Chapter1

**INTRODUCTION**

An Inventory management system or a warehouse management system (WMS) is software and processes that allow organizations to control and administer warehouse operations from the time goods or materials enter a warehouse until they move out. Operations in a warehouse include [inventory management](https://searcherp.techtarget.com/definition/inventory-management), picking processes and auditing. For example, a WMS can provide visibility into an organization's inventory at any time and location, whether in a facility or in transit. It can also [manage supply chain](https://searcherp.techtarget.com/definition/supply-chain-management-SCM) operations from the manufacturer or wholesaler to the warehouse, then to a retailer or [distribution centre](https://whatis.techtarget.com/definition/distribution-center). Although a WMS is complex and expensive to implement and run, organizations gain benefits that can justify the complexity and costs. Implementing a WMS can help an organization reduce labour costs, improve inventory accuracy, improve flexibility and responsiveness, decrease errors in picking and shipping goods, and improve customer service. Modern warehouse management systems operate with real-time data, allowing the organization to manage the most current information on activities like orders, shipments, receipts and any movement of goods. An inventory management system is the combination of technology (hardware and software) and processes and procedures that oversee the monitoring and maintenance of stocked products, whether those products are company assets, raw materials and supplies, or finished products ready to be sent to vendors or end consumers. A complete inventory management system consists of:

Maintaining a central database

Associated information about the products

Process and policies for labelling, documentation and reporting

Product Functions of Inventory Management System would consist of:

*Administrator*

In this system, administrator is the merchant who can add, modify and delete the products and items. Administrator could control the status of the items. He/she has every right to modify the already entered or newly entered products and brands. He/she can view the system every day and could take a necessary action based on the situations of the warehouse. Administrator is logged in using admin username and password. Administrator can generate invoice.

*Customer*

In this system, customer is the user who can buy the required products from the administrator and could retrieve the invoice from the administrator using the customer contact. The system would calculate the total price, including the discount, GST and other required fields and could be shown to the user. When purchase is done then the invoice is sent to customer.

*Current System Architecture*

“Guckenheimer” (an on-site inventory management system for a restaurant) follows a system where the basic resources list needs to be manually calculated at the end of a certain time period by the staff. They must accordingly check the inventory levels for determining if they are below the threshold level then orders are processed to the vendors. This sort of system not only leaves a lot of room for human error, but is also incredibly time consuming. The lack of a centralized database also creates an issue when it comes to keeping track of inventory levels as well as past trends in ingredient requirements. The system also relies on human intuition and guesswork to place the correct orders for the following week, which will not be as precise as an algorithm designed for this purpose.

This has shown a huge number of bugs before a decade of time and has a loss of about 5000$. In our search, we have found that this has happened due to database entry constraints and the previous entries were repeated. To avoid and eradicate it, our design would be robust and database is tightly viewed.

Chapter 2

**RELATED WORK**

## 2.1 INVENTORY MANAGEMENT SYSTEM

Inventory management systems track goods through the entire supply chain or the portion of it a business operates in. That covers everything from production to retail, warehousing to shipping, and all the movements of stock and parts between.

Practically, it means a business can see all the small moving parts of its operations, allowing it to make better decisions and investments. Different inventory managers focus on different parts of the supply chain—though small businesses are usually more interested in the ordering and sales end of the chain.

Because of their wide variation in scope, inventory management systems also vary widely in cost. Software Advice’s [Front Runners report for inventory management](http://www.softwareadvice.com/inventory-management/#top-products) gives you an idea of what that price/features mix looks like.

## 2.2 SOFTWARE FEATURES

Let’s start at the beginning and work our way toward the point of sale.

Again, the software you choose to manage your inventory could offer any, or all, of these features, depending on your needs and budget. Each industry has its unique requirements, but all inventory management systems will likely include:

* Barcoding
* Reporting tools
* Inventory forecasting
* Inventory alerts
* Accounting tools or tie-ins

### 2.3 CREATING INVENTORY MANAGEMENT SYSTEM

The manufacturing process revolves around work orders and bills of materials. Inventory management at this point in the process is all about tying work orders and bills to your existing materials. That means you’ll find:

* Materials tracking
* Inventory levels for parts and finish products
* Automatic reordering
* Integrations with ERP or maintenance software

By tracking the work you’re doing and the materials required to produce that work, you, well, make money. It’s close to impossible to run a profitable manufacturing business without some sort of inventory management system in place.

The only real question is how much of the process you want to track. Do you need to automate reordering, tie into a warehouse management system, or track uptime for your machinery? Those are all possible extensions within the manufacturing inventory management system.

### 2.4 MANAGING AN INVENTORY

Managing a warehouse is a different beast and requires different system features. Warehouses are most interested in where an item is. Of course, having the right number on hand is important, but the real value in inventory management in a warehouse setting is being able to find something quickly.

You’ll commonly see features including:

* Advanced barcode systems supporting QR and other standards
* Multiple location support
* Shelf and bin tracking systems
* Order picking support

From there, you can order more inventory and pick your existing inventory for customer orders.

One of the major benefits of software-based warehouse management is that it frees you from the confines of your human mind.

Here’s what I mean by that:

You think the letter paper and legal paper should be next to each other in the warehouse because—well, obviously they should be. You want paper, so you head to aisle Z4. Voila! Paper. What you don’t realize is that letter paper is often ordered at the same time as staples. Letter paper is almost never ordered at the same time as legal paper.

In reality, you should be stocking the legal paper over in Z6, and staples and letter paper together in Z4.

This makes it harder to conceptualize the warehouse, but it makes picking stock easier and faster. Warehouse inventory management software can help you uncover those types of issues.

You can also set restocking levels so you’re never caught without a popular item. Actually, it’s more likely that a good system will keep you from being caught without an unpopular item—the one you don’t notice is out or low until a huge order comes in, and it’s too late.

### 2.5 RETAIL INVENTORY MANAGEMENT SOFTWARE

In its final iteration, inventory management systems help retailers keep their businesses running smoothly and profitably. From [our friends at Software Advice](http://www.softwareadvice.com/retail/inventory-management-software-comparison/#definition):

A good inventory system can keep shelves as full as they need to be. At its extreme, this is called “just-in-time” inventory management, which prioritizes minimizing cash tied up in product over maximizing inventory buffer.

There’s a lot more going on in this scenario, so you’ll see:

* Order picking support
* Automatic reorder
* Inventory forecasting tools
* Barcoding support

Of course, you can set all your reorder points based on your business’s goals. Maybe you want to have five times the buffer of your competitors—that’s up to you.

## 2.6 FINDING THE RIGHT INVENTORY MANAGEMENT SYSTEM

Once you have a list of your required features in place, you can find the system that’s going to work right for you. Capterra, conveniently, has an [inventory management software directory](https://www.capterra.com/inventory-management-software/) with over 275 options. You can filter by user rating, feature set, and number of users.

You can also check out [our list of free options](https://blog.capterra.com/the-top-5-free-inventory-software-systems/), which might be a perfect option for companies just starting out or those with some technical knowledge floating around. Find whatever option works for you, put it in place, and start working smarter.

Chapter 3

**SYSTEM ANALYSIS**

System analysis is conducted for the purpose of studying a system or its parts in order to identify its objectives. It is a problem-solving technique that improves the system and ensures that all the components of the system work efficiently to accomplish their purpose.

**3.1 SOFTWARE REQUIREMENT SPECIFICATION**

A **software requirements specification** (SRS) is a detailed description of a software system to be developed with its functional and non-functional requirements. The SRS is developed based the agreement between customer and contractors. It may include the use cases of how user is going to interact with software system. The software requirement specification document consistent of all necessary requirements required for project development. To develop the software system, we should have clear understanding of Software system. To achieve this, we need to continuous communication with customers to gather all requirements.

A good SRS defines the how Software System will interact with all internal modules, hardware, communication with other programs and human user interactions with wide range of real-life scenarios. Using the Software requirements specification (SRS) document on QA lead, managers create test plan. It is very important that testers must be cleared with every detail specified in this document in order to avoid faults in test cases and its expected results.

**3.2 PURPOSE**

The objective of inventory management system is to manage the warehouse of a merchant and to arrange the items such that it would easy for a merchant to serve the items to the customers who have an order. It will also facilitate keeping all the records of the products, categories, brands, price, stock, date, status (active or inactive) and action (delete or edit). The merchant can create an invoice and send it to customer. When a customer orders a product then it would be easy to find the item in the warehouse and could easy pack the item to delivery. When a multiple product is to be delivered to same customer all the price calculations are done along with the discounts, service charges, delivery charges and GST by the system and then invoice is generated. When a merchant gets a new product from the companies then he could add the new category, parent category, brand, stock and price to the database. When product data is kept active then the customer can order the product else customer could not order a product.

**3.3 PRODUCT SCOPE**

Without Inventory Management System it would be so difficult for the merchants to maintain the warehouse and the products of the warehouse. Inventory Management System will store all the information about the products and items of the warehouse including the product information such as categories, brands, price, stock, dates, status and actions.

Login Module: It will help for authentication of the accounts. Accounts which have valid login id and password can only be logged into the respective accounts.

Search Module: It will help for the searching of the products all around the warehouse and could give back the result of the rack number of the product.

Sign Up Module: It will help the new clients to create an account to use the system

Admin Module: This module is used to have all features and warehouse control to administrator. Thus he/she could manage the products and items present.

**3.4 OBJECTIVES**

Objectives of inventory management system is to make administrator to control the outcomes of a company to a warehouse and deliver the products and items to customers. The customers and administrator would have some common actions but an administrator would have more authority over the system we design. Our main objective is to maintain a database of a company products and manage them till the customer purchases the products form a warehouse. A [**warehouse management system**](https://www.skuvault.com/blog/skuvault-warehouse-management-software) **(WMS)** is a type of software used to manage operations in a warehouse including[inventory managemen](https://www.skuvault.com/blog/warehouse-management-inventory-management)**t**, picking processes, and auditing. A WMS is an instrumental part of the supply chain. The supply chain’s primary goal is to track and control products. Such as, tracking movement of products, and controlling inventory actions. **A warehouse management system streamlines and facilitates processes and a product’s journey throughout the warehouse**; for example, such systems will direct and optimize picking by utilizing real-time inventory data to determine the proper warehouse location to retrieve the product from. More precisely, **warehouse management involves the receipt, storage, and movement of goods.**Normally, finished goods are accounted for, but [kits may also be managed by a sophisticated WMS.](https://www.skuvault.com/warehouse-management-system/features/catalog)

**3.5 EXISTING SYSTEM**

The Guckenheimer Warehouse, is located at 125 First Avenue in the Downtown neighbourhood of Pittsburgh, Pennsylvania. The building served as the warehouse for the Guckenheimer barrels, which were distilled in Freeport, Pennsylvania, until prohibition caused the company to shut down. This warehouse uses an inventory management system known as Guckenheimer manager which has a huge database. This system was developed by Common Gateway Interface and Sybase database system.

Limitations:

* This warehouse system is used by the company and has a huge response time for huge number of inputs.
* The inventory management has unable to store the items in correct order, a decade before and has a huge variation in delivering the items to customers and has got a huge loss of 5000$.

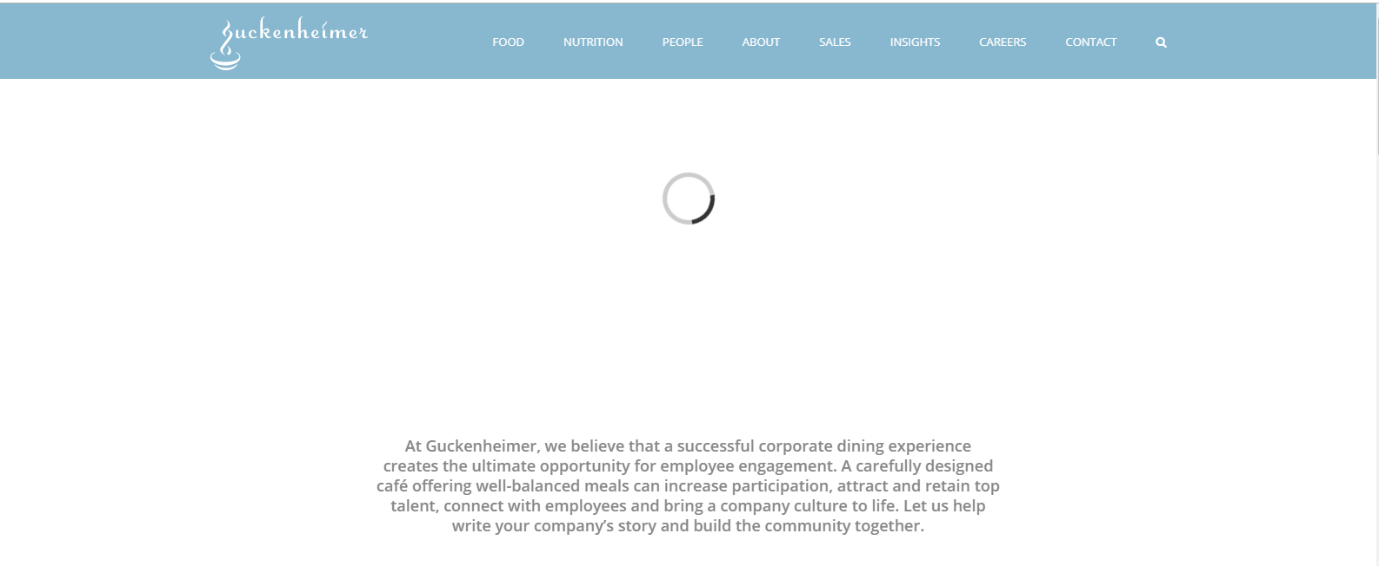


Fig. 3.5 Image of Guckenheimer System taking huge response time

**3.6 PROPOSED SYSTEM**

The main limitation of the existing system is removed in our system. Our system would have a less response time for any number of inputs. For the system to be active all the time we were using the HTML 5.0 which has shown a very less response time in the Silicon Valley research centre. For the database system we are using an advanced version of MySQL database version 8.0 which has shown a memory acquittance in dynamic manner. This would be helpful in managing the tables and when the rows are modified by the many different users at a time then would take help of synchronization to modify the rows. Thus, there would be no collision of data in the system.

**3.7 ADVANTAGES**

* Inventory Management system we have proposed would take the less response time compared to the CGI with Sybase System.
* The database management would be secured and manipulating of data would be better and lightweight.

**3.8 PROBLEM STATEMENT**

The main Problem for an inventory management system is to maintain the accounts and costs produced. There are many systems for accounting but there are no many systems which would consider both products in an inventory and costs of products in detail. Thus, it was a problem for many warehouses that there are products and the cost would go down in some extent in sales of products manually in stores. Thus, we have to create an inventory management system which would deal with inventory as well as shopping for customers.

**3.9 FUNCTIONAL REQUIREMENTS**

**3.9.1 PRODUCT CATEGORIZATION**

In our system products are well classified and could be accessed by the admin very easily.

**3.9.2 STOCK INQUIRIES**

The stock of the products can be notified and stored in databases and can easily modified based on the orders and status of the goods present in the warehouse. Thus, stocks of the products can be found using the queries that would run behind the system to inquire the stock.

**3.9.3 ADMINISTRATOR MANAGED INVENTORY**

The inventory can be completely managed by the administrator and can be easily accessed. The customers would have a limited access to the system and can easily place the order.

**3.9.4 INVOICE GENERATION**

For the system when order is confirmed by the administrator then the invoice is easily generated and is shared to the customer.

**3.10 NON-FUNCTIONAL REQUIREMENTS**

**3.10.1 SAFETY REQUIREMENTS**

The products and items of the merchants should be kept safe and while delivering the products to the customers the items should be safely delivered and also correct products must be delivered.

**3.10.2 SECURITY REQUIREMENTS**

The data used by the system must be secured without infecting with the virus, trojans, malware or worms. When a database is crashed, it would be so difficult to retrieve the data back. So, to avoid this situation a data backup should be kept.

**3.10.3 USER INTERFACES**

For a user to use the software easily we need to provide interface so easily that no computer knowledge should be required to operate the software.

**3.10.4 HARDWARE INTERFACES**

Processor: Intel Pentium 4 or above

RAM: 1GB or above

HDD: 12GB or above

Monitor: 12inch or above

Mouse: Scroll or Optical Mouse

Keyboard: Standard QWERTY keyboard

Printer: \*Optional

**3.10.5 SOFTWARE INTERFACES**

Connections between the front and backend.

In this system we use Apache Tomcat for front-end.

We use MySQL for Back-end.

**3.10.6 COMMUNICATION INTERFACES**

Client (customer) will use HTTP/HTTPS protocol.

Client (admin) will use HTTP/HTTPS protocol.

Chapter 4

**SYSTEM DESIGN**

Object oriented design is concerned with developing an object-oriented model of a software system to implement the identified requirements. It is the process of defining the components, interfaces, objects, classes, attributes and operations that will satisfy the requirements. Typically start with the candidate objects defined during the analysis, but add much more rigor to their definition. Then add or change objects as needed to refine a solution. The designer’s goal is how the outputs to be produced and in what format samples of the output are also presented. The processing phases are handled through the program construction and testing. Finally details related to justification of the system and an estimate of the impact of the candidate system on the user and the organization are documented and evaluated by management as a step towards implementation. The importance of software design can be stated in a single word “Quality”. Design provides us with representations of software that can be assessed for quality. Design is the only way that can be able to accurately translate a customer’s requirements into a finished software product or system without design risk, building an unstable system, that might fail it if small changes are made or may be difficult to test, or one who’s quality can’t be tested. So, it is essential phase in the development of a software product.

Object-oriented design can yield the following benefits:

**Maintainability** through simplified mapping to the problem domain, which provides for less analysis effort, less complexity in system design, easier verification by the user.

**Reusability** of the design artefacts, which saves time and cost.

**Productivity** gains through direct mapping of features of Object-Oriented Programming Languages.

**4.1 UML DESIGN**

UML stands for Unified Modelling Language. Taking SRS document of analysis as input to the design phase drawn UML diagrams. The UML is only language so is just one part of the software development method. The UML is process independent, although optimally it should be used in a process that should be driven, architecture-centric, iterative, and incremental. The UML is language for visualizing, specifying, constructing, documenting the articles in a software-intensive system. A modelling language is a language whose vocabulary and rules focus on the conceptual and physical representations of the system. A modelling language such as the UML is thus a standard language for software blueprints. The UML is a graphical language, which consists of all interesting systems. There are also different structures that can transcend what can be represented in a programming language.

There are different diagrams in UML

**4.1.1. USE CASE DIAGRAM**

Use Case during requirement elicitation and analysis to represent the functionality of the system. Use Case describes a function provided by the system that yields a visible result for an actor. The identification of actors and use cases result in the definition of the boundary of the system i.e., differentiating the tasks accomplished by the system and the tasks accomplished by its environment. The actors are outside the boundary of the system, whereas the Use cases are inside the boundary of the system. Use Cases describe the behaviour of the system as seen from the actor’s point of view. It describes the function provided by the system as a set of events that yield a visible result for the actor.

**USE CASE DIAGRAM FOR USER**

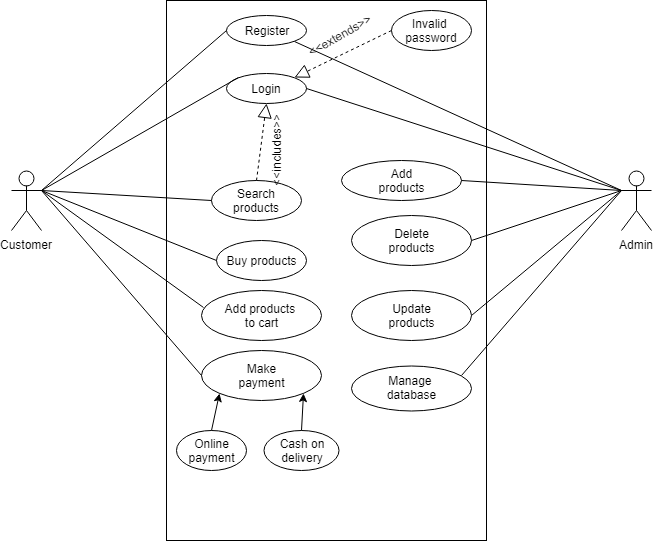


Fig. 4.1.1. Use Case diagram

**4.1.2 CLASS DIAGRAM**

Class diagrams model class structure and contents using design elements such as classes, packages and objects. Class diagram describe three different perspectives when designing a system- conceptual, specification and implementation. Classes are composed of three things: name, attributes, and operations. Class diagrams also display relationships such as containment, inheritance, associations etc. The association relationship is most common relationship in a class diagram. The association shows the relationship between instances of classes.

**CLASS DIAGRAM**

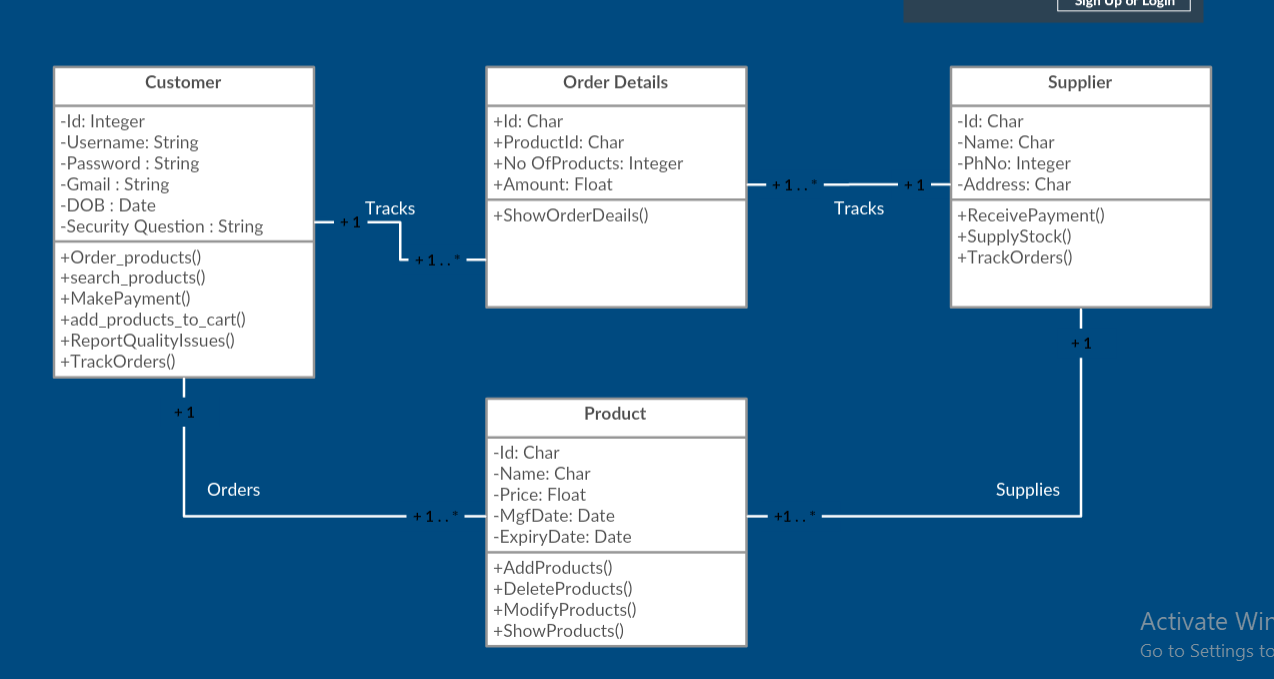


Fig. 4.1.2. Class Diagram

**4.1.3 INTERACTION DIAGRAMS**

**SEQUENCE DIAGRAMS**

Sequence diagram displays the time sequence of the objects participating in the interaction. This consists of the vertical dimension (time) and horizontal dimension (different objects).

**OBJECT:** Object can be viewed as an entity at a particular point in time with a specific value and as a holder of identity that has different values over time.

**ACTOR:** An Actor represents a coherent set of roles that users of a system play when interacting with the use cases of the system.

**MESSAGE:** A message is sending of a signal from one sender object to other receiver object(s).

**SEQUENCE DIAGRAM FOR USER**

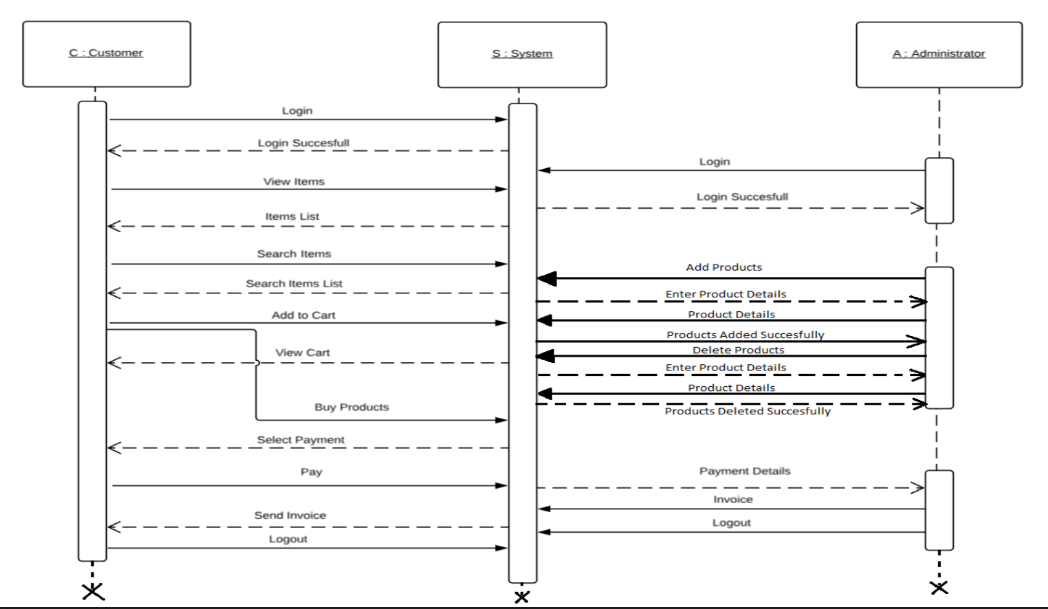


Fig 4.1.3.1. Sequence diagram for User

**COLLABORATION DIAGRAMS**

Collaboration diagram displays an interaction organized around the objects and their links to one another. Numbers are used to show the sequence of messages.

**COLLABORATION DIAGRAM FOR USER**

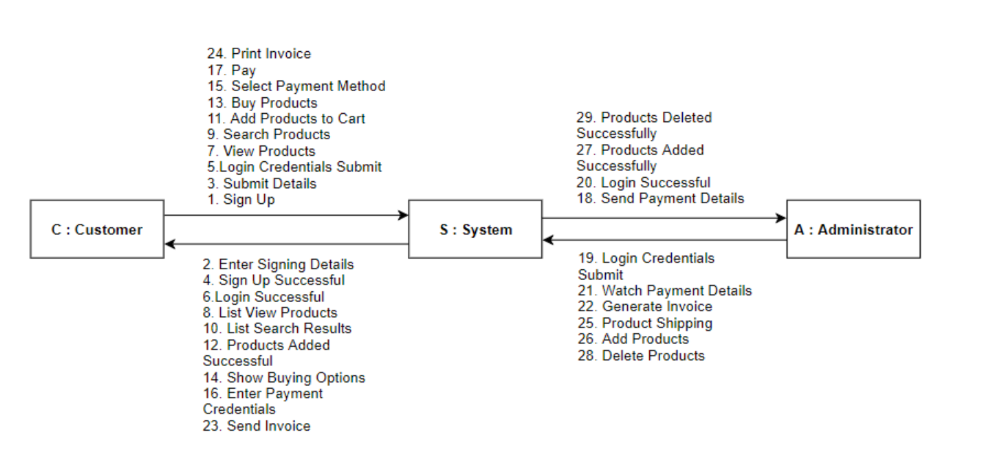


Fig.4.1.3.2. Collaboration diagram for User

**4.1.4 ACTIVITY DIAGRAM**

An activity diagram shows the flow from activity to activity. An activity is a going non-atomic execution with in a state machine. An activity results in some action, results in a change of state or return of a value.

Activity Diagram commonly contains

1. Activity states and action states.
2. Transitions.
3. Objects, it may contain nodes and constraints.

**ACTIVITY STATES AND ACTION STATES:** An executable atomic computation is called action state, which cannot be decomposed. Activity state is non atomic, decomposable and takes some duration to execute.

**TRANSITION:** It is the path from one state to the next state, represented as simple directed line.

**BRANCHING:** When an alternate path exists, branching arises which is represented by open diamond. It has an incoming transition, two or more outgoing transitions.

**FORKING AND JOINING:** The synchronization bar when split one flow into two or more flows is called fork. When two or more flows are combined at synchronization bar, the bar is called join.

**SWIM LANES:** Group work flow is called swim lanes. All groups are portioned by vertical solid lines. Each swim lane specifies locus of activities and has a unique name. Each swim lane is implemented by one or more classes. Transition may occur between objects across swim lanes.

**Activity diagram**

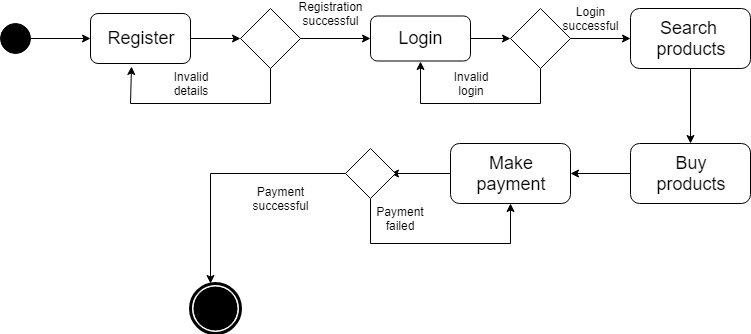


Fig. 4.6 Activity diagram

Chapter 5

**IMPLEMENTATION**

Implementation is the stage where the theoretical design is turned into a working system. The most crucial stage in achieving a new successful system and in giving confidence on the system for the users that will work efficiently and effectively. The system will be implemented only after thorough testing and if it is found to work according to the specification.

**5.1. OVERVIEW OF SOFTWARES USED**

Product Inventory System is developed using[HTML, PHP](http://www.php.net/), CSS and bootstrap. Talking about the project, it contains an admin side and a user where a user can maintain transactions easily. Additionally, the admin plays an important role in the management of this system. In this project, the user has to perform all the main functions from the admin side.

**HTML**:

Hypertext Mark-up Language (HTML) is the standard [mark-up language](https://en.wikipedia.org/wiki/Markup_language) for creating [web pages](https://en.wikipedia.org/wiki/Web_page) and [web applications](https://en.wikipedia.org/wiki/Web_application). With [Cascading Style Sheets](https://en.wikipedia.org/wiki/Cascading_Style_Sheets) (CSS) and [JavaScript](https://en.wikipedia.org/wiki/JavaScript), it forms a triad of [cornerstone](https://en.wikipedia.org/wiki/Cornerstone) technologies for the [World Wide Web](https://en.wikipedia.org/wiki/World_Wide_Web). [Web browsers](https://en.wikipedia.org/wiki/Web_browser) receive HTML documents from a [web server](https://en.wikipedia.org/wiki/Web_server) or from local storage and [render](https://en.wikipedia.org/wiki/Browser_engine) the documents into multimedia web pages. HTML describes the structure of a web page [semantically](https://en.wikipedia.org/wiki/Semantic_Web) and originally included cues for the appearance of the document. [HTML elements](https://en.wikipedia.org/wiki/HTML_element) are the building blocks of HTML pages. With HTML constructs, [images](https://en.wikipedia.org/wiki/HTML_element#Images_and_objects) and other objects such as [interactive forms](https://en.wikipedia.org/wiki/Fieldset) may be embedded into the rendered page. HTML provides a means to create [structured documents](https://en.wikipedia.org/wiki/Structured_document) by denoting structural [semantics](https://en.wikipedia.org/wiki/Semantics) for text such as headings, paragraphs, lists, [links](https://en.wikipedia.org/wiki/Hyperlink), quotes and other items. HTML elements are delineated by tags, written using [angle brackets](https://en.wikipedia.org/wiki/Bracket#Angle_brackets). Tags such as <img /> and <input /> directly introduce content into the page. Other tags such as <p> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page. HTML can embed programs written in a [scripting language](https://en.wikipedia.org/wiki/Scripting_language) such as [JavaScript](https://en.wikipedia.org/wiki/JavaScript), which affects the behaviour and content of web pages. Inclusion of CSS defines the look and layout of content. The [World Wide Web Consortium](https://en.wikipedia.org/wiki/World_Wide_Web_Consortium) (W3C), maintainer of both the HTML and the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997.

**CSS**:

Cascading Style Sheets (CSS) is a style sheet language used to describe the presentation semantics (that is, the look and formatting) of a document written in a mark-up language. Its most common application is to style web pages written in HTML and XHTML, but the language can be applied to any kind of XML document, including SVG and XUL. CSS is designed primarily to enable the separation of document content (written in HTML or a similar mark-up language) from document presentation, including elements such as the colours, fonts, and layout. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple pages to share formatting, and reduce complexity and repetition in the structural content (such as by allowing for table less web design).

**PHP**:

PHP started out as a small open source project that evolved as more and more people found out how useful it was. Rasmus Lerdorf unleashed the first version of PHP way back in 1994. PHP is a recursive acronym for "PHP: Hypertext Pre-processor". PHP is a server-side scripting language that is embedded in HTML. It is used to manage dynamic content, databases, session tracking, even build entire e-commerce sites. It is integrated with a number of popular databases, including MySQL, PostgreSQL, Oracle, Sybase, Informix, and Microsoft SQL Server. PHP is pleasingly zippy in its execution, especially when compiled as an Apache module on the Unix side. The MySQL server, once started, executes even very complex queries with huge result sets in record-setting time. PHP supports a large number of major protocols such as POP3, IMAP, and LDAP. PHP4 added support for Java and distributed object architectures (COM and CORBA), making n-tier development a possibility for the first time. PHP is forgiving: PHP language tries to be as forgiving as possible. PHP Syntax is C-Like.

**BOOTSTRAP**:

Bootstrap is a [free and open source](https://whatis.techtarget.com/definition/Free-and-open-source-software-FOSS-or-free-libre-open-source-software-FLOSS) [front end](https://whatis.techtarget.com/definition/front-end) development framework for the creation of websites and [web apps](https://searchsoftwarequality.techtarget.com/definition/Web-application-Web-app). The Bootstrap framework is built on [HTML](https://www.theserverside.com/definition/HTML-Hypertext-Markup-Language), [CSS](https://www.theserverside.com/definition/cascading-style-sheet-CSS), and JavaScript ([JS](https://www.theserverside.com/definition/JavaScript)) to facilitate the development of [responsive](https://whatis.techtarget.com/definition/responsive-design), [mobile-first](https://searchmobilecomputing.techtarget.com/definition/mobile-first) sites and apps.

Responsive design makes it possible for a web page or app to detect the visitor’s screen size and orientation and automatically adapt the display accordingly; the mobile first approach assumes that [smartphones](https://searchmobilecomputing.techtarget.com/definition/smartphone), [tablets](https://searchmobilecomputing.techtarget.com/definition/tablet-PC) and task-specific [mobile apps](https://whatis.techtarget.com/definition/mobile-app) are employees' primary tools for getting work done and addresses the requirements of those technologies in design.

Bootstrap includes user interface components, layouts and JS tools along with the framework for implementation. The software is available precompiled or as [source code](https://searchmicroservices.techtarget.com/definition/source-code).

Chapter 6

**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.

Software testing is an important element of the software quality assurance and

represents the ultimate review of specification, design and coding. The increasing feasibility

of software as a system and the cost associated with the software failures are motivated forces for well-planned through testing.

**6.1 TESTING OBJECTIVES**

These are several rules that can save as testing objectives they are:

* Testing is a process of executing program with the intent of finding an error.
* A good test case is one that has a high probability of finding an undiscovered error.

**6.2 TYPES OF TESTING**

In order to make sure that the system does not have errors, the different levels of testing

strategies that are applied at differing phases of software development are:

**6.3 UNIT TESTING**

Unit Testing is done on individual modules as they are completed and become executable. It

is confined only to the designer’s requirements. Unit testing is different from and should be preceded

by other techniques, including:

Inform debugging

Code inspections

Each module can be tested using the following two Strategies:

**6.4 BLACK BOX TESTING**

In this strategy some test cases are generated as input conditions that fully execute all

functional requirements for the program. This testing has been used to find error in the

following categories:

Incorrect or missing functions.

Interface errors.

Errors in data structure or external database access.

Performance error.

Initialization and termination errors.

In this testing only the output is checked for correctness.

The logical flow of the data is not checked.

**6.5 WHITE BOX TESTING**

In this the test cases are generated on the logic of each module by drawing flow

graphs of that module and logical decisions are tested on all the cases. It has been uses to

generate the test cases in the following cases:

Guarantee that all independent paths have been executed.

Execute all loops at their boundaries and within their operational bounds.

Execute internal data structures to ensure their validity.

**6.6 INTEGRATING TESTING**

Integration testing ensures that software and subsystems work together a whole. It

tests the interface of all the modules to make sure that the modules behave properly when

integrated together. It is typically performed by developers, especially at the lower, module-

to-module level. Testers become involved at the higher levels.

**6.7 SYSTEM TESTING**

Involves in - house testing of the entire system before delivery to the user. The aim is to

satisfy the user the system meets all requirements of the client&#39;s specifications. It is conducted by the

testing organization if a company has one. Test data may range from hand generated to production.

Requires test scheduling to plan and organize:

Inclusion of changes/fixes.

Test data to use

One common approach is graduated testing: as system testing progresses and

(hopefully) fewer and fewer defects are found, the code is frozen for testing for increasingly

longer time periods.

**6.8 ACCEPTANCE TESTING**

It is a pre-delivery testing in which entire system is tested at client’s site on real world data to find errors.

**6.8.1 USER ACCEPTANCE TEST (UAT)**

Beta testing: Acceptance testing in the customer environment.

Requirements traceability:

Match requirements to test cases.

Every requirement has to be cleared by at least one test case.

Display in a matrix of requirements vs. test cases.

**6.9 TEST CASES**

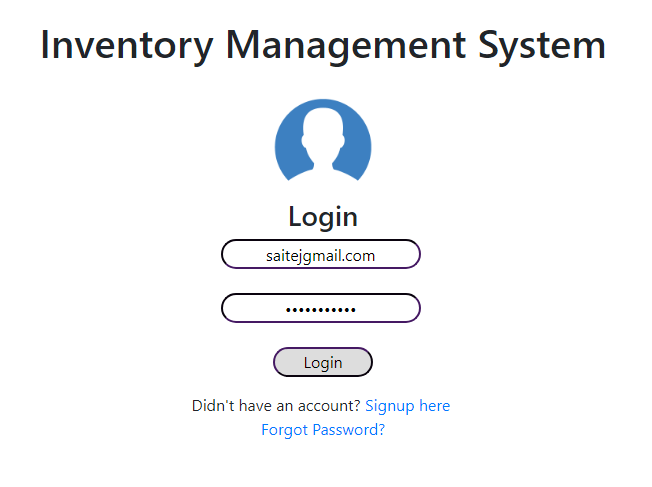
In general, a test case is a set of test data and test programs and their expected results.

A test case in software engineering normally consists of a unique identifier, requirement references from a design specification, preconditions, events, a series of steps (also known as actions) to follow, input, output and it validates one or more system requirements and generates a pass or fail.

**6.5.1 TESTCASE 1**

**OBJECTIVE: INCORRECT LOGIN DETAILS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Item Number | Test Condition | Input Specification | Output Specification | Pass / Fail |
| 1 | Checking Whether Username and Password are correct or not | If username and password are not correct | Display Account Doesn’t exist | Pass |



**TESTCASE OUTPUT:**

Fig. 6.5.1.2 Inputs of Test Case for Invalid Login Details

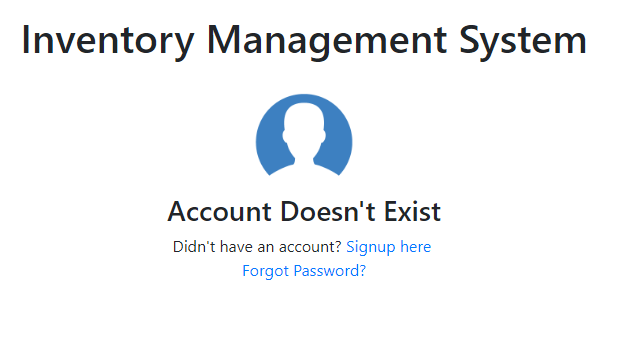
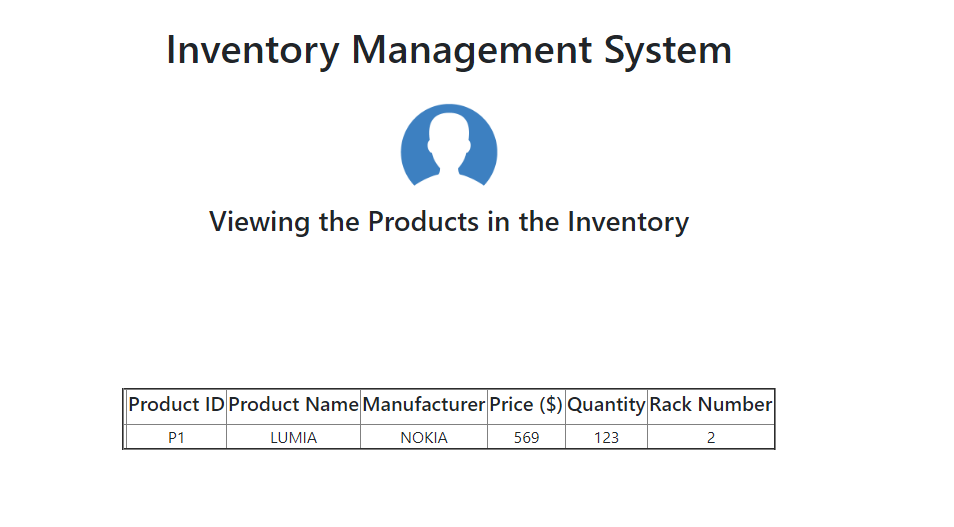


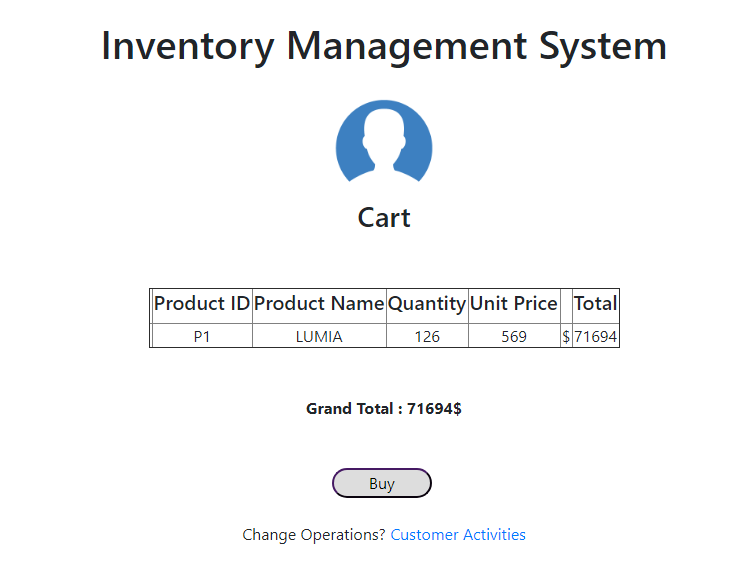
Fig. 6.5.1.1 Result of Test Case for Invalid Login Details

**6.5.2 TESTCASE 2**

**OBJECTIVE: BUYING PRODUCTS BEYOND STOCK**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Item Number | Test Condition | Input Specification | Output Specification | Pass / Fail |
| 2 | Checking Whether Products can be bought beyond the stock | If Products quantity in stock are more than in inventory | Display Demand is greater than Supply | Pass |

**TESTCASE OUTPUT:**

 Fig. 6.5.2.1 Showing Products in Inventory

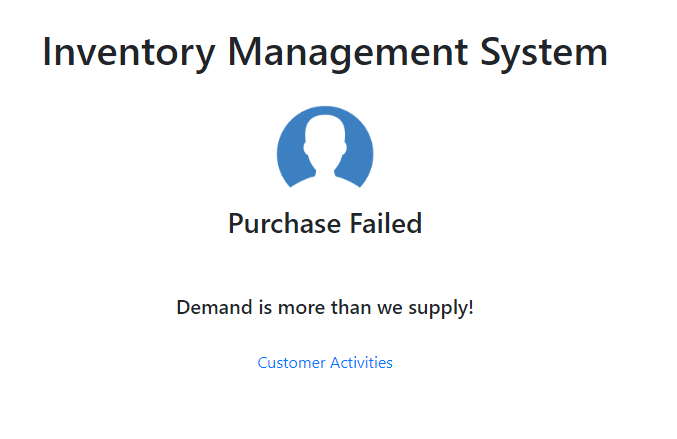
 Fig. 6.5.2.2 Showing Added Products in Cart

Fig. 6.5.2.3. When bought we get “Demand is more than we supply”

**6.5.3 TESTCASE 3**

**OBJECTIVE: SEARCHING PRODUCTS NOT IN STOCK**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Item Number | Test Condition | Input Specification | Output Specification | Pass / Fail |
| 3 | Searching the products are not in stock | Products entered to search bar are not present in stock | Display No Products Found! | Pass |

**TESTCASE OUTPUT:**

Fig. 6.5.3.1 Viewing Inventory Products

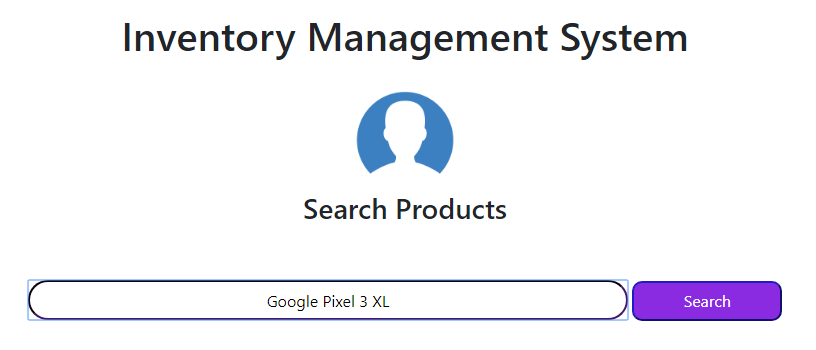


Fig. 6.5.3.2 Searching for the Products not in inventory



Fig. 6.5.3.3 Showing result as No Products Found!

**6.5.3 TESTCASE 4**

**OBJECTIVE: SEARCHING PRODUCTS NOT IN STOCK**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Item Number | Test Condition | Input Specification | Output Specification | Pass / Fail |
| 4 | Searching the products are not in stock | Products entered to search bar are not present in stock | Display No Products Found! | Pass |

Chapter 7

**RESULT ANALYSIS**

When we have compared with the response and time complexity taken by the already been implemented existing system, we have seen that our system has performed extremely well. The time complexity is very less and has almost reduced the (approx. as measured with clock in seconds at 1Mbps internet speed) 40% of response time. The data manipulation tests have also been passed by the great number of inputs and data storage has also been managed by our system.

This has been achieved by our system practically and the main implementation to reach the products to users directly from inventory also been implemented which can be easily understood by the user. Thus, we could say easily that our systems have reached to solutions to the problem statement we have observed.

Server connection is also successful since we have tested in simulation of Apache Tomcat Version 5.0.

When a client needs to be connected then he could easily use the http protocol to connect with server and then use the webpage to display the results.

Chapter 8

**FUTURE ENHANCEMENT AND CONCLUSIONS**

Our project presented an enhancement in inventory management system from present system. System Analysis has shown that our system is secure and more comfortable than the present system which use Sybase Database and Common Gateway System. Performance has also been increased since we have used the latest versions of web technology to build as well as to test the system.

**8.1 Future Enhancement**

In terms of responsiveness, we can increase the responsiveness with the next version of technology which would increment in near future.

The images could be easily added to the products we display at the client’s home page by embedding the images with the MySQL Database.

Login Protocol can be upgraded to use the Python since the enhancement would be happened by integrating the web technologies implementation with Python.

Java Script can be added to the web page for the dynamicity of web pages.

The functionalities like adding the products, removing the products can be implemented using the prompts.

Chapter 9

**REFERENCES**

* + Research paper on Inventory management system by Punam Khobragade\*, Roshni Selokar\*, Rina Maraskolhe\* and Prof.Manjusha Talmale – International Research Journal of Engineering and Technology (IRJET)

**APPENDIX**

<DOCTYPE html>

<html>

<head>

<meta charset="utf8" description="Home page of Project"/>

<title>Inventory Management System</title>

<style>

body

{

color:black;

font:sans-serif;

background: white;

}

input[type="text"],input[type="password"],input[type="email"],input[type="date"]

{

text-align:center;

border-radius:20px;

height:30px;

width:200px;

border-color:#451963;

background:none;

transition:250ms;

}

input[type="submit"]

{

text-align:center;

border-radius:20px;

height:30px;

width:100px;

border-color:#451963;

transition:250ms;

}

input[type="text"]:focus,input[type="password"]:focus

{

width:250px;

height:35px;

}

input[type="submit"]:hover

{

width:110px;

border-color:#1D1DC2;

color:white;

background-color:blueviolet;

}

</style>

<link rel="stylesheet" href="css/bootstrap.css"></link>

</head>

<body>

<center>

<br>

<br>

<br>

<h1>Inventory Management System</h1>

<br>

<img src="user.png" style="height:100px;width:100px;">

<h3>Login</h3>

<form action="login.php" method="POST">

<input type="text" placeholder="Username" name="username">

<br><br>

<input type="password" placeholder="Password" name="password">

<br>

<br>

<input type="submit" value="Login">

</form>

<h7>Didn't have an account? <a href="http://localhost/Project/Signup.html">Signup here</a></h7>

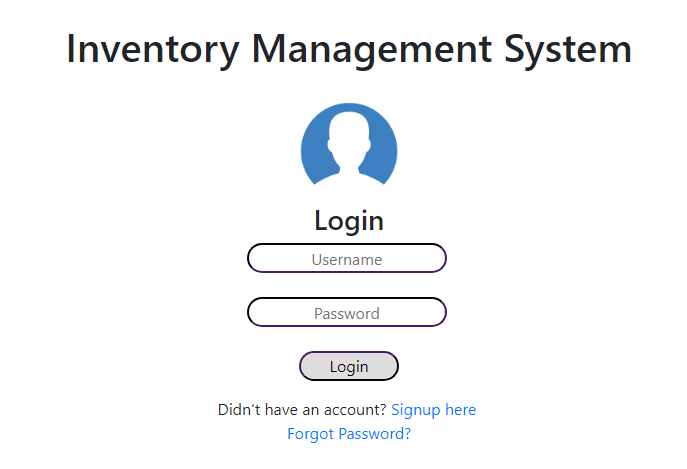
<br>

<h7><a href="http://localhost/Project/forget.html">Forgot Password?</a></h7>

</center>

</body>

</html>



<DOCTYPE html>

<html>

<head>

<meta charset="utf8" description="Home page of Project"/>

<title>Inventory Management System</title>

<style>

body

{

color:black;

font:sans-serif;

background: white;

}

input[type="text"],input[type="password"],input[type="email"],input[type="date"]

{

text-align:center;

border-radius:20px;

height:30px;

width:200px;

border-color:#451963;

background:none;

transition:250ms;

}

input[type="submit"]

{

text-align:center;

border-radius:20px;

height:30px;

width:100px;

border-color:#451963;

transition:250ms;

}

input[type="text"]:focus,input[type="password"]:focus

{

width:250px;

height:35px;

}

input[type="submit"]:hover

{

width:110px;

border-color:#1D1DC2;

color:white;

background-color:blueviolet;

}

</style>

<link rel="stylesheet" href="css/bootstrap.css"></link>

</head>

<body>

<center>

<br>

<br>

<br>

<h1>Inventory Management System</h1>

<br>

<img src="user.png" style="height:100px;width:100px;">

<h3>Sign Up</h3>

<br>

<form action="signup.php" method="POST">

<input type="text" placeholder="Full Name" name="Name" required>

<br><br>

<input type="email" placeholder="Email Address/Username" name="username" required>

<br><br>

<input type="password" placeholder="Password" name="password" required>

<br><br>

<input type="password" placeholder="Confirm Password" name="cpassword" required>

<br><br>

<input type="Date" placeholder="Date of Birth" name="dob" required>

<br><br>

<label>Security Question: What is your <b>birth place?</b> </label>

<br>

<input type="text" placeholder="Security Answer" name="security" required>

<br><br>

<input type="submit" value="Signup">

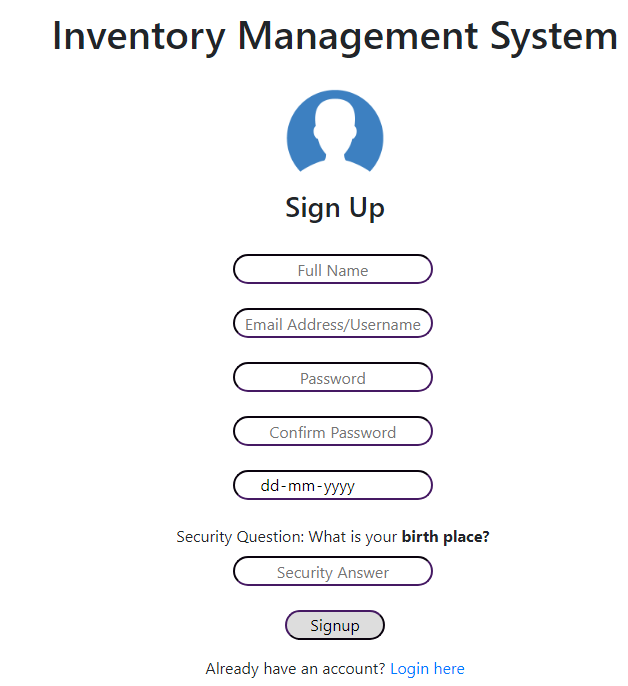
</form>

<h7>Already have an account? <a href="http://localhost/Project/home.html"> Login here </a></h7>

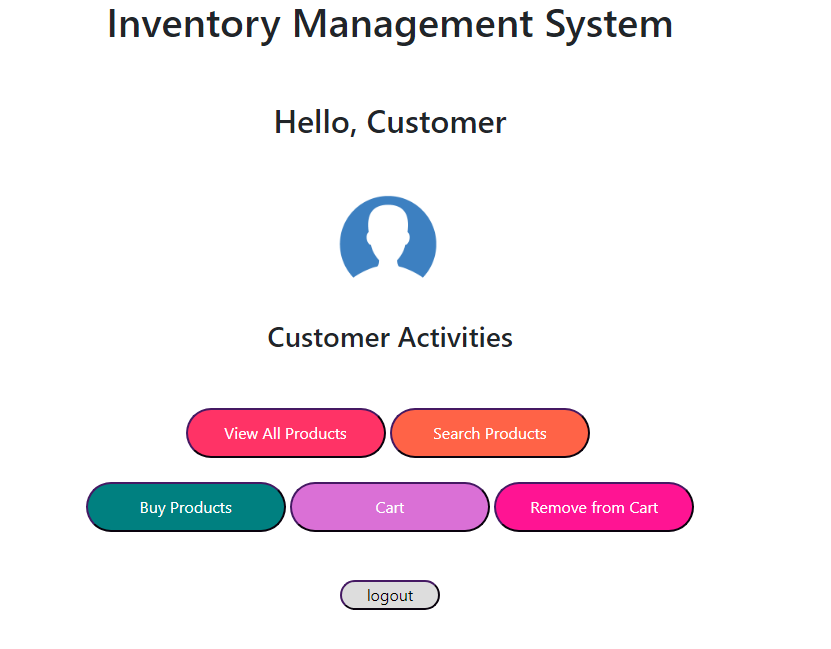
</center>

</body>

</html>



Customer Home Page:



Administrator Home Page:

