1.ABSTRACT

Commodity management is key factor in success of any organization. Commodity means any idle resource of an enterprise it also means stock on hand at a given time. So, the function of commodity management is to find sufficient amount of stocks that will fulfil the demand without causing over stocks. This research paper is focused on various aspects of commodity management as well as the benefits of implementing commodity management in organization. Commodity is commonly used to indicate materials like raw materials, finished, semi-finished, packing, spares and others stocked in order to meet an expected demand or distribution. The basic purpose of commodity analysis, whether in manufacturing, distribution, retail or services, is to specify when items should be ordered and how last the order should be. In distribution, commodity is classified as in transit meaning that it is being moved in the system and warehouse which is commodity in a warehouse or distribution center retail sites carry commodity for immediate sale to customer. In services commodity generally refers to the tangible goods to be sold and the supplies necessary.

2.INTRODUCTION

The commodity management system is the mix of innovation that involves both hardware and software along with cycles and techniques that manage the observing and upkeep of loaded items, regardless of whether those items are organization resources, crude materials, and supplies, or completed items fit to be shipped off merchants or end customers. A commodity management system comprises of a system for distinguishing each stock thing and its related data, for example, standardized identification marks or resource labels; hardware instruments for perusing standardized tag marks, for example, handheld standardized tag scanners or cell phones with standardized identification filtering applications; Commodity the board programming, which gives a focal information base and perspective for all stock, combined with the capacity to examine information, create reports, gauge future interest, and that's just the beginning; Processes and strategies for marking, documentation, and announcing. This ought to incorporate a stock administration strategy like Just in Time, ABC Analysis, First-In First-Out (FIFO), Stock Review, or another demonstrated philosophy; People who prepared to follow these strategies and cycles. Materials Management [1] is related to organizing, getting, taking care of and giving the reasonable material of right quality, perfect sum at right spot in fortunate time to coordinate and schedule the creation development in an integrative course for a mechanical undertaking. Stock Management is fundamentally the technique by which an affiliation is given the items and undertakings that it needs to achieve its objectives of buying, amassing and improvement of materials. Stock organization systems are critical to how associations track and control inventories. Having the option to measure stock in a fortunate and definite manner is fundamental for having consistent business exercises since stock is routinely one of the greatest current assets on an association's bookkeeping report. Stock is a summary for items and materials, or those product and materials themselves, held open in stock by a business. A perfect commodity management system will mention to you what product is available, what is on hand when it will show, and what you've sold. With such a framework, you can design buys wisely and rapidly perceive the quick things you need to reorder and the sluggish things you should write down or uncommonly advance. [2]Some retailers track stock utilizing a manual label framework, which can be refreshed day by day, week after week, or even month to month. In a manual label framework, you eliminate sticker prices from the item at the place to checkout. You at that point crosscheck the labels against the actual stock to sort out what you have sold. A key stock organization measure is restoration.

3 . SYSTEM REQUIREMENTS

3.1 SOFTWARE AND HARDWARE REQUMENTS

The software requirements are description of features and functionalities of the target system. Requirements convey the expectations of users from the software product. The requirements can be obvious or hidden, known or unknown, expected or unexpected from client's point of view.

3.2 SOFTWARE REQUMENTS

Operating System:	Windows XP(min)
Front End:	Java 1.2 or Above (AWT, Swings, Files)
IDE	MyEclipse, Eclipse

3.3 HARDWARE REQUMENTS

3.1.1	Processor	-	intel i3,CORE -2
3.1.2	Speed	-	1.1 Ghz and above
3.1.3	RAM	-	8 GB
3.1.4	SSD	-	1 TB
3.1.5	Key Board	-	Standard Windows Keyboard(QWERTY)
3.1.6	Mouse	-	Two or Three Button Mouse
3.1.7	Monitor	-	SVGA

4.SOFTWARE'S USED FOR PROJECT DEVELOPMENT

4.1INTRODUCTION TO JAVA

Java is a programming language created by James Gosling from Sun Microsystems (Sun) in 1991. The target of Java is to write a program once and then run this program on multiple operating systems. The first publicly available version of Java (Java 1.0) was released in 1995. Sun Microsystems was acquired by the Oracle Corporation in 2010. Oracle has now the statesmanship for Java. In 2006 Sun started to make Java available under the GNU General Public License (GPL). Oracle continues this project called OpenJDK.

Over time new enhanced versions of Java have been released. The current version of Java is Java 1.8 which is also known as Java 8.

Java is defined by a specification and consists of a programming language, a compiler, core libraries and a runtime (Java virtual machine) The Java runtime allows software developers to write program code in other languages than the Java programming language which still runs on the Java virtual machine. The Java platform is usually associated with the Java virtual machine and the Java core libraries.

The Java language was designed with the following properties:

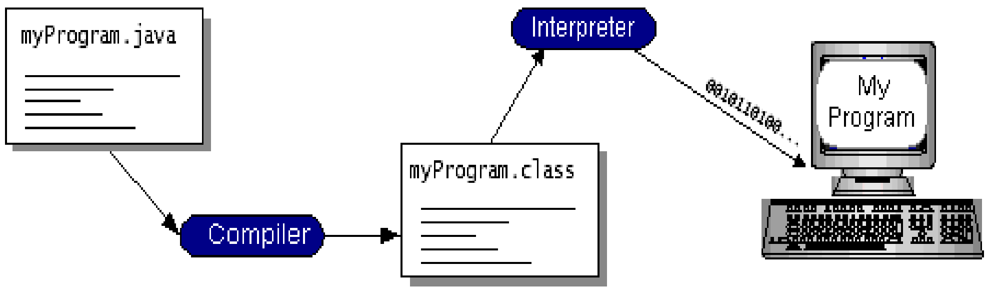
Platform independent: Java programs use the Java virtual machine as abstraction and do not access the operating system directly. This makes Java programs highly portable. A Java program (which is standard-compliant and follows certain rules) can run unmodified on all supported platforms, e.g., Windows or Linux.

Object-orientated programming language: Except the primitive data types, all elements in Java are objects.

Strongly-typed programming language: Java is strongly-typed, e.g., the types of the used variables must be pre-defined and conversion to other objects is relatively strict, e.g., must be done in most cases by the programmer.

Interpreted and compiled language: Java source code is transferred into the bytecode format which does not depend on the target platform. These bytecode instructions will be interpreted by the Java Virtual machine (JVM). The JVM contains a so called Hotspot-Compiler which translates performance critical bytecode instructions into native code instructions.

Automatic memory management: Java manages the memory allocation and de-allocation for creating new objects. The program does not have direct access to the memory. The so-called garbage collector automatically deletes objects to which no active pointer exists.
The Java syntax is similar to C++. Java is case-sensitive, e.g., variables called myValue and myvalue are treated as different variables.
With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called Java byte codes —the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works.



4.2IDE

4.2. IDE 4.2.1. Apache NetBeans IDE NetBeans IDE is an open-source integrated development environment. NetBeans IDE supports development of all Java application types (Java SE (including JavaFX), Java ME, web, EJB and mobile applications) out of the box. Among other features are an Ant-based project system, Maven support, refactorings, version control (supporting CVS, Subversion, Git, Mercurial and Clearcase). Modularity: All the functions of the IDE are provided by modules. Each module provides a welldefined function, such as support for the Java language, editing, or support for the CVS versioning system, and SVN. NetBeans contains all the modules needed for Java development in a single download, allowing the user to start working immediately. Modules also allow NetBeans to be extended. Ne New features, such as support for other programming languages, can be added by installing additional modules. For instance, Sun Studio, Sun Java Studio Enterprise, and Sun Java Studio Creator from Sun Microsystems are all based on the NetBeans IDE. License: The IDE is licensed under the Apache License 2.0. Previously, from July 2006 through 2007, NetBeans IDE was licensed under Sun's Common Development and Distribution License (CDDL), a license based on the Mozilla Public License (MPL). In October 2007, Sun announced that NetBeans would henceforth be offered under a dual license of the CDDL and the GPL version 2 licenses, with the GPL linking exception for GNU Classpath.[15] Oracle has donated NetBeans Platform and IDE to the Apache Foundation where it underwent incubation and graduated as a top level project

4.3 SDLC LIFE CYCLE

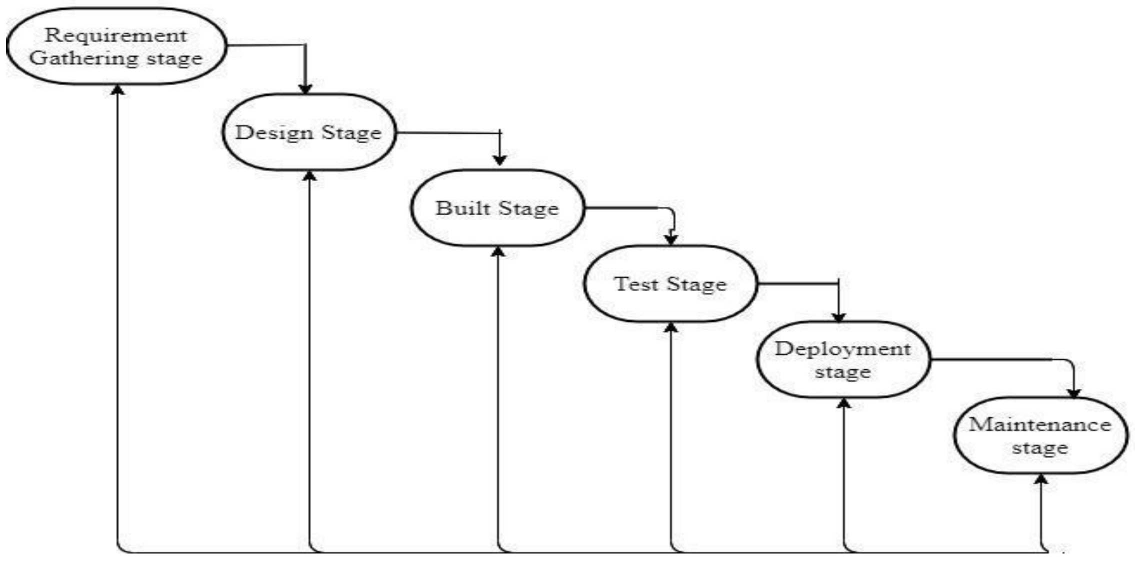
4.3.1 WATER FALL MODEL

The waterfall model is a popular version of the systems development life cycle model for software engineering. Often considered the classic approach to the systems development life cycle, the waterfall model describes a development method that is linear and sequential. Waterfall development has distinct goals for each phase of development. Imagine a waterfall on the cliff of a steep mountain. Once the water has flowed over the edge of the cliff and has begun its journey down the side of the mountain, it cannot turn back. It is the same with waterfall development. Once a phase of development is completed, the development proceeds to the next phase and there is no turning back Source.

Water fall model was the first process model to be introduced and widely used in the software engineering to ensure the success of the project.

The Waterfall model has got 6 phases and after the completion of each phase, the next phase will be started.

The phases are Requirement, Design, Coding and Implementation, Testing, Deployment and Maintenance. This is a very old model and the drawback is even if slight changes needs to be done it has to be started from the beginning.



Water Fall Model

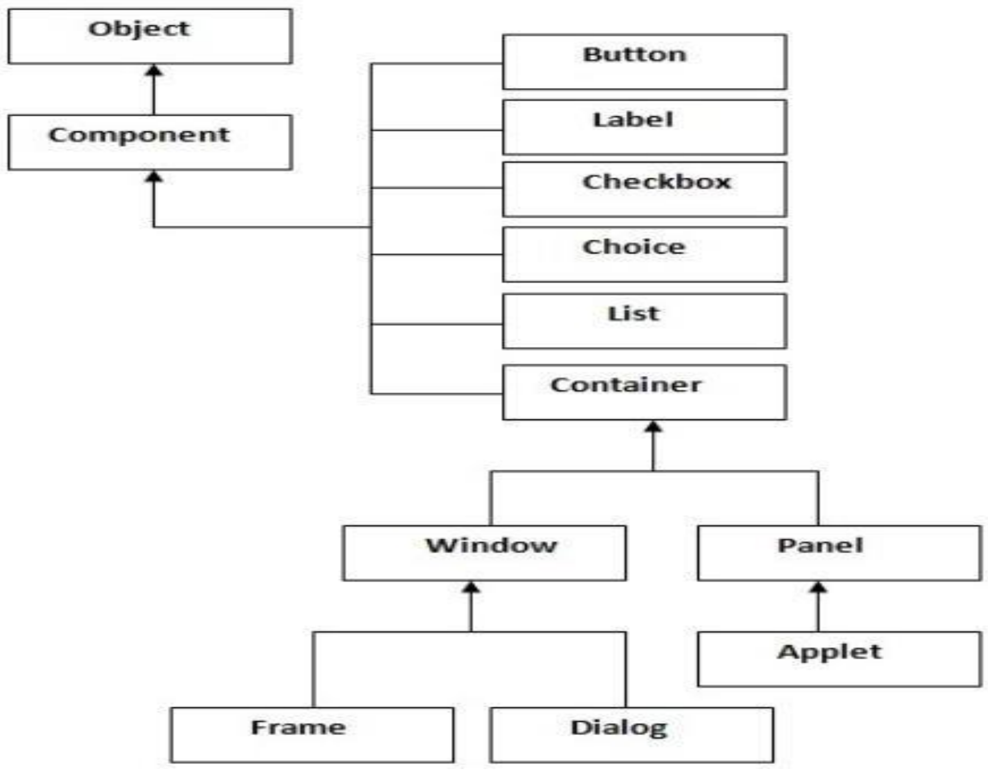
Different Phases of Water-Fall Models

Requirement Gathering stage	During this phase, detailed requirements of the software system to be developed are gathered from client
Design Stage	Plan the programming language like Java, PHP, .net or database like Oracle, MySQL, etc. Or other high-level technical details of the project
Built Stage	After design stage, it is built stage, that is nothing but coding the software
Test Stage	In this phase, you test the software to verify that it is built as per the specifications given by the client
Deployment stage	Deploy the application in the respective environment
Maintenance stage	Once your system is ready to use, you may later require change the code as per customer request

4.4 Java AWT

Java AWT components are platform-dependent i.e. components are displayed according to the view of operating system. AWT is heavyweight i.e. its components uses the resources of system. The java.awt package provides classes for AWT api such as TextField, Label, TextArea, RadioButton, CheckBox, Choice, List etc.

4.4.1. Java AWT Hierarchy.



4.4.2. Java AWT Hierarchy

The hierarchy of Java AWT classes are given below.

Container: The Container is a component in AWT that can contain another component like buttons, textfields, labels etc. The classes that extend Container class are known as container such as Frame, Dialog and Panel.

Window: The window is the container that have no borders and menu bars. You must use frame, dialog or another window for creating a window.

Panel: The Panel is the container that doesn't contain title bar and menu bars. It can have other components like button, text field etc.

Frame: The Frame is the container that contain title bar and can have menu bars. It can have other components like button, text field etc.

4.5 Swings

Swing is a GUI widget toolkit for Java. It is part of Oracle's Java Foundation Classes (JFC) – an API for providing a graphical user interface (GUI) for Java programs.
Swing was developed to provide a more sophisticated set of GUI components than the earlier Abstract Window Toolkit (AWT). Swing provides a native look and feel that emulates the look and feel of several platforms, and also supports a pluggable look and feel that allows applications to have a look and feel unrelated to the underlying platform. It has more powerful and flexible components than AWT. In addition to familiar components such as buttons, check boxes and labels, Swing provides several advanced components such as tabbed panel, scroll panes, trees, tables, and lists.
Unlike AWT components, Swing components are not implemented by platform-specific code. Instead, they are written entirely in Java and therefore are platform-independent. The term

"lightweight" is used to describe such an element.
Swing API is set of extensible GUI Components to ease developer's life to create JAVA based Front End/ GUI Applications. It is build upon top of AWT API and acts as replacement of AWT API as it has almost every control corresponding to AWT controls. Swing component follows a Model-View-Controller architecture to fulfill the following criterias.

"lightweight" is used to describe such an element.
Swing API is set of extensible GUI Components to ease developer's life to create JAVA based Front End/ GUI Applications. It is build upon top of AWT API and acts as replacement of AWT API as it has almost every control corresponding to AWT controls. Swing component follows a Model-View-Controller architecture to fulfill the following criterias

- A single API is to be sufficient to support multiple look and feel.
- API is to model driven so that highest level API is not required to have the data.
- API is to use the Java Bean model so that Builder Tools and IDE can provide better services to the developers to use it.

4.5.1 MVC Architecture

- Swing API architecture follows loosely based MVC architecture in the following manner.
- A Model represents component's data.
- View represents visual representation of the component's data.
- Controller takes the input from the user on the view and reflects the changes in Component's data.
- Swing component have Model as a seperate element and View and Controller part are clubbed in User Interface elements. Using this way, Swing has pluggable look-and-feel architecture.

4.5.2 Swing features

- **Light Weight –** Swing component are independent of native Operating System's API as Swing API controls are rendered mostly using pure JAVA code instead of underlying operating system calls.

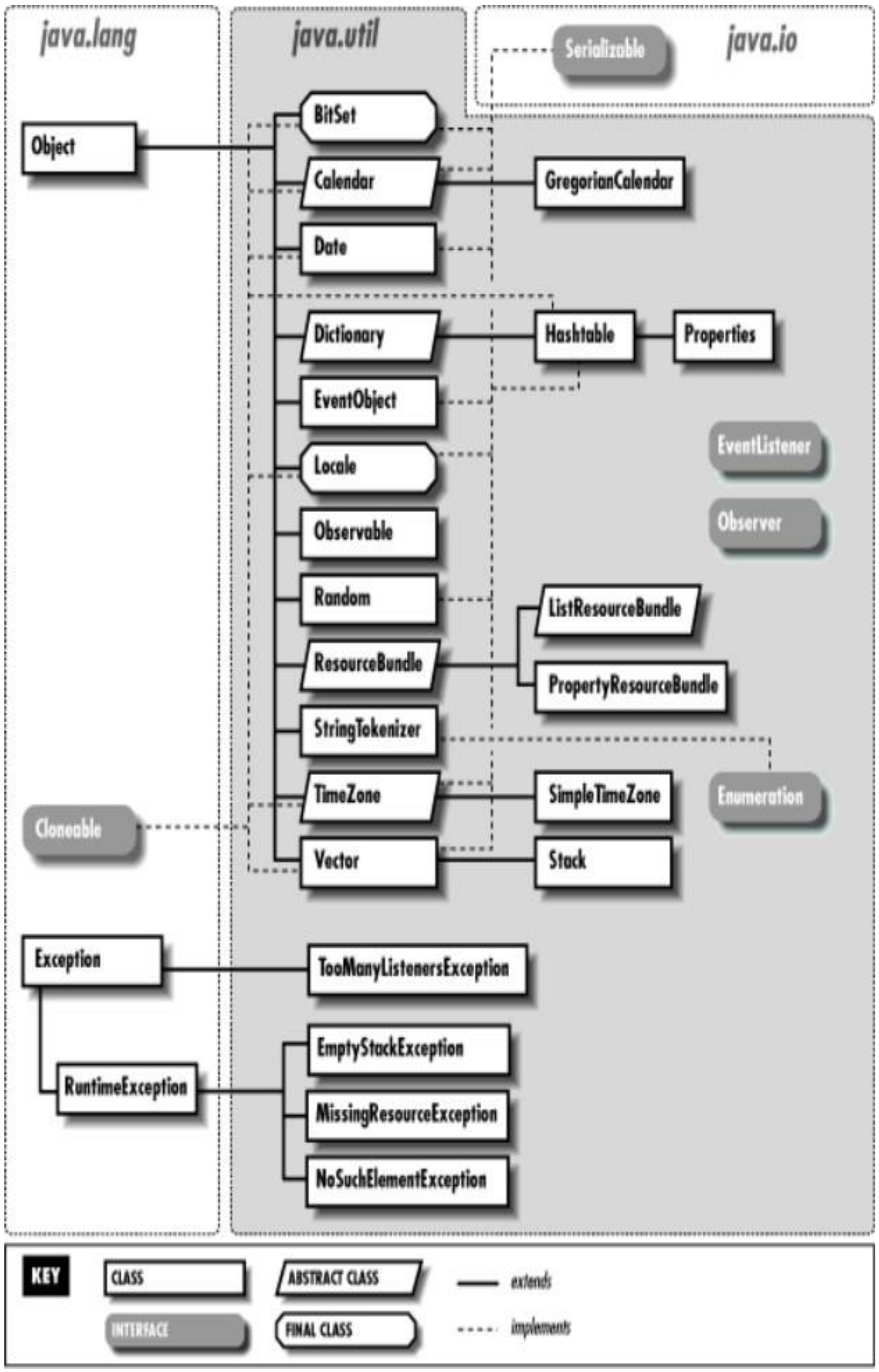
- **Rich controls –** Swing provides a rich set of advanced controls like Tree, TabbedPane, slider, colorpicker, table controls
- **Highly Customizable –** Swing controls can be customized in very easy way as visual apperance is independent of internal representation.
- **Pluggable look-and-feel-** SWING based GUI Application look and feel can be changed at run time based on available values.

4.6 Java.util

The package **java.util** contains a number of useful classes and interfaces. Although the name of the package might imply that these are utility classes, they are really more important than that. In fact, Java depends directly on several of the classes in this package, and many programs will find these classes indispensable. The classes and interfaces in **java.util** include:

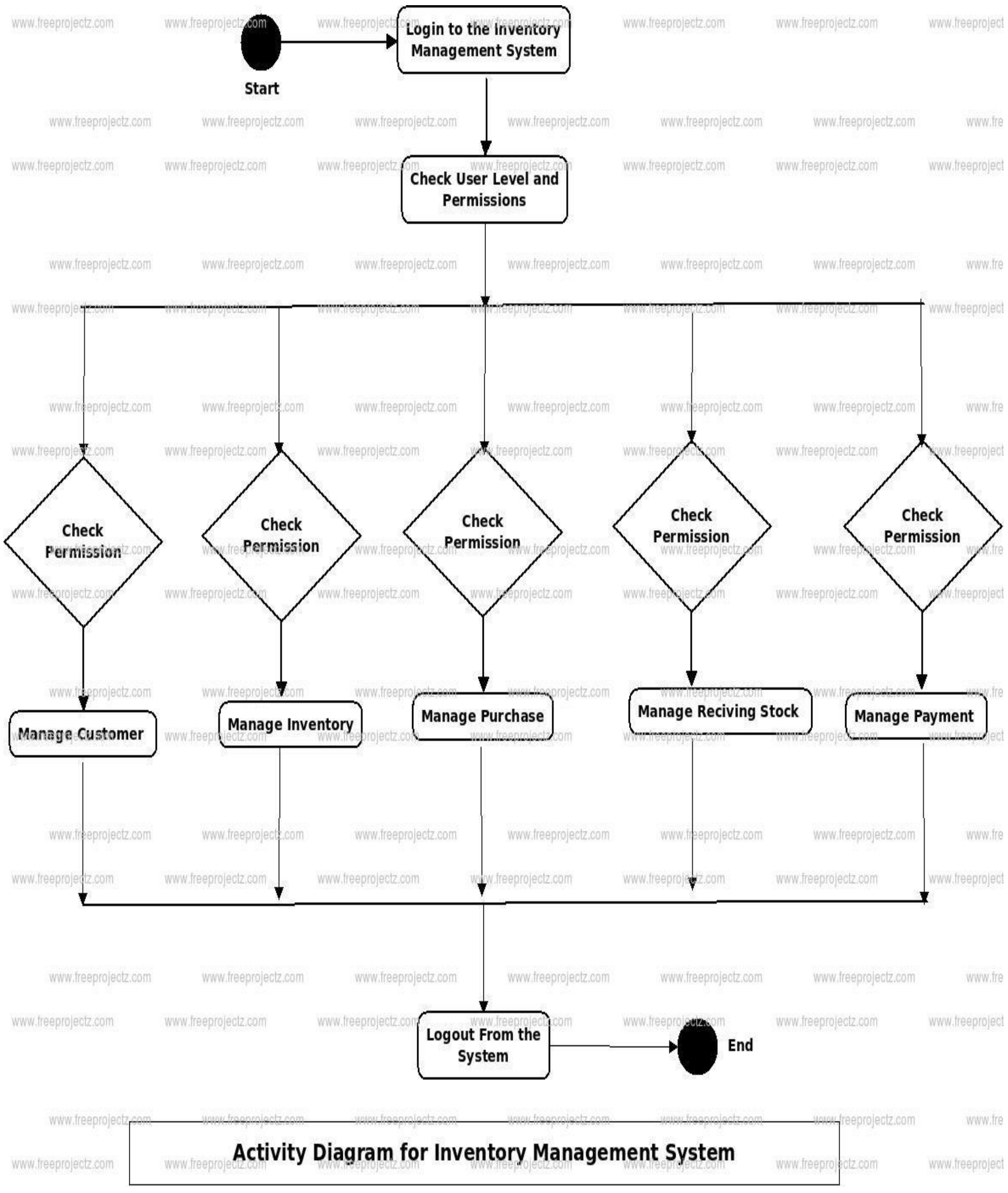
- The Hashtable class for implementing hashtables, or associative arrays.
- The Vector class, which supports variable-length arrays.
- The Enumeration interface for iterating through a collection of elements.
- The StringTokenizer class for parsing strings into distinct tokens separated by delimiter characters.
- The EventObject class and the EventListener interface, which form the basis of the new AWT event model in Java 1.1.
- The Locale class in Java 1.1, which represents a particular locale for internationalization purposes.
- The Calendar and TimeZone classes in Java. These classes interpret the value of a Date object in the context of a particular calendar system.
- The ResourceBundle class and its subclasses, ListResourceBundle and PropertyResourceBundle, which represent sets of localized data in Java 1.1.

4.6.1 The class hierarchy for the java.util package

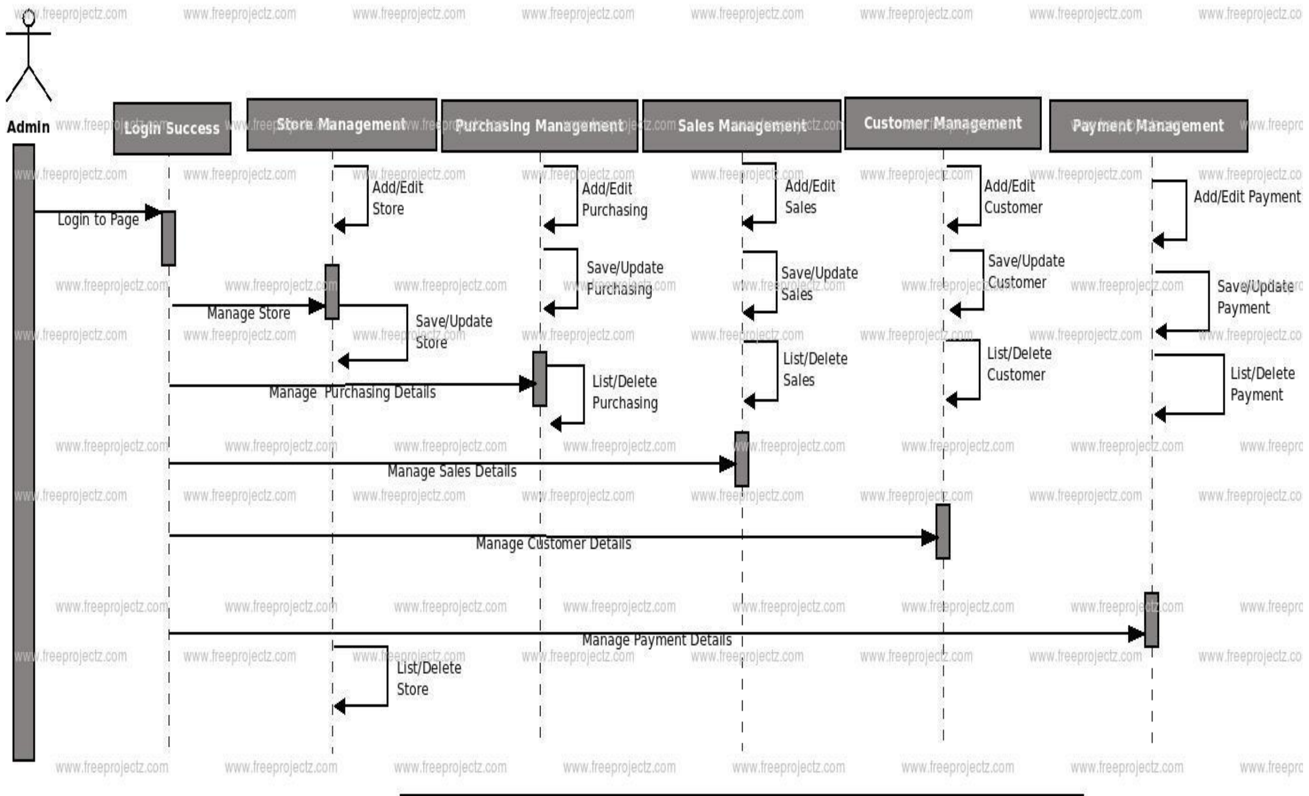


5 .SYSTEM DESIGN

5.1 ACTIVITY DIAGRAM

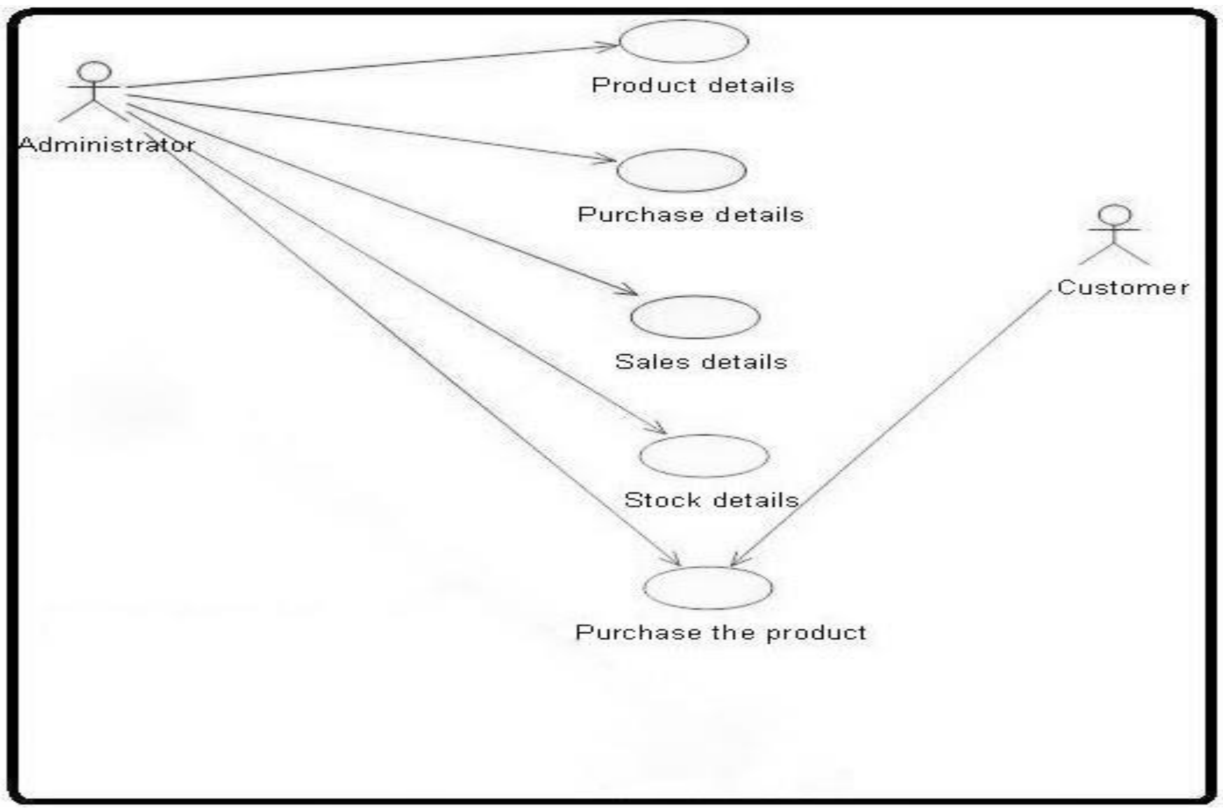


5.2 SEQUENCE DIAGRAM



Sequence Diagram of Super Market Management System

5.3 USE CASE DIAGRAM



6.SAMPLE CODE

6.1HOMEPAGE CODE

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
    <title>Courier</title>
    <style type="text/css">
        .style1
        {
            width: 100%;
            font-weight: bold;
            border: 6px solid GRAY;
            background-color: green;
        }
        .style2
        {
            font-size: xx-large;
            color: yellow;
        }
        .style3
        {
            width: 60%;
            border: 4px solid PINK;
        }
        a
        {
            color:yellow;
            font-weight:bold;
        }
    </style>
</head>
<body background="images/92e97314b6be31e1781561d268fbb833.jpg">

    <form NAME="HOME" ACTION="HOMEPAGE" METHOD="GET">
    <div>

        <table cellpadding="10" cellspacing="0" width="80%" class="style1">
            <tr>
                <td class="style2">
               <center>COMMODITY MANAGEMENT SYSTEM</center></td>
            </tr>
        </table>

    </div>
    <hr color="red" size="10">

     <table  align="center" cellpadding="6"  width="80%" bgcolor="maroon">
    <tr>
```

```
            <td>
               <a href="ViewCustomer.jsp">CUSTOMER</a>
            </td>
            <td>
               <a href="ViewEmployee.jsp">EMPLOYEE</a>
            </td>
            <td>
               <a href="ViewPharmacy.jsp">PRODUCTS</a>
            </td>
            <td>
               <a href="ViewSales.jsp">SALES</a>
            </td>

            <td>
               <a href="adminlog.jsp"> Admin </a>
            </td>
</tr>
   </table>
   <center>
   <img src="images/img3.jpg">
   </center>
   </form>
   </body>
</html>
```

6.2LOGIN PAGE CODE

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head>
   <title></title>
   <link href="css/templatemo_style.css" rel="stylesheet" type="text/css" />
   <style type="text/css">
      .style1
      {
         width: 30%;
         border: 4px solid #FD9E9E;
         background-color: #FFFFFF;
      }
      #Text1
      {
         width: 189px;
      }
      #Password1
      {
         width: 184px;
      }
   </style>
</head>
<body>
```

```
<table cellpadding="10" border="5" width="100%"  bgcolor="purple"
    style="font-size:30px; color:yellow; font-weight: bolder;" align="center">
<tr>
<td>
```

6.3 LOGIN FOR THE USER

```
</td>
</tr>
</table>
<hr size="10" color="green">


   <form name="f1" action="loginaction.jsp">
   <table cellpadding="10" class="style1">
      <tr>
         <td>
            UserName</td>
         <td>
            <input id="Text1" type="text" name="t1" /></td>
      </tr>
      <tr>
         <td>
            Password</td>
         <td>
            <input id="Password1" type="password"  name="t2"/></td>
      </tr>
      <tr>
<td colspan="2" style="text-align: center">
```

```
            <input id="Submit1" type="submit" value="submit" /><input
    id="Reset1"
                type="reset" value="reset" /><br />

            </td>
        </tr>
    </table>
        <img src="images/img1.jpg" width="50%">

        <a href="Register.jsp"> SIGN UP HERE</a>

    </form>
</body>
</html>
```

6.4 ADMIN LOGIN PAGE

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Courier</title>

</head>
<body background="images/login.jpg">

<form method="post" action="adminc.jsp">
<table cellpadding="4" cellspacing="2"  width="100%" border="10px"
    style="font-color:yellow; background-color:blue"; font-size:25px; blue;border-
    style:ridge;border-color:yellow;">
      <tr>
    <td align="center"><h1>Admin Controll</h1></td>
  </tr>
<br>
<hr size="15" color="black"/>
<table cellpadding="4" cellspacing="2"  width="50%" border="10px"
   align="center"   style="font-color:seagreen; background-color:pink"; font-
   size:25px; blue;border-style:ridge;border-color:#eeacfb;">
<tr>
            <td><center>UserName</center></td>
            <td><input type="text" name="username" /></td>
          </tr>
          <tr>
            <td><center>Password</center></td>
            <td><input type="password" name="password"  /></td>
          </tr>
<tr>
<td colspan="2" align="center">
<input type="submit" value="Submit" />
<input type="reset" value="Reset"/>
</td>
</tr>
</table>

<hr size="15" color="black"/>
</form>
</body>
</html>
```

6.5 VIEW SALES

```
page contentType="text/html" pageEncoding="UTF-8"%>
 < page import="java.sql.*" %>
< page import="java.io.*" %>
<!DOCTYPE html>
<html>
    <head>
<title></title>

</head>
<body bgcolor="#999966">



    <FORM action="sales.jsp" method="get">
    <div>
 <table cellpadding="15" cellspacing="5" bgcolor="purple" cellspacing="1"
width="100%" style="font-family:Times New Roman; font-size:35px; font-
weight:bolder; color:yellow;" >
    <tr><td align="center">
     Sale Details Submission
    </td></tr>
    </table>
    <hr color="red" size="1" />
     <img src="images/img11.jpg">
  <table cellpadding="15" align="right" width="40%" border="1"  bgcolor="gray"
style="font-family:Times New Roman; font-size:18px; font-weight:bolder;
color:maroon">
    <tr><td align="center">
     Sale_ID
    </td>
 <td align="center">
<input type="text" name="txt0"  size="40" style="height: 31px" />
 </td></tr>
   <tr><td align="center">
  Customr Name
 </td>

 <td align="center">
<input type="text" name="txt1"  size="40" style="height: 31px" />
 </td></tr>
        <tr><td align="center">
  productName
 </td>

 <td align="center">
<input type="text" name="txt2"  size="40" style="height: 31px" />
 </td></tr>
        <tr><td align="center">
 Price
 </td>

 <td align="center">
<input type="text" name="txt3"  size="40" style="height: 31px" />
 </td></tr>
       <tr>
       <td align="center" colspan="2">
       <input type="submit" value="Submit">
       <input type="reset" value="Reset">
       </table>
<%
```

```
 String txt0 = request.getParameter("txt0");
String txt1 = request.getParameter("txt1");
String txt2 = request.getParameter("txt2");
String txt3 = request.getParameter("txt3");

/* Create string of connection url within specified
format with machine name,
 port number and database name. Here machine name id
 localhost and database name is student. */
 String connectionURL = "jdbc:mysql://localhost:3306/pharmacy";
      // declare a connection by using Connection interface
 Connection connection = null;
    // declare object of Statement interface that uses for

  PreparedStatement pstatement = null;
     // Load JBBC driver "com.mysql.jdbc.Driver"
  Class.forName("com.mysql.jdbc.Driver").newInstance();
      int updateQuery = 0;

       // check if the text box is empty
       if(txt1!=null && txt2!=null && txt3!=null){
                        // check if the text box having only blank spaces
           if(txt1!="" && txt2!="" && txt3!="") {
                      try {
          /* Create a connection by using getConnection()
          method that takes parameters of string type
          connection url, user name and password to connect
                to database. */
          connection = DriverManager.getConnection
          (connectionURL, "root", "root");
                     // sql query to insert values in the secified table.
```

```
              String queryString = "INSERT INTO
sales(id,customer_name,product_name,price) VALUES (?,?, ?, ?)";
                         /* createStatement() is used for create statement
              object that is used for
                    sending sql statements to the specified database. */
              pstatement = connection.prepareStatement(queryString);
               pstatement.setString(1, txt0);
              pstatement.setString(2, txt1);
                                pstatement.setString(3, txt2);
                                pstatement.setString(4, txt3);

                     updateQuery = pstatement.executeUpdate();
                            if (updateQuery != 0) { %>
                     <br>
                     <TABLE style="background-color: #E3E4FA;"
                   WIDTH="30%" border="1">
                       <tr><th>Data submitted successfully <a href='adminhome.jsp'>
Home</a></th></tr>
                        </table>
              <%
              }
            }
            catch (Exception ex) {
            out.println("Unable to connect to batabase.");

               }
            finally {
                // close all the connections.
                pstatement.close();
                connection.close();
            }
            }
          }
%>
  </FORM>

    </div>

    </form>
</body>
</html>
```

7.TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the
Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

7.1 TYPES OF TESTING

7.1.1UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

7.1.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components

7.1.3 FUNCTIONAL TEST

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input	: identified classes of valid input must be accepted.
Invalid Input	: identified classes of invalid input must be rejected.
Output	: identified classes of application outputs must be exercised.
Systems/Procedures	: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined

.

7.1.4 SYSTEM TEST

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

7.1.5WHITE BOX TESTING

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

7.1.6 BLACK BOX TESTING

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

7.2 UNIT TESTING

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.
Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

- **Test objectives**
  - ➢ All field entries must work properly.
  - ➢ Pages must be activated from the identified link.
  - ➢ The entry screen, messages and responses must not be delayed.

- **Features to be tested**

1. No dublicate entries should be allowed
2. All links should take the user to the user the correct page
3. No duplicate entries should be allowed

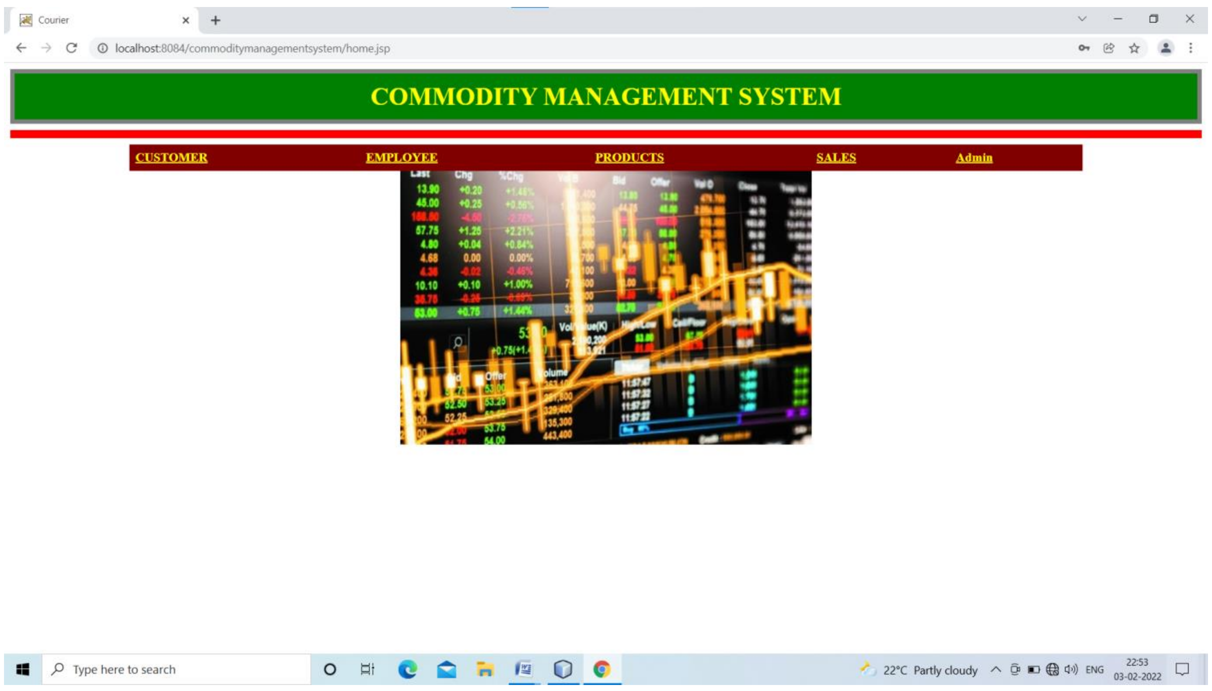
- **Integration Testing**

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

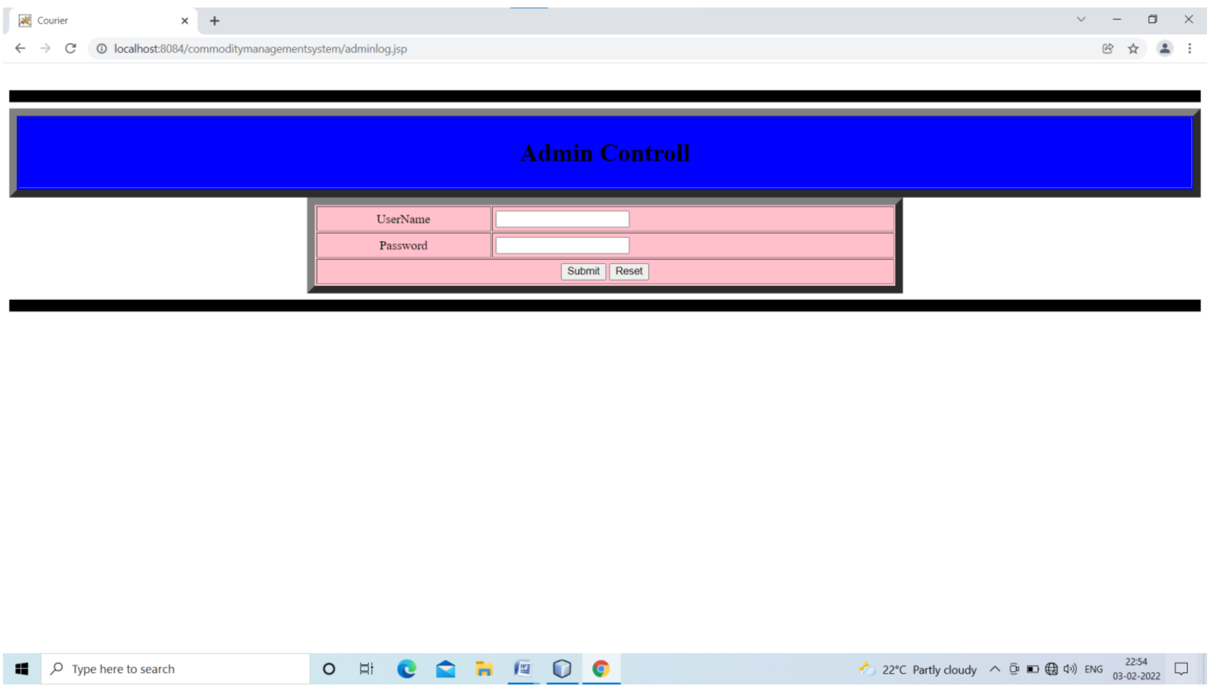
The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

8.OUTPUT SCREEN SHORTS

- **HOMEPAGE**



- **ADMIN PAGE**

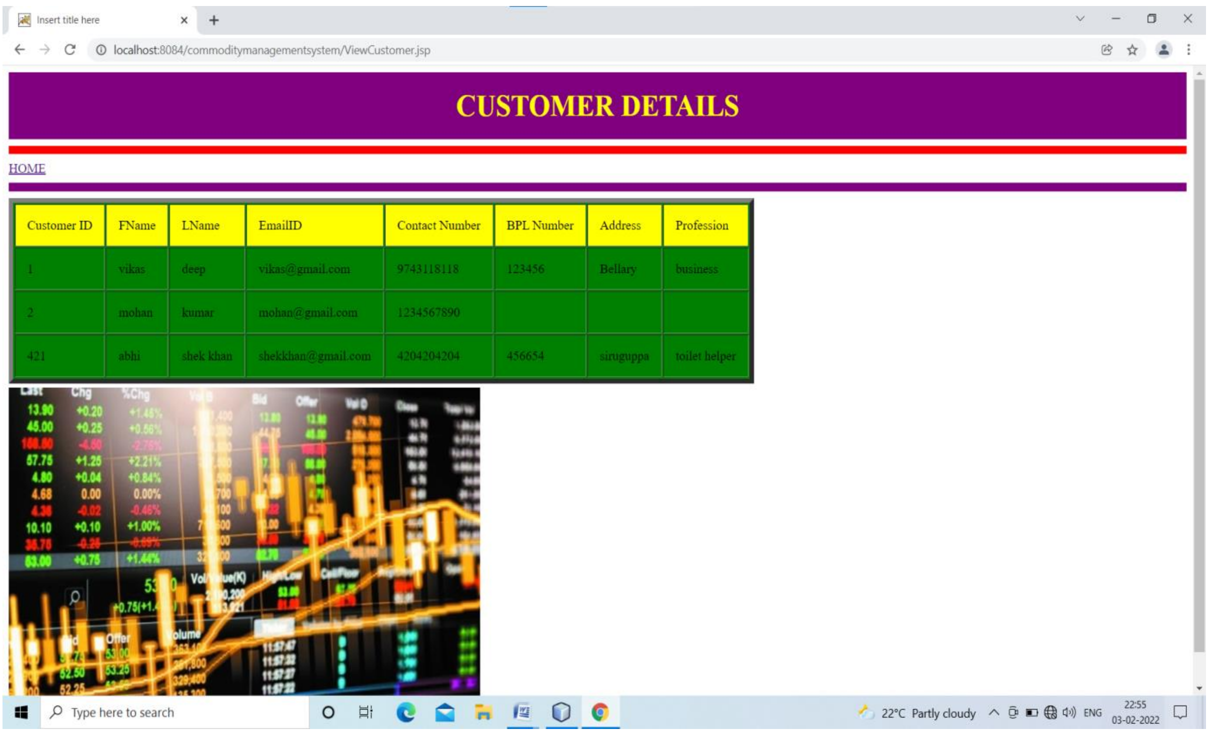


Admin Controll

UserName	
Password	

Submit Reset

- **Customer details**

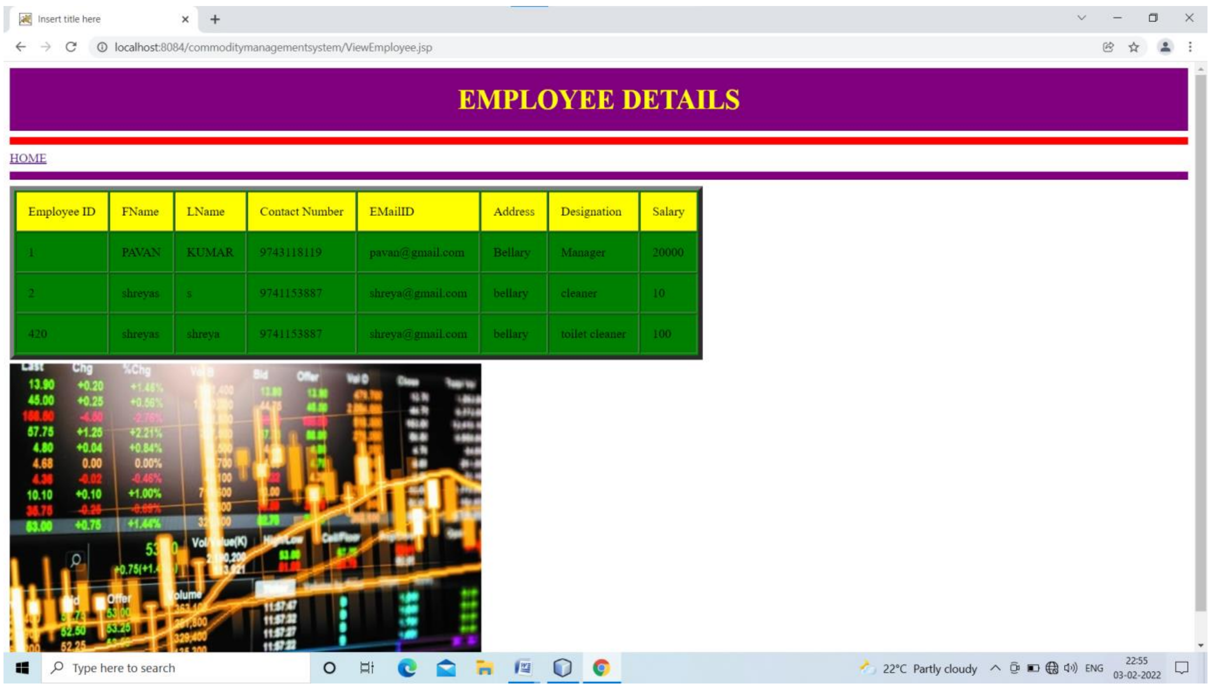


CUSTOMER DETAILS

HOME

Customer ID	FName	LName	EmailID	Contact Number	BPL Number	Address	Profession
1	vikas	deep	vikas@gmail.com	9743118118	123456	Bellary	business
2	mohan	kumar	mohan@gmail.com	1234567890			
421	abhi	shek khan	shekkhan@gmail.com	4204204204	456654	siruguppa	toilet helper

- **Employee details**

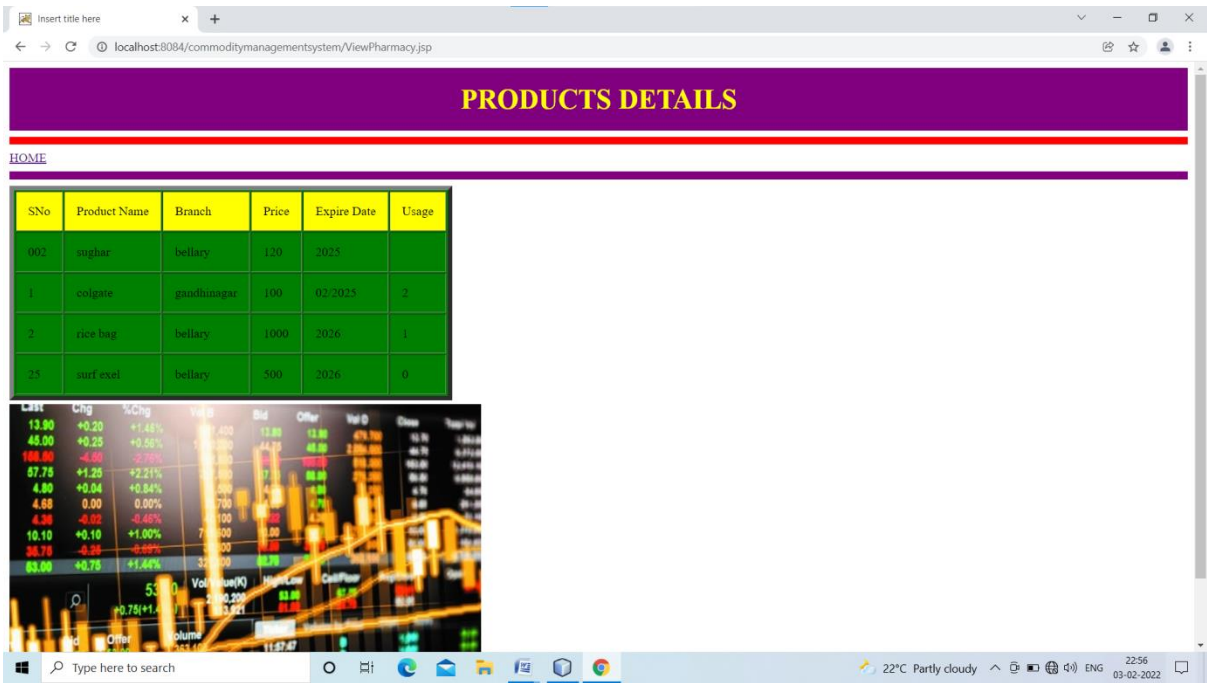


EMPLOYEE DETAILS

HOME

Employee ID	FName	LName	Contact Number	EMailID	Address	Designation	Salary
1	PAVAN	KUMAR	9743118119	pavan@gmail.com	Bellary	Manager	20000
2	shreyas	s	9741153887	shreya@gmail.com	bellary	cleaner	10
420	shreyas	shreya	9741153887	shreya@gmail.com	bellary	toilet cleaner	100

- **Product details**



PRODUCTS DETAILS

HOME

SNo	Product Name	Branch	Price	Expire Date	Usage
002	sughar	bellary	120	2025	
1	colgate	gandhinagar	100	02/2025	2
2	rice bag	bellary	1000	2026	1
25	surf exel	bellary	500	2026	0

9.CONCLUSION

Exact commodity management is critical to maintaining a profitable product business. Following stock consistently can help keep away from stock blunders and different issues. A solid commodity management programming could diminish the danger of overselling, saves expenses, keep away from stock-outs just as overabundance stocks, and builds the benefit. It likewise advances great terms with sellers and providers just as clients. This thus prompts greater efficiency, great quality, sensible costs, and a decent benefit which prompts a mutually beneficial arrangement for both the customers and vendors.

10.FUTURE ENHANCEMENTS

Accurate stock administration is basic to keeping a beneficial item business. Following stock reliably can help avoid stock goofs and various issues. A strong stock administration programming could decrease the risk of overselling, saves costs, avoid stock-outs similarly as excess stocks, and fabricates the advantage,
Security can be enhanced.