1 INTRODUCTION

1.1 PROJECT OVERVIEW

College Store Management System (CSMS) is web application for automate College Store. It focuses quick sale of stationery items for students according to their needs. CSMS provide a powerful search facility for searching items that are available in our College Store. Students can order the items if they required. All users (College Students and Staffs) can order and payment done through Credit/ Debit card. This system has a unique feature that enable the users to buy product without registration, for identifying each student/staff through admission number/staff id. It is very user friendly and promote cashless transaction.

1.2 PROJECT SPECIFICATION

CSMS is web application for automate college store. The CSMS management system helps you to maintain all records of college store. This CSMS will handle all type of transactions concerning purchase, storage and issue of stock items. CSMS include three modules, they are admin module, store keeper module and user module. They are:

1. Admin Module

The admin is the overall controller of the system. Admin can manage store keepers, view items, view tenders, and view sale details.

2. Store keeper Module

Store keeper can manage items, item delivery, view user requested items and view sale details.

3. User Module

User can search and buy items from CSMS with online payment. User can also request new item which does not exist in store.

2 SYSTEM STUDY

2. SYSTEM ANALYSIS

2.1 INTRODUCTION

System analysis is a process of gathering and interpreting facts, diagnosing problems and the information to recommend improvements on the system. It is a problem-solving activity that requires intensive communication between the system users and system developers. System analysis or study is an important phase of any system development process. The system is studied to the minute's detail and analysed. The system analyst plays the role of the interrogator and dwells deep into the working of the present system. The system is viewed as a whole and the input to the system are identified. The outputs from the organizations are traced to the various processes. System analysis is concerned with becoming aware of the problem, identifying the relevant and decisional variables, analysing and synthesizing the various factors and determining an optimal or at least a satisfactory solution or program of action.

A detailed study of the process must be made by various techniques like interviews, questionnaires etc. The data collected by these sources must be scrutinized to arrive to a conclusion. The conclusion is an understanding of how the system functions. This system is called the existing system. Now the existing system is subjected to close study and problem areas are identified. The designer now functions as a problem solver and tries to sort out the difficulties that the enterprise faces. The solutions are given as proposals. The proposal is then weighed with the existing system analytically and the best one is selected. The proposal is presented to the user for an endorsement by the user. The proposal is reviewed on user request and suitable changes are made. This is loop that ends as soon as the user is satisfied with proposal.

Preliminary study is the process of gathering and interpreting facts, using the information for further studies on the system. Preliminary study is problem solving activity that requires intensive communication between the system users and system developers. It does various feasibility studies. In these studies, a rough figure of the system activities can be obtained, from which the decision about the strategies to be followed for effective system study and analysis can be taken.

2.2 EXISTING SYSTEM

Now the students have to go to the store and buy items through cash payment. So, this is very difficult for the users to buy items. Also, the students were not allowed to buy items during class hours. This creates very big queue in front of the store and many of the students will not get the items they needed. The existing system thus got lot of problems for the people as nowadays all are very busy and also all are using online sites for their all basic needs. In case of store keeper, who has difficult to manage items in store such as find stock of each item, sale details etc. And also, all are wish to get everything within their reach. In such case the existing system is obviously causes lot of disadvantages.

2.3 DRAWBACKS OF EXISTING SYSTEM

Drawbacks of existing system are,

- The process of existing system is done through paper work.
- The Store is getting rush.
- This system does not provide the option to order in advance.
- Time consuming.
- Difficult to create and maintain consistent data
- Difficulty to find item is available or not, price etc.
- We need to keep money with us.
- Difficult to find buyers for used items.

2.4 PROPOSED SYSTEM

The proposed system provides a powerful search option for items that are available in college store. Students and staff can buy the items through online if they required. It will be a simple platform for users to buy stationary items for their huge needs. For all users the payment can be done through online. The main features include:

- Powerful search option for customers
- User friendly
- Reduce the wastage of time
- Cashless transaction
- Workload of the storekeeper reduced
- No paper usage

2.5 ADVANTAGES OF PROPOSED SYSTEM

The system is very simple in design and to implement. The system requires very low system resources and the system will work in almost all configurations. It has got following features:

- It will have a centralized item details.
- Paper work is reduced.
- Powerful search option for customers
- User friendly
- Reduce the wastage of time
- Cashless transaction
- Workload of the storekeeper reduced

2.6 USER CHARACTERISTICS

There are three users in CSMS. They are Admin, Store keeper and user (Students/staffs).

ADMIN

The admin is the overall controller of the system. The admin can add Store employees and also, he can monitor and control or access the entire system functionalities.

STORE KEEPER

Store keeper can viewing orders and delivery of ordered items, manage items in store such as update stock, add new category, item type and item, viewing sale details and viewing item request from customers.

USERS (Students/staffs)

All students and staffs can search items from store, buy the items and request new items to store. Identifying each student/ staff through admission number/ Staff id. No cash payment is allowed. All payment can be done through online Credit/ Debit card payment methods.

3 REQUIREMENT ANALYSIS

3.1 FEASIBILITY STUDY

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spend on it. Feasibility study lets the developer foresee the future of the project and the usefulness. A feasibility study of a system proposal is according to its workability, which is the impact on the organization, ability to meet their user needs and effective use of resources. Thus, when a new application is proposed it normally goes through a feasibility study before it is approved for development.

The document provides the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as Technical, Economic and Operational feasibilities. The following are its features:

3.1.1 Economical Feasibility

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require.

The following are some of the important financial questions asked during preliminary investigation:

- The costs conduct a full system investigation.
- The cost of the hardware and software.
- The benefits in the form of reduced costs or fewer costly errors.

Since the system is developed as part of project work, there is no manual cost to spend for the proposed system. Also, all the resources are already available, it gives an indication of the system is economically possible for development.

3.1.2 Technical Feasibility

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs and procedures. Having identified an outline system, the investigation must go on to suggest the type of equipment, required method developing the system, of running the system once it has been designed.

Technical issues raised during the investigation are:

- Does the existing technology sufficient for the suggested one?
- Can the system expand if developed?

The project should be developed such that the necessary functions and performance are achieved within the constraints. The project requires High Resolution Scanning device and utilizes Cryptographic techniques. Through the technology may become obsolete after some period of time, due to the fact that newer version of same software supports older versions, the system may still be used. So, there are minimal constraints involved with this project. The system has been developed using PHP in front end and MySQL in server in back end, the project is technically feasible for development.

3.1.3 Behavioural Feasibility

This includes the following questions:

- Is there sufficient support for the users?
- Will the proposed system cause harm?

The project would be beneficial because it satisfies the objectives when developed and installed. All behavioural aspects are considered carefully and conclude that the project is behaviourally feasible. CSMS, GUI is simple so that users can easily use it. CSMS site is simple enough so that no training is needed.

3.2 SYSTEM SPECIFICATION

3.2.1 Hardware Specification

Processor	-	Pentium IV/AMD Dual core
RAM	-	2 GB
Hard disk	-	500 GB

3.2.2 Software Specification

Front End	-	PHP
Backend	-	MYSQL
Client on PC	-	Windows XP or above
Technologies used	-	JS, HTML5, AJAX, JQuery, PHP, CSS

3.3 SOFTWARE DESCRIPTION

3.3.1 PHP and Features

PHP is a server-side scripting language designed primarily for web development but also used as a general-purpose programming language. Originally created by Rasmus Lerdorf in 1994, the PHP reference implementation is now produced by The PHP Development Team. PHP originally stood for *Personal Home Page*, but it now stands for the recursive acronym *PHP: Hypertext Preprocessor*.

PHP code may be embedded into HTML code, or it can be used in combination with various web template systems, web content management systems and web frameworks. PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as a Common Gateway Interface (CGI) executable. The web server combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page. PHP code may also be executed with a command line interface (CLI) and can be used to implement standalone graphical applications.

The standard PHP interpreter, powered by the Zend Engine, is free software released under the PHP License. PHP has been widely ported and can be deployed on most web servers on almost every operating system and platform, free of charge.

The PHP language evolved without a written formal specification or standard until 2014, leaving the canonical PHP interpreter as a *de facto* standard. Since 2014 work has gone on to create a formal PHP specification.

HTML

HTML, which stands for Hyper Text Markup Language, is the predominant markup language for web pages. HTML is the basic building-blocks of web pages. A markup language is a set of markup tags, and HTML uses markup tags to describe web pages. The purpose of a web browser is to read HTML documents and compose them into visual or audible web pages. The browser does not display the HTML tags, but uses the tags to interpret the content of the page.

HTML elements form the building blocks of all websites. HTML allows images and objects to be embedded and can be used to create interactive forms. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. It can embed scripts in languages such as JavaScript which affect the behaviour of HTML web pages. Web browsers can also refer to Cascading Style Sheets (CSS) to define the appearance and layout of text and other material. The W3C, maintainer of both the HTML and the CSS standards, encourages the use of CSS over explicitly presentational HTML markup.

3.3.2 MySQL

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation. The MySQL Web site provides the latest information about MySQL software.

- **MySQL is a database management system.**

  A Database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

- **MySQL databases are relational.**

  A Relational Database stores data in separate tables rather than putting all the data in one big storeroom. The database structures are organized into physical files optimized for speed. The logical model, with objects such as databases, tables, views, rows, and columns, offers a flexible programming environment. You set up rules governing the relationships between different data fields, such as one-to-one, one-to-many, unique, required or optional, and "pointers" between different tables. The database enforces these rules, so that with a well-designed database, your application never sees inconsistent, duplicate, orphan, out-of-date, or missing data.

  The SQL part of "MySQL" stands for "Structured Query Language". SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax.

  SQL is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual, "SQL92" refers to the standard released in 1992, "SQL:1999" refers to the standard released in 1999, and "SQL:2003" refers to the current version of the standard.

  We use the phrase "the SQL standard" to mean the current version of the SQL Standard at any time.

- **MySQL software is Open Source.**

  Open Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL software uses the GPL (GNU General Public License), to define what you may and may not do with the software in different situations. If you feel uncomfortable with the GPL or need to embed MySQL code into a commercial application, you can buy a commercially licensed version from us. See the MySQL Licensing Overview for more information.

- **The MySQL Database Server is very fast, reliable, scalable, and easy to use.**

  If that is what you are looking for, you should give it a try. MySQL Server can run comfortably on a desktop or laptop, alongside your other applications, web servers, and so on, requiring little or no attention. If you dedicate an entire machine to MySQL, you can adjust the settings to take advantage of all the memory, CPU power, and I/O capacity available. MySQL can also scale up to clusters of machines, networked together.

  MySQL Server was originally developed to handle large databases much faster than existing solutions and has been successfully used in highly demanding production environments for several years. Although under constant development, MySQL Server today offers a rich and useful set of functions. Its connectivity, speed, and security make MySQL Server highly suited for accessing databases on the Internet.

- **MySQL Server works in client/server or embedded systems.**

  The MySQL Database Software is a client/server system that consists of a multi-threaded SQL server that supports different backends, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs).
  We also provide MySQL Server as an embedded multi-threaded library that you can link into your application to get a smaller, faster, easier-to-manage standalone product.

- **A large amount of contributed MySQL software is available.**

  MySQL Server has a practical set of features developed in close cooperation with our users. It is very likely that your favourite application or language supports the MySQL Database Server.

4 SYSTEM DESIGN

4.1 INTRODUCTION

Design is the first step into the development phase for any engineered product or system. Design is a creative process. A good design is the key to effective system. The term

"Design" is defined as "the process of applying various techniques and principles for the purpose of defining a process or a system in sufficient detail to permit its physical realization". It may be defined as a process of applying various techniques and principles for the purpose of defining a device, a process or a system in sufficient detail to permit its physical realization. Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm that is used. The system design develops the architectural detail required to build a system or product. As in the case of any systematic approach, this software too has undergone the best possible design phase fine tuning all efficiency, performance and accuracy levels. The design phase is a transition from a user oriented document to a document to the programmers or database personnel.

System design goes through two phases of development: Logical and Physical Design.

4.2 ARCHITECTURAL DESIGN

This section describes the components of the CSMS. The admin, student and staff can access the CSMS System through internet using their Laptop, Smart Phone, Tablet or Desktop Computer. The System's Application program processes the user's request and provides the required services by taking data from the system database.

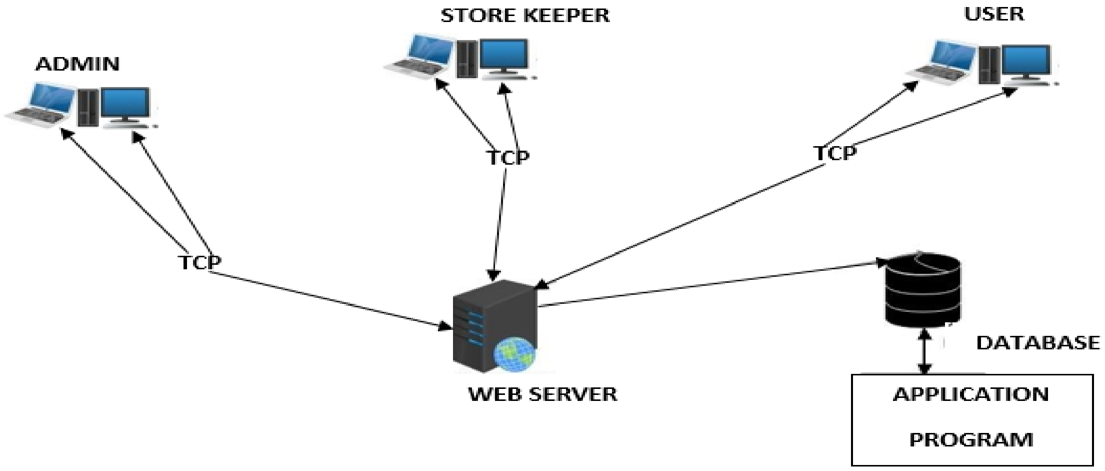


Figure 1: Architectural Design

4.3 MODULE DESIGN

ADMIN MODULE

The admin is the overall controller of the system. The admin can add Store employees and also, he can monitor and control or access the entire system functionalities. Functionalities are,

- Manage Store keepers.
- Viewing the sales.
- Viewing items.
- Viewing tenders.

USER MODULE

All students and staffs can search items from store, buy the items and request new items to store. Identifying each student/ staff through admission number/ Staff id. No cash payment is allowed. All payment can be done through online Credit/ Debit card payment methods. Functionalities are,

- Buy product.
- Search items.
- Request new items.

STORE KEEPER MODULE

Store keeper can viewing orders and delivery of ordered items, manage items in store such as update stock, add new category, item type and item, viewing sale details and viewing item request from customers. Functionalities are,

- Managing items.
- Viewing sale details.
- Viewing orders and delivery.
- Viewing item request.
- Change Password.

4.4 DATA FLOW DIAGRAM

Data Flow Diagram (DFD) is a diagram that describes the flow of data and the processes that change data throughout a system. It's a structured analysis and design tool that can be used for flowcharting in place of or in association with information. Oriented and process oriented system flowcharts. When analysts prepare the Data Flow Diagram, they specify the user needs at a level of detail that virtually determines the information flow into and out of the system and the required data resources. This network is constructed by using a set of symbols that do not imply physical implementations. The Data Flow Diagram reviews the current physical system, prepares input and output specification, specifies the implementation plan etc.

The purpose of the design is to create architecture for the evolving implementation and to establish the common tactical policies that must be used by desperate elements of the system. We begin the design process as soon as we have some reasonably completed model of the behaviour of the system. It is important to avoid premature designs, wherein develop designs before analysis reaches closer. It is important to avoid delayed designing where in the organization crashes while trying to complete an unachievable analysis model.

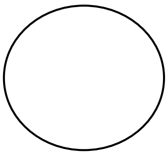
Throughout the project, the context flow diagrams, data flow diagrams and flow charts have been extensively used to achieve the successful design of the system. In our opinion, "efficient design of the data flow and context flow diagrams helps to design the system successfully without much major flaws within the scheduled time". This is the most complicated part in a project. In the designing process, our project took more than the activities in the software life cycle. If we design a system efficiently with all the future enhancements, the project will never become junk and it will be operational.

The data flow diagrams were first developed by Larry Constantine as way for expressing system requirements in graphical form. A data flow diagram also known as "bubble chart" has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. It functionally decomposes the requirement specification down to the lowest level. DFD depicts the information flow, the transformation flow and the transformations that are applied as data move from input to output. Data Flow Diagram is quite effective, especially when the required design is unclear and the user and analyst need a notational language for communication. It is used to model the system components such as the system process, the data used by the process, any external entities that interact with the system and information flows in the system.

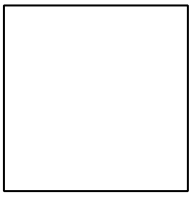
Four basic symbols are used to construct data flow diagrams. They are symbols that represent data source, data flows, and data transformations and data storage. The points at which data are transformed are represented by enclosed figures, usually circles, which are called nodes.

Main symbols used in the data flow diagram are:

1. Circle represents a process that transforms incoming data flows in to outgoing data flows.



2. A square defines a source and destination of system data.



3. Arrow identifies data in motion.

4. An open rectangle defines a data store, data at rest or temporary repository of data.

Steps to Construct Data Flow Diagrams:

Four steps are commonly used to construct a DFD:

- Process should be named and numbered for easy reference. Each name should be representative of the process.
- The destination of flow is from top to bottom and from left to right.
- When a process is exploded in to lower level details they are numbered.
- The names of data stores, sources and destinations are written in capital letters.
- **Rules for constructing a Data Flow Diagram**

- Arrows should not cross each other.
- Squares, circles and files must bear names.
- Decomposed data flow squares and circles can have same names.
- Draw all data flow around the outside of the diagram.

Data Flow Diagrams of CSMS

Level -0 data flow diagram of CSMS

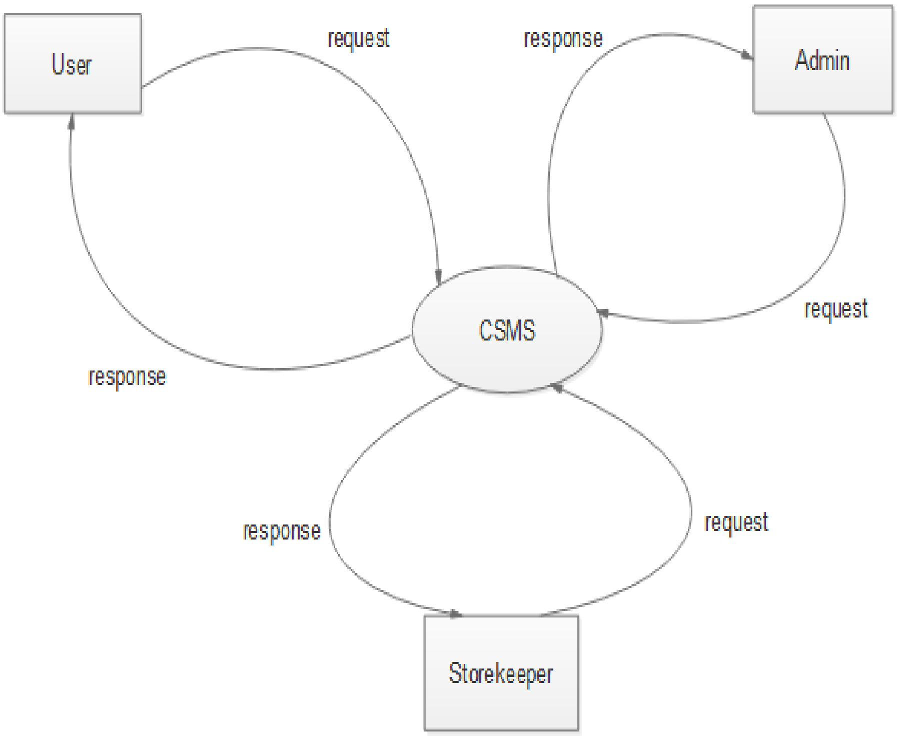


Figure 2: Level -0 data flow diagram of CSMS

Level -1 data flow diagram of Storekeeper module

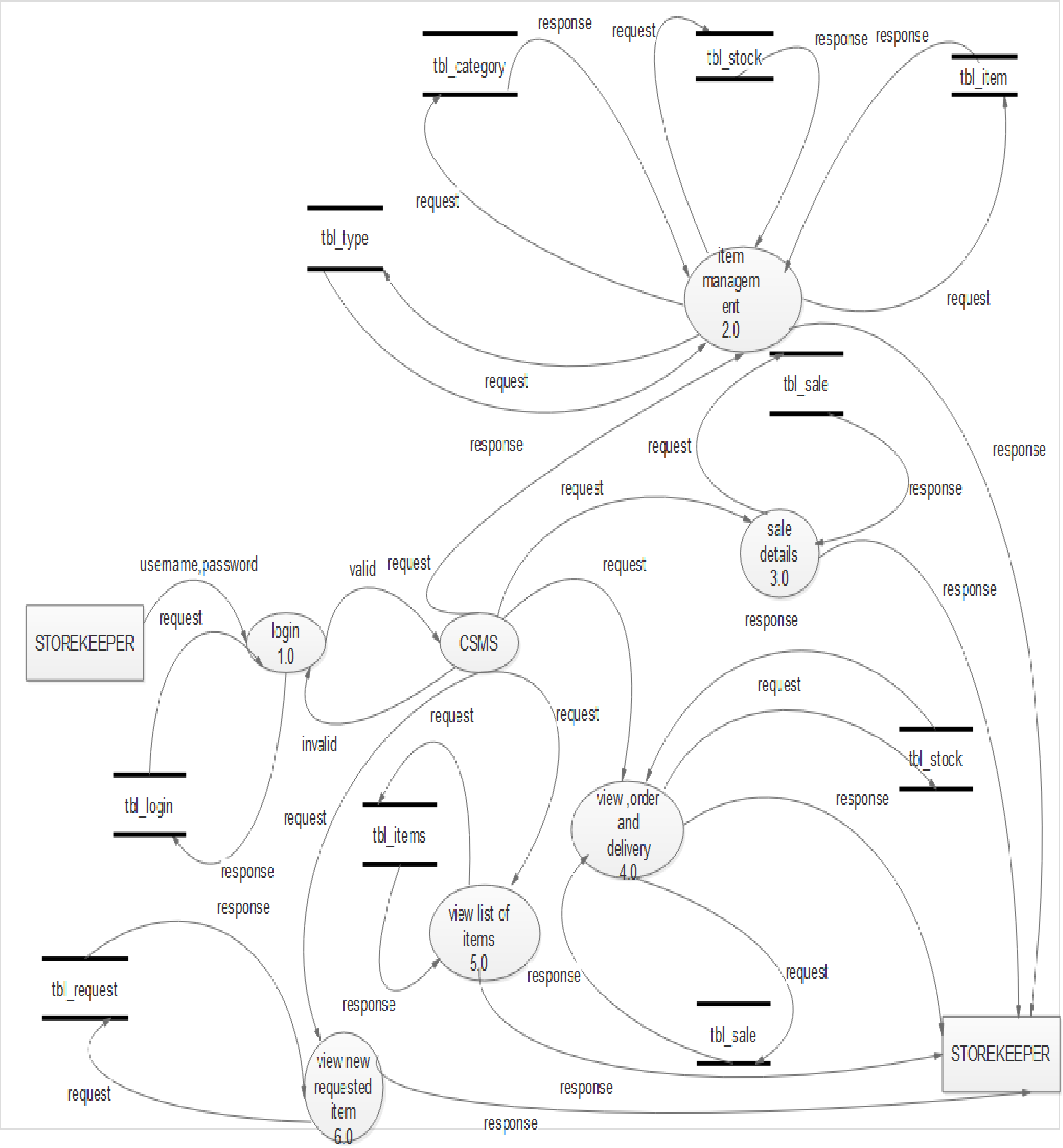


Figure 3: Level -1 data flow diagram of Storekeeper module

Level -1 data flow diagram of Admin module

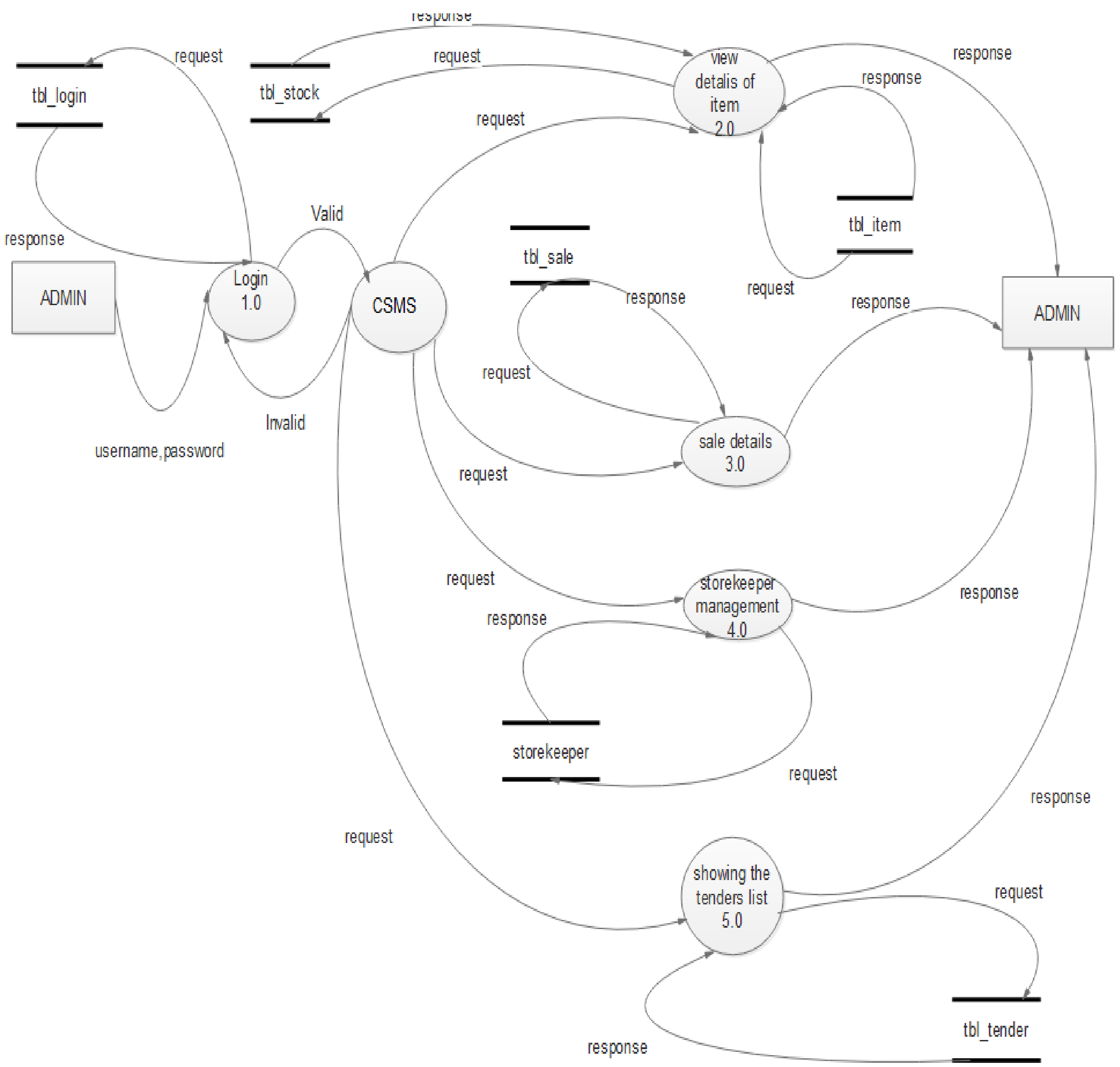


Figure 4: Level -1 data flow diagram of Admin module

Level -1 data flow diagram of User module

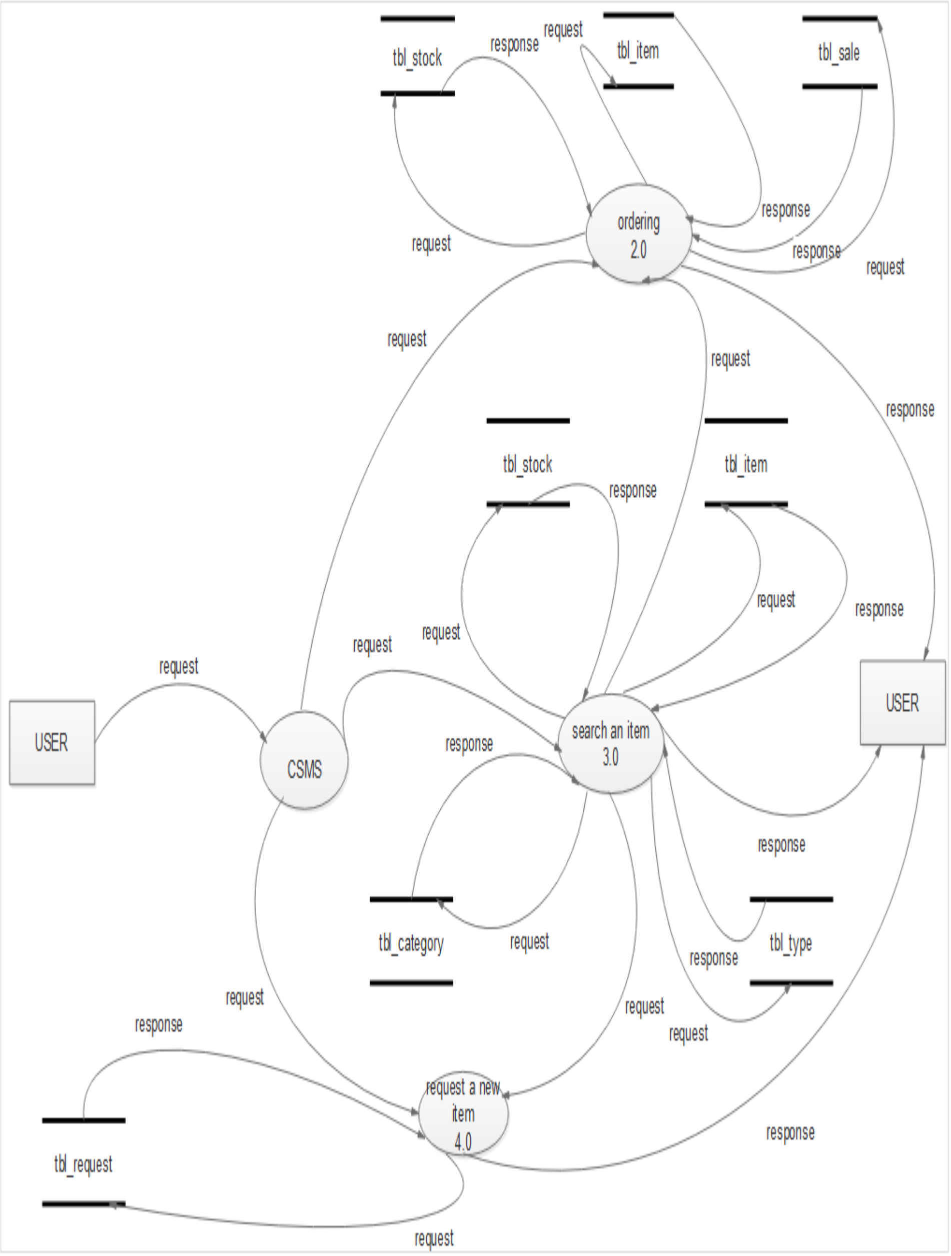


Figure 5: Level -1 data flow diagram of User module

Level -2 data flow diagram of item management in Storekeeper module

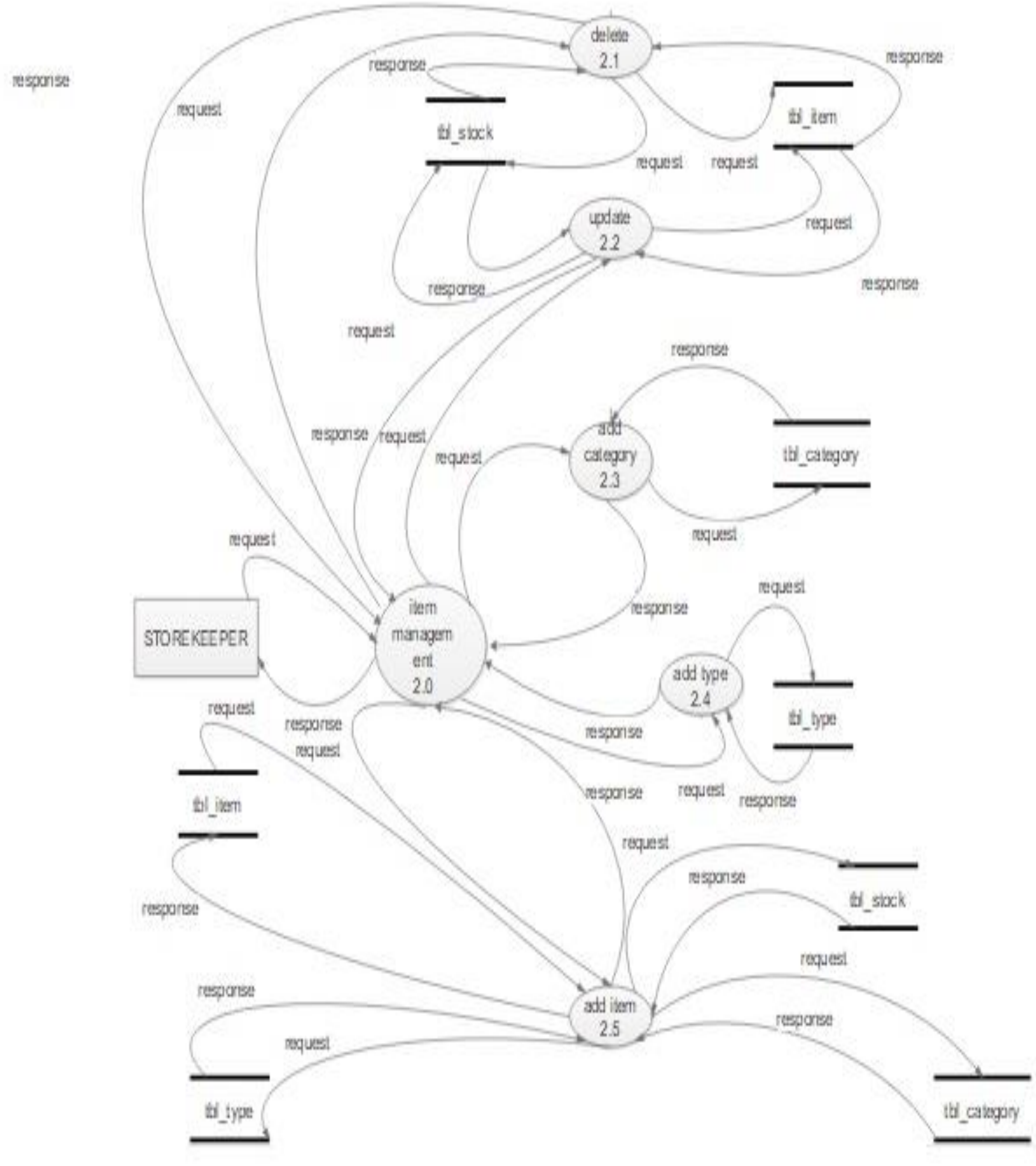


Figure 6: Level -2 data flow diagram of item management in Storekeeper module

Level -2 data flow diagram of Ordering in User module

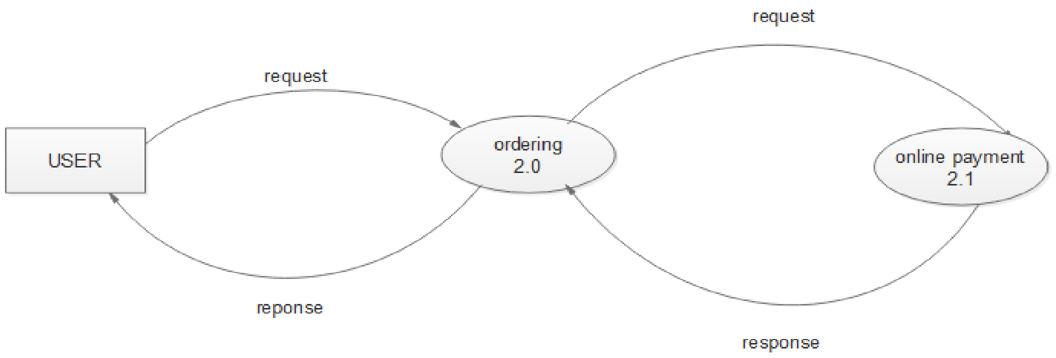


Figure 7: Level -2 data flow diagram of Ordering in User module

4.5 UML DIAGRAM

UML, or the Unified Modelling Language, is a modelling toolkit that guides the creation and notation of many types of diagrams, including behaviour diagrams, interaction diagrams, and structure diagrams.

4.5.1 USE CASE DIAGRAM

A use case diagram is a dynamic or behaviour diagram in UML. Use case diagrams model the functionality of a system using actors and use cases. Use cases are a set of actions, services, and functions that the system needs to perform. In this context, a "system" is something being developed or operated, such as a web site. The "actors" are people or entities operating under defined roles within the system.

Use case diagrams are valuable for visualizing the functional requirements of a system that will translate into design choices and development priorities. They also help identify any internal or external factors that may influence the system and should be taken into consideration. They provide a good high-level analysis from outside the system.

Use case diagrams specify how the system interacts with actors without worrying about the details of how that functionality is implemented.

Use case Diagram for Admin

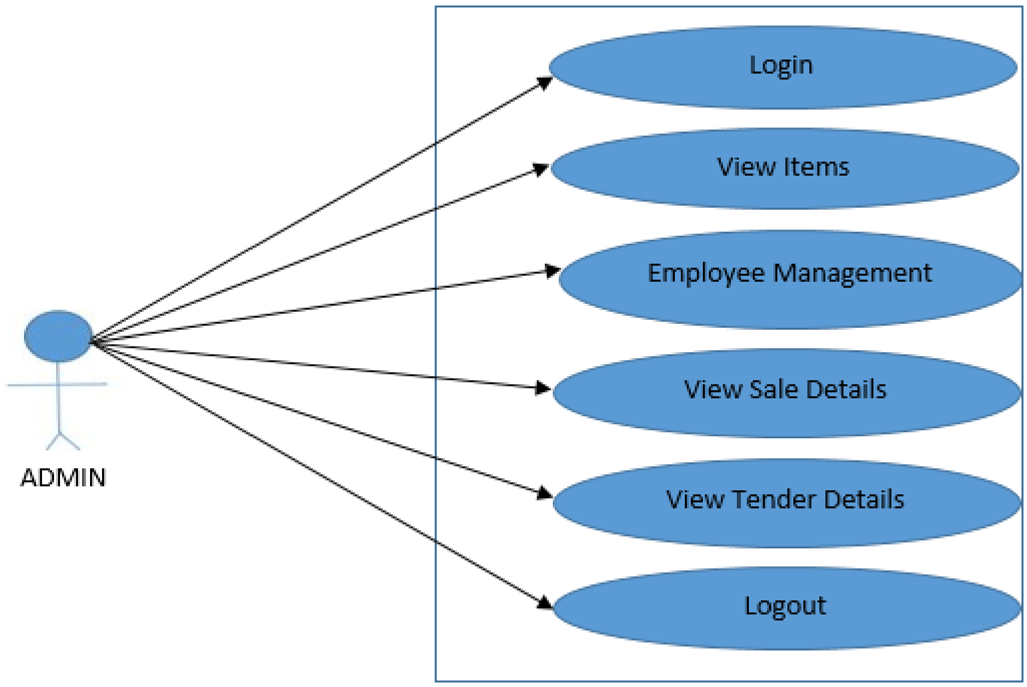


Figure 8: Use case Diagram for Admin

Use case Diagram for User

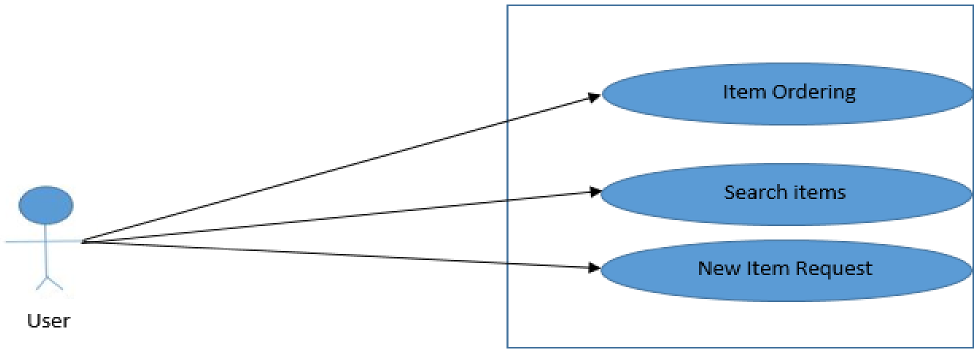


Figure 9: Use case Diagram for User

Use case Diagram for Store Keeper

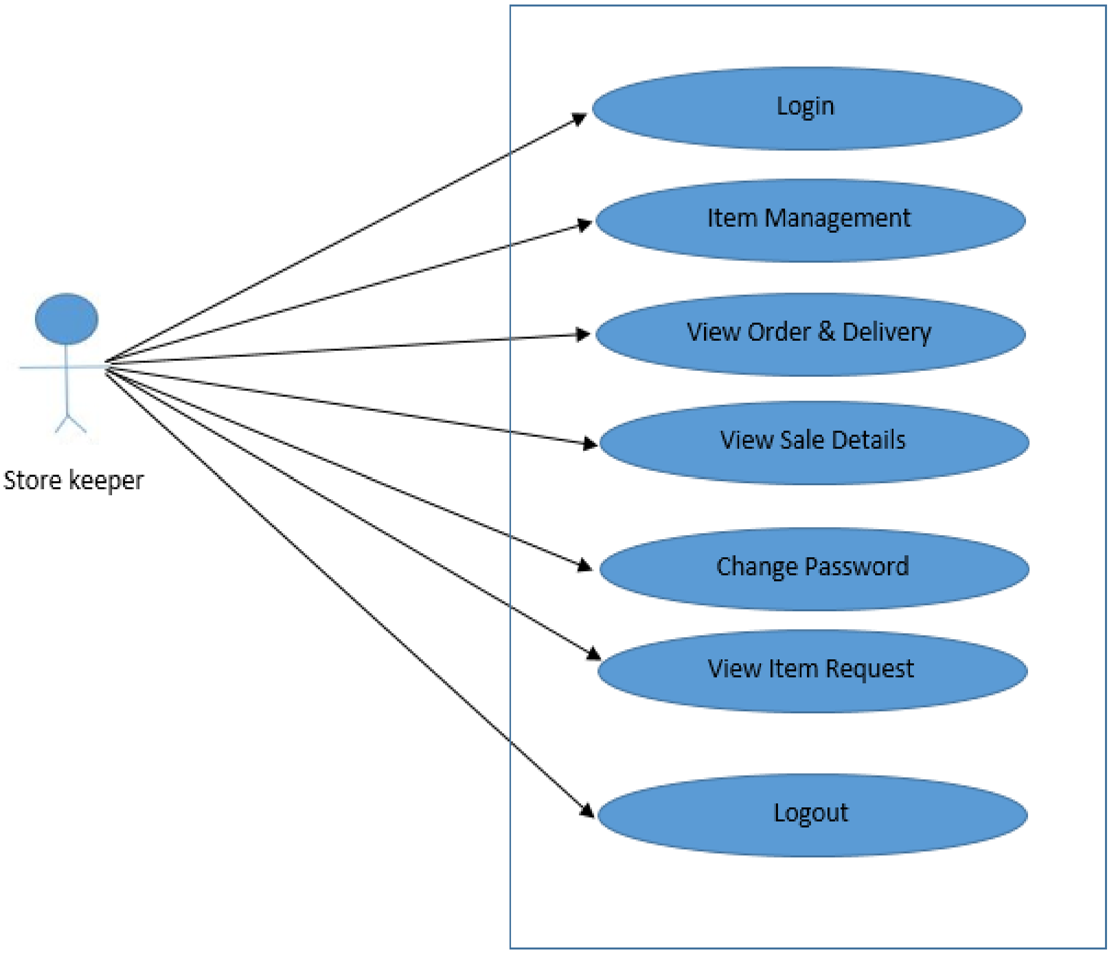


Figure 10: Use case Diagram for Store Keeper

4.6 SEQUENCE DIAGRAM

To understand what a sequence diagram is, it's important to know the role of UML. UML, or the Unified Modelling Language, is a modelling toolkit that guides the creation and notation of many types of diagrams, including behaviour diagrams, interaction diagrams, and structure diagrams. Sequence diagrams are a kind of interaction diagram, because they describe how and in what order a group of objects works together. These diagrams are used by software developers and business people alike to understand requirements for a new system or to document an existing process. Sequence diagrams are sometimes known as event diagrams or event scenarios.

Sequence diagram for user module in CSMS

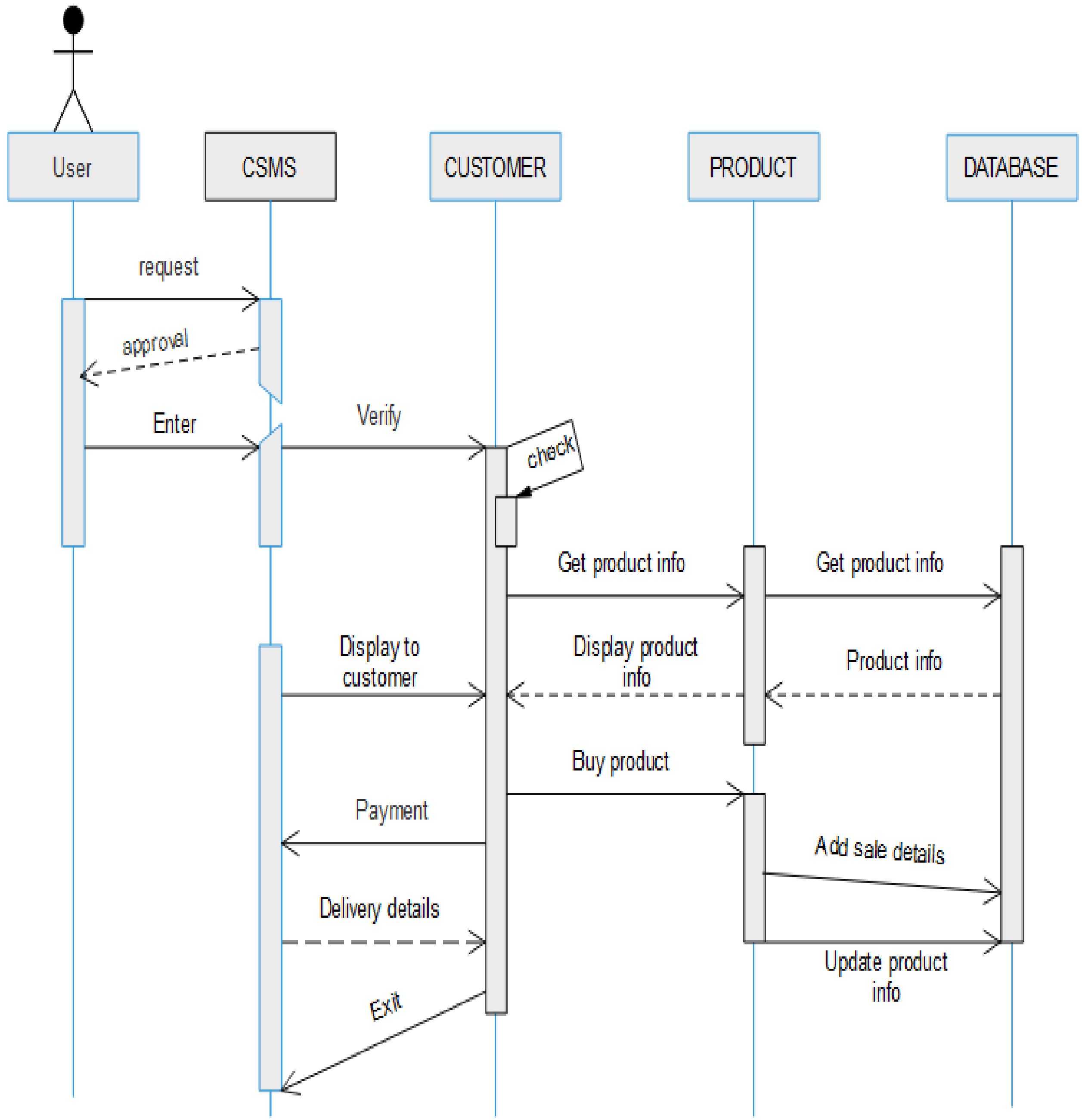


Figure 11: Sequence diagram for user module in CSMS

4.7 USER INTERFACE DESIGN

UI for Home page

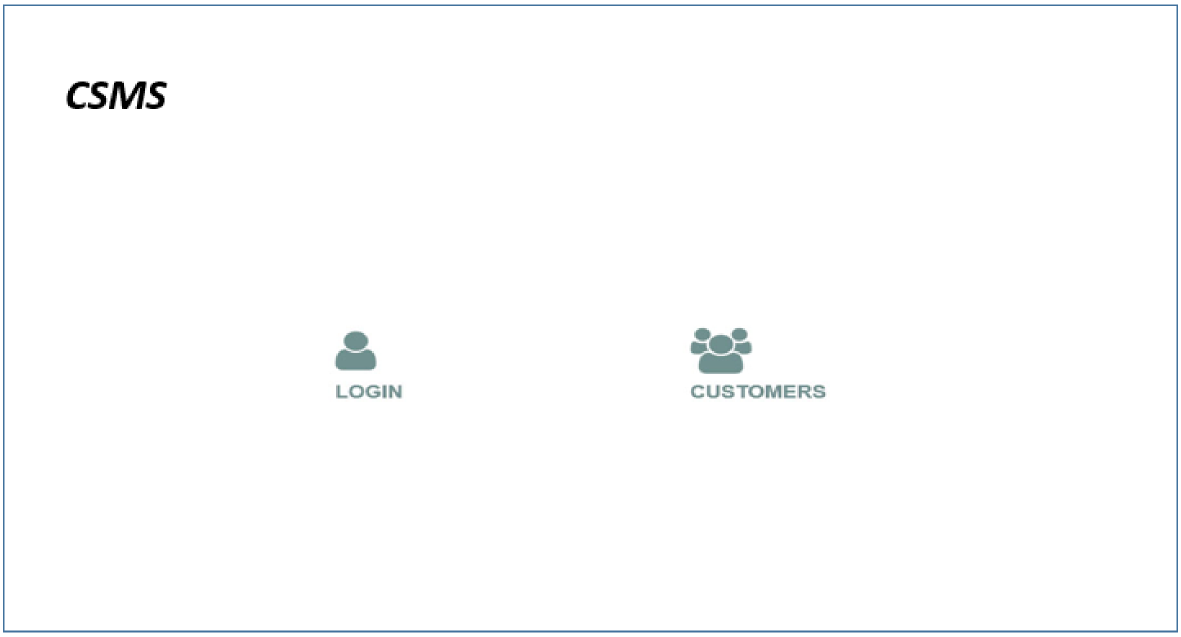


Figure 12: UI for Home page

UI for Login page

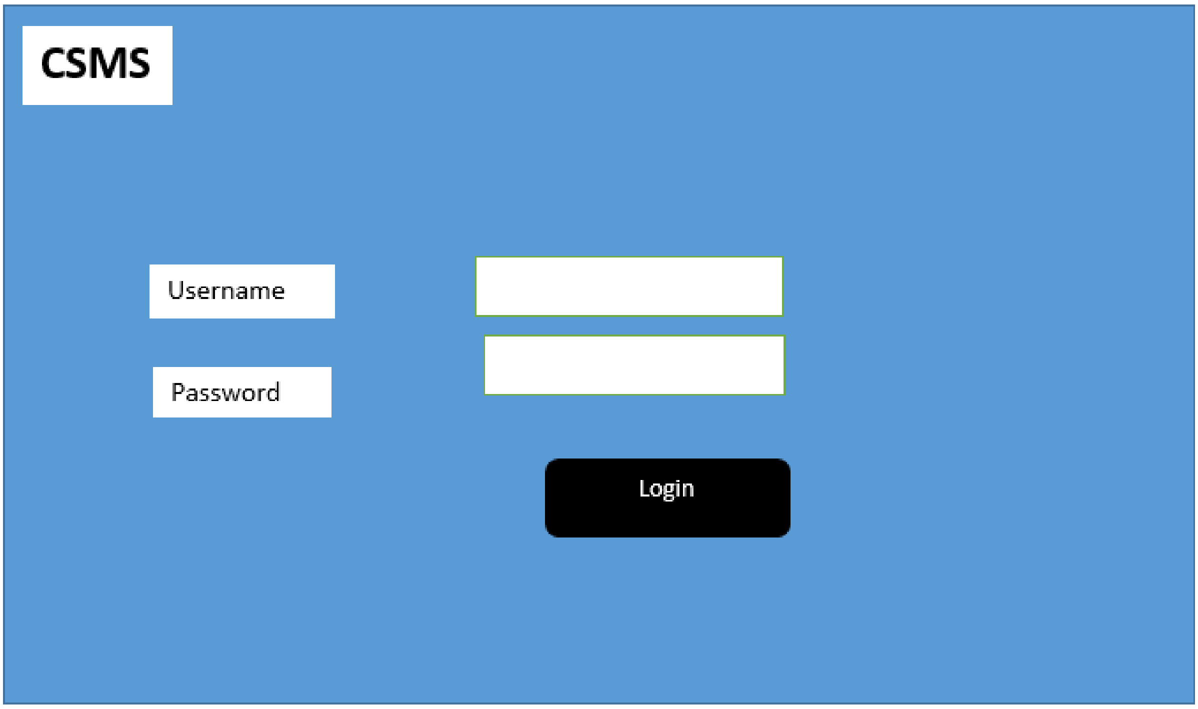


Figure 13: UI for Login page

UI for Buy Products page

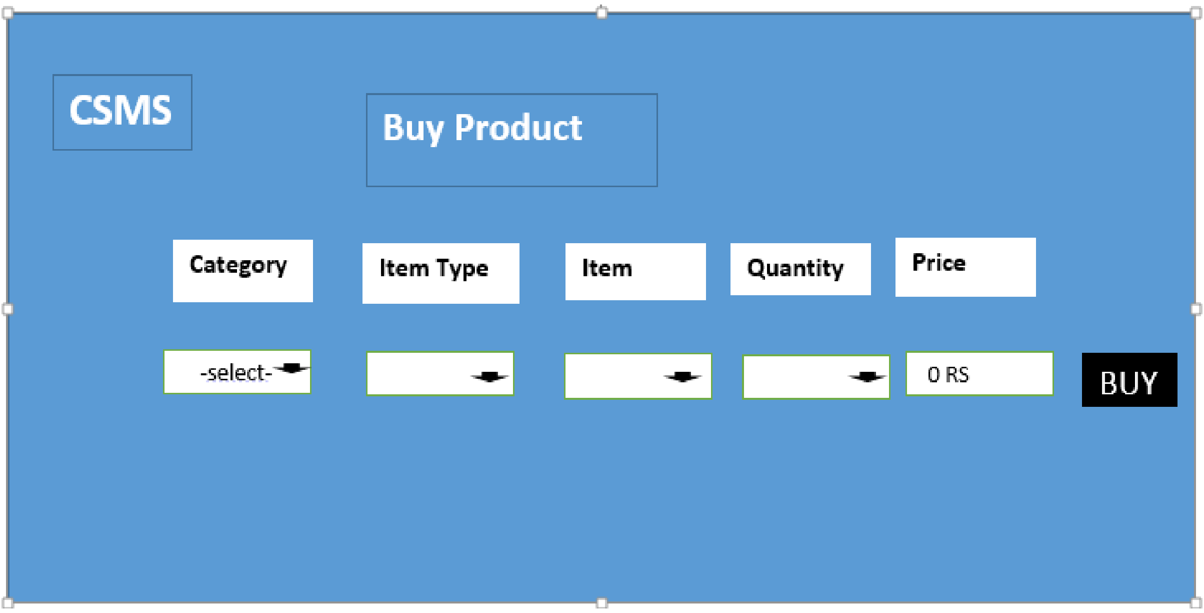


Figure 14: UI for Buy Products page

UI for Search Products page

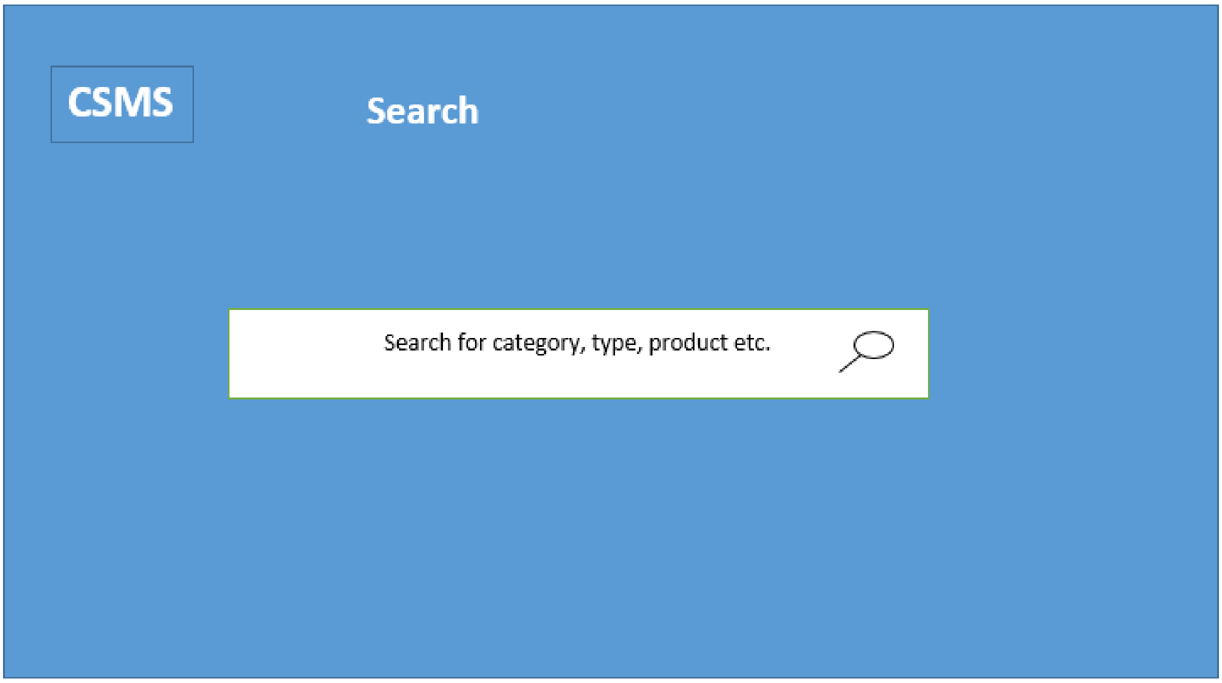


Figure 15: UI for Search Products page

4.8 DATABASE DESIGN

A database is an organized mechanism that has the capability of storing information through which a user can retrieve stored information in an effective and efficient manner. The data is the purpose of any database and must be protected.

The database design is a two-level process. In the first step, user requirements are gathered together and a database is designed which will meet these requirements as clearly as possible. This step is called Information Level Design and it is taken independent of any individual DBMS.

In the second step, this Information level design is transferred into a design for the specific DBMS that will be used to implement the system in question. This step is called Physical Level Design, concerned with the characteristics of the specific DBMS that will be used. A database design runs parallel with the system design. The organization of the data in the database is aimed to achieve the following two major objectives.

- Data Integrity
- Data independence

4.8.1 Relational Database Management System (RDBMS)

A relational model represents the database as a collection of relations. Each relation resembles a table of values or file of records. In formal relational model terminology, a row is called a tuple, a column header is called an attribute and the table is called a relation. A relational database consists of a collection of tables, each of which is assigned a unique name. A row in a tale represents a set of related values.

Relations, Domains & Attributes

A table is a relation. The rows in a table are called tuples. A tuple is an ordered set of n elements. Columns are referred to as attributes. Relationships have been set between every table in the database. This ensures both Referential and Entity Relationship Integrity. A domain D is a set of atomic values. A common method of specifying a domain is to specify a data type from which the data values forming the domain are drawn. It is also useful to specify a name for the domain to help in interpreting its values.

Every value in a relation is atomic, that is not decomposable.

Relationships

- Table relationships are established using Key. The two main keys of prime importance are Primary Key & Foreign Key. Entity Integrity and Referential Integrity Relationships can be established with these keys.
- Entity Integrity enforces that no Primary Key can have null values.
- Referential Integrity enforces that no Primary Key can have null values.
- Referential Integrity for each distinct Foreign Key value, there must exist a matching Primary Key value in the same domain. Other key are Super Key and Candidate Keys.

4.8.2 Normalization

Data are grouped together in the simplest way so that later changes can be made with minimum impact on data structures. Normalization is formal process of data structures in manners that eliminates redundancy and promotes integrity. Normalization is a technique of separating redundant fields and breaking up a large table into a smaller one. It is also used to avoid insertion, deletion, and updating anomalies. Normal form in data modelling use two concepts, keys and relationships. A key uniquely identifies a row in a table. There are two types of keys, primary key and foreign key. A primary key is an element or a combination of elements in a table whose purpose is to identify records from the same table. A foreign key is a column in a table that uniquely identifies record from a different table. All the tables have been normalized up to the third normal form.

As the name implies, it denotes putting things in the normal form. The application developer via normalization tries to achieve a sensible organization of data into proper tables and columns and where names can be easily correlated to the data by the user. Normalization eliminates repeating groups at data and thereby avoids data redundancy which proves to be a great burden on the computer resources. These include:

- Normalize the data.
- Choose proper names for the tables and columns.
- Choose the proper name for the data.

First Normal Form

The First Normal Form states that the domain of an attribute must include only atomic values and that the value of any attribute in a tuple must be a single value from the domain of that attribute. In other words, 1NF disallows “relations within relations” or “relations as attribute values within tuples”. The only attribute values permitted by 1NF are single atomic or indivisible values. The first step is to put the data into First Normal Form. This can be donor by moving data into separate tables where the data is of similar type in each table. Each table is given a Primary Key or Foreign Key as per requirement of the project. In this we form new relations for each non-atomic attribute or nested relation. This eliminated repeating groups of data. A relation is said to be in first normal form if only if it satisfies the constraints that contain the primary key only.

Second Normal Form

According to Second Normal Form, for relations where primary key contains multiple attributes, no non-key attribute should be functionally dependent on a part of the primary key. In this we decompose and setup a new relation for each partial key with its dependent attributes. Make sure to keep a relation with the original primary key and any attributes that are fully functionally dependent on it. This step helps in taking out data that is only dependent on a part of the key. A relation is said to be in second normal form if and only if it satisfies all the first normal form conditions for the primary key and every non-primary key attribute of the relation is fully dependent on its primary key alone.

Third Normal Form

According to Third Normal Form, Relation should not have a non-key attribute functionally determined by another non-key attribute or by a set of non-key attributes. That is, there should be no transitive dependency on the primary key. In this we decompose and set up relation that includes the non-key attributes that functionally determines other non-key attributes. This step is taken to get rid of anything that does not depend entirely on the Primary Key. A relation is said to be in third normal form if only if it is in second normal form and more over the non-key attributes of the relation should not be depend on another non-key attribute.

4.8.3 TABLE DESIGN

Table No. 1: tb_login

NO	**Field Name**	**Data Type**	**Description**
1	**Userid**	**Integer(11)**	**Primary key**
2	**username**	**Varchar(20)**	**Username of user for login**
3	**password**	**Varchar(20)**	**Password of the user**

Table No. :2 tb_log_type

NO	**Field Name**	**Data Type**	**Description**
1	**username**	**Varchar(20)**	**Primary key**
2	**log_type**	**Integer(2)**	**Value of log_type=1 then admin,** **log_type=0 then storekeeper.**

Table No. 3: tb_category

NO	**Field Name**	**Data Type**	**Description**
1	Category_id	Integer(11)	Primary key,which represents each category's id
2	Category	Varchar(50)	Represents the each category
3	Status	Integer(11)	Represents the status of type of category

Table No. 4: tb_type

No	**Field Name**	**Data Type**	**Description**
1	type_id	Integer(11)	Primary key
2	Category_id	Integer(11)	Foreign key(tb_category)
3	Type	Varchar(50)	Type of an item
4	Status	Integer(11)	Status disable or enable item

Table No. 5: tb_item

NO	**Field Name**	**Data Type**	**Description**
1	**item_id**	**Integer(11)**	**Primary key, which represents the id of each item**
2	**type_id**	**Integer(11)**	**Foreign key(from tb_type)**
3	**item_name**	**Varchar(80)**	**Name of the item**
4	**cost_price_unit**	**Integer(40)**	**Cost price of an item**
5	**selling_price_unit**	**Integer(40)**	**Selling price of an item**
6	**Status**	**Integer(11)**	**Status help to enable or disable an item**

Table No. 6: tb_stock

NO	Field Name	Data Type	Description
1	**Stock_id**	**Integer(11)**	**Primary key**
2	**item_id**	**Integer(11)**	**Foreign key(tb_item)**
3	**available_qty**	**Integer(11)**	**Available quantity of an item**

Table No. 7: tb_storekeeper

NO	Field Name	Data Type	Description
1	Sk_id	Integer	Primary key, which is the id of each storekeeper
2	Name	Varchar(20)	Name of the storekeeper
3	Address	Varchar(30)	Address of the storekeeper
4	Gender	Varchar(5)	Gender of the storekeeper
5	Email	Varchar(30)	Email id of the storekeeper
6	Phone	Varchar(10)	Phone no of the storekeeper
7	Username	Varchar(80)	Username of the storekeeper

Table No. 8: tb_tender

NO	Field Name	Data Type	Description
1	**tender_id**	**Integer(11)**	**Primary key**
2	**tender_name**	**Varchar(50)**	**Name of the tender**
3	**tender_phone**	**Varchar(15)**	**Phone no of the tender**
4	**category**	**Varcahr(50)**	**Category of an item**

Table No. 9: tb_sale

NO	Field Name	Data Type	Description
1	sale_id	Integer(11)	Primary key
2	admission_no	Varchar(20)	Foreign key(web service)
3	item_id	Integer(11)	Foreign key(tb_item)
4	Date	Varchar(15)	Date
5	quantity	Integer(11)	Quantity of purchasing item
6	Price	Integer(20)	Price of the purchased item or items
7	Status	Integer(11)	Status value=0 then user ordered Status value=1 then user paid their payment Status value=2 then delivered

Table No. 10*:* tb_request

No	Field Name	Data Type	Description
1	**request_id**	**Integer(11)**	**Primary key**
2	**admission_no**	**Integer(11)**	**Foreign key(web service)**
3	**category**	**Varchar(50)**	**Item’s category**
4	**type**	**Varchar(50)**	**Item’s type**
5	**item**	**Varchar(50)**	**Item name**
6	**date**	**Varchar(20)**	**Requesting date**

5 TESTING

5.1 INTRODUCTION

Software Testing is the process of executing software in a controlled manner, in order to answer the question - Does the software behave as specified? Software testing is often used in association with the terms verification and validation. Validation is the checking or testing of items, includes software, for conformance and consistency with an associated specification. Software testing is just one kind of verification, which also uses techniques such as reviews, analysis, inspections, and walkthroughs. Validation is the process of checking that what has been specified is what the user actually wanted.

Validation **:** Are we doing the right job?
Verification: Are we doing the job right?
Software testing should not be confused with debugging. Debugging is the process of analysing and localizing bugs when software does not behave as expected. Although the identification of some bugs will be obvious from playing with the software, a methodical approach to software testing is a much more thorough means for identifying bugs.
Debugging is therefore an activity which supports testing, but cannot replace testing.

Other activities which are often associated with software testing are static analysis and dynamic analysis. Static analysis investigates the source code of software, looking for problems and gathering metrics without actually executing the code. Dynamic analysis looks at the behavior of software while it is executing, to provide information such as execution traces, timing profiles, and test coverage information.

Testing is a set of activity that can be planned in advanced and conducted systematically. Testing begins at the module level and work towards the integration of entire computers based system. Nothing is complete without testing, as it vital success of the system testing objectives, there are several rules that can serve as testing objectives. They are:

Testing is a process of executing a program with the intent of finding an error.

- A good test case is one that has high possibility of finding an undiscovered error.
- A successful test is one that uncovers an undiscovered error.

If a testing is conducted successfully according to the objectives as stated above, it would uncover errors in the software. Also testing demonstrate that the software function appears to be working according to the specification, that performance requirement appears to have been met.

There are three ways to test program.

- For correctness
- For implementation efficiency
- For computational complexity

Test for correctness are supposed to verify that a program does exactly what it was designed to do. This is much more difficult than it may at first appear, especially for large programs.

5.2 TEST PLAN

A test plan implies a series of desired course of action to be followed in accomplishing various testing methods. The Test Plan acts as a blue print for the action that is to be followed. The software engineers create a computer program, its documentation and related data structures. The software developers is always responsible for testing the individual units of the programs, ensuring that each performs the function for which it was designed. There is an independent test group (ITG) which is to remove the inherent problems associated with letting the builder to test the thing that has been built. The specific objectives of testing should be stated in measurable terms. So that the mean time to failure, the cost to find and fix the defects, remaining defect density or frequency of occurrence and test work-hours per regression test all should be stated within the test plan.

The levels of testing include:

- Unit testing
- Integration Testing
- Data validation Testing
- Output Testing

5.2.1 Unit Testing

Unit testing focuses verification effort on the smallest unit of software design – the software component or module. Using the component level design description as a guide, important control paths are tested to uncover errors within the boundary of the module.

The relative complexity of tests and uncovered scope established for unit testing. The unit testing is white-box oriented, and step can be conducted in parallel for multiple components. The modular interface is tested to ensure that information properly flows into and out of the program unit under test. The local data structure is examined to ensure that data stored temporarily maintains its integrity during all steps in an algorithm's execution. Boundary conditions are tested to ensure that sall statements in a module have been executed at least once. Finally, all error handling paths are tested.

Tests of data flow across a module interface are required before any other test is initiated. If data do not enter and exit properly, all other tests are moot. Selective testing of execution paths is an essential task during the unit test. Good design dictates that error conditions be anticipated and error handling paths set up to reroute or cleanly terminate processing when an error does occur. Boundary testing is the last task of unit testing step. Software often fails at its boundaries.

Unit testing was done in Sell-Soft System by treating each module as separate entity and testing each one of them with a wide spectrum of test inputs. Some flaws in the internal logic of the modules were found and were rectified. After coding each module is tested and run individually. All unnecessary code where removed and ensured that all modules are working, and gives the expected result.

5.2.2 Integration Testing

Integration testing is systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing. The objective is to take unit tested components and build a program structure that has been dictated by design. The entire program is tested as whole. Correction is difficult because isolation of causes is complicated by vast expanse of entire program. Once these errors are corrected, new ones appear and the process continues in a seemingly endless loop.

After performing unit testing in the System all the modules were integrated to test for any inconsistencies in the interfaces. Moreover, differences in program structures were removed and a unique program structure was evolved.

5.2.3 Validation Testing or System Testing

This is the final step in testing. In this the entire system was tested as a whole with all forms, code, modules and class modules. This form of testing is popularly known as Black Box testing or System tests.

Black Box testing method focuses on the functional requirements of the software. That is, Black Box testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements for a program.

Black Box testing attempts to find errors in the following categories; incorrect or missing functions, interface errors, errors in data structures or external data access, performance errors and initialization errors and termination errors.

5.2.4 Output Testing or User Acceptance Testing

The system considered is tested for user acceptance; here it should satisfy the firm's need. The software should keep in touch with perspective system; user at the time of developing and making changes whenever required. This done with respect to the following points:

- Input Screen Designs,
- Output Screen Designs,

The above testing is done taking various kinds of test data. Preparation of test data plays a vital role in the system testing. After preparing the test data, the system under study is tested using that test data. While testing the system by which test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

5.3 TEST CASES

5.3.1 Test Case 1

Project Name: CSMS	
CSMS Test Case 1	
Test Case ID : Test Case 1	Test Designed by: Sojan Sunny
Test Priority(Low/Medium/High):Medium	Test Designed date: 20-01-2018
Module Name: Login screen	Test Executed by: Mrs. Shelly Shiju George

Test Title: Verify login with valid username and Password	Test Execution date: 20-01-2018
Description: Test the CSMS Login Page	
Pre-conditions: User has Username and Password	
Dependencies	

Step	Test Steps	Test Data	Expected Result	Actual Result	Status(Pass/Fail)	Note
1	Navigation to login page		Navigated to login page	Navigated to login page	Pass	
2	Provide Valid username	Username: myadmin	User should be able to login	Logged in and the User is navigated to Admin home page	Pass	
3	Provide valid password	Password: Myadmin1				
4	Click on login Button					

Post conditions:

User is validated with database and successfully login to account. The account session details are logged in database

5.3.2 Test Case 2

Project Name: CSMS	
CSMS Test Case 2	
Test Case ID: Test Case 2	Test Designed by: Sojan Sunny
Test Priority(Low/Medium/High):Medium	Test Designed date: 27-03-2018
Module Name: User Modules	Test Executed by: Mrs. Shelly Shiju George
Test Title: Buy Product	Test Execution date: 27-03-2018
Description: Buy Product	
Pre-conditions: User data should be correct and provide all necessary data	
Dependencies	

Step	Test Steps	Test Data	Expected Result	Actual Result	Status(Pass/Fail)	Note
1	Navigation to buy page		Navigated to buy page	Navigated to buy page	Pass	
2	Provide	Admission number, select category and type of item , select item and select quantity	Displayed price and could click buy Button. After button clicked, we shall see receipt.	Displayed price and could click buy Button. After button clicked, we could see receipt.	Pass	
3	Click on Buy	Validations/ Required				

	Button					
Post conditions: User is validated with database.						

5.3.3 Test Case 3

Project Name: CSMS	
CSMS Test Case 3	
Test Case ID: Test Case 3	Test Designed by: Sojan Sunny
Test Priority(Low/Medium/High):Medium	Test Designed date: 17-04-2018
Module Name: Storekeeper module	Test Executed by: Mrs. Shelly Shiju George
Test Title: Update item’s stock	Test Execution date: 17-04-2018
Description: Test the update item’s stock	
Pre-conditions: Stock data of each item should be correct and storekeeper must login	
Dependencies	

Step	Test Steps	Test Data	Expected Result	Actual Result	Status(Pass/Fail)	Note
1	Navigation to stock update page		Navigated to stock update page	Navigated to stock update page	Pass	
2	Provide Valid data	New Data	updated value should be shown	updated value should be shown	Pass	
3	Click on Update	Validations				

	Button					
Post conditions: New stock data is validated with database and other validations.						

6 IMPLEMENTATION

6.1 INTRODUCTION

Implementation is the stage of the project where the theoretical design is turned into a working system. It can be considered to be the most crucial stage in achieving a successful new system gaining the users confidence that the new system will work and will be effective and accurate. It is primarily concerned with user training and documentation. Conversion usually takes place about the same time the user is being trained or later. Implementation simply means convening a new system design into operation, which is the process of converting a new revised system design into an operational one.

At this stage the main work load, the greatest upheaval and the major impact on the existing system shifts to the user department. If the implementation is not carefully planned or controlled, it can create chaos and confusion.

Implementation includes all those activities that take place to convert from the existing system to the new system. The new system may be a totally new, replacing an existing manual or automated system or it may be a modification to an existing system. Proper implementation is essential to provide a reliable system to meet organization requirements. The process of putting the developed system in actual use is called system implementation. This includes all those activities that take place to convert from the old system to the new system. The system can be implemented only after through testing is done and if it is found to be working according to the specifications. The system personnel check the feasibility of the system. The more complex the system being implemented, the more involved will be the system analysis and design effort required to implement the three main aspects: education and training, system testing and changeover.

The implementation state involves the following tasks:

- Careful planning.
- Investigation of system and constraints.
- Design of methods to achieve the changeover.
- Training of the staff in the changeover phase.

6.2 IMPLEMENTATION PROCEDURES

Implementation of software refers to the final installation of the package in its real environment, to the satisfaction of the intended uses and the operation of the system. In many organizations someone who will not be operating it, will commission the software development project. In the initial stage people doubt about the software but we have to ensure that the resistance does not build up, as one has to make sure that:

- The active user must be aware of the benefits of using the new system.
- Their confidence in the software is built up.
- Proper guidance is imparted to the user so that he is comfortable in using the application.

Before going ahead and viewing the system, the user must know that for viewing the result, the server program should be running in the server. If the server object is not up running on the server, the actual process won’t take place.

6.2.1 User Training

User training is designed to prepare the user for testing and converting the system. To achieve the objective and benefits expected from computer based system, it is essential for the people who will be involved to be confident of their role in the new system. As system becomes more complex, the need for training is more important. By user training the user comes to know how to enter data, respond to error messages, interrogate the database and call up routine that will produce reports and perform other necessary functions.

6.2.2 Training on the Application Software

After providing the necessary basic training on computer awareness the user will have to be trained on the new application software. This will give the underlying philosophy of the use of the new system such as the screen flow, screen design type of help on the screen, type of errors while entering the data, the corresponding validation check at each entry and the ways to correct the date entered. It should then cover information needed by the specific user/ group to use the system or part of the system while imparting the training of the program on the application. This training may be different across different user groups and across different levels of hierarchy.

6.2.3 System Maintenance

Maintenance is the enigma of system development. The maintenance phase of the software cycle is the time in which a software product performs useful work. After a system is successfully implemented, it should be maintained in a proper manner. System maintenance is an important aspect in the software development life cycle. The need for system maintenance is for it to make adaptable to the changes in the system environment. Software maintenance is of course, far more than "Finding Mistakes".

7 CONCLUSION AND FUTURE ENHANCEMENTS

7.1 CONCLUSION

College Store Management System (CSMS) is web application for automate College Store. It focuses quick sale of stationery items for students according to their needs. The software reduces the time consumption and the manual efforts of buying items from a store. It will be a simple platform for users to access services for their huge needs.
The benefits, we can obtain from the new system are:

- Timely and accurate information will be available
- Reduced big queue at the store front.
- Online payment improves easiness.
- Better customer Satisfaction.
- Reduce Human Effort.
- The access time and process time is highly reduced
- Quick data view
- Error free output

The proposed system is expected to replace manual system and provide more efficient performance and services.

7.2 FUTURE ENHANCEMENTS

The system is designed in such a way that the payment to the store should be done in online mode. The system can be further extended to calculate profit, check demand and check most purchased student. In future, give more importance to admin and storekeeper modules because can add more functionalities to these modules.

8 BIBLIOGRAPHY

8.1 REFERENCES:

- Gary B. Shelly, Harry J. Rosenblatt, “*System Analysis and Design*”, 2009.
- Roger S Pressman, “*Software Engineering*”, 1994.
- Pankaj Jalote, “So*ftware engineering*: a precise approach”, 2006.
- James lee and Brent ware Addison, “Open source web development with LAMP”, 2003
- IEEE Std 1016 Recommended Practice for Software Design Descriptions.

8.2 WEBSITES:

- www.w3schools.com
- www.jquery.com
- http://homepages.dcc.ufmg.br/~rodolfo/es-1-03/IEEE-Std-830-1998.pdf
- www.agilemodeling.com/artifacts/useCaseDiagram.htm

APPENDIX

I. SAMPLE CODE

index.php

```
<?php
session_start();
$_SESSION['admin']='';
$_SESSION['sk']='';
?>
<html>
    <head>
    <meta charset="UTF-8">
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
        <title>CSMS</title>
        <link rel="stylesheet" type="text/css" href="bootstrap-3.3.7-
dist/css/bootstrap.min.css">
         <link rel="stylesheet" href="bootstrap-3.3.7-dist/css/bootstrap-theme.min.css">
         <link rel="stylesheet" href="bootstrap-3.3.7-dist/css/bootstrap.css">
         <link rel="stylesheet" href="bootstrap-3.3.7-dist/css/bootstrap-theme.css">

         <link rel="stylesheet" type="text/css" href="css/mystyle.css">
         <script src="jquery-3.3.1.min.js"></script>
         <script src="bootstrap-3.3.7-dist/js/bootstrap.min.js"></script>

         <script type="text/javascript">

</script>


    </head>
    <body>
        <div  id="main">
            <div class="container-fluid">
            <br>
            <div class="row">
                 <div class="col-sm-6">
                     <h1 style="color: deeppink;font: fantasy">
                         CSMS
                         </h1>
                <br>
                     <a href="home.php"><h2><b>HOME</b></h2></a>
            </div>
            <div class="col-xs-12">

            </div>

            </div>
            <div class="row"></div>
            <div class="row"></div>
        <div class="row"></div>
        <div class="row"><div class="col-sm-4"></div><div class="col-sm-4"><h2 
style="color: brown;">LOGIN</h2>
              <div class="row">
                 <form method="post" autocomplete="off">
                 <label>USERNAME</label>&nbsp;<input type="text" name="uname" 
id="uname" placeholder="Username" required><br/><br/>
                 <label>PASSWORD</label>&nbsp;<input type="password" 
name="password" id="password"  placeholder="Password" required><br/><br/>
                 <input type="submit" name="login" id="login"  class="btn btn-success" 
value="LOGIN >>" style="color: #ffffff ">
                 <br>
                    <?php
      // put your code here
      if($_POST){
$con=mysqli_connect("localhost","root","","csms");
        if($con)
        {
        $usr=$_POST['uname'];
        $pass=$_POST['password'];
        $r=mysqli_query($con,"select * from tb_login where BINARY username='$usr' 
and BINARY password='$pass';");
        if(mysqli_num_rows($r)==1)
        {
           session_start();
           $tp=mysqli_query($con,"select * from tb_log_type where username='$usr';");
              $row=mysqli_fetch_array($tp);
               if($row['log_type']==1){
                  $_SESSION['admin']=$usr;
               echo("<script>location.href = '"."admin/"."';</script>");}
               else{
                   $_SESSION['sk']=$usr;
                  echo("<script>location.href = '"."storekeeper/"."';</script>");
               }
        }
        else{
           echo 'Incorrect username/password';
        }
      }
}
      ?>
                 </form>
              </div>

           </div>
           </div>
           <div class="col-sm-4"></div>

        </div>
      </div>
      </div>
    <script src="jquery-3.3.1.min.js"></script>
    <script src="bootstrap-3.3.7-dist/js/bootstrap.min.js"></script>
    <script src="bootstrap-3.3.7-dist/js/bootstrap.js"></script>
    <script src="bootstrap-3.3.7-dist/js/npm.js"></script>
   </body>
</html>
```

buy.php

```
<?php
session_start();
?>
<html>
   <head>
   <meta charset="UTF-8">
   <meta charset="utf-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1">
      <title>CSMS</title>
      <link rel="stylesheet" type="text/css" href="../bootstrap-3.3.7-
dist/css/bootstrap.min.css">
      <link rel="stylesheet" href="../bootstrap-3.3.7-dist/css/bootstrap-theme.min.css">
      <link rel="stylesheet" href="../bootstrap-3.3.7-dist/css/bootstrap.css">
      <link rel="stylesheet" href="../bootstrap-3.3.7-dist/css/bootstrap-theme.css">
       <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">
      <link rel="stylesheet" type="text/css" href="../css/mystyle.css">
      <link rel="stylesheet" type="text/css" href="../css/font-awesome.min.css">
      <script src="../jquery-3.3.1.min.js"></script>
       <script src="../bootstrap-3.3.7-dist/js/bootstrap.min.js"></script>

       <script type="text/javascript">
 function gettype(str) {
   if (str == "") {
      document.getElementById("itemtype").innerHTML = "";
      return;
   } else {
      if (window.XMLHttpRequest) {
         // code for IE7+, Firefox, Chrome, Opera, Safari
         xmlhttp = new XMLHttpRequest();
      } else {
         // code for IE6, IE5
         xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
      }
      xmlhttp.onreadystatechange = function() {
         if (this.readyState == 4 && this.status == 200) {
            document.getElementById("itemtype").innerHTML = this.responseText;
         }
      };
      xmlhttp.open("GET","../storekeeper/sel_type_item.php?q="+str,true);
      xmlhttp.send();
   }
}
function getitem(str) {
    if (str == "") {
        document.getElementById("item").innerHTML = "";
        return;
    } else {
        if (window.XMLHttpRequest) {
            // code for IE7+, Firefox, Chrome, Opera, Safari
            xmlhttp = new XMLHttpRequest();
        } else {
            // code for IE6, IE5
            xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
        }
        xmlhttp.onreadystatechange = function() {
            if (this.readyState == 4 && this.status == 200) {
                document.getElementById("item").innerHTML = this.responseText;
            }
        };
        xmlhttp.open("GET","sel_item.php?q="+str,true);
        xmlhttp.send();
    }
}
function getqty(str) {
  // var str=document.getElementById('item').value;
    if (str == "") {
        document.getElementById("qty").innerHTML = "";
        return;
    } else {
        if (window.XMLHttpRequest) {
            // code for IE7+, Firefox, Chrome, Opera, Safari
            xmlhttp = new XMLHttpRequest();
        } else {
            // code for IE6, IE5
            xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
        }
        xmlhttp.onreadystatechange = function() {
            if (this.readyState == 4 && this.status == 200) {
                document.getElementById("qty").innerHTML = this.responseText;
            }
        };
        xmlhttp.open("GET","qty.php?q="+str,true);
        xmlhttp.send();
    }
}
</script>
  <script>
    function valid_admno() {

  var str=document.getElementById('admno').value;
  //alert(str);
    if (str == "") {
        //document.getElementById("qty").innerHTML = "";
        return;
    } else {
        if (window.XMLHttpRequest) {
         // code for IE7+, Firefox, Chrome, Opera, Safari
         xmlhttp = new XMLHttpRequest();
      } else {
         // code for IE6, IE5
         xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
      }
      xmlhttp.onreadystatechange = function() {
         if (this.readyState == 4 && this.status == 200) {
            if(this.responseText=="no")
            {
               alert("Enter valid  Admission Number");
               document.getElementById('admno').value="";
            }

         }
      };
      xmlhttp.open("GET","checkadmission_no.php?q="+str,true);
      xmlhttp.send();
   }
}
 </script>

   <script>
     function get(){
var q = $("#qty").val();
var item = $("#item").val();
//alert(q);
$.ajax({
   url: 'cal_price.php',
   data: {
      'qty': q,
      'itm': item
   },
    type: "GET",
      dataType: "html",
   success: function(data) {
      //alert(data.toString());
    // var result = $('<label />').append(data).find('#price').html();
    $('#price').html(data.toString());

    document.getElementById('pri').value=data.toString();
   }
});
     }
      </script>
         <script type = "text/javascript" >
          history.pushState({ page: 1 }, "Title 1", "");
window.onhashchange = function (event) {
  window.location.hash = "";
};
window.addEventListener('popstate', function () {
   history.pushState(null, null, document.URL);
});
```

```
</script>
  </head>
  <body>
    <div  id="main">
      <div class="container-fluid">

      <div class="row">
         <div class="col-sm-6">
           <h1 style="color: deeppink;font: fantasy">
             <img src='../img/csms.jpg' style="width:30%;height: 10% "/>
             </h1>

      </div>
      <div class="col-xs-12">

      </div>

      </div>
        <div class="row">

           <div class="col-sm-8" style="position: absolute;left:450px;top:100px;">
               <h1 style="font-size:40px; color:#ff0a45 "> Buy products</h1>
            </div>
        </div>
        <br>

          <div class="row">
  <div class="col-sm-4">
    <a href="buy.php">
   <span class="glyphicon glyphicon-shopping-cart" style="font-size:48px;color:tomato"></span>
   <h4 style="color: red;font-size: 15px;">BUY</h4>

   </a>
     <a href="search.php">
   <span class="glyphicon glyphicon-search" style="font-size:48px;color:tomato"></span>
   <h4 style="color:red;font-size: 15px;">SEARCH</h4>

    </a>
    <a href="request.php">
   <i class="fa fa-pencil-square-o" style="font-size:48px;color:tomato"></i>
   <h4 style="color: red;font-size: 15px;">REQUEST</h4>

  </a>
   </div>
            <div class="col-lg-8" style="position: absolute;left:200px;top:150px;">
               <i class="glyphicon glyphicon-plus-sign fa-2x" style="color:blue"></i>

               <table class="table">
```

```
                    <?php

                    if($_POST){ ?>
                    <thead class="thead-dark">
                    <tr style="background-color:white;">
                    <th style="font-size: 20px; color: red;">Category</th>
                    <th style="font-size: 20px; color: red;">Type</th>
                    <th style="font-size: 20px; color: red;">Product</th>
                   <th style="font-size: 20px; color: red;">Quantity</th>
                   <th style="font-size: 20px; color: red;">Price</th>
                    </tr>
                    </thead>
                    <tr style="background-color: #000\9;font-size: 20px; color: blue;">
                    <form method="post" action="receipt.php" autocomplete="off">


                        <input type="hidden" name="admno" value="<?php echo
$_SESSION['admino']; ?>">
                        <td style="font-size: 15px;">
                <select id="category" name="category_id"
onchange="gettype(this.value)" >
                <?php  $con=mysqli_connect("localhost","root","","csms");
        if($con)
        {
       $r=mysqli_query($con,"select * from tb_category;");

       echo ' <option value="0">-Select-</option>';
       while($rw= mysqli_fetch_array($r))
       {
          echo '<option value='.$rw[category_id].'>'.$rw[category].'</option>';

       }
     }
     ?></select>
                        </td>
                        <td style="font-size: 15px;">

                <select id="itemtype" name="type_id" onchange="getitem(this.value)">
                </select>
                        </td>
                        <td style="font-size: 15px;">

                           <select id="item" name="item" onchange="getqty(this.value)"
>
                </select>
                        </td>
                        <td style="font-size: 15px;">
                           <select name="qty"  id="qty" onchange="get()"
required="true" ></select>
                        </td>
                        <td id="price" style="color:black; font:bold; font-size: 20px;">

                        </td>
```

```
                         <input type='hidden' id="pri" name="pri">
                         <td>
                            <input type="submit" class="btn btn-danger" name="buy"
style="color: white;font-size: 20px;font:bold;" value="BUY">
                         </td>
    </form>

                   </tr>
                      <?php }
                      else {

                      ?>
                      <thead>
                         <tr style="background-color:white;">
                      <th style="font-size: 20px; color: red;">Admission no</th>
                      <th style="font-size: 20px; color: red;">Category</th>
                      <th style="font-size: 20px; color: red;">Type</th>
                      <th style="font-size: 20px; color: red;">Product</th>
                     <th style="font-size: 20px; color: red;">Quantity</th>
                     <th style="font-size: 20px; color: red;">Price</th>
                         </tr>
                      </thead>
                      <tr style="color: blue;">
                      <form method="post" action="receipt.php" autocomplete="off">

                      <td style="font-size: 15px;">
                         <input type="text" name="admno" id="admno"
onchange="valid_admno()" placeholder="Admission No" required>
                         </td><td style="font-size: 15px;">
                <select id="category" name="category_id"
onchange="gettype(this.value)" >
                <?php  $con=mysqli_connect("localhost","root","","csms");
        if($con)
        {
       $r=mysqli_query($con,"select * from tb_category;");

       echo ' <option value="0">-Select-</option>';
       while($rw= mysqli_fetch_array($r))
       {
          echo '<option value='.$rw[category_id].'>'.$rw[category].'</option>';

       }
     }
     ?></select>
                         </td>
                         <td style="font-size: 15px;">

                <select id="itemtype" name="type_id" onchange="getitem(this.value)">
                </select>
                         </td>
                         <td style="font-size: 15px;">
```

```
                    <select id="item" name="item" onchange="getqty(this.value)"
>
            </select>
                  </td>
                  <td style="font-size: 15px;">
                    <select name="qty"  id="qty" onchange="get()"
required="true" ></select>
                  </td>
                  <b>
                  <td id="price" style="color:black; font:bold; font-size: 20px;">

                  </td></b>
                  <input type='hidden' id="pri" name="pri">
                  <td>
                    <input type="submit" class="btn btn-danger" name="buy"
style="color:white;font-size: 20px;font:bold;" value="BUY">
                  </td>
   </form>

              </tr>
              <?php }


                ?>

     <br><br>            </table>

            </div>
 </div>
 </div>
   </div>
   <script src="jquery-3.3.1.min.js"></script>
   <script src="bootstrap-3.3.7-dist/js/bootstrap.min.js"></script>
   <script src="bootstrap-3.3.7-dist/js/bootstrap.js"></script>
   <script src="bootstrap-3.3.7-dist/js/npm.js"></script>
  </body>
</html>
```

II. SCREEN SHOTS

Home page

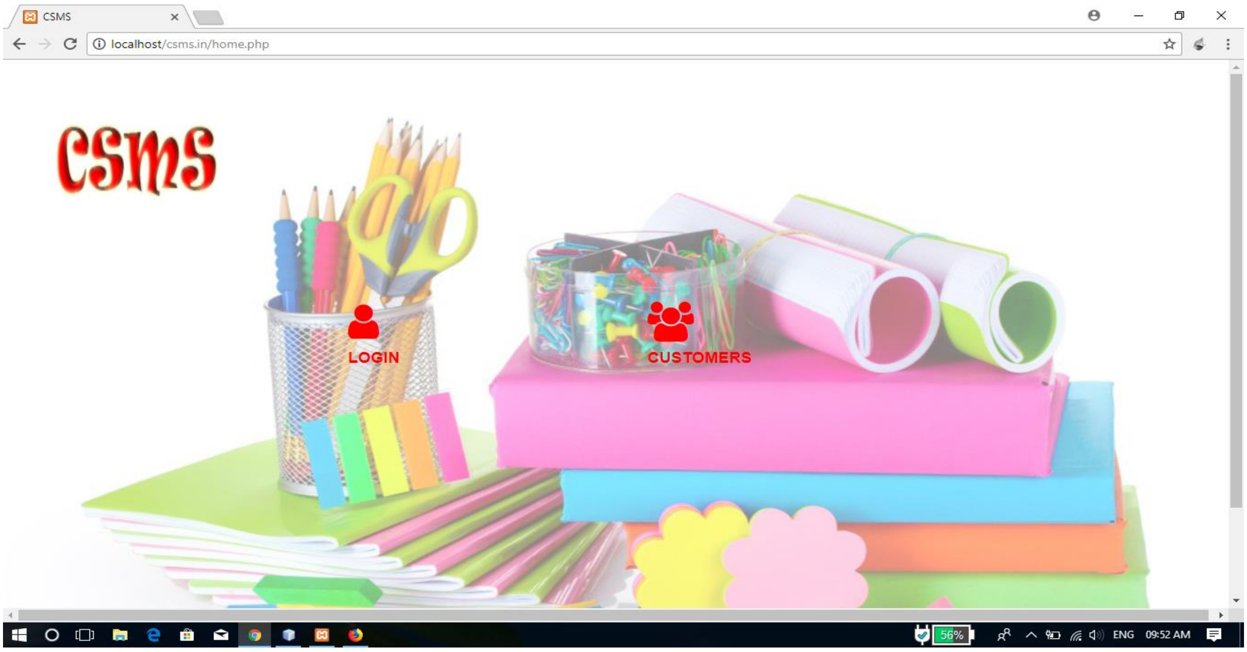


Figure 16: Home page

Index page of user

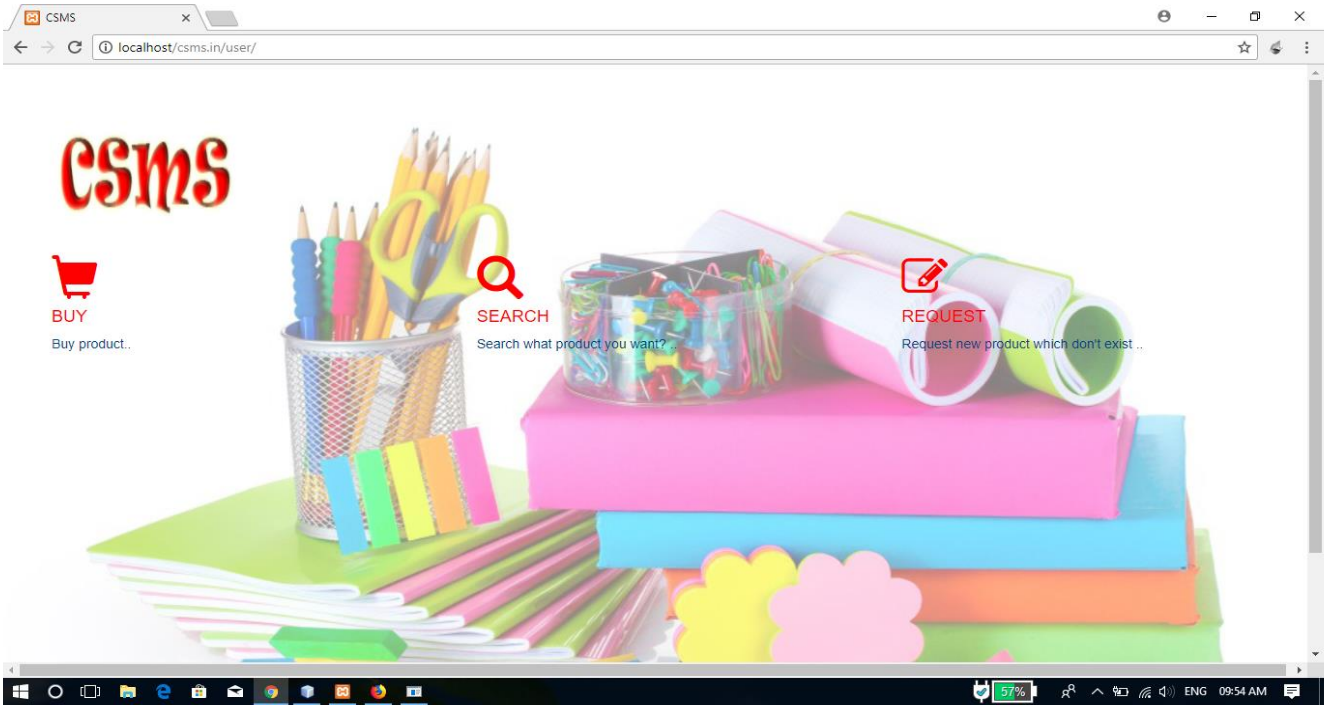
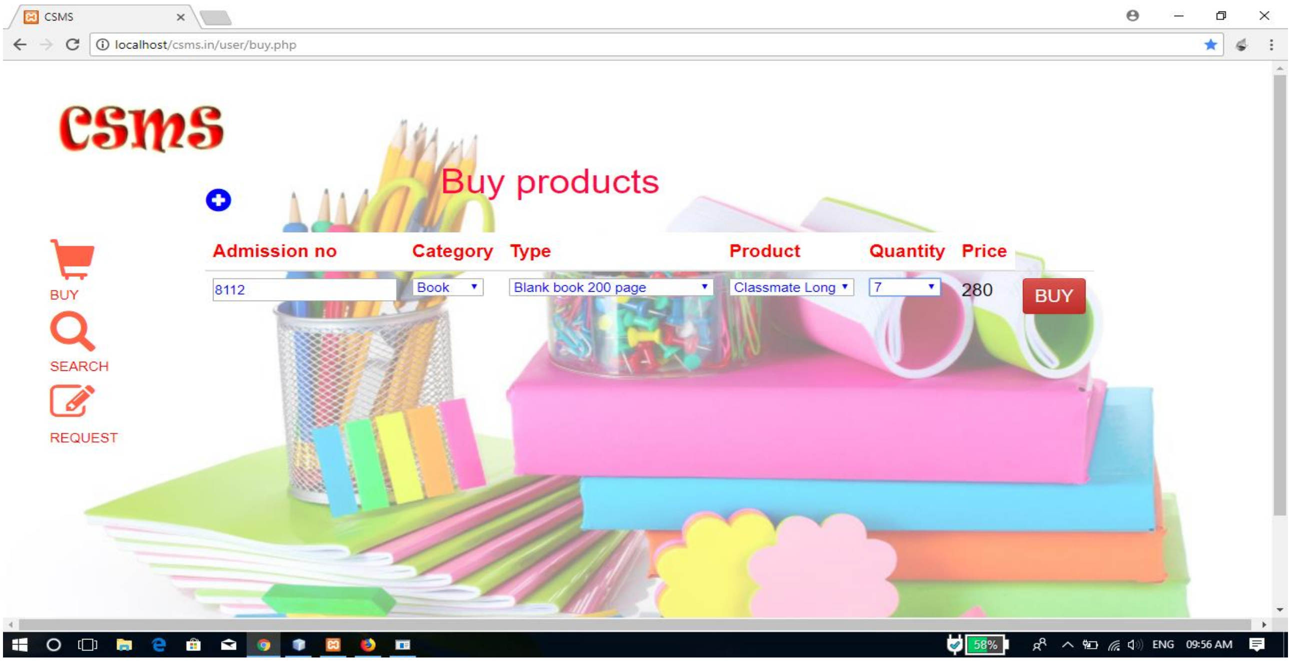


Figure 17: Index page of user

Buy product page

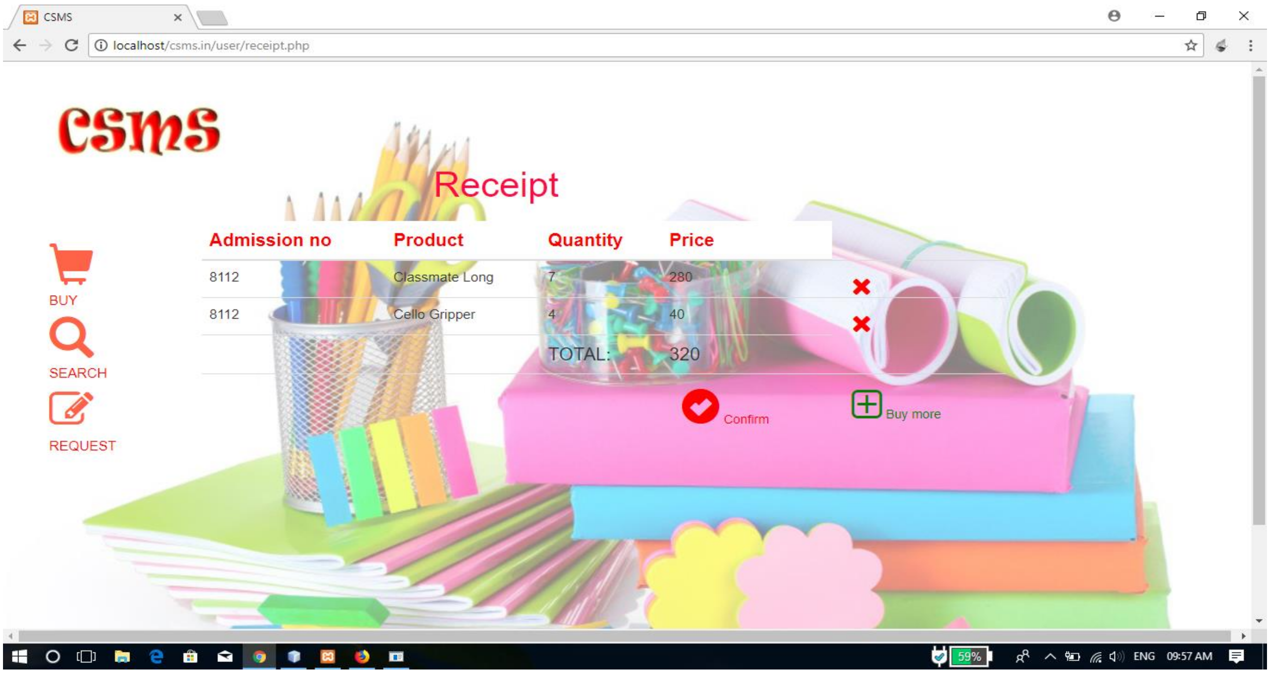


Buy products

Admission no	Category	Type	Product	Quantity	Price	
8112	Book	Blank book 200 page	Classmate Long	7	280	BUY

Figure 18: Buy product page

Receipt page

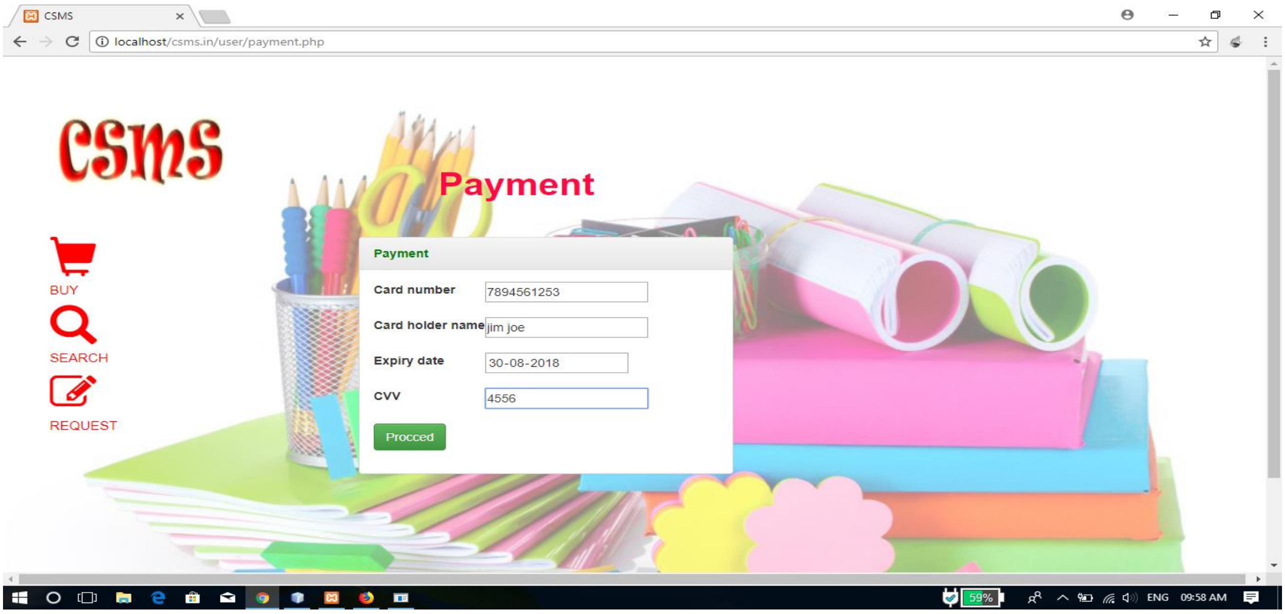


Receipt

Admission no	Product	Quantity	Price
8112	Classmate Long	7	280
8112	Cello Gripper	4	40
		TOTAL:	320

Figure 19: Receipt page

Payment page



Payment

Card number 7894561253

Card holder name jim joe

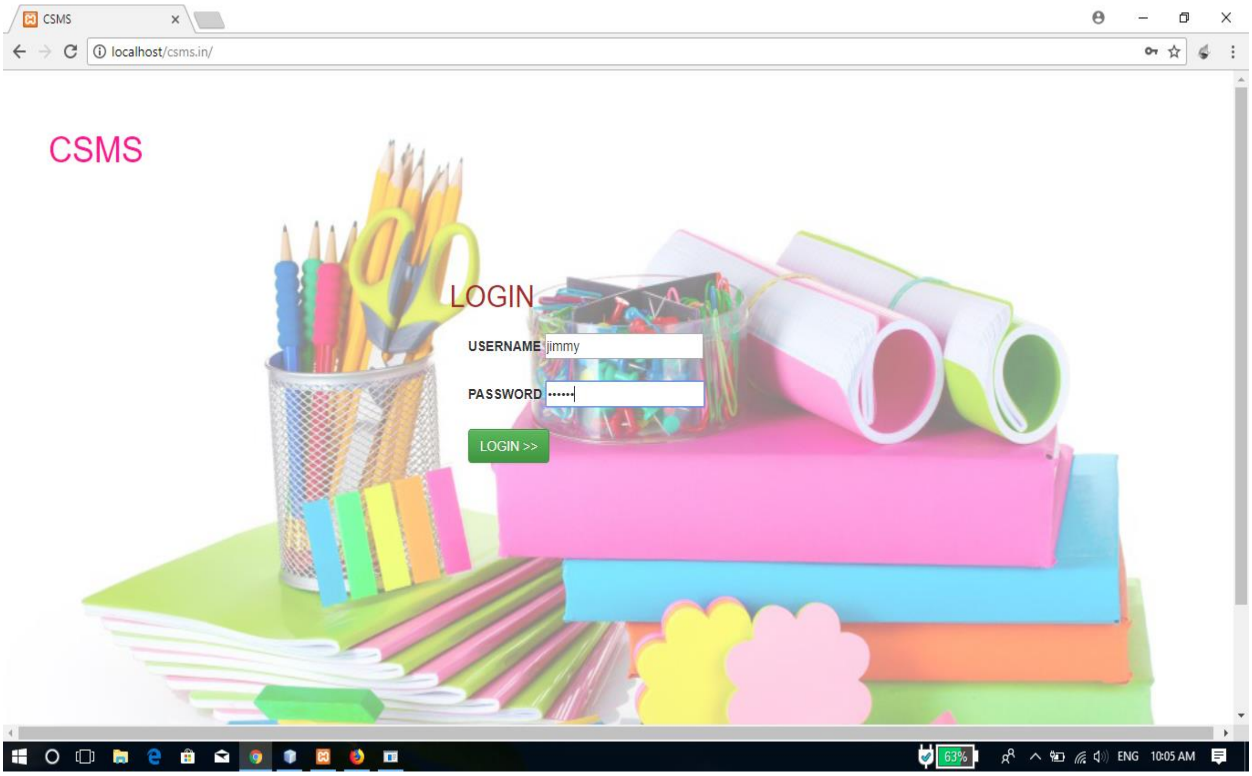
Expiry date 30-08-2018

CVV 4556

Procced

Figure 20: Payment page

Login page for Admin and Storekeeper



LOGIN

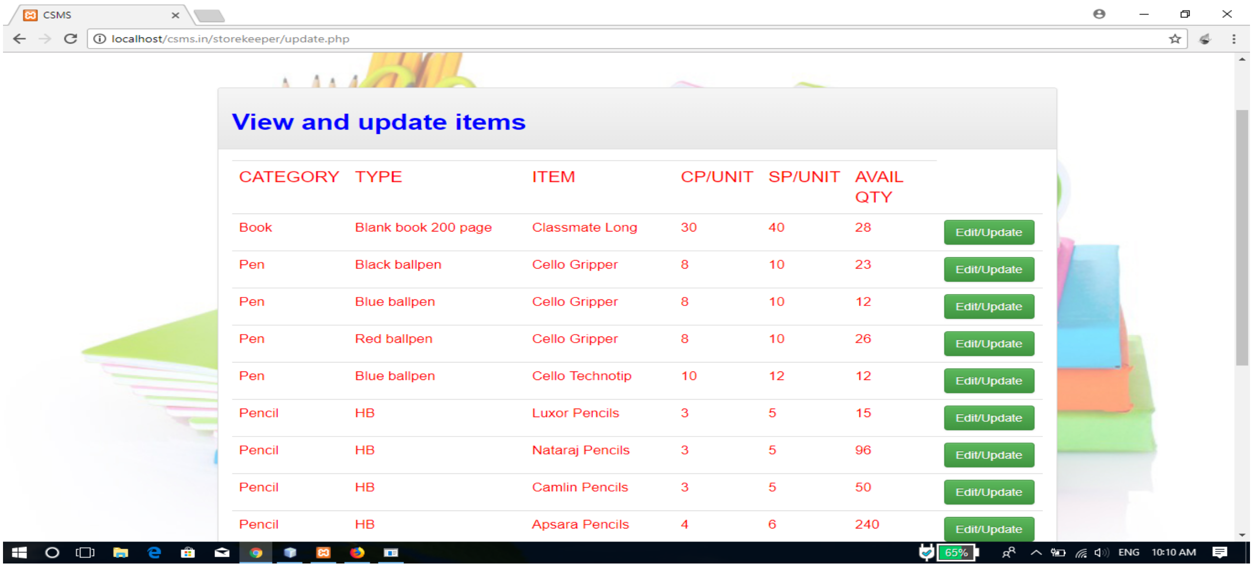
USERNAME jimmy

PASSWORD ••••••

LOGIN >>

Figure 21: Login page for Admin and Storekeeper

View and update items

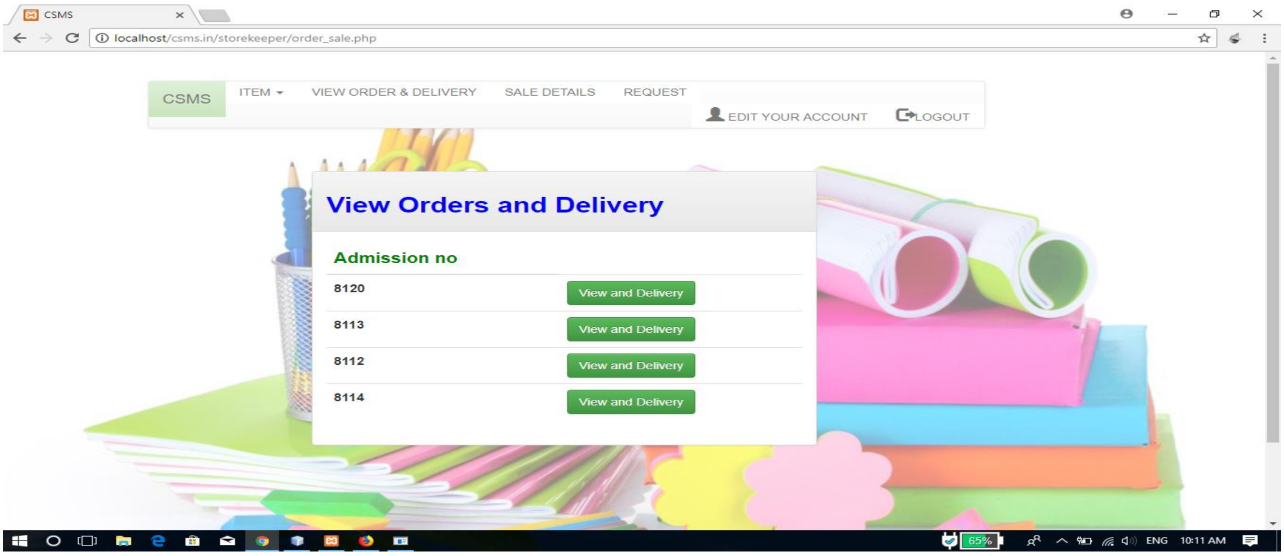


View and update items

CATEGORY	TYPE	ITEM	CP/UNIT	SP/UNIT	AVAIL QTY	
Book	Blank book 200 page	Classmate Long	30	40	28	Edit/Update
Pen	Black ballpen	Cello Gripper	8	10	23	Edit/Update
Pen	Blue ballpen	Cello Gripper	8	10	12	Edit/Update
Pen	Red ballpen	Cello Gripper	8	10	26	Edit/Update
Pen	Blue ballpen	Cello Technotip	10	12	12	Edit/Update
Pencil	HB	Luxor Pencils	3	5	15	Edit/Update
Pencil	HB	Nataraj Pencils	3	5	96	Edit/Update
Pencil	HB	Camlin Pencils	3	5	50	Edit/Update
Pencil	HB	Apsara Pencils	4	6	240	Edit/Update

Figure 22: View and update items

View Orders and Delivery

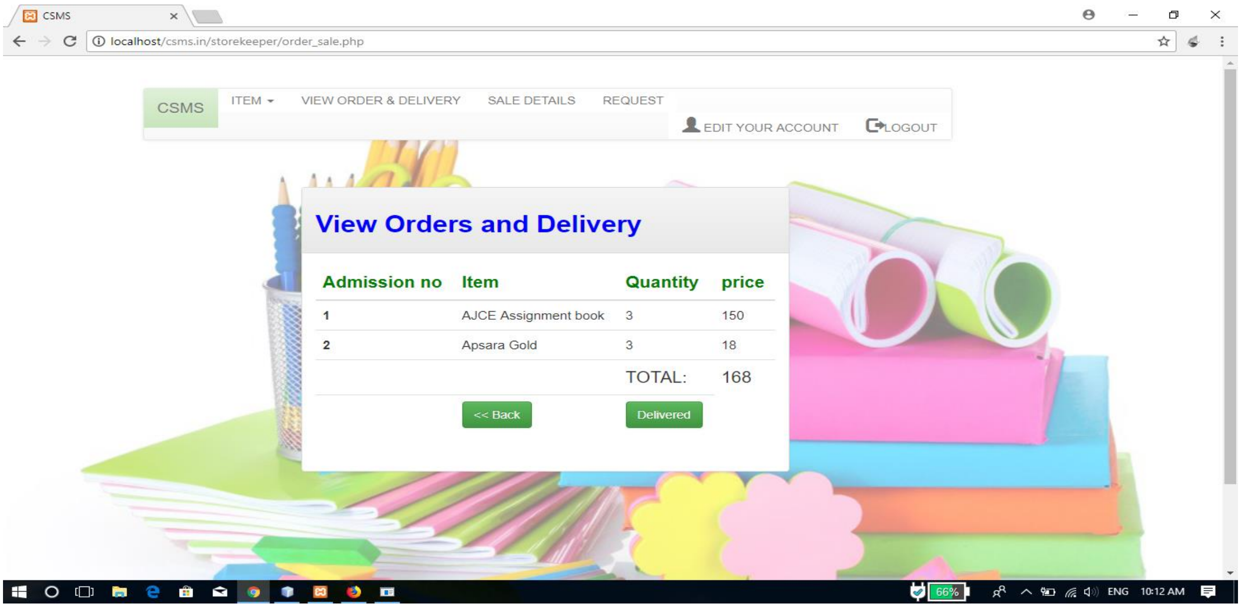


View Orders and Delivery

Admission no	
8120	View and Delivery
8113	View and Delivery
8112	View and Delivery
8114	View and Delivery

Figure 23: View Orders and Delivery

View Order and Delivery (View order details of each user)



View Orders and Delivery

Admission no	Item	Quantity	price
1	AJCE Assignment book	3	150
2	Apsara Gold	3	18
		TOTAL:	168

Figure 24: View Order and Delivery (View order details of each user)