer Details Table



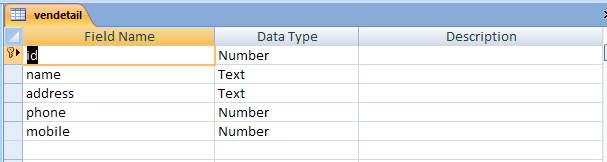
Staff Details Table



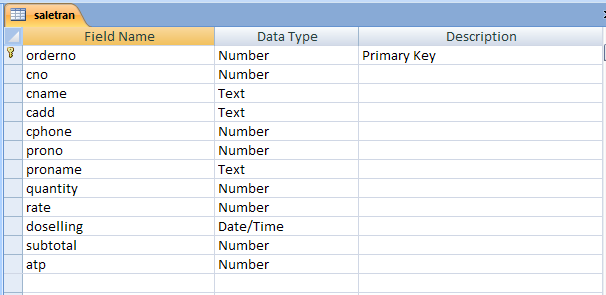
Product Details Table



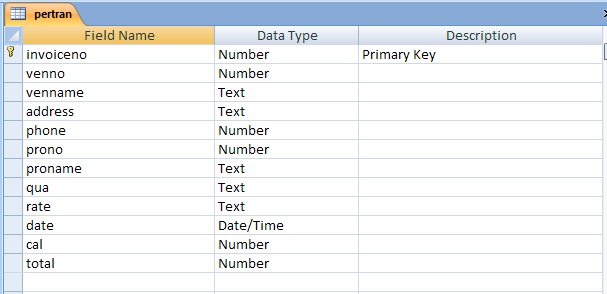
Vendor Details Table



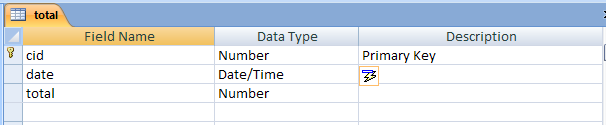
Sales Transaction Table



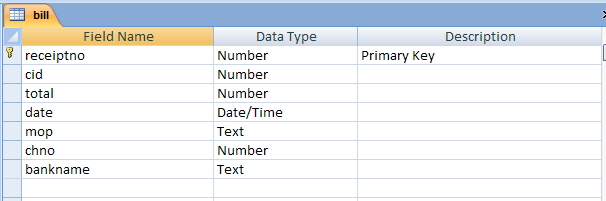
Purchase Transaction Table



Total Table



Bill Table



Program Description

frmsplash:-

It provides a splash screen showing the system and its name.

frmlogin:-

This form shows username and password to enables MDI\_form.

frmmain:-

It is the main form having manus.

frmctab:-

This form view all the customer

frmaddcus:-

For adding Customer

frmsearchcus:-

For searching Customer

frmvtab:-

This form view all the Vendor

frmaddcven:-

For adding Vendor

frmsearchven:-

For searching Vendor

frmstab:-

This form view all the Staff

frmaddstaff:-

For adding Staff

frmsearchstaff:-

For searching Staff

frmptab:-

This form view all the Product

frmaddpro:-

For adding Product

frmsearchpro:-

For searching Product

frmptran:-

This form contain Purchase Transaction (from vendor)

Frmstran:-

This for contain Sales Transaction (to costomer)

Frmbill:-

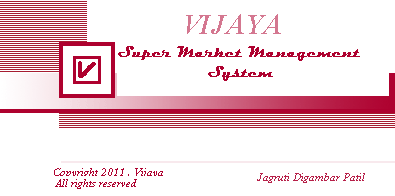
This form contain bill for particular customer

Naming Conventions

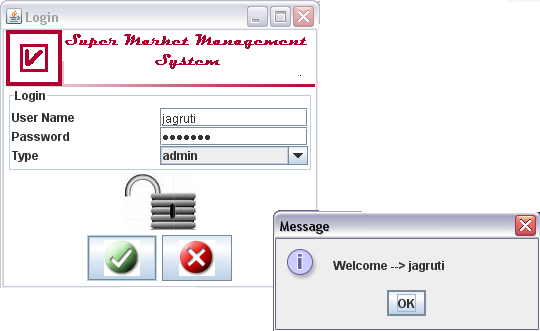
|  |  |
| --- | --- |
| **Name** | **Conventions** |
| JFram | frm |
| JText Field | txt |
| JButton | Btn |
| JComboBox | Cbo |
| JPanel | jpanel |
| JLabel | lbl |
| JScrollPane | jsp |
| JTable | Jt |
| JGroupBox | jgb |

Screen Layout

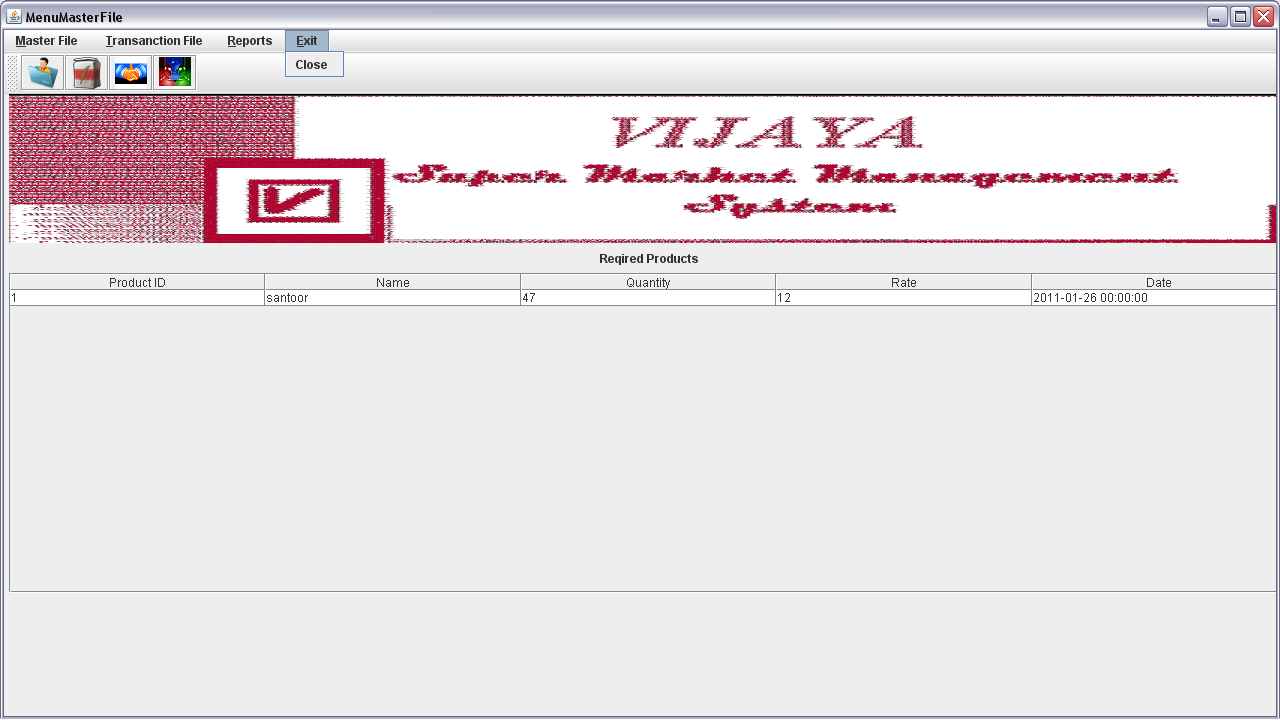
**Splash Screen:-**



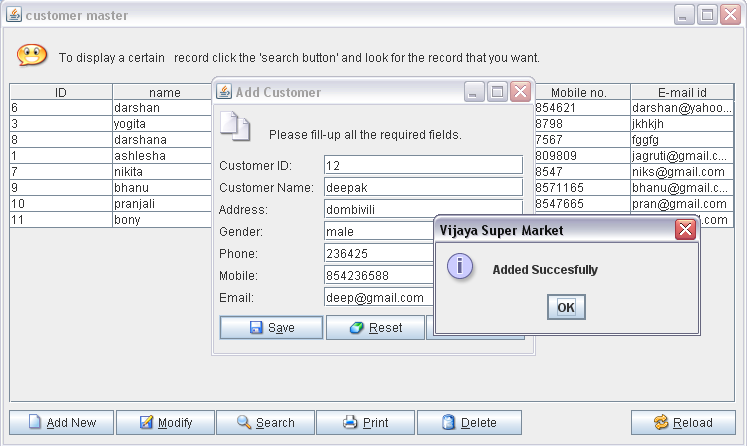
**Login:-**



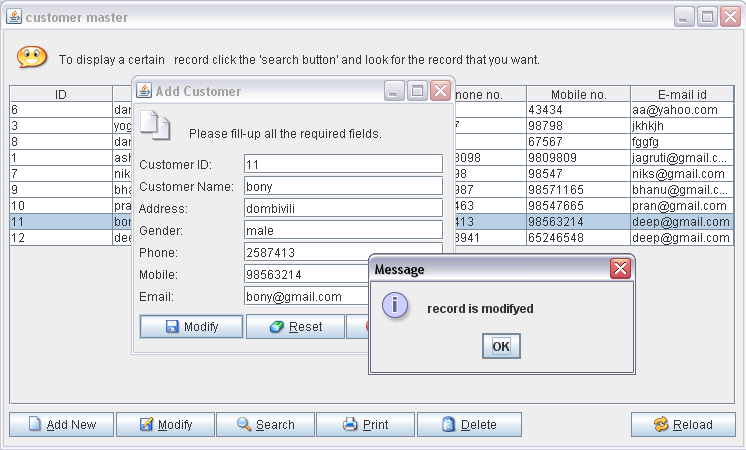
**MDI Form:-**



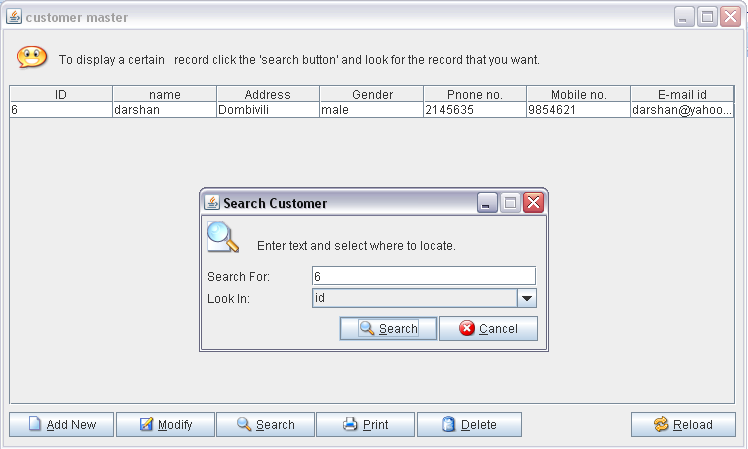
**Frmaddcus:-**



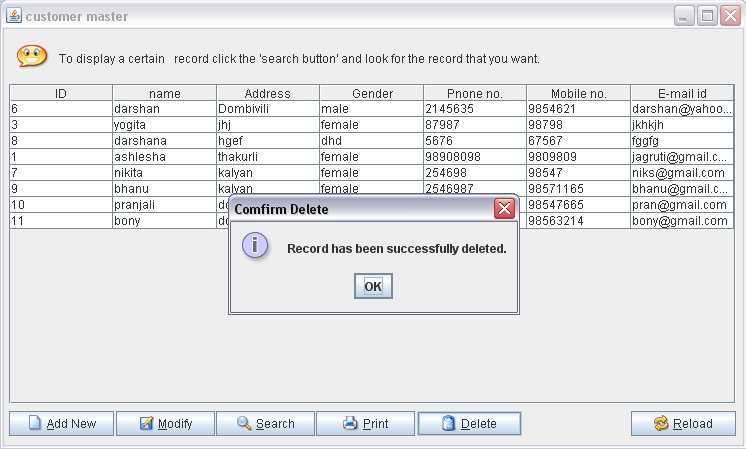
Modify Customer



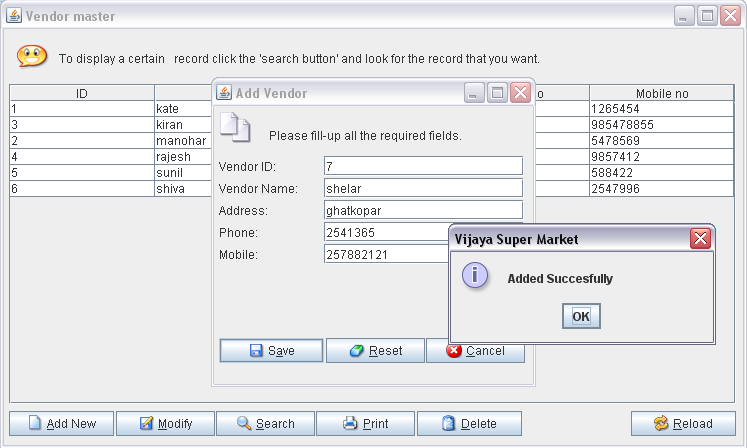
**frmsearchcust**



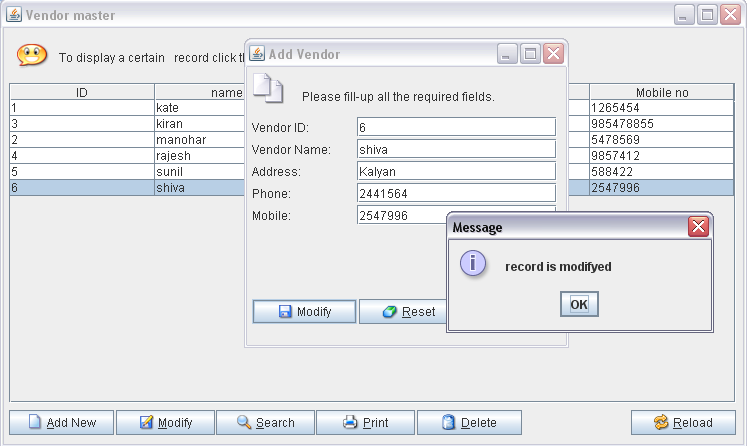
Deleting from frmctab



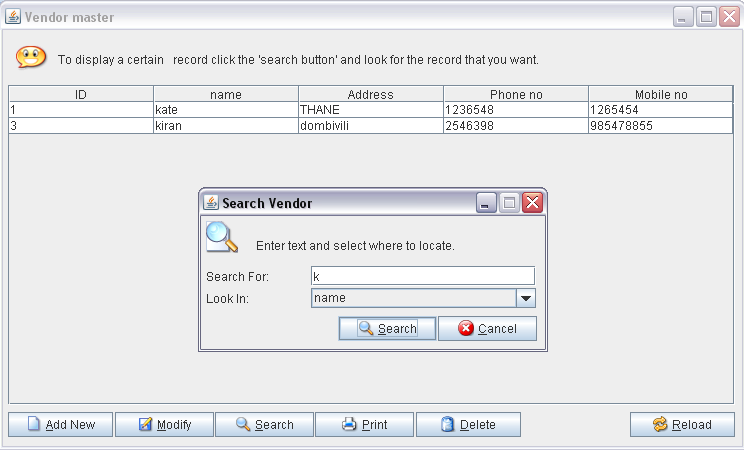
frmaddven



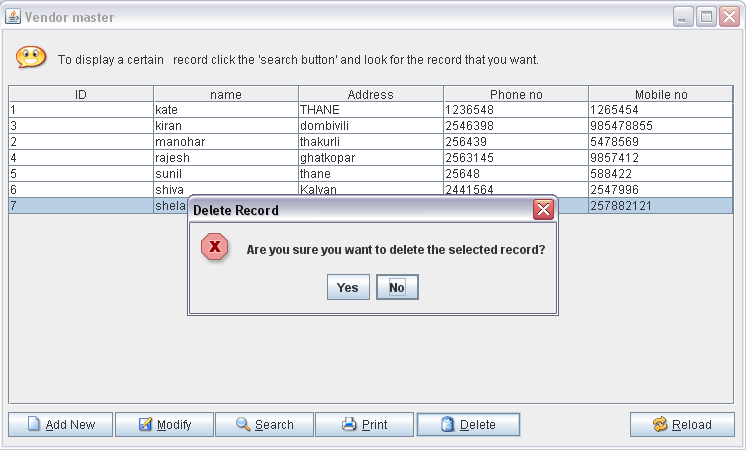
Modify Vendor



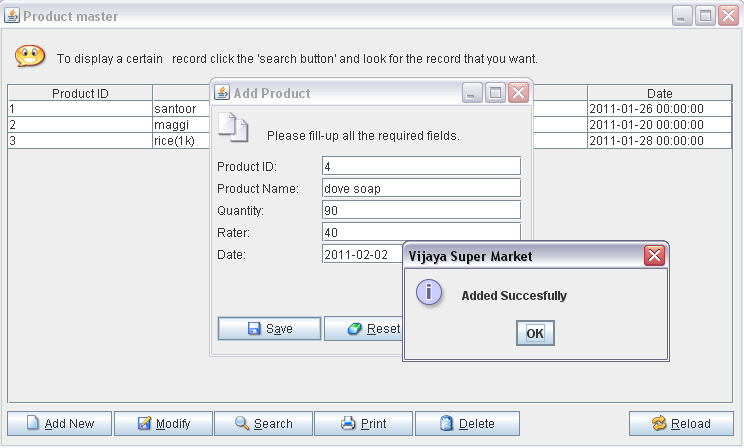
frmsearchven



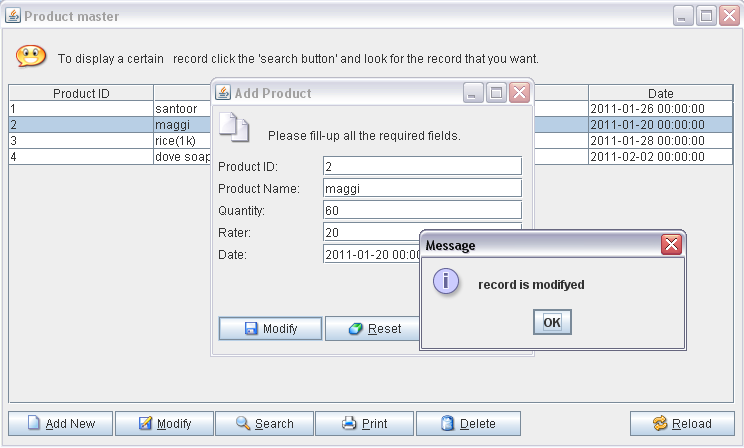
Deleting from frmvtab



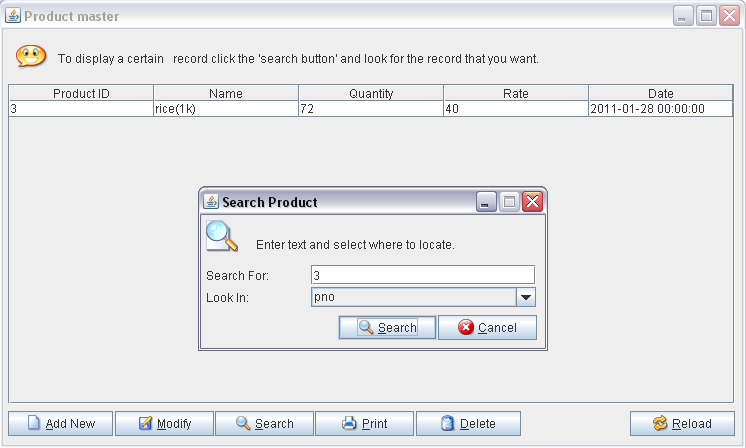
frmaddpro



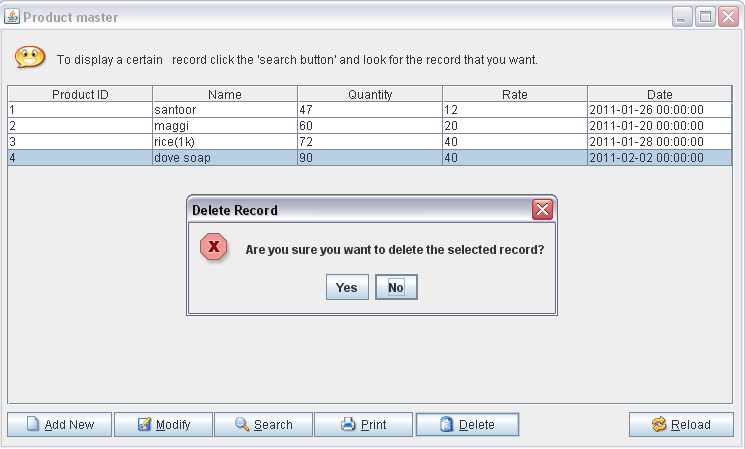
Modify Product



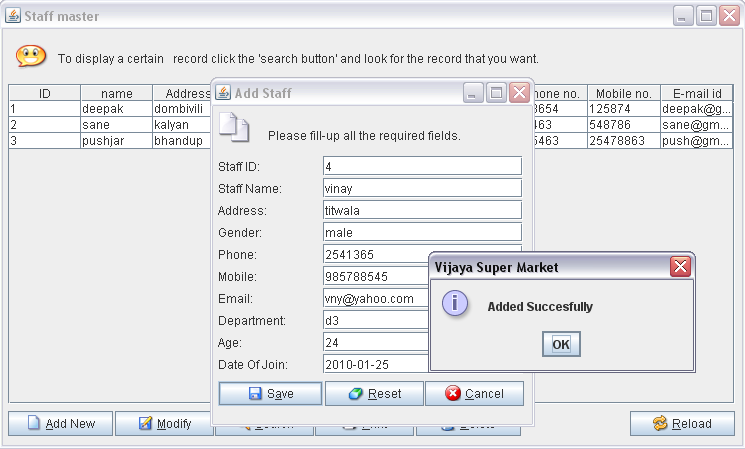
frmsearchpro



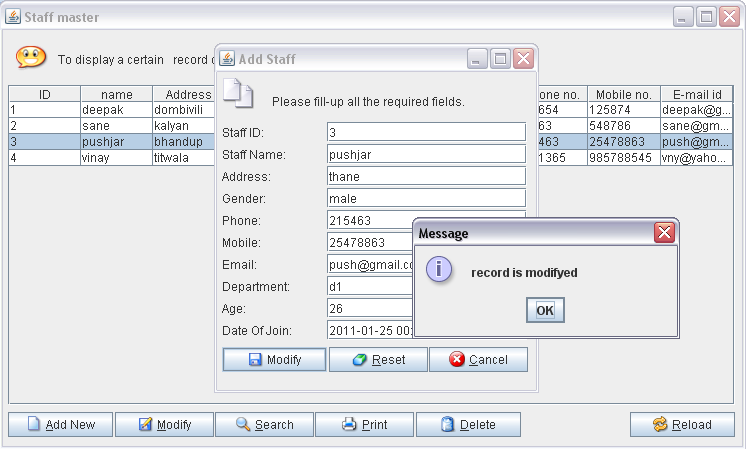
Deleting from frmptab



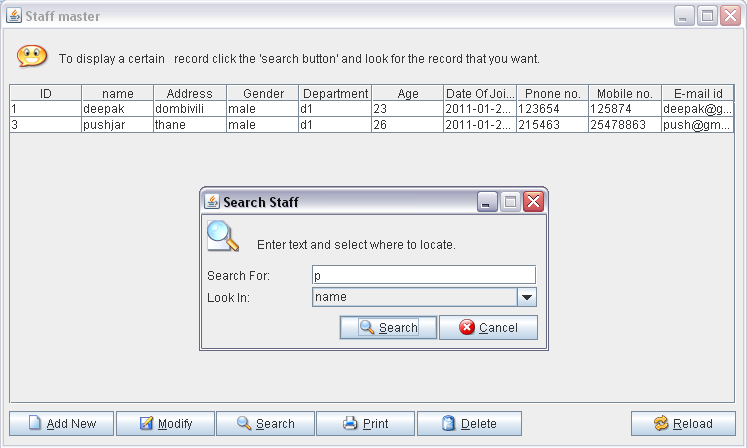
frmaddstaff



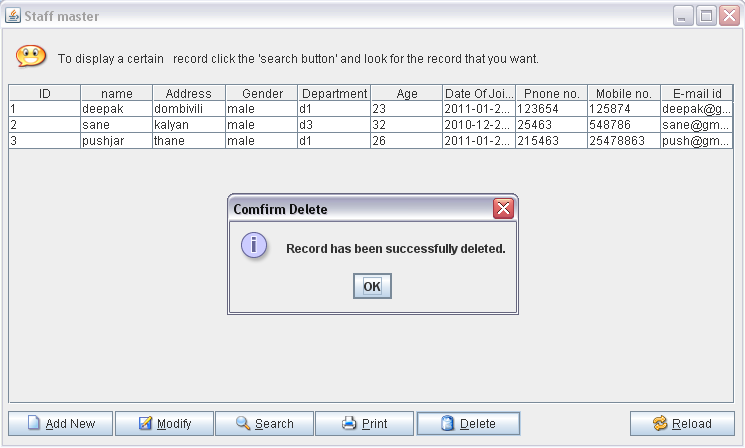
Modify Staff



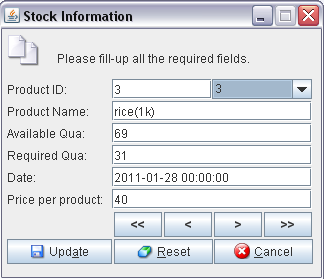
frmsearchstaff



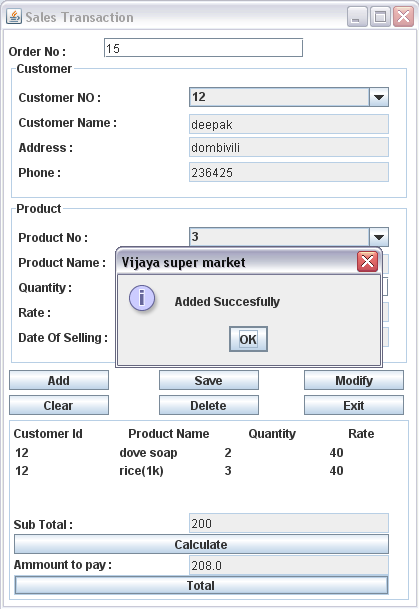
Deleting from frmptab



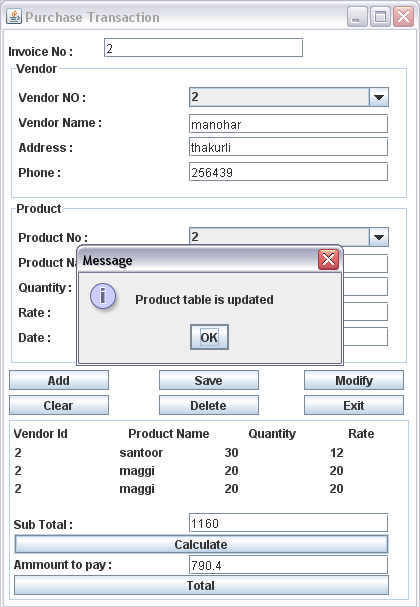
frmstkinfo



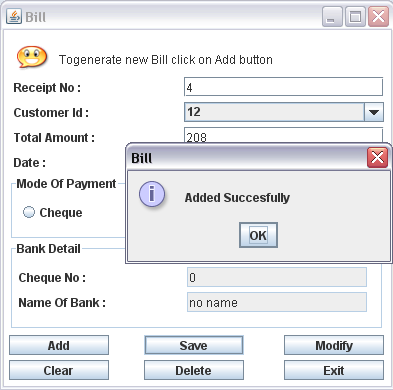
frmstran

****

frmptran

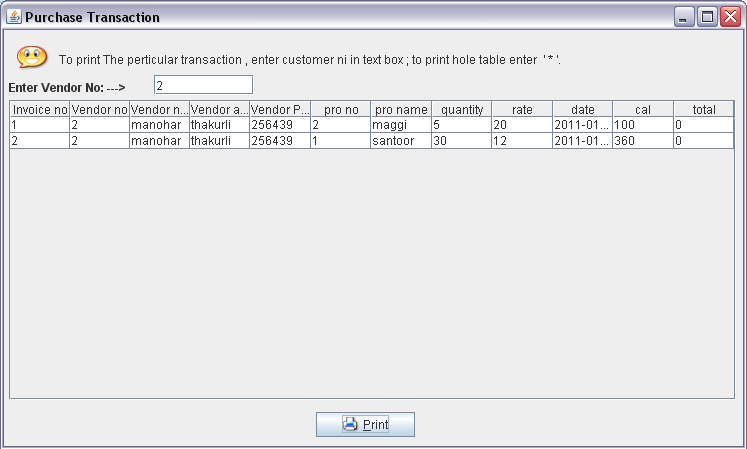


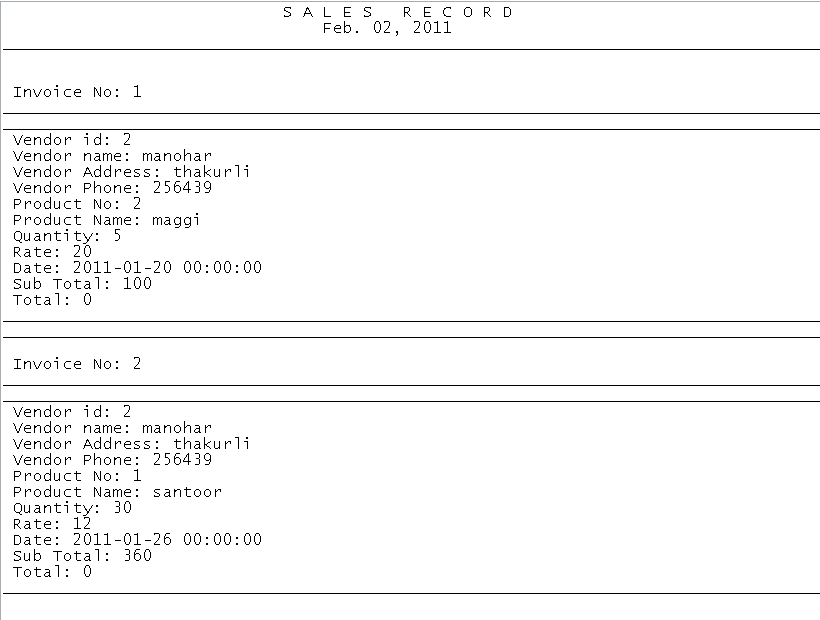
frmbill



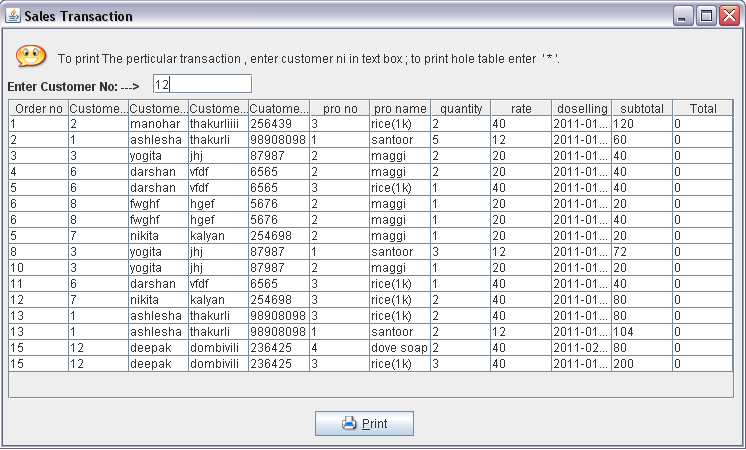
Report Layout

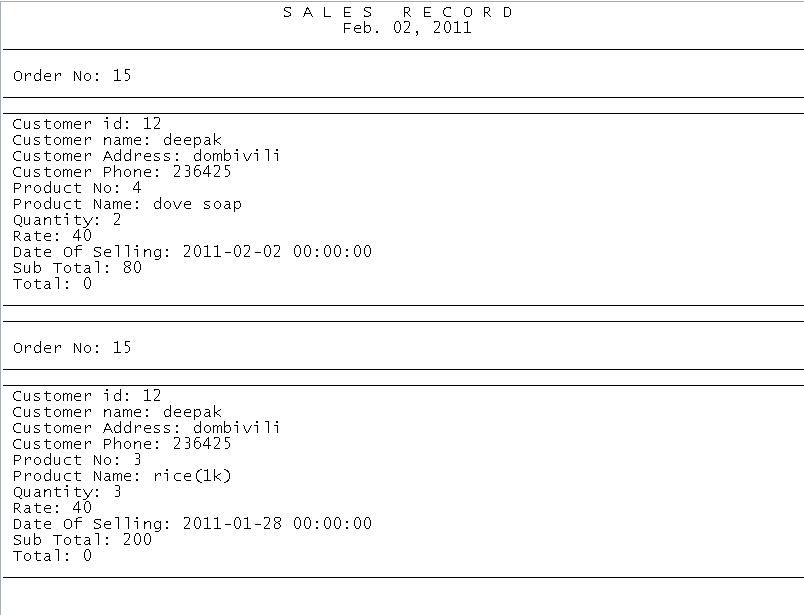
frmperrep



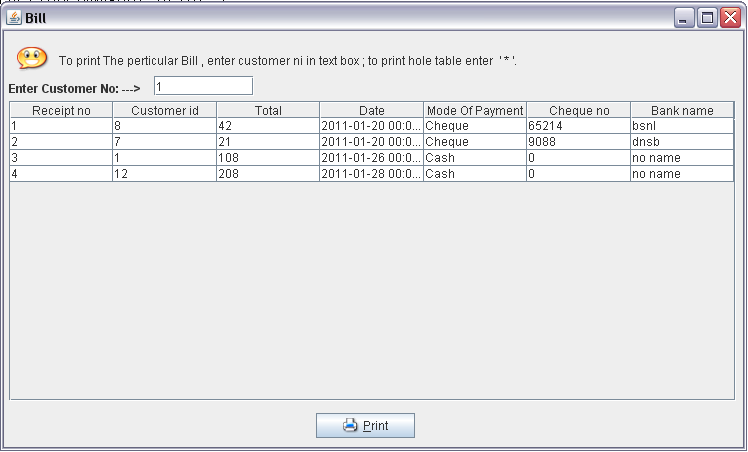


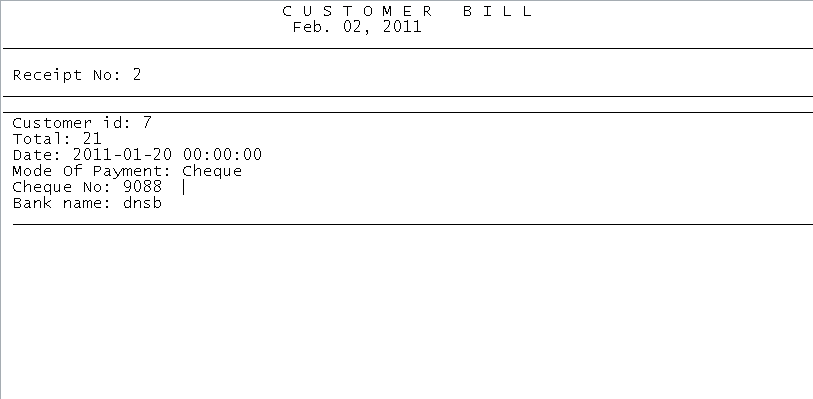
frmsalerep





frmbillrep





Validations

**Testing:-**

Software testing is the process used to help, identify the correctness, completeness, security, quality of developed computer software. Testing is a process of technical investigation performed on behalf of stake holders that is intended to reveal quality of related information about the product with respect to the context in which it is intended to operate. This includes, but is not limited to the process of executing a program or application with the intend of finding errors.

**Test Levels:**

**Unit Testing**: Testing of the minimal software components and sub-components or modules by the programmer.

**Integration Testing**: Exposes defects in the interface and interaction between integrated components (modules).

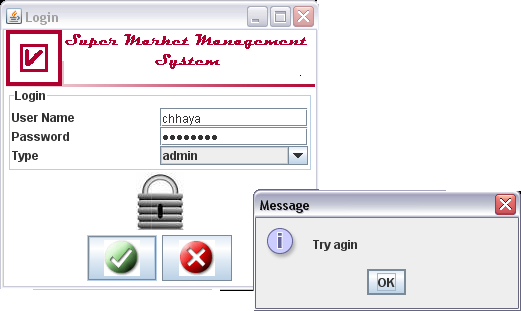
**Functional Testing**: Tests the product according to programmable work.

**System Testing**: Test an integrated system to verify /validate that it makes it requirements.

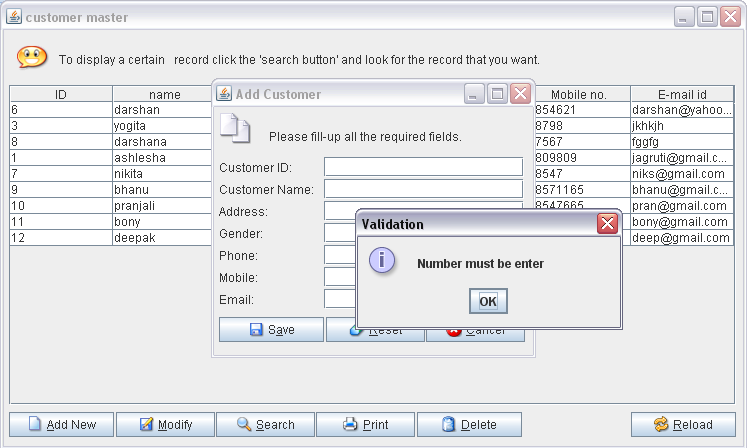
**Acceptance Testing**: Can be conducted by the client. It allows the end user or customer or client to decide whether or not to accept the product.

Acceptance testing may be performed after the testing and before the implementation phase.

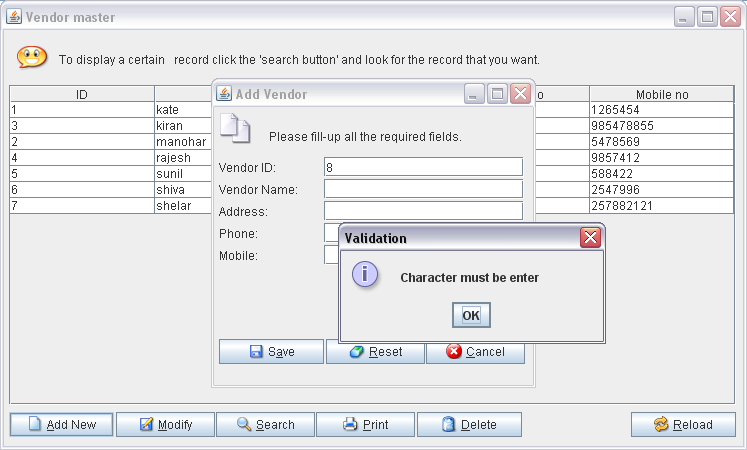
Login Validation



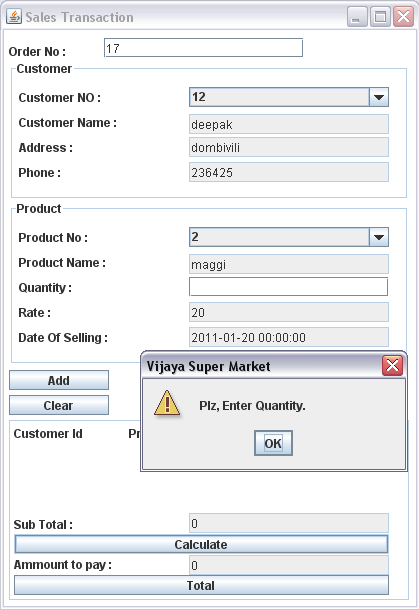
Validation for number



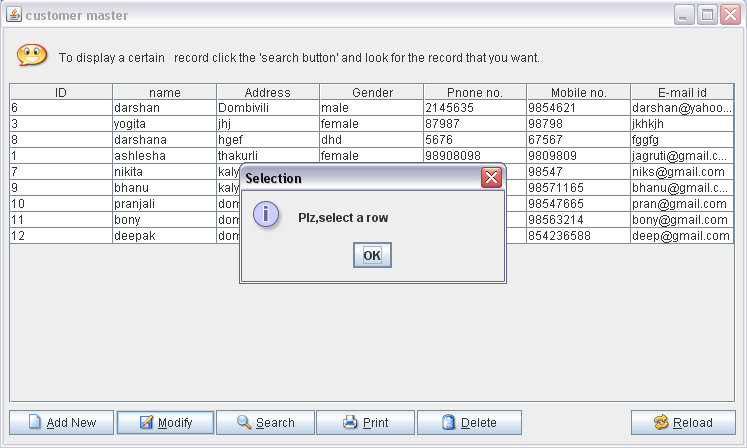
Validation for Character



Empty field validation



Selection Validation



TestData

Customer Master

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Validation** | id | name | address | gender | phone | mobile |
| **Valid** | Numeric | String | String | String | Numeric | Numeric |
| **Invalid** | String | Numeric | Numeric | Numeric | String | String |

Staff Master

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Validation** | id | name | address | gender | Dep | Age | |
| **Valid** | Numeric | String | String | String | String | Numeric | |
| **Invalid** | String | Numeric | Numeric | Numeric | Numeric | | String | |
| **Validation** | Doj | Phone | Mobile |
| **Valid** | Numeric | Numeric | Numeric |
| **Invalid** | String | String | String |

Product Master

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Validation** | id | name | quantity | rate | date |
| **Valid** | Numeric | String | Numeric | Numeric | Numeric |
| **Invalid** | String | Numeric | String | String | String |

Vendor Master

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Validation** | id | name | address | gender | phone | mobile |
| **Valid** | Numeric | String | String | String | Numeric | Numeric |
| **Invalid** | String | Numeric | Numeric | Numeric | String | String |

Total

|  |  |  |  |
| --- | --- | --- | --- |
| **Validation** | cid | Date | total |
| **Valid** | Numeric | Numeric | Numeric |
| **Invalid** | String | String | String |

Sale Transaction

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Validation** | Orderno | Custno | Custname | Custadd | Custphone | Prono |
| **Valid** | Numeric | Numeric | String | String | Numeric | Numeric |
| **Invalid** | String | String | Numeric | Numeric | String | String |
| **Validation** | Proname | Quantity | Rate | Date |
| **Valid** | String | Numeric | Numeric | Numeric |
| **Invalid** | Numeric | String | String | String |

Purchase Transaction

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Validation** | Invoice | Id | Name | Add | Phone | Proname |
| **Valid** | Numeric | Numeric | String | String | Numeric | String |
| **Invalid** | String | String | Numeric | Numeric | String | Numeric |
| **Validation** | Prono | Quantity | Rate | Date |
| **Valid** | Numeric | Numeric | Numeric | Numeric |
| **Invalid** | String | String | String | String |

Bill

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Validation** | Receiptno | Custid | Total | Date | Mop | Chequeno |
| **Valid** | Numeric | Numeric | Numeric | Numeric | String | Numeric |
| **Invalid** | String | String | String | String | Numeric | String |
| **Validation** | bankname |
| **Valid** | String |
| **Invalid** | Numeric |

Source Code

Frmsplash:-

import java.awt.\*;

import java.awt.event.\*;

import java.awt.Color;

import javax.swing.\*;

public class splash extends JWindow

{

JProgressBar current;

JPanel p2,p3,p4;

int num = 0;

splash()

{

Container d=getContentPane();

d.setLayout(new BorderLayout());

UIManager.put("ProgressBar.selectionBackground",Color.BLUE);

UIManager.put("ProgressBar.selectionForeground",Color.RED);

UIManager.put("ProgressBar.foreground",new Color(105,105,105));

current = new JProgressBar(0,2000);

current.setSize(10,20);

current.setValue(0);

current.setStringPainted(true);

current.setVisible(false);

JLabel l2=new JLabel(new ImageIcon("image\\Splash.gif"),JLabel.CENTER);

p3=new JPanel(new BorderLayout());

p3.setBackground(Color.white);

p3.add(BorderLayout.CENTER,l2);

p3.add(BorderLayout.SOUTH,current);

d.add(BorderLayout.CENTER,p3);

JPanel p1=new JPanel(new GridLayout(2,1));

p1.setBackground(new Color(220,220,220));

Toolkit tk=Toolkit.getDefaultToolkit();

Dimension d1=tk.getScreenSize();

int w=d1.width;

int h=d1.height;

setSize(400,200);

setVisible(true);

setLocation(w/4,h/4);

}

public void iterate() {

while (num<2000) {

current.setValue(num);

try {

Thread.sleep(150);

} catch (InterruptedException e) { }

num += 95;

}new Login();

dispose();

}

public static void main(String[]args)

{

splash s1=new splash();

s1.iterate();

}

}

Frmctab

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

import java.sql.\*;

import java.util.\*;

import java.awt.PrintJob;

import java.text.\*;

import javax.swing.table.DefaultTableModel;

public class Ctab extends JFrame

{

Toolkit kit=Toolkit.getDefaultToolkit();

Dimension screenSize=kit.getScreenSize();

int sh=screenSize.height;

int sw=screenSize.width;

public static int v,h;

public static Vector ch,row;

public static JTable jt1;

public static DefaultTableModel model;

public static JScrollPane s1;

static Connection con;

static ResultSet rs;

static Statement st;

public int a=0;

public String c;

static Container JPContainer;

//................................Method...............

public static void reloadRecord(String s)

{

try{

removeRecord();

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

Connection conn=DriverManager.getConnection("jdbc:odbc:super");

Statement stt=conn.createStatement();

ResultSet rss=stt.executeQuery(s);

while(rss.next())

{

String a=rss.getString("id");

//System.out.println(a);

String b=rss.getString("name");

String c=rss.getString("address");

String d=rss.getString("gender");

String e=rss.getString("phone");

String f=rss.getString("mobile");

String g=rss.getString("email");

String[] addrec={a,b,c,d,e,f,g};

model.addRow(addrec);

}

}catch(Exception e){System.out.println(e);}

}

//................................method...............

public static void removeRecord()

{

try{

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

Connection conn=DriverManager.getConnection("jdbc:odbc:super");

Statement stt=conn.createStatement();

ResultSet rss=stt.executeQuery("Select \* from Custdetail");

while(rss.next())

{

String a=rss.getString("id");

//System.out.println(a);

String b=rss.getString("name");

String c=rss.getString("address");

String d=rss.getString("gender");

String e=rss.getString("phone");

String f=rss.getString("mobile");

String g=rss.getString("email");

String[] addrec={a,b,c,d,e,f,g};

model.removeRow(0);

}

}catch(Exception e){}

}

//................................Start of Cunstructor...............

public Ctab()

{

super("customer master");

model = new DefaultTableModel();

JPContainer=getContentPane();

JPContainer.setLayout(null);

JLabel JLPicture1 = new JLabel(new ImageIcon("image/helper.png"));

JLabel JLHelpText = new JLabel("To display a certain \n record click the 'search button' and look for the record that you want.");

JButton JBAddNew = new JButton("Add New",new ImageIcon("image/new.png"));

JButton JBModify = new JButton("Modify",new ImageIcon("image/modify.png"));

JButton JBSearch = new JButton("Search",new ImageIcon("image/search.png"));

JButton JBPrint = new JButton("Print",new ImageIcon("image/print.png"));

JButton JBDelete = new JButton("Delete",new ImageIcon("image/delete.png"));

JButton JBReload = new JButton("Reload",new ImageIcon("image/reload.png"));

//-- Add the JLPicture1

JLPicture1.setBounds(5,5,48,48);

JPContainer.add(JLPicture1);

//-- Add the JLHelpText

JLHelpText.setBounds(55,5,570,48);

JLHelpText.setFont(new Font("Dialog", Font.PLAIN, 12));

JPContainer.add(JLHelpText);

try{

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

Connection conn=DriverManager.getConnection("jdbc:odbc:super");

Statement stt=conn.createStatement();

ResultSet rss=stt.executeQuery("Select \* from Custdetail");

model.addColumn("ID");

model.addColumn("name");

model.addColumn("Address");

model.addColumn("Gender");

model.addColumn("Pnone no.");

model.addColumn("Mobile no.");

model.addColumn("E-mail id");

while(rss.next())

{

String a=rss.getString("id");

System.out.println(a);

String b=rss.getString("name");

String c=rss.getString("address");

String d=rss.getString("gender");

String e=rss.getString("phone");

String f=rss.getString("mobile");

String g=rss.getString("email");

String[] addrec={a,b,c,d,e,f,g};

model.addRow(addrec);

}

jt1 = new JTable(model);

jt1.setBackground(Color.white);

v=ScrollPaneConstants.VERTICAL\_SCROLLBAR\_AS\_NEEDED;

h=ScrollPaneConstants.HORIZONTAL\_SCROLLBAR\_AS\_NEEDED;

s1=new JScrollPane(jt1,v,h);

s1.setBounds(5,55,727,320);

JPContainer.add(s1);

st.close();

con.close();

}catch(Exception e){}

//-- Add the JBAddNew

JBAddNew.setBounds(5,382,105,25);

JBAddNew.setFont(new Font("Dialog", Font.PLAIN, 12));

JBAddNew.setMnemonic(KeyEvent.VK\_A);

JBAddNew.setActionCommand("add");

JPContainer.add(JBAddNew);

//-- Add the JBModify

JBModify.setBounds(112,382,99,25);

JBModify.setFont(new Font("Dialog", Font.PLAIN, 12));

JBModify.setMnemonic(KeyEvent.VK\_M);

JBModify.setActionCommand("modify");

JPContainer.add(JBModify);

//-- Add the JBSearch

JBSearch.setBounds(212,382,99,25);

JBSearch.setFont(new Font("Dialog", Font.PLAIN, 12));

JBSearch.setMnemonic(KeyEvent.VK\_S);

JBSearch.setActionCommand("search");

JPContainer.add(JBSearch);

//-- Add the JBPrint

JBPrint.setBounds(312,382,99,25);

JBPrint.setFont(new Font("Dialog", Font.PLAIN, 12));

JBPrint.setMnemonic(KeyEvent.VK\_P);

JBPrint.setActionCommand("print");

JPContainer.add(JBPrint);

//-- Add the JBDelete

JBDelete.setBounds(413,382,105,25);

JBDelete.setFont(new Font("Dialog", Font.PLAIN, 12));

JBDelete.setMnemonic(KeyEvent.VK\_D);

JBDelete.setActionCommand("delete");

JPContainer.add(JBDelete);

//-- Add the JBReload

JBReload.setBounds(627,382,105,25);

JBReload.setFont(new Font("Dialog", Font.PLAIN, 12));

JBReload.setMnemonic(KeyEvent.VK\_R);

JBReload.setActionCommand("reload");

JPContainer.add(JBReload);

setSize(747,450);

setVisible(true);

setLocation((sw - 747)/2,((sh-450)/2)-45);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

JBAddNew.addActionListener(new ActionListener()

{

public void actionPerformed(ActionEvent ae)

{

a=0;

new Addcus(a,"");

}

});

JBSearch.addActionListener(new ActionListener()

{

public void actionPerformed(ActionEvent ae)

{

new Searchcus();

}

});

JBModify.addActionListener(new ActionListener()

{

public void actionPerformed(ActionEvent ae)

{

try{

a=1;

if(jt1.getValueAt(jt1.getSelectedRow(),jt1.getSelectedColumn())!=null)

{

c="Select \* from Custdetail where id = " + jt1.getValueAt(jt1.getSelectedRow(),0);

new Addcus(a,c);

}

}catch(Exception e){

JOptionPane.showMessageDialog(Ctab.this,"Plz,select a row","Selection",JOptionPane.INFORMATION\_MESSAGE);

}

}

});

JBDelete.addActionListener(new ActionListener()

{

public void actionPerformed(ActionEvent ae)

{try{

if(jt1.getValueAt(jt1.getSelectedRow(),jt1.getSelectedColumn())!=null)

{

String ObjButtons[] = {"Yes","No"};

int PromptResult = JOptionPane.showOptionDialog(null,"Are you sure you want to delete the selected record?","Delete Record",JOptionPane.DEFAULT\_OPTION,JOptionPane.ERROR\_MESSAGE,null,ObjButtons,ObjButtons[1]);

if(PromptResult==0){

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

Connection con=DriverManager.getConnection("jdbc:odbc:super");

Statement st=con.createStatement();

//ResultSet rs=

st.executeUpdate("Delete from Custdetail where id = " + jt1.getValueAt(jt1.getSelectedRow(),0));

model.removeRow(jt1.getSelectedRow());

JOptionPane.showMessageDialog(null,"Record has been successfully deleted.","Comfirm Delete",JOptionPane.INFORMATION\_MESSAGE);

}

}

}catch(Exception e){JOptionPane.showMessageDialog(Ctab.this,"Plz,select a row","Selection",JOptionPane.INFORMATION\_MESSAGE);}

}

});

JBReload.addActionListener(new ActionListener()

{

public void actionPerformed(ActionEvent ae)

{

try{

reloadRecord("Select \* from Custdetail");

}catch(Exception e){System.out.println(e);}

}

});

JBPrint.addActionListener(new ActionListener()

{

public void actionPerformed(ActionEvent ae)

{

try{

if(jt1.getValueAt(jt1.getSelectedRow(),jt1.getSelectedColumn()) != null){

clsPublicMethods PrintingClass = new clsPublicMethods();

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

con=DriverManager.getConnection("jdbc:odbc:super");

Statement st1=con.createStatement();

ResultSet rsPrint = st1.executeQuery("SELECT \* FROM Custdetail WHERE id =" + jt1.getValueAt(jt1.getSelectedRow(),0) );

if(rsPrint.next()==true){

String RecordToPrint = "";

java.util.Date CurrentDate = new java.util.Date();

SimpleDateFormat SDFDateFormatter = new SimpleDateFormat("MMM. dd, yyyy",Locale.getDefault());

RecordToPrint += " C U S T O M E R R E C O R D \n";

RecordToPrint += " " + SDFDateFormatter.format(CurrentDate).toString() + "\n\n\n";

RecordToPrint += "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n\n";

RecordToPrint += " Customer ID: " + rsPrint.getString("id") + "\n";

RecordToPrint += "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

RecordToPrint += "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n";

RecordToPrint += " name: " + rsPrint.getString("name") + "\n";

RecordToPrint += " Address: " + rsPrint.getString("address") + "\n";

RecordToPrint += " Gender: " + rsPrint.getString("gender") + "\n";

RecordToPrint += " Phone: " + rsPrint.getString("phone") + "\n";

RecordToPrint += " Mobile: " + rsPrint.getString("mobile") + "\n";

RecordToPrint += " Email: " + rsPrint.getString("email") + "\n";

RecordToPrint += "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n";

PrintingClass.printRecord(RecordToPrint,Ctab.this);

CurrentDate=null;

SDFDateFormatter=null;

RecordToPrint=null;

}else{

JOptionPane.showMessageDialog(null,"The selected record has been change since last modified. Please click the 'Reload' button and try again!","No record to print",JOptionPane.WARNING\_MESSAGE);

}

//Dispose the variable

rsPrint=null;

}

}catch(Exception sqlE){System.out.println(sqlE);}

/\*if(sqlE.getMessage() != null){

System.out.println(sqlE.getMessage());

}else{

JOptionPane.showMessageDialog(null,"Please select a record in the list to print.","No Record Selected",JOptionPane.INFORMATION\_MESSAGE);

}\*/

}

});

}

//................................Start of Cunstructor...............

public static void main(String args[])

{

new Ctab();

}

}

System Implementation

Implementation

The following steps were carried out in implementation phase.

* Conduct Training: The training was conducted for the employees of the company to make them familiar with the system.
* Bug fixing and documentation: Any errors that occurred were solved and documented.
* Install the system: The system was then installed.

Maintenance

The maintenance of software is the time period in which the software is software product performs useful work. Maintenance activities involve making enhancement activities to the, adapting product to new environment and correcting problems. Software enhancement may involve providing new functional capabilities, improving user displays and modes of interaction.

Adaptation of software to a new environment may involve moving the software to a different machine. Problem correction involves modification and revalidation of software to correct errors. The four types of maintenance activities are described below:

🡪 Corrective Maintenance

🡪 Preventive Maintenance

🡪 Perfective Maintenance

🡪 Adaptive Maintenance

Corrective Maintenance:

Corrective maintenance can be defined as the [maintenance](http://en.wikipedia.org/wiki/Maintenance) which is required when an item has failed or worn out, to bring it back to working order.

Corrective maintenance is the most commonly used maintenance approach, but it is easy to see its limitations. When equipment fails, it often leads to downtime in production, and sometime it causes spreading of damage to other parts. In most cases this is costly business. Also, if the equipment needs to be replaced, the cost of replacing it alone can be substantial.

Adaptive Maintenance:

Adaptive maintenance is an activity that modifies software to properly interface with the changing environment.

Perfective Maintenance:

Perfective maintenance is performed to satisfy user requests such as new Capabilities, modifications to existing functions and general enhancements.

Preventive Maintenance:

Preventive maintenance occurs when software is changed to improve future maintainability or to provide a better basic for future enhancements.

Conclusion

An attempt is made in all its earnest towards the successful completion of the project. This system was verified with valid as well as with invalid data.

This system is user friendly since it has been developed in Visual Studio 2008, a successful GUI environment. Since the connection can be extended to any database. The control will be more powerful.

Connecting it to any type of database extends the development control. Any suggestions for future development of the system are welcome.

Upgrading the system can be done without affecting the proper functioning of system.

BIBLIOGRAPHY

**References:**

Books:

COMPLETE REFERENCE - HEWELLT SCHILDT

JAVA PROGRAMMING - BALGURUSWAMY

JAVA BEGINNERS - WROX PUBLICATION

HEADSTART IN JAVA - S. KATHY

Websites:

www.java.sun.com

* java.sun.com/products/java-media/jmf/2.1.1/specdownload.html
* java.sun.com/products/java-media/jmf/2.1.1/documentation.html
* java.sun.com/products/java-media/jmf/2.1.1