ABSTRACT

Web app is made for helping people to find solution to their home appliances. If there are any problems with home appliances like fan, refrigerator, TV etc., The Web app will guide the user through the solution. Users are required to go through a questionnaire in the ‘Common Queries’ section for specific appliances. The questionnaire will help the user to diagnose the issue, find possible solutions and find expert assistance if required.

LIST OF FIGURES

S NO	TABLE NO	TABLE NAME	PAGE NO

13	7.1.1.7	Rating Table	

LIST OF TABLES

14	7.1.1.8	Staff Table	
15	7.1.1.9	Stock Report Table	
16	7.1.1.10	User Table	

S NO	ABBREVATION	EXPANSION
1	QR	Quick Response
2	AOSP	Android Open Source Project
3	GMS	Google Mobile Service
4	AEP	Automated Error Prevention
5	SRS	Software Requirement Specification
6	GHz	GigaHertz
7	RDBMS	Relational Database Managment System
8	DBMS	Database Managment System
9	ER	Entity Relationship
10	DFD	Data Flow Diagram

LIST OF ABBREVATIONS

SHOP N SWIFT 1 INTRODUCTION

1.1 INTRODUCTION

Electronic home appliances has become inevitable part of our homely needs. It has become almost impossible to fulfill our daily needs without them. This shows us, how important is it for us to learn about them, give them proper care and provide them proper assistance when required. The pandemic has made people difficult to deal with problems and malfunctioning of such home appliances. It is not easy to get expert assistance and spare parts. The assistance they find are often exploitative, non-trustworthy and unsafe, especially during the pandemic. The objective of this project is to provide them proper, trustworthy, non-exploitative, safe and secure assistance. The web app will provide easy and user friendly solutions for specific home appliances. Priority will be given to the methods which can be dealt by the user themselves. Also, the web app will be providing expert assistance if required. The experts in the web app will be properly screened and evaluated.

1.2 PURPOSE OF THE PROJECT

The purpose of the project is make easy shopping. Currently for purchasing the products from the shops the user need more time to pay the bill because of the reason of long queue, due to this the customers can loss the time in the waiting problem. This project intended to make a smart shopping application. As well as better role for customer. There is no role for client for an existing system .Also for faster transaction and less man power consumption. This system will reduce the time. There is no waiting problem for customer. This is a novel method of collaborating ease in online shopping and the sense of security money wise as well as for customer satisfaction while doing shopping offline. It provides methods to change the quantity of product/s purchased and edit the list. Along with this the customer would be informed about the on-going offers in the store. Payment can be according to customer convenience.

1.3 SCOPE OF THE PROJECT

The system has the capability for easy integration with other systems. Currently for purchasing the products from the shops the user need more time to pay the bill because of the reason of long queue, due to this the customers can loss the time in the waiting problem. This project intended to make a smart shopping application. I have found that there must be a need for application which can help the users to select the products by QR scanning; each product is added to cart and also view the bill. Here we are implementing this system to make offline shopping easier. Scan to cart mechanism provides customer participation. Provides intelligent guidance based on budget.

2

SYSTEM ANALYSIS

2.1 SYSTEM ANALYSIS

System Analysis is concerned with analyzing, designing, implementing and evaluating information system in our organization. It is carried out to make the system more effective either by modification or by substantial redesign. In system analysis we identify the problem, study the alternative solution and select the most suitable solution, which meet the technical economic and social demands for analysis, various tools such as dataflow diagram, interviews on site observation, questionnaires etc., are used. System analysis process is also called a life cycle methodology since it relates to four significant phases in life cycle of all information system.

They are

1. System Analysis / Study Phase.
2. System Design / Design phase.
3. System Development / Development Phase.
4. Testing and implementation / Operation Phase.

All activities associated with each life cycle phase must be performed managed and documented. So system analysis is the performance, management and documentation of the activities related to the four life cycle phases of a computer based system.

The basic concept behind the project ”ShopnSwift” presents a novel method of collaborating ease in online shopping and the sense of security money wise as well as for customer satisfaction while doing shopping offline. In Offline mode, the customer needs to physically pick up his purchase, carry cash, credit/debit cards along with them and wait in the long queue to make payments.

2.2 STUDY PHASE

A detailed study to determine whether, to what extent, and how automatic data-processing equipment should be used; it usually includes an analysis of the existing system and the design of the new system, including the development of system specifications which provide a Systems analysis is a process of collecting factual data, understand the processes involved, identifying problems and recommending feasible suggestions for improving the system functioning. This involves studying the business processes, gathering operational data, understand the information flow, finding out bottlenecks and evolving solutions for overcoming the weaknesses of the system so as to achieve the organizational goals. System Analysis also includes subdividing of complex

process involving the entire system, identification of data store and manual processes. The major objectives of systems analysis are to find answers for each business process:

What is being done?

How is it being done?

Who is doing it?

When is he doing it? Why is it being done?

How can it be improved?

2.3 EXISTING SYSTEM

An Existing system refers to the system that is being followed till now. For an existing system barcodes are often intended for consumer use where using a barcode device, a consumer can take an image of a barcode on a product. In this method the barcode must be read using computer vision techniques and barcode can hold information, it makes this vision task in consumer scenarios unusually challenging. Barcode decoder can give the vision algorithm feedback, and develop a progressive strategy of the product. Consequently, the existing methods are time consuming. There is no role for client and waiting problem for customer.

2.4 PROPOSED SYSTEM

In the proposed system, we are using Multiplexing and Demultiplexing algorithm for recognizes QR code image using smart phones to provide various services that can recognize the authenticity of any product. So QR code verifies products by capturing it through the smart phone, and then decodes the item. The user will scan the item which he wants to purchase with the help of scanner provided by this app. After scanning of the item a web service will get called which will create a connection with the database of the shop. As the connection is established, the user is now synched with the database and information related to that item is provided to him. In this whole procedure the overall time of scanning of individual items is saved and thus reducing the time of the shopping. The proposed system make easy shopping, better role for customer, faster transaction and less man power consumption.

2.5 FEASIBLITY STUDY

A feasibility study aim is to objectively and rationally uncover the strengths and weakness of an existing business or proposed venture opportunities and threats present in an environment. The resources required to carry through, and ultimately the prospects for success. In simplest terms, the two criteria to judge feasibility are cost required and value to be attained. The conducted feasibility study and found out that the proposed system is feasible. Feasibility study is to find whether the proposed system is feasible in all aspects compared to existing system. Feasibility

study may be documented as a report to higher effective of the top level management and can be included as an appendix to the system specification feasibility and risk analysis is related in any ways.

2.5.1 TECHNICAL FEASIBILITY

The technical feasibility centers on the existing system and what extend it can support the proposed addition. The technical feasibility assessment is focused on gaining understand of the propose system. It is an evaluation of hardware and software requirements and how it meets the need of proposed system. It is concerned with using the hardware and software and to what extend it can support the new system. Technical feasibility of a project simply means whether our project is technically feasible or not, in the existing system. The also checks whether all the specifications of our project are feasible. In our case our aim is to develop a user suit application of ShopnSwift.

2.5.2 OPERATIONAL FEASIBILITY

The study mainly focuses on how the system satisfies the user needs. The system should be easy to operate by the user and it is also flexible to the requirements. Operational feasibility is a measure of how well a proposed system solves the problem, and taking the advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of the system development.

2.5.3 ECONOMICAL FEASIBILITY

The economic feasibility is the most important and frequently used method for evaluating the effective-ness of the proposed system. It is very essential because the main goal of the proposed system is to have economically better result along with increased efficiency. Cost benefit analysis is usually performed for this purpose. The innovation of the system has much influence on the economical side of the user. The new system can perform more additional features than the existing system.

2.6 MODULE DESCRIPTION

Modules and their description

2.6.1 -ADMIN

2.6.2 -SALES

2.6.3 -STORE

2.6.4 -USER

2.6.1 Functions of ADMIN

Staff Management

Product management

View stock report

View bill reports

View rating

Admin has access to the admin page by using username and password provided for admin. Only admin has the privilege to manage the details of staff that are registered through the system. The other functions of admin are, admin can add and manage products for the shop, admin can view the stock report of sales. Also admin can access or view the bill report generated and admin can also view the rating from users.

2.6.2 Functions of SALES

Verification code entry

Sent transaction and cash entry

View reports

View stock

Sales registration and allocations are maintained by the admin. The sales modules has the functions such as : Sales module can verify the code entry for the products, can manage transactions and cash entry, also the sales module can view the reports and view the stock details.

2.6.3 Functions of STORE

View out of stock product

QR code generation

Product stock updation

Store module has function such as store module can view the out of stock details of product and also can update the stock of product details. Also the store module has given the privilege to generate the QR code for each products.

2.6.4 Functions of USER

Scan product using android application

View bill

Ratings

A new user can register through the android application by providing essential details, later they can login to the application by using the username and password. After the successful login ,the user can then scan the product using the scanner provided in the application to view the product details and can add the desired product to cart. The user can also view the bill details and the user can give the rating for each product.

2.7 PROJECT SCHEDULING

Project planning is the process of defining objectives and scope, goals and milestones (deliverables), and assigning tasks and budgetary resources for each step. A good plan is easily shareable with everyone involved, and it's most useful when it's revisited regularly. Planning is needed to identify desired goals, reduce risks, avoid missed deadlines, and ultimately deliver the agreed product, service or result. The project schedule indicates what needs to be done, which resources must be utilized, and when the project is due. In short, it's a timetable that outlines start and end dates and milestones that must be met for the project to be completed on time.

Project scheduling is just as important as cost budgeting as it determines the timeline, resources needed, and reality of the delivery of the project. Project managers that have experience are better able to properly dictate the tasks, effort and money required to complete a project. Project scheduling provides details such as start and end date of the project, milestones and tasks for the project. In addition it specifies the resources (such as people, equipment, and facilities) required to complete the project and the dependencies of tasks of the project on each other. The purpose of project plan is to define all the technique, procedure and methodologies will be used in the project to assure timely delivery of the software that meets specified requirements within project resources. This will be reviewing and auditing the software products and activities to verify that they comply with the applicable procedure and standards and providing the software project and other appropriate managers with the results of these reviews and audits.

S NO	PHASES	START DATE	END DATE	NO. OF DAYS
1	STUDY PHASE	25-10-2020	31-10-2020	7 DAYS
2	DESIGN PHASE	14-11-2020	26-11-2020	13 DAYS
3	DEVELOPMENT PHASE	05-12-2020	24-12-2020	20 DAYS
4	TESTING	15-012021	19-01-2021	5 DAYS
5	IMPLEMENTATION	03-02-2021	04-02-2021	2 DAY

Table 2.7.1 Project Scheduling

TOTAL NO. OF DAYS: 47 DAYS

SYSTEM REQUIREMENT SPECIFICATION

3.1 INTRODUCTION

The Software Requirements Specification (SRS) document describes all data, functional and behavioral requirements of the software under production or development.

3.2 SOFTWARE SPECIFICATION

A software requirements specification (SRS) is a document that describes what the software will do and how it will be expected to perform. It also describes the functionality the product needs to fulfill all stakeholders (business, users) needs.

- Operating system : Windows 7 or above, Android
- Technology Used : Python
- IDE : PyCharm / Android Studio • Framework : Flask
- Database : MySQL

3.3 HARDWARE SPECIFICATION

Computer hardware specifications are technical descriptions of the computer's components and capabilities. Processor speed, model and manufacturer. Processor speed is typically indicated in gigahertz (GHz). Computer hardware specifications are technical descriptions of the computer's components and capabilities.

- Processor : Pentiun IV or above
- RAM : Min 1 GB
- Hard Disk : 10 GB or more

3.4 TECHNOLOGY SPECIFICATION

This application is based on web as well as mobile. So it is necessary to use a technology which is capable of providing the network facilities to the application. In this application we use HTML as a frontend and MySQL as a backend and for mobile we use android OS.

PYTHON:

Python is an interpreted, object oriented, high level programming language with dynamic semantics. Its high level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for rapid application development as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed. Python's features include –

Easy-to-learn − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.

Easy-to-read − Python code is more clearly defined and visible to the eyes. Easy-to-maintain − Python's source code is fairly easy-to-maintain.

A broad standard library − Python's bulk of the library is very portable and crossplatform compatible on UNIX, Windows, and Macintosh.

Interactive Mode − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.

Portable − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.

Extendable − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.

Databases − Python provides interfaces to all major commercial databases.

Scalable − Python provides a better structure and support for large programs than shell scripting.

GUI Programming − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix

FLASK:

Flask is a micro web framework written in Python. It is classified as a micro framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools. Extensions are updated far more regularly than the core Flask program. Some of the features of flask are the following,

Contains development server and debugger

Integrated support for unit testing

RESTful request dispatching

Support for secure cookies (client side sessions)

Unicode-based

Google App Engine compatibility

MySQL:

MySQL is an open-source relational database management system (RDBMS).The MySQL development project has made its source code source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. MySQL is offered under two different editions: the open source MySQL Community Server and the proprietary Enterprise Server. MySQL Enterprise Server is differentiated by a series of proprietary extensions which install as server plugins, but otherwise shares the version numbering system and is built from the same code base.

ANDROID:

Android is an open source and Linux-based Operating System for mobile devices such as smart phones and tablet computers first developed by a Silicon Valley company by the name of Android Inc which Google bought in 2005. On November 5, 2007, the Open Handset Alliance, a consortium of technology companies including Google, device manufacturers such as HTC, Motorola and Samsung, wireless carriers such as Sprint and T-Mobile, and chipset makers such as Qualcomm and Texas Instruments, unveiled itself, with a goal to 9 develop "the first truly o pen and comprehensive platform for mobile devices" and the Android distribution was announced by them. The core Android source code is known as Android Open Source Project

(AOSP) and is primarily licensed under the Apache License. Android is also associated with a suite of proprietary software developed by Google, called Google Mobile Services (GMS) that very frequently comes pre-installed in devices, which usually includes the Google Chrome web browser and Google Search and always includes core apps for services such as Gmail, as well as the application store and digital distribution platform Google Play, and associated development platform. Android includes middleware, libraries and APIs written in C and application software running on an application framework which includes Java-compatible libraries based on Apache Harmony. Android

uses the Dalvik virtual machine with just-in-time compilation to run compiled Java code. The first commercial Android device launched in September 2008.

4

DESIGN PHASE

4.1 INTRODUCTION

A software design is a meaningful engineering representation of some software product that is to be built. A design can be traced to the customer's requirements and can be assessed for quality against predefined criteria.

During the design process the software requirements model is transformed into design models that describe the details of the data structures, system architecture, interface, and components. Each design product is reviewed for quality before moving to the next phase of software development

Design forms a blue print of the system and adds how the components relate to each other. The design phase proceeds accordingly to an ordinary sequence of steps, beginning with review and assignment of task and ending with package design. Design phase is the life cycle phase in which the detailed design of the system selected in the study phase is accomplished. A smooth transition from the study phase to design is necessary because the design phase continues the activities in the earlier phase. The first step in the design phase is to design the database and then input and output within predefined guidelines.

4.2 ARCHITECTURAL DESIGN

Architecture is an overall structure of a system. Architecture takes into consideration the overall working of the system. Large system can be decomposed into sub-systems that provide some related set of services. The initial design process of identifying these sub-systems and establishing a framework for sub-system control and communication is called architecture design.

Architecture design usually comes before detailed system specification. Architecture decomposition is necessary to structure and organize the specification. There is no generally accepted process depends on application knowledge and on the skill and intuition of the system architect**.**

4.2.1 DATA FLOW DIAGRAM

Data Flow Diagram is the graphical description of the system's data and how the processes transform the data. Data Flow diagram depicts information flow, and the transform that are applied as data move from the input to output. It is the starting point of the design phase that functionally decomposes the requirement specifications down to the lowest level of details. Thus a DFD describes what data flows (logical) rather than how they are processed.

Unlike detailed flowchart, Data Flow Diagrams do no supply detailed description of the modules but graphically describes a system's data and how the data interacts with the system. A DFD consists of a series of bubble joined by lines. The bubble represents data transformation and lines represent data flow in the system. So DFD is also called Bubble Chart.

BASIC DFD SYMBOLS:

To Construct a Data Flow Diagram with the help of following symbols.

- Arrow
- Circles
- Open End Box
- Squares

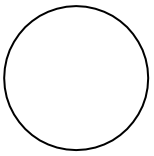
ARROW

An arrow identifies the data flow in motion. It is a pipeline through which information is flown like the rectangle in the Flow Chart.

Data may flow a source to a processor and from a data store or process. An arrow line depicts the flow, with the arrowhead pointing in the direction of flow.

CIRCLE

Circle stands for process that converts the data into information



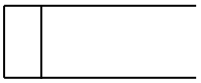
A process represents transformation where incoming data flow is changed into outgoing flows.

RECTANGLE

A Rectangle defines a source or destination of system data. A source is a person or a part of organization, which enters or receives information from the system but is considered to be outside the context of the data flow model.

OPEN END BOX

An Open End Box represents a data store, data at rest or temporary reposition of data.



A graphical picture of the logical steps and sequence involved in a procedure or a program is called a flow chart. Unlike detailed flow chart, Data Flow Diagram does not supply detailed description of the modules but graphically describes a system's data how the data interact with the system.

SIX RULES FOR CONSIDERING DATA FLOW DIAGRAM

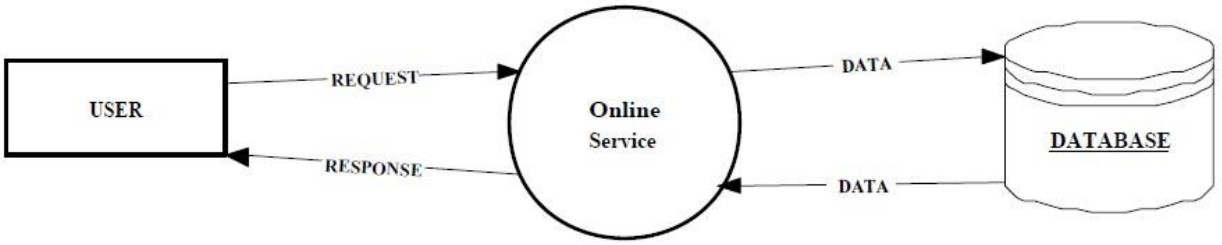
- Arrows should not cross each other
- Squares, circles and Data Store must have names
- Decomposed data flow squares and circles can have the same names.
- Choose meaningful names for data flow
- Draw all data flows around the outside of the diagram.
- Control information such as record count, password and validation Requirement is not relevant to Data Flow Diagram.

DIFFERENT LEVELS OF DATA FLOW DIAGRAMS:

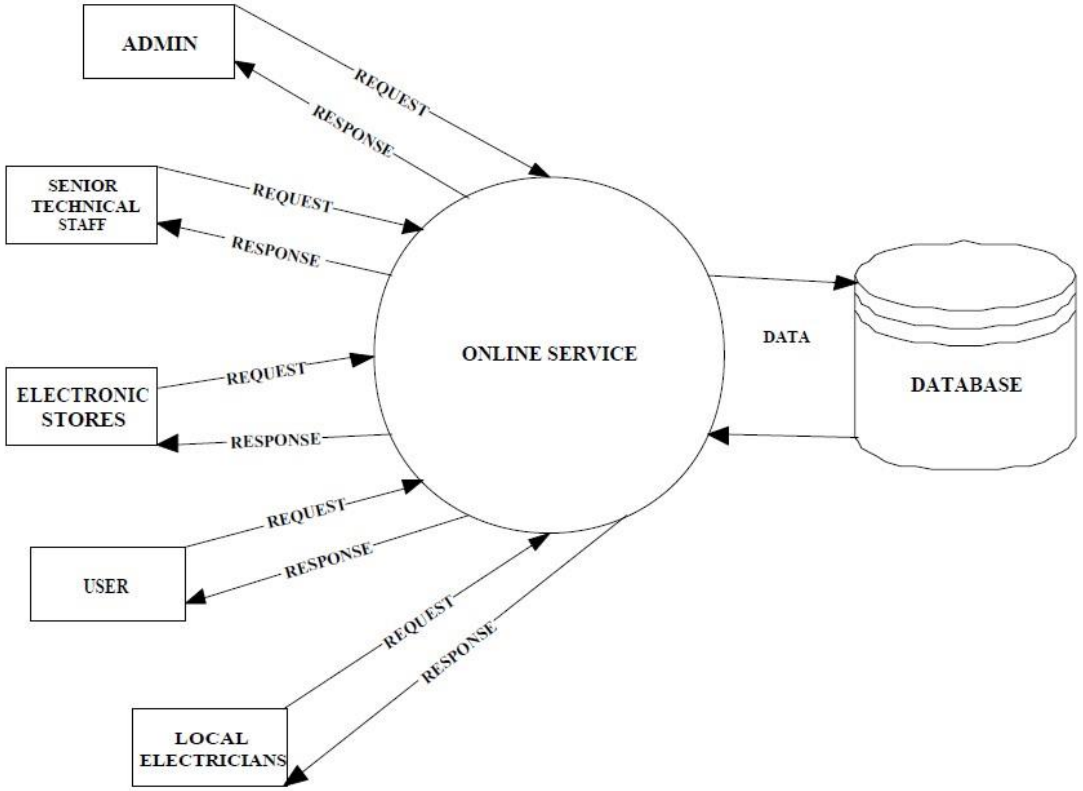
A dataflow diagram as a diagram that depicts data source, data sinks, data storage and processes performed on data as nodes, and logical flow of data as links between the nodes. Generally DFD are used as a design notation to represent architectural design and top level design specification.

DFD represent the system in hierarchical manner with the one top level and many lower level diagrams with each representing separate parts of the system. Since diagrammatic representations are easier to interpret as compared to the technical descriptions, the non-technical users can also understand the system details clearly.

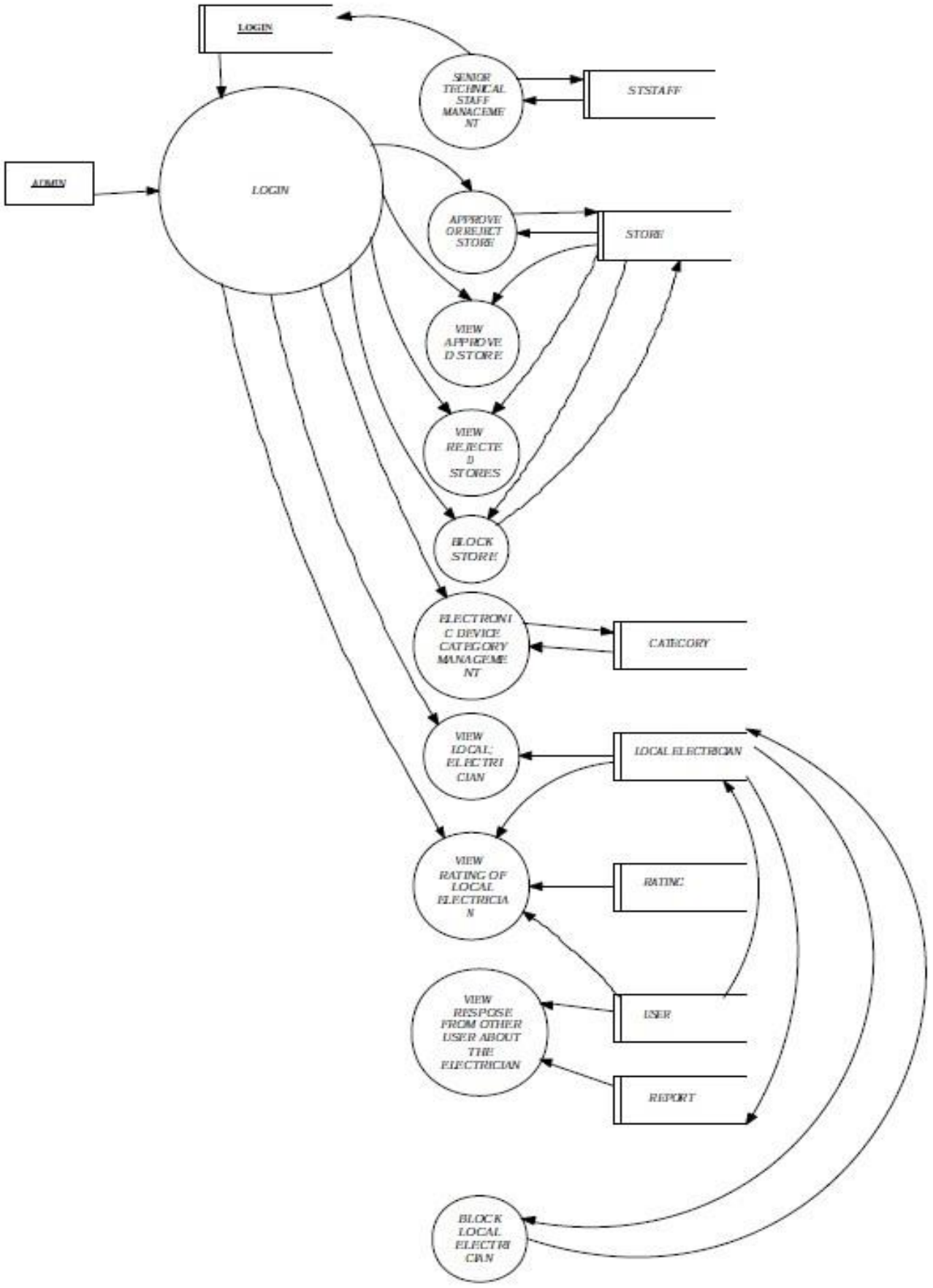
LEVEL 0



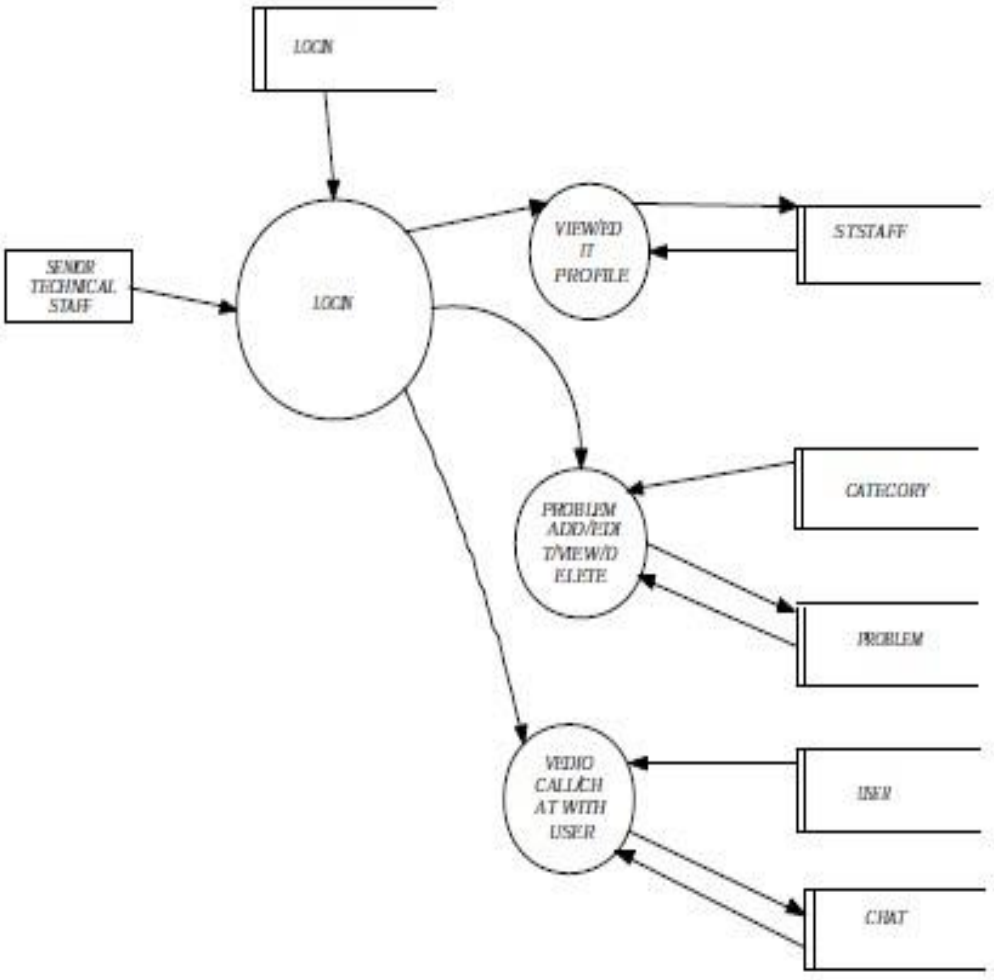
LEVEL 1



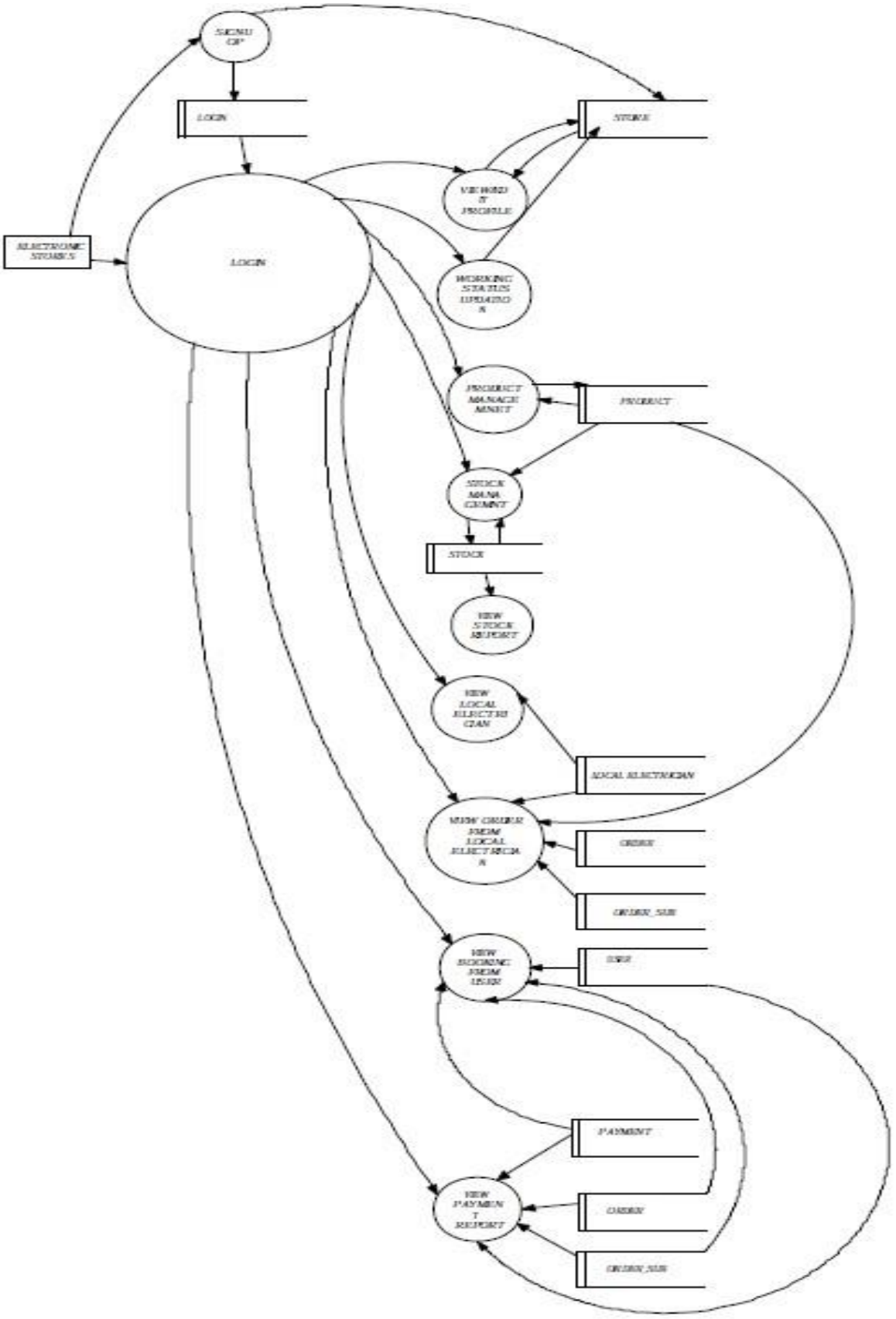
LEVEL 2


LOGIN
ADMIN
LOGIN
SENIOR TECHNICAL STAFF MANAGEMENT
STSTAFF
APPROVE OR REJECT STORE
STORE
VIEW APPROVED STORE
VIEW REJECTED STORES
BLOCK STORE
ELECTRONIC DEVICE CATEGORY MANAGEMENT
CATEGORY
VIEW LOCAL ELECTRICIAN
LOCAL ELECTRICIAN
VIEW RATING OF LOCAL ELECTRICIAN
RATING
VIEW RESPOSE FROM OTHER USER ABOUT THE ELECTRICIAN
USER
REPORT
BLOCK LOCAL ELECTRICIAN

LEVEL 3

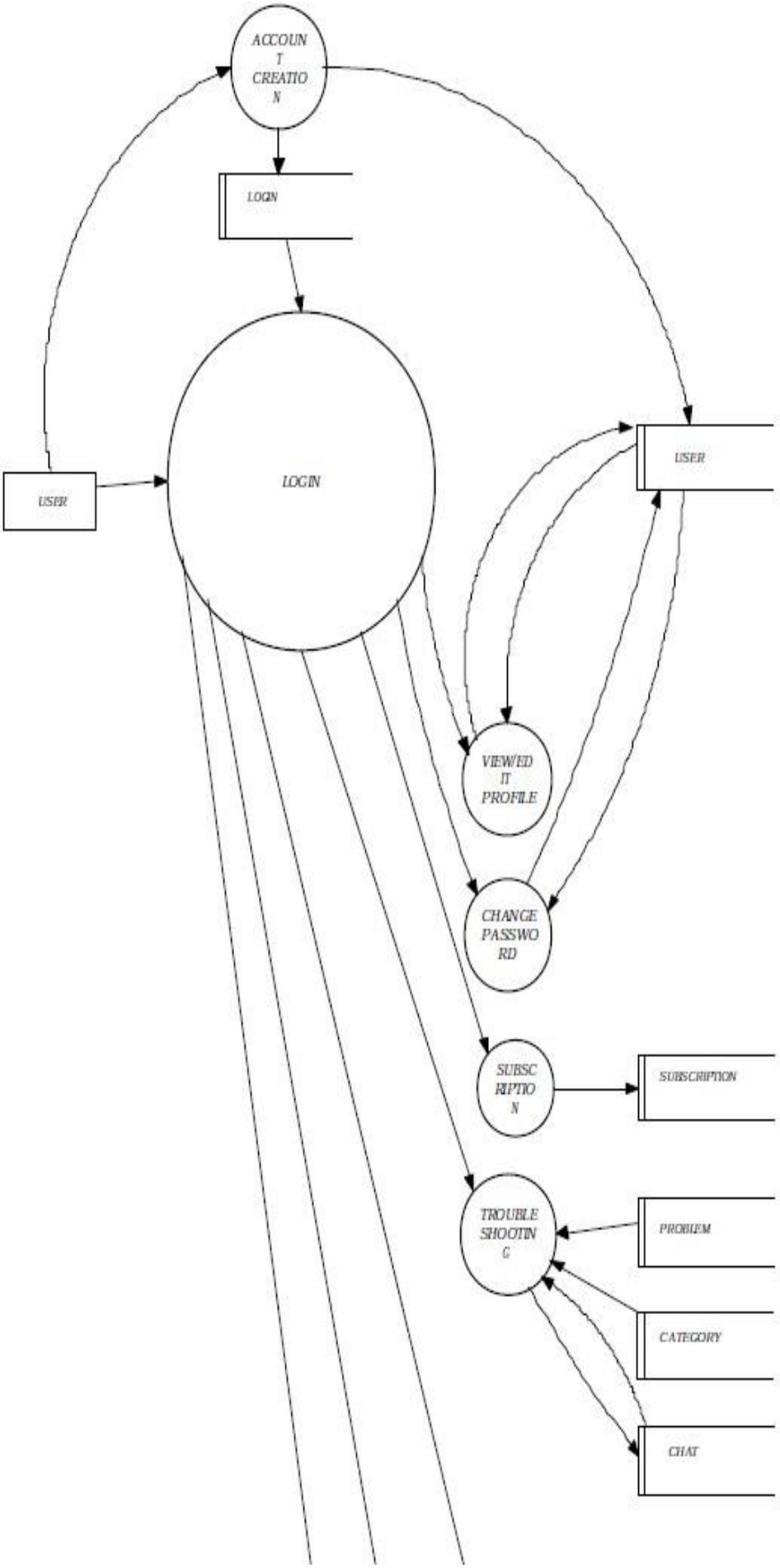

LOGIN
SENIOR TECHNICAL STAFF
LOGIN
VIEW/ED IT PROFILE
STSTAFF
PROBLEM ADD/EDIT/VIEW/DELETE
CATEGORY
PROBLEM
VEDIO CALL/CHAT WITH USER
USER
CHAT

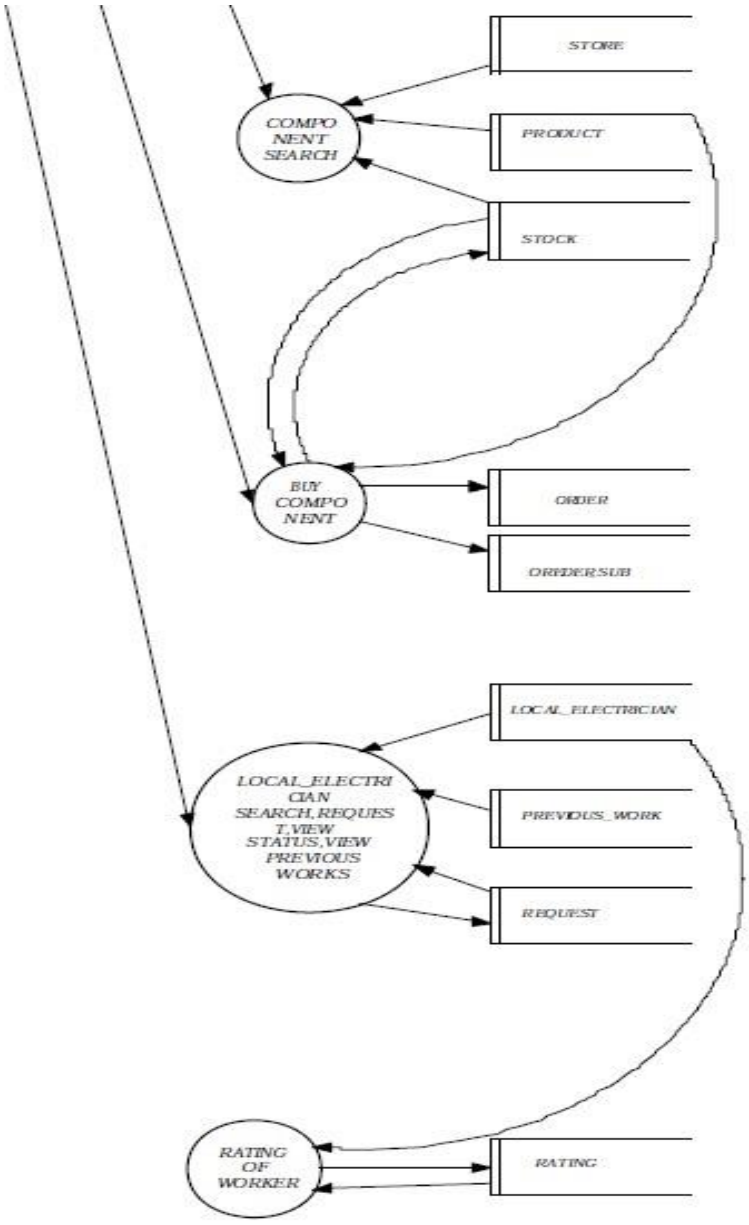
LEVEL 4


SIGNU
OF
LOGIN
STOCK
VECHILD
E
PROFILE
ELECTRONIC
STOCKS
LOGIN
WORKING
STATUS
UPDATIO
N
PROJECT
MANAGE
MENT
PROJECT
STOCK
MAIN
GEMNT
STOCK
NEW
STOCK
REPORT
NEW
LOCAL
ELECTRI
CIAN
LOCAL ELECTRICIAN
NEW ORDER
FROM
LOCAL
ELECTRICIA
N
ORDER
ORDER_SUB
NEW
BOOKING
FROM
USER
USER
PAYMENT
NEW
PAYMEN
T
REPORT
ORDER
ORDER_SUB

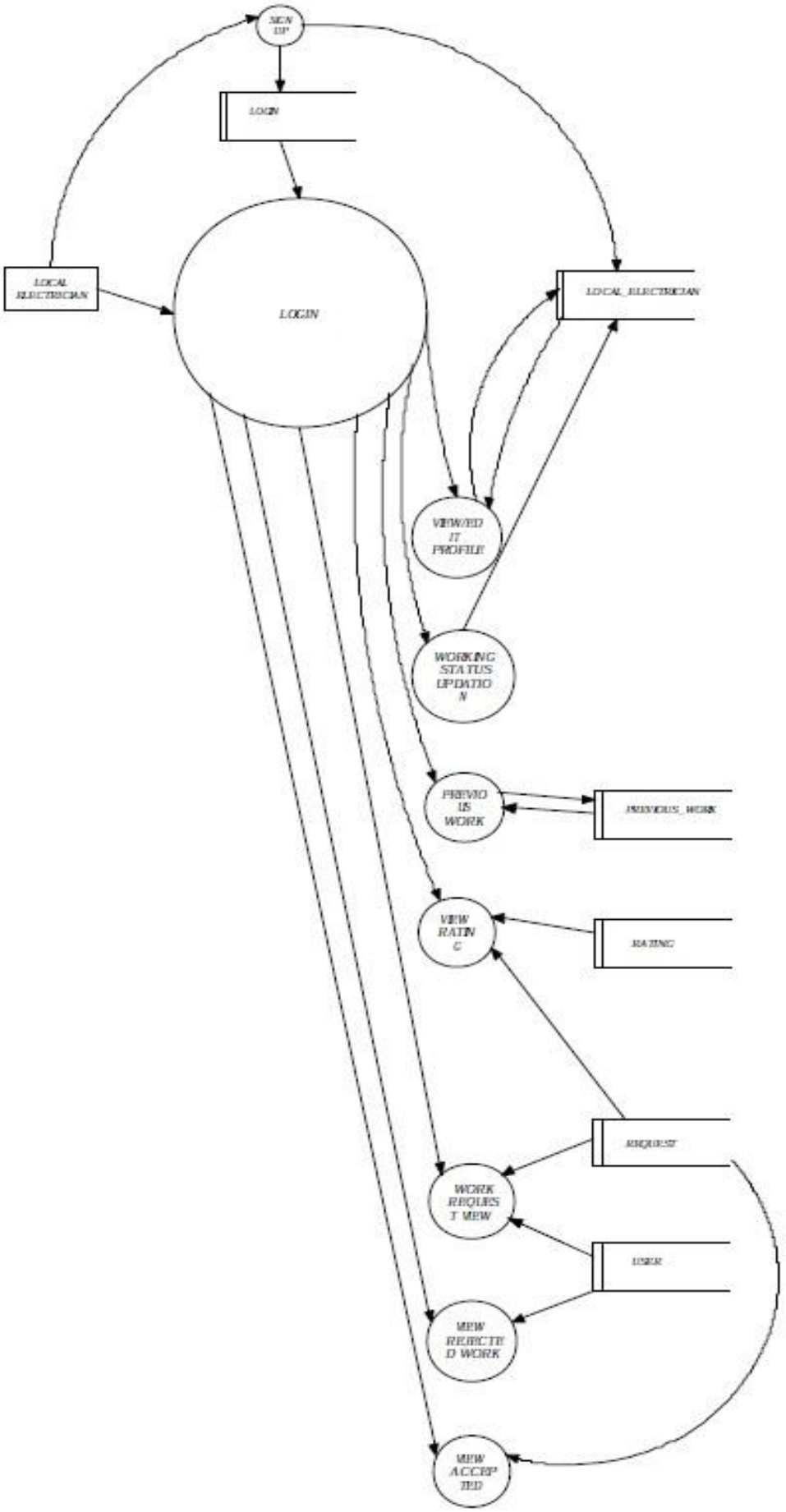
LEVEL 5

LEVEL 6


ACCOUNT CREATION
LOGIN
USER
LOGIN
USER
VIEWED IT PROFILE
CHANGE PASSWORD
SUBSCRIPTION
SUBSCRIPTION
TROUBLE SHOOTING
PROBLEM
CATEGORY
CHAT


STORE
COMPO
NENT
SEARCH
PRODUCT
STOCK
BUY
COMPO
NENT
ORDER
ORDERSUB
LOCAL_ELECTRICIAN
LOCAL_ELECTRI
CIAN
SEARCH,REQUES
T,VIEW
STATUS,VIEW
PREVIOUS
WORKS
PREVIOUS_WORK
REQUEST
RATING
OF
WORKER
RATING

LEVEL 6


SIGN UP
LOGIN
LOCAL ELECTRICIAN
LOGIN
LOCAL_ELECTRICIAN
VIEWED IT PROFILE
WORKING STATUS UPDATIO N
PREVIO US WORK
PREVIOUS_WORK
VIEW RATIN C
RATING
REQUEST
WORK REQUES T VIEW
USER
VIEW REJECTE D WORK
VIEW ACCEP TED

4.2.2.1 INPUT DESIGN

Input design is the process of converting the user-originated inputs to a computer based format. The design for handling input specifies how data are accepted for computer processing. Input design is a part of overall system design that needs careful attention and if includes specifying the means by which actions are taken.

A system user interacting through a workstation must be able to tell the system whether to accept input produce a report system design or end processing. The collection of input data is considered to be the most expensive part of the.

Since the inputs have to be planned in such a manner so as to get the relevant information extreme care is taken to obtain the information. If the data going into the system is incorrect then processing and outputs will magnify these errors.

The major activities carried out are

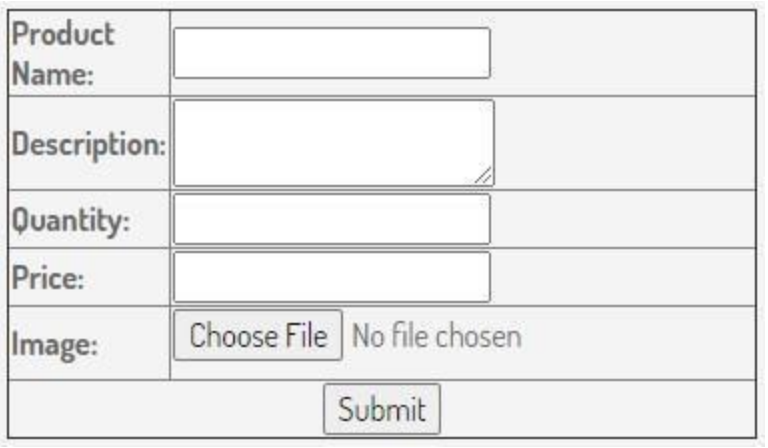
- Collection of needed data from the source.
- Conversion of data into computer application form.
- Verification of converted data.
- Checking data for accuracy.

The goal of designing input data is to make the automation as easy and free from errors as possible. For providing a good input design for the application easy data input and selection features are adopted. The input design requirements such as user friendliness, consistent format and interactive dialogue for giving the right message and help the user at right time are also considered also considered for the development of this project.

The following are the input of the system:

a) LOGIN SCREEN

b) ADD PRODUCT FORM



c) STAFF REGISTRATION FORM

4.2.2.2 OUTPUT DESIGN

The output design is done so that the result of processing could be committed to the user and to provide a hard copy of these results and evaluations for later consultations.

Effective output design will improve the clarity and performance of outputs. Output design phase of the system is concerned with the convergence of information's to the end user-friendly manner. The output design should be efficient, intelligible so that system relationship with the end user is improved and there by enhancing the process of decision making.

There are three types of outputs

- External output.
- Internal output
- Operational output.

External Output

These outputs are prepared and given outside the organisation. That is modified database.

Internal Output

These outputs are prepared exclusively for the people inside the organisation.

Operational Output

These outputs are used purely within the members of computer department itself. The output us the primary purpose of any system. The output design is discussed before other aspects of the design because the output is based on the client point of view.

The following are the major output screens used for web portal for public services:

a) VIEW PRODUCT

PRODUCT	DESCRIPTION	PRICE	QUANTITY	IMAGE	
					DELETE
					DELETE
					DELETE

b) VIEW STAFF

FIRST NAME	LAST NAME	EMAIL	PHONE	GENDER	
					DELETE
					DELETE
					DELETE

c) VIEW STOCK

PRODUCT NAME	DESCRIPTION	PRICE	QUANTITY	IMAGE

d) VIEW RATING

USERNAME	PHONE	RATING

e) VIEW BILL REPORT

BILL NO	TOTAL	DATE	
			VIEW DETAILS
			VIEW DETAILS
			VIEW DETAILS

f) STOCK UPDATION

PRODUCT NAME	DESCRIPTION	PRICE	QUANTITY	
				UPDATE
				UPDATE
				UPDATE

DATABASE DESIGN

Database design runs in parallel with the application design. As we collect information about what is to be done, we will obviously collect information about what data needs to be entered, stored, message and printed on reports. One of the main tasks in building a new system is determining the contents and structure of the database. The type of retrieval and recording required by the user and availability of inputs determine what data has to be stored. The database is a collection of related data.

A comprehensive database should contain all the information necessary to manage the enterprise. Database can be implemented directly using file management program or a database management system. The objectives in establishing a database can be summarized as:

- Eliminate redundant data as much as possible.
- Integrate data files.

- Incorporate changes easily and quickly.
- Lower the cost of storing and retrieving data.
- Improve accuracy, integrity and consistency.
- Data independence.
- Privacy and security.

The various steps involved in database design are given below:

4.2.3.1 Organizing the data

As we gather the data, we will organize it into tables and begin to work out its connections. Our first goal is to identify the tables and the columns within the tables. **4.2.3.2 Identify the key**

The next step in database design is to identify the key that is used as a primary key. A primary key is a column, which uniquely identifies a row in a table.

Uniqueness is the key property of a primary key, so we have to choose it carefully. Then identify the foreign keys that are used. A foreign key is a key that is a column in a table, which acts as a primary key for another table. So after identifying the primary keys and the foreign keys, go to the next step of database design.

SQL

SQL is characterized as a free, fast, reliable open source relational database. It does lack some sophistication and facilities, but it has an active development team and, as it goes from release to release, more capabilities are added. At certain times there will be a trade- off between speed and capabilities, and the SQL teams intend to keep their database engine fast and reliable. It has a good feature set.

SQL is a multithreaded, multi-user, SQL Database Management System (DBMS) with more than six million installations.

4.2.3.3 NORMALISATION

Data structuring is refined through the process called normalization. The basic objective of normalization is to reduce the data redundancy, which means that information is stored only once. There are several normal forms, they are

FIRST NORMAL FORM

A relation is said to be in first normal form if and only if all the attribute values are atomic.

In the First normal form;

- All the key attributes are defined.
- There are no repeating groups in the table. In other words, each row/column intersection can contain one and only one not a set of values
- All the attributes are dependent on the primary key.

SECOND NORMAL FORM

To be in second normal form a table must be in first normal form and no attribute of the table should be functionally dependent on any part of the primary key A table is in second normal form if:

- It is in 1 NF
- It include no PARTIAL DEPENENCIES; that is no attribute is dependent on a portion of the primary key

FIELDNAME	DATATYPE	SIZE	CONSTRAINTS
Id	Int		Primary key
User name	Varchar	50	
Password	Varchar	50	
Type	Varchar	50	

THIRD NORMAL FORM

To be in third normal form a table must be in second normal form and no attribute of the table should be transitively functionally dependent on the primary key.

- It is in 2NF
- It contains no transitive dependencies.

4.2.4 TABLE DESIGN & DESCRIPTION

1) Table name: Login

Description: Used to store login details

FIELD NAME	DATATYPE	SIZE	CONSTRAINTS

login_id	Int		Primary Key, Not Null, Auto Increment
username	Varchar	100	
password	Varchar	100	
user_type	Varchar	100	
price	Varchar	100	

2) Table name: Bill Reports

Description: used to store the bill report

FIELD NAME	DATATYPE	SIZE	CONSTRAINTS
bill_report	Int		Primary Key, Not Null, Auto Increment
user_id	Int		
date	Date		
amount	Int		

3) Table name: Bill item

FIELD NAME	DATATYPE	SIZE	CONSTRAINTS
bill_item_id	Int		Primary Key, Not Null, Auto Increment
bill_report_id	Int		
product_id	Int		
quantity	Int		
price	Int		

Description: used to store the bill

4) Table name : Cart Items

Description: used to store Cart details

FIELD NAME	DATATYPE	SIZE	CONSTRAINTS

id	Int		Primary Key, Not Null, Auto Increment
uid	Int		
pid	Int		
quantity	Int		
price	Int		

5) Table name: Product

Description: used to store Product details

FIELD NAME	DATATYPE	SIZE	CONSTRAINTS
product_id	Int		Primary Key, Not Null, Auto Increment
name	Varchar	100	
description	Varchar	500	
image	Varchar	200	
quantity	Int		
price	Int		

6) Table name: Staff

Description: used to store Staff details

FIELD NAME	DATATYPE	SIZE	CONSTRAINTS
satff_id	Int		Primary Key, Not Null, Auto Increment
login_id	Int		
first_name	Varchar	100	
last_name	Varchar	100	
gender	Varchar	100	
email	Varchar	100	
phone	Bigint	50	

7) Table name: Stock Report

Description: used to store Stock details

FIELD NAME	DATATYPE	SIZE	CONSTRAINTS
stock_id	Int		Primary Key, Not Null, Auto Increment
product_id	Int		
stock	Int		

8) Table name: User

Description: used to store User details

FIELD NAME	DATATYPE	SIZE	CONSTRAINTS
usert_id	Int		Primary Key, Not Null, Auto Increment
login_id	Int		
first_name	Varchar	100	
last_name	Varchar	100	
email	Varchar	100	
phone_number	Bigint	50	
gender	Varchar	100	

9) Table name: Rating

Description: used to store Rating

FIELD NAME	DATATYPE	SIZE	CONSTRAINTS
r_id	Int		Primary Key, Not Null, Auto Increment
user_id	Int		
rating	Varchar	200	
date	Date		

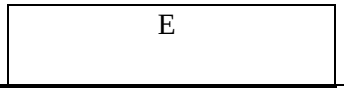
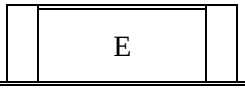
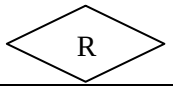
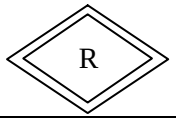
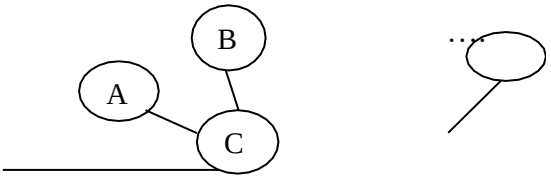
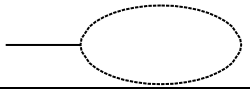
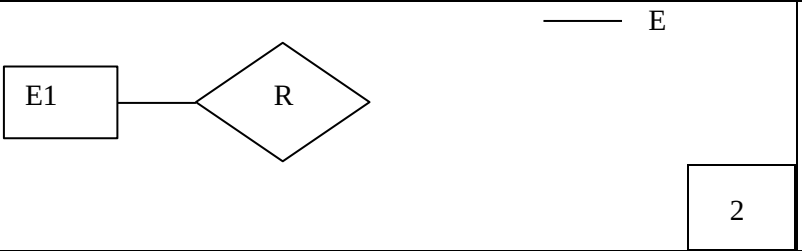
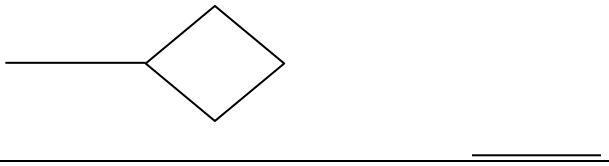
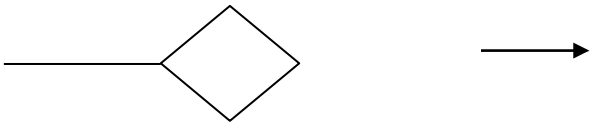
10) Table name: Payment

Description: used to store Payment details

FIELD NAME	DATATYPE	SIZE	CONSTRAINTS
name	Varchar	50	
phone	Bigint	50	Primary Key, Not Null
amount	Int		

4.3 ER DIAGRAM DESCRIPTION

A model is a collection of conceptual tools for describing data, data relationship, data semantics, and consistency constrains. The E-R data model is a high-level conceptual data model that provides the designers with a collection of conceptual tool for describing the structure of a database. The E-R model perceives the real world as consisting of basic objects, called entities, and relationship among these objects. It developed to facilitate the database description by allowing specification of schema that represents the overall logical structure of a database. An ER diagram can overall logical structure of a database graphically. E-R diagrams are simple and clear. An E-R diagram consists of the following major components.

SYMBOL	MEANING
	Entity Set
	Weak Entity
	Relationship Set
	Identifying relationship for weak entity
	Attribute
	Key attribute
	Multivalued attribute
	Composite attribute
	Derived attribute
	Total participation of E2 in R
	One-to-one relationship
	Many-to-one relationship

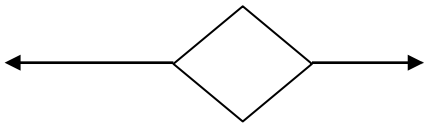
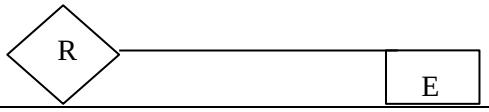
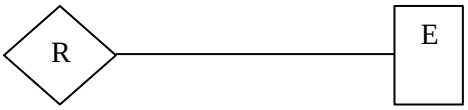
	Many-to-many relationship
1.h 	Cardinality limit
Role name 	Role indicator

Table 4.3.1 ER Diagram And Description

4.4 ER DIAGRAM

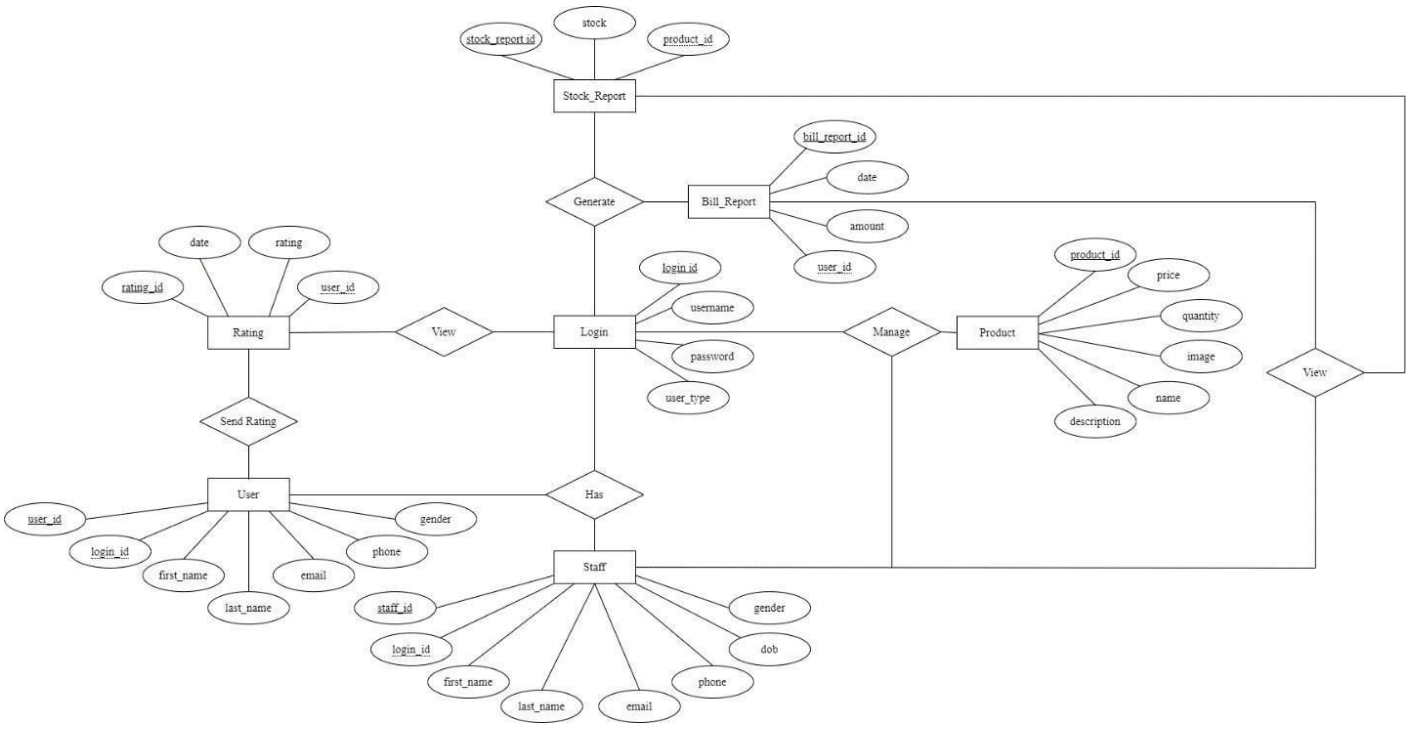


Fig: 4.4.1 ER diagram of Whole System

5

CODING

5.1 CODING METHODOLOGY

This methodology refers to a set of well-documented procedures and guidelines used in the analysis, design, and implementation of programs. Coding methodology includes a diagrammatic notation for documenting the results of the procedure. It also includes an objective set (ideally quantified) of criteria for determining whether the results of the procedure are of the desired quality. The steps to use coding' methodology are listed below.

1. The software development team begins its work by reviewing and understanding the design and requirements specification documents. These documents are essential for understanding user requirements and creating a framework for the software code.

2. In case the software development team is unable to understand user requirements correctly and further clarification is required, the queries are sent back to the user. In addition, the software development team also returns the requirements that are understood by them.

3. After the requirements are clearly understood by the software development team, the design and specifications are implemented in source code, supporting files, and the header files. Note that while writing the software code, the coding style guidelines should be followed. In some cases, there may be a proposal of change in hardware or software specifications. However, the requests for change are implemented only after the approval of the user.

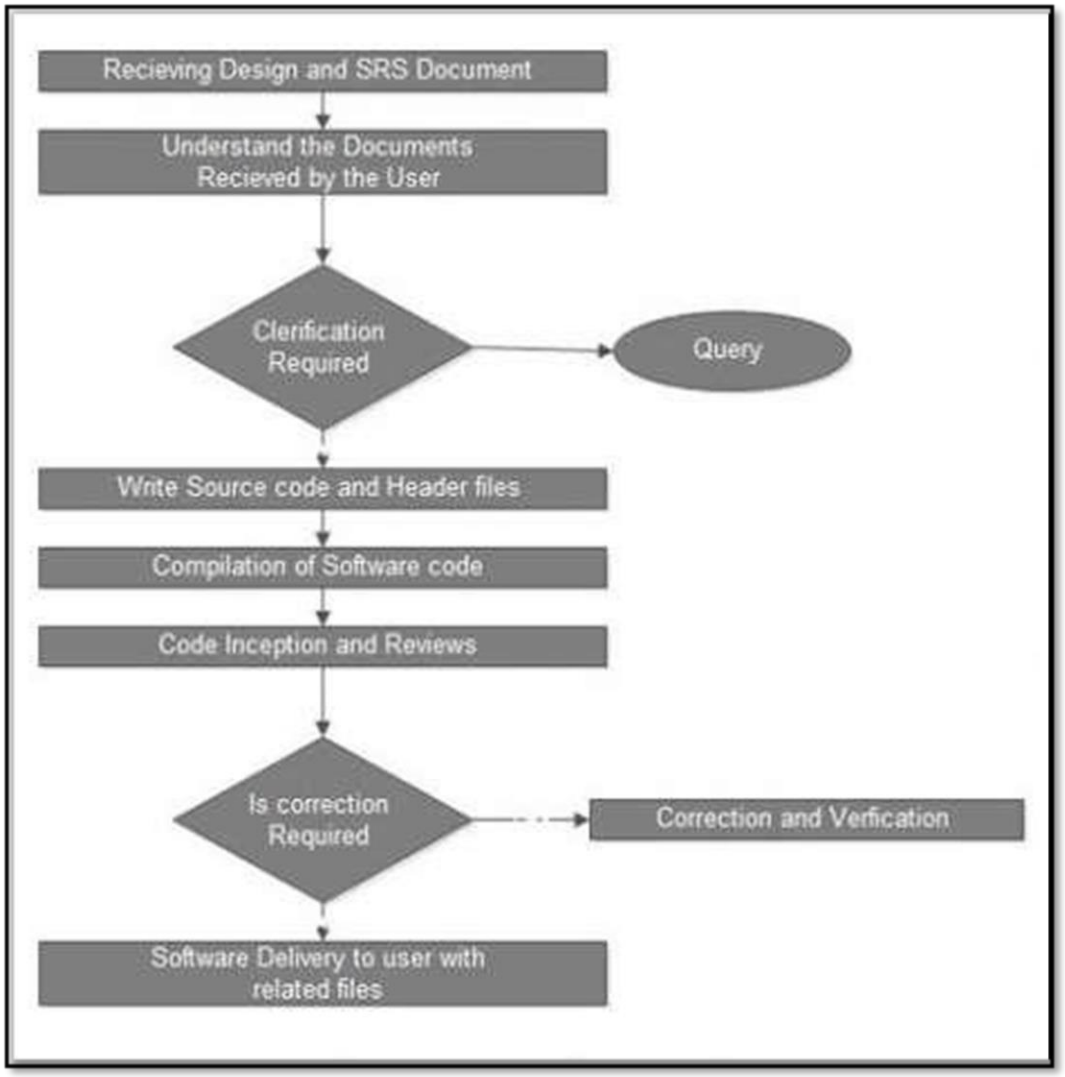
4. When the software code is completely written, it is compiled along with other required files.

5. Code inspection and reviews are conducted after the compilation. These methods are used to correct and verify errors in the software code.

6. Software testing is carried out to detect and correct errors in each module of the software code.

7. After the software code is tested, the software is delivered to the user along with the relevant code files, header files, and documentation files.

In Software Coding Process further change and clarifications are required in the design or SRS, the software development team raises a query, which is sent to the user with the document containing what the software development team understood from the documents sent by the user. Changes are made only when the user has a positive response to the queries raised by the software development team.



FLOWCHART OF SOFTWARE CODING

5.2 CODING GUIDELINES

Writing an efficient software code requires a thorough knowledge of programming. This knowledge can be implemented by following a coding stylewhich comprises several guidelines

that help in writing the software code efficiently and with minimum errors. These guidelines, known as **coding guidelines,** are used to implement individual programming language constructs, comments, formatting, and so on. These guidelines, if followed, help in preventing errors, controlling the complexity of the program, and increasing the readability and understandability of the program.

A set of comprehensive coding guidelines encompasses all aspects of code development. To ensure that all developers work in a harmonized manner (the source code should reflect a harmonized style as a single developer had written the entire code in one session), the developers should be aware of the coding guidelines before starting a software project. Moreover, coding guidelines should state how to deal with the existing code when the software incorporates it or when maintenance is performed.

Since there are numerous programming languages for writing software codes, each having different features and capabilities, coding style guidelines differ from one language to another. However, there are some basic guidelines which are followed in all programming languages. These include naming conventions, commenting conventions, and formatting conventions.

There are certain rules for naming variables, functions and methods in the software code. These naming conventions help software; developers in understanding the use of a particular variable or function. The guidelines used to assign a name to any variable, function, and method are listed below.

1. All the variables, functions, and methods should be assigned names that make the code more understandable to the reader. By using meaningful names, the code can be selfexplanatory, thus, minimizing the effort of writing comments for variables. For example, if two variables are required to refer to 'sales tax' and 'income tax', they should be assigned names such as 'sales Tax' and 'income Tax'.
2. For names, a full description in a commonly spoken language (for example, English) should be used. In addition, the use of abbreviations should be avoided. For example, variable names like 'contact Number' and 'address' should be used instead of 'cno' and 'add'.

3. Short and clear names should be assigned in place of long names. For 'example, 'multiply The Two Numbers' can be shortened to 'multiply Numbers' as it is clear and short enough to be expressed in reasonable length.

In every programming language, there is a different naming convention for variables and constants in the software code. The commonly used conventions for naming variables and constants are listed in Table.

Variable Naming Conventions	Constant Naming Conventions
The variable names should be in camel case letters starting with a lower case letter. For example, use 'total Amount' instead of 'Total Amount'.	All the names of constants should be in upper case. In case the name of constant is too long, it should be separated by an underscore. For example, sales tax rate should be written as 'SALES_TAX_RATE'.
The temporary storage variables that are restricted to a segment of code should be short. For example, the variable 'temp' can be used for a temporary variable. It is important to note that a single temporary variable should not be reused in the same program. For example, variables 'i', j', or 'k' are declared while using loops.	The use of literal should be avoided. Literal numbers such as '15'used in the software code confuses the reader. These numbers are counted as integers and result in wrong output of the program. However, the numbers '0' and '1' can be used as constants.
The use of numbers in naming variables should be avoided. For example, 'First Number' should be used instead of 'Number 1'.	

As with variables and constants, there are some guidelines that should be followed while naming functions in the software code. These conventions are listed below.

- The names of functions should be meaningful and should describe the purpose of the function with clarity and briefness. Like variables, the names should be self-explanatory so that no additional description about the task of that function is required.

- The function name should begin with a verb. For example, the verb 'display' can be used for the function that displays the output on the screen. In case the verb itself is not descriptive, an additional noun or adjective can be used with the verb. For example, the function name 'add Marks' should be used to clarify the function and its purpose.

- In case the function returns a Boolean value, the helping verbs 'is' and 'has' should be used as prefixes for the function name. For example, the function name 'is Deposited' or 'has Deposited' should be used for functions that return true or false values.

Comments are helpful in proper understanding of the code segment used in program. Commenting conventions should be used efficiently to make the code easy to grasp. Generally, two types of commenting conventions are used: file header comments and trailing comments.
File header comments are useful in providing information related to a file as a whole and comprise identification information such as date of creation, Dame of the creator, and a brief description of the software code.

Trailing comments are used to provide explanation of a single line of code. These comments are used to clarify the complex code. These also specify the function of the abbreviated variable names that are not clear. In some languages, trailing comments are used with the help of a double slash (//).The commenting conventions that are commonly followed in the software code are listed below.

- Comments should not be used to include information that is clearly understandable from the software.
- Comments should be used with important segments of code and code segments that are difficult to understand. • Comments should be separated from the code to enhance readability of the software code.

Formatting (way of arranging a program in order to enhance readability) consists of indentation, alignment, and use of white spaces in the program. Consistency plays an important role while formatting a program in an organized way. A program with consistent formatting makes the code easier to read and understand. The commonly used formatting conventions are listed below.

Indentation: This refers to one or more spaces left at the beginning of statements in the program. Indentation is useful in making the code easily readable. However, the spaces used for indentation should be followed in the entire program. The guidelines that are commonly followed while indenting a program are listed below.

White spaces: These improve readability by minimizing the compactness of the code. Some of the guidelines for proper usage of spaces within the code are listed below.

- Indentation should be used to highlight a nested block. Some nested blocks can be made with the help of 'if-else' and 'do-while' loops.
- Indentation is required if the statement is large enough to fit in a single line.
- Indentation should b consistent at the beginning and at the end of the braces in the program.
- There should be a space after placing a comma between two function arguments.
- There should be no space between a function name and parenthesis.
- There should be spaces to align the operators vertically to emphasize program structure and semantics.

Implementing Coding Guidelines

If coding guidelines are used in a proper manner, errors can be detected at the time of writing the software code. Such detection in early stages helps in increasing the performance of the software as well as reducing the additional and unplanned costs of correcting and removing errors. Moreover, if a well-defined coding guideline is applied, the program yields a software system that is easy to comprehend and maintain. Some of the coding guidelines that are followed in a programming language are listed below.

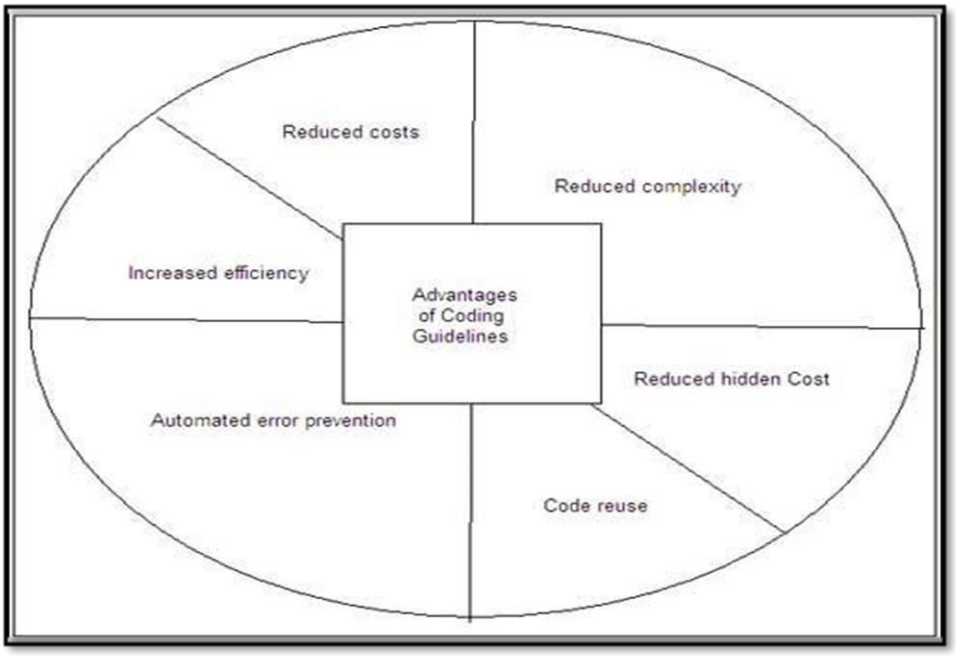
- All the codes should be properly commented before being submitted to the review team.
- All curly braces should start from a new line.
- All class names should start with the abbreviation of each group. For example, AA and CM can be used instead of academic administration and course management, respectively.
- Errors should be mentioned in the following format: [error code]: [explanation]. For example, 0102: null pointer exception, where 0102 indicates the error code and null pointer exception is the name of the error.
- Every 'if statement should be followed by a curly braces even if there exists only a single statement.
- Every file should contain information about the author of the file, modification date, and version information.

Similarly, some of the commonly used coding guidelines in a database (organized collection of information that is systematically organized for easy access and analysis) are listed below.

- Table names should start with TBL. For example, TBL_STUDENT.
- If table names contain one word, field names should start with the first three characters of the name of the table. For example, STU_FIRSTNAME.
- Every table should have a primary key.
- Long data type (or database equivalent) should be used for the primary key.

Advantages of Coding Guidelines

Coding guidelines supplement the language standard by defining acceptable and unacceptable usage of the programming language used. Acceptable usage avoids troublesome situations while unacceptable usage is conducive to errors or leads to misunderstanding of the written code. Properly implemented coding guidelines help the developer to limit program complexity, establish the basis for code review, and guard against compiler and common programming errors. Other advantages associated with coding guidelines are listed below and depicted.



- **Increased efficiency:** Coding guidelines can be used effectively to save time spent on gathering unnecessary details. These guidelines increase the efficiency of the software team while the software development phase is carried out. An efficient software code is fast and economical. Software coding guidelines are used to increase efficiency by making the team productive, thus, ensuring that the software is delivered to the user on time

- **Reduced costs:** Coding guidelines are beneficial in reducing the cost incurred on the software project. This is possible since coding guidelines help in detecting errors in the early stages of the software development. Note that if errors are discovered after the software is delivered to the user, the process of rectifying them becomes expensive as additional costs are incurred on late detection, rework, and retesting of the entire software code.

- **Reduced costs:** Coding guidelines are beneficial in reducing the cost incurred on the software project. This is possible since coding guidelines help in detecting errors in the early stages of the software development. Note that if errors are discovered after the software is delivered to the user, the process of rectifying them becomes expensive as additional costs are incurred on late detection, rework, and retesting of the entire software code.

- **Reduced hidden costs:** Coding guidelines, if adhered to in a proper manner, help to achieve a high-quality software code. The software quality determines the efficiency of the software. Software quality is the degree to which user requirements are accomplished in the software along with conformity to standards. Note that if quality is not considered while developing the software, the cost for activities such as fixing errors, redesigning the software, and providing technical support increases considerably

- **Code reuse:** Using coding guidelines, software developers are able to write a code that is more robust and create individual modules of the software code. The reason for making separate code segment is to enable reusability of the modules used in the software. A reusable module can be used a number of times in different modules in one or more software.

- **Automated error prevention:** The coding guidelines enable Automated Error Prevention (AEP).This assures that each time error occurs in software, the software development activity is improved to prevent similar errors in future. AEP begins with detecting errors in the software, isolating its cause, and then searching the cause of error generation. Coding guidelines are useful in preventing errors as they allow implementation of requirements that prevent the most common and damaging errors in the software code.

6

TESTING

6.1 INTRODUCTION

Software testing is critical element of software quality assurance and represent the ultimate review of the specification, design and coding. System testing makes a logical assumption that all the part of the system is correct; the goal will be successfully achieved.

Testing is a set of activity that can be planned in advance and conducted**.** Systematically, this is aimed at ensuring that the system works accurately and efficiently before live operations commences,

- Testing is the process of correcting a program with intend of finding an error.
- A good test case is one that has high probability of finding a yet undiscovered error. A successful test is one that uncovers a yet undiscovered error.

Testing Objectives

There are several rules that can serve as testing objectives,

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

Testing is vital to the success of the system. System testing makes a logical assumption that if all parts of the system are subject to variety of tests on-line response, volume, stress, recovery and security and usability tests. A series of tests are performed before the system is ready for user acceptance testing.

6.2 TESTING STRATEGIES

- White Box Testing
- Black Box Testing

6.2.1 WHITE BOX TESTING

White box testing is also known as code testing. The code testing strategy checks for the correctness of the every statement in the program. To follow this strategy, there should be cases that result in execution of every instruction in the program or module, which is every path in the program, is tested. The test cases should be guaranteed that independent paths within module are executed once.

- Exercise all logical decision on their true or false sides.
- Execute all loops at their boundaries and within their operational bounds.

This testing strategy, on the face of it, sounds exhaustive. If every statement in the program is checked for its validity, there does not seem to be much scope of error.

6.2.2 BLACK BOX TESTING

Black box testing is also known as specification testing. To perform black box testing, the analyst examines the specification taking what the program or module should do and how it should perform on the various conditions and submitted for processing. By examine the result, the analyst can examine whether the program performs according to the specified requirements.

Testing objectives are:

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

6.3 TYPES OF TESTING

Different types of testing are,

- Unit testing
- Integration testing
- System testing
- Validation testing
- User acceptance testing

6.3.1 UNIT TESTING

Unit testing enables a programmer to detect error in coding. A unit test focuses verification of the smallest unit of software design. This testing was carried out during the coding itself. Among the more common errors in computation are:

- Misunderstood or incorrect arithmetic precedence
- Mixed mode operations.
- Incorrect initialization.
- Precision inaccuracy. • Incorrect symbolic representation of an expression.

This testing was carried out during the coding itself. In this testing step, each module is going to be work satisfactorily as the expected output from the module.

Project aspect:

The front-end design consists of various forms. They were tested for data acceptance. Similarly, the back-end also tested for successful acceptance and retrieval of data.

The test causes are prepared for each module. Here testing are done in order to verifying whether the user can registered to the system successfully or not.

The following are some of the test cases for unit testing,

The test cases for each module tabulated below:

TEST CASE ID	TEST CASE DESCRIPTION	EXPECTED RESULTS	ACTUAL RESULTS
1	Giving Username and Password	Correct Username and Password	Successfull
2	Giving Username and Password	Invalid Username and Password	Failed

Table 6.3.1.1 Test Cases for Admin Module

TEST CASE ID	TEST CASE DESCRIPTION	EXPECTED RESULTS	ACTUAL RESULTS
1	Giving Username and Password	Correct Username and Password	Successfull
2	Giving Username and Password	Invalid Username and Password	Failed

Table 6.3.1.2 Test Case of Sales Module

TEST CASE ID	TEST CASE DESCRIPTION	EXPECTED RESULTS	ACTUAL RESULTS
1	Giving Username and Password	Correct Username and Password	Successfull
2	Giving Username and Password	Invalid Username and Password	Failed

Table 6.3.1.3 Test Case of Store Module

TEST CASE ID	TEST CASE DESCRIPTION	EXPECTED RESULTS	ACTUAL RESULTS
1	Giving Username and Password	Correct Username and Password	Successfull

2	Giving Username and Password	Invalid Username and Password	Failed

Table 6.3.1.4 Test Cases for User Module

6.3.2 INTEGRATION TESTING

Through each program work individually, they should work after linking together. This is referred to as interfacing. Data may be lost across the interface; one module can have adverse effect on the other. Subroutines after linking may not do the desired function expected by the main routine. Integration testing is the systematic technique for constructing the program structure while at the same time conducting test to uncover errors associated with the interface. Using integrated test plan prepared in the design phase of the system development as a guide, the integration test was carried out. All the errors found in the system were corrected for the next testing step.

Project aspect:

After connecting the back-end and the front-end as whole module, the data entered in the front-end once submitted were successfully entered in the database. On request, data were successfully retrieved in to forms.

6.3.3 SYSTEM TESTING

After performing the integration testing, the next step is output testing of the proposed system. No system could be useful if it doesn't produce the required output in a specified format. The output generated are displayed by the system under consideration and then tested by comparing with the format require by the user. Here the output format is considered in to two ways, one in on-screen and other in printed format. **Project aspect:** The entire project was tested and found successful.

6.3.4 VALIDATION TESTING

The user has to work with the system and check whether the project meets his needs. In the validation checking, the user works with the beta version of the software.

Project aspects:

User enters the appropriate data and results was checked and validated Validation checks are performed on the following field: **Text Field:**

The text field can contain only the number of characters lesser than or equal to its size. The text fields are alphanumeric in some tables. Incorrect entry always flashes an error message.

Numeric Field:

The numeric field can contain only numbers from 0-9. An entry of any characters flashes an error message. The individual modules are checked for accuracy and what it has to perform. Each module subjected to test run along with the sample data. The individually tested modules are integrated into single system. Testing involves executing the real data information used in the program. The existence of any program defect is inferred from the output. The testing should be planned so that all the requirements are individually tested.

Email validator:

The email field can contain @ , .com /.in. The email fields are checked for these symbols. The email field does not contain these symbols entry always flashes an error message.

6.3.5 USER ACCEPTANCE TEST

User acceptance of a system is a key factor of the success of any system. The system under consideration was tested for user acceptance by running a prototype of the software. **Project aspects:**

An alpha version is demonstrated to the users. Their suggestions are recorded.

7

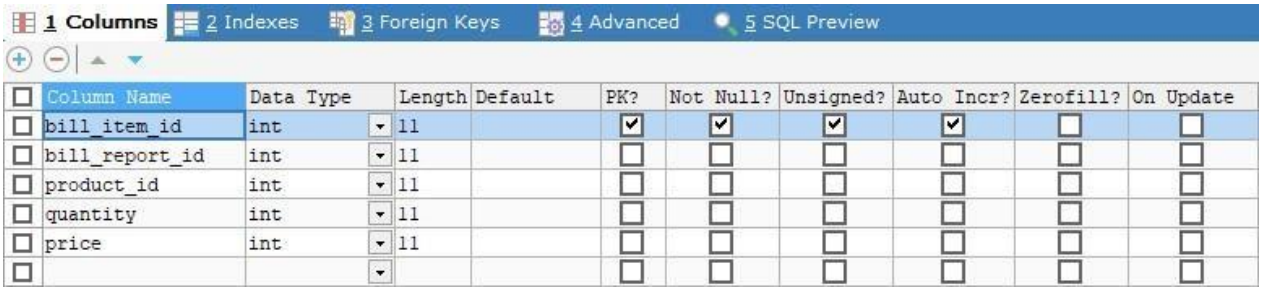
APPENDIX

7.1 SCREEN SHOT

7.1.1 DATABASE DESIGN

TABLES:

7.1.1.1 BILL ITEM TABLE



Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	On Update
bill_item_id	int	11		☑	☑	☑	☑	☐	☐
bill_report_id	int	11		☐	☐	☐	☐	☐	☐
product_id	int	11		☐	☐	☐	☐	☐	☐
quantity	int	11		☐	☐	☐	☐	☐	☐
price	int	11		☐	☐	☐	☐	☐	☐
				☐	☐	☐	☐	☐	☐

7.1.1.2 BILL REPORT TABLE

1 Columns | 2 Indexes | 3 Foreign Keys | 4 Advanced | 5 SQL Preview

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	On Update
bill_report	int	11		☑	☑	☐	☑	☐	☐
user_id	int	11		☐	☐	☐	☐	☐	☐
date	date			☐	☐	☐	☐	☐	☐
amount	int	11		☐	☐	☐	☐	☐	☐
				☐	☐	☐	☐	☐	☐

7.1.1.3 CART TABLE

1 Columns | 2 Indexes | 3 Foreign Keys | 4 Advanced | 5 SQL Preview

Move up (Alt+Up)

Column	Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	On Update
id	int	11		☑	☑	☐	☑	☐	☐
uid	int	11		☐	☐	☐	☐	☐	☐
pid	int	11		☐	☐	☐	☐	☐	☐
quantity	int	11		☐	☐	☐	☐	☐	☐
price	int	11		☐	☐	☐	☐	☐	☐
				☐	☐	☐	☐	☐	☐

7.1.1.4 LOGIN TABLE

1 Columns | 2 Indexes | 3 Foreign Keys | 4 Advanced | 5 SQL Preview

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	On Update
login_id	int	11		☑	☑	☐	☑	☐	☐
username	varchar	100		☐	☐	☐	☐	☐	☐
password	varchar	100		☐	☐	☐	☐	☐	☐
user_type	varchar	100		☐	☐	☐	☐	☐	☐
				☐	☐	☐	☐	☐	☐

7.1.1.5 PAYMENT TABLE

1 Columns | 2 Indexes | 3 Foreign Keys | 4 Advanced | 5 SQL Preview

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	On Update
name	varchar	50		☐	☐	☐	☐	☐	☐
phone	bigint	50		☑	☑	☐	☐	☐	☐
amount	bigint	50		☐	☐	☐	☐	☐	☐
				☐	☐	☐	☐	☐	☐

7.1.1.6 PRODUCT TABLE

1 Columns | 2 Indexes | 3 Foreign Keys | 4 Advanced | 5 SQL Preview

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	On Update
product_id	int	11		☑	☑	☐	☑	☐	☐
name	varchar	100		☐	☐	☐	☐	☐	☐
description	varchar	500		☐	☐	☐	☐	☐	☐
image	varchar	200		☐	☐	☐	☐	☐	☐
quantity	int	11		☐	☐	☐	☐	☐	☐
price	int	11		☐	☐	☐	☐	☐	☐
				☐	☐	☐	☐	☐	☐

7.1.1.7 RATING TABLE

1 Columns | 2 Indexes | 3 Foreign Keys | 4 Advanced | 5 SQL Preview

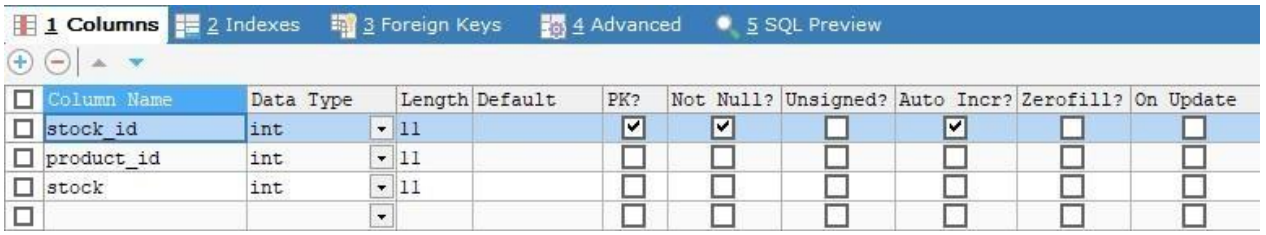
Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	On Update
r_id	int	11		☑	☑	☐	☑	☐	☐
user_id	int	11		☐	☐	☐	☐	☐	☐
rating	varchar	200		☐	☐	☐	☐	☐	☐
date	date			☐	☐	☐	☐	☐	☐
				☐	☐	☐	☐	☐	☐

7.1.1.8 STAFF TABLE

1 Columns | 2 Indexes | 3 Foreign Keys | 4 Advanced | 5 SQL Preview

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	On Update
staff_id	int	11		☑	☑	☐	☑	☐	☐
login_id	int	11		☐	☐	☐	☐	☐	☐
first_name	varchar	100		☐	☐	☐	☐	☐	☐
last_name	varchar	100		☐	☐	☐	☐	☐	☐
gender	varchar	100		☐	☐	☐	☐	☐	☐
email	varchar	100		☐	☐	☐	☐	☐	☐
phone	bigint	50		☐	☐	☐	☐	☐	☐
				☐	☐	☐	☐	☐	☐

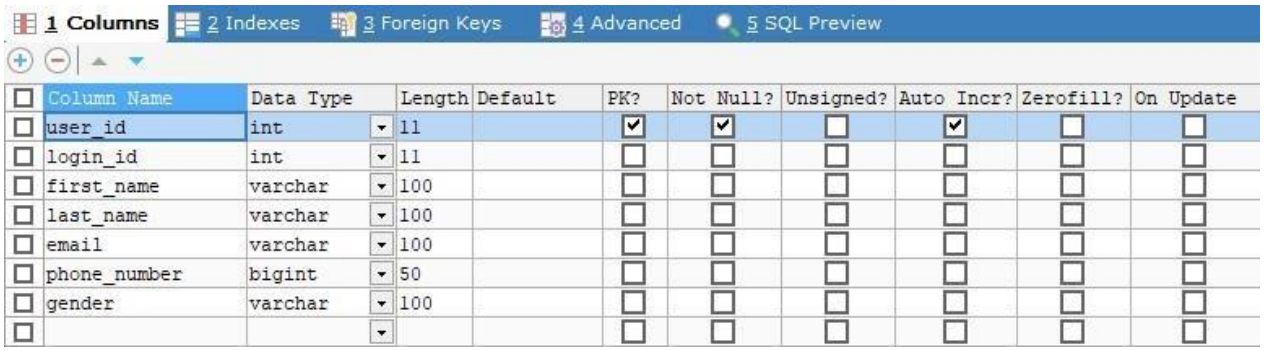
7.1.1.9 STOCK REPORT TABLE



1 Columns | 2 Indexes | 3 Foreign Keys | 4 Advanced | 5 SQL Preview

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	On Update
stock_id	int	11		☑	☑	☐	☑	☐	☐
product_id	int	11		☐	☐	☐	☐	☐	☐
stock	int	11		☐	☐	☐	☐	☐	☐
				☐	☐	☐	☐	☐	☐

7.1.1.10 USER TABLE



1 Columns | 2 Indexes | 3 Foreign Keys | 4 Advanced | 5 SQL Preview

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	On Update
user_id	int	11		☑	☑	☐	☑	☐	☐
login_id	int	11		☐	☐	☐	☐	☐	☐
first_name	varchar	100		☐	☐	☐	☐	☐	☐
last_name	varchar	100		☐	☐	☐	☐	☐	☐
email	varchar	100		☐	☐	☐	☐	☐	☐
phone_number	bigint	50		☐	☐	☐	☐	☐	☐
gender	varchar	100		☐	☐	☐	☐	☐	☐
				☐	☐	☐	☐	☐	☐

7.1.2 SCREEN LAYOUTS

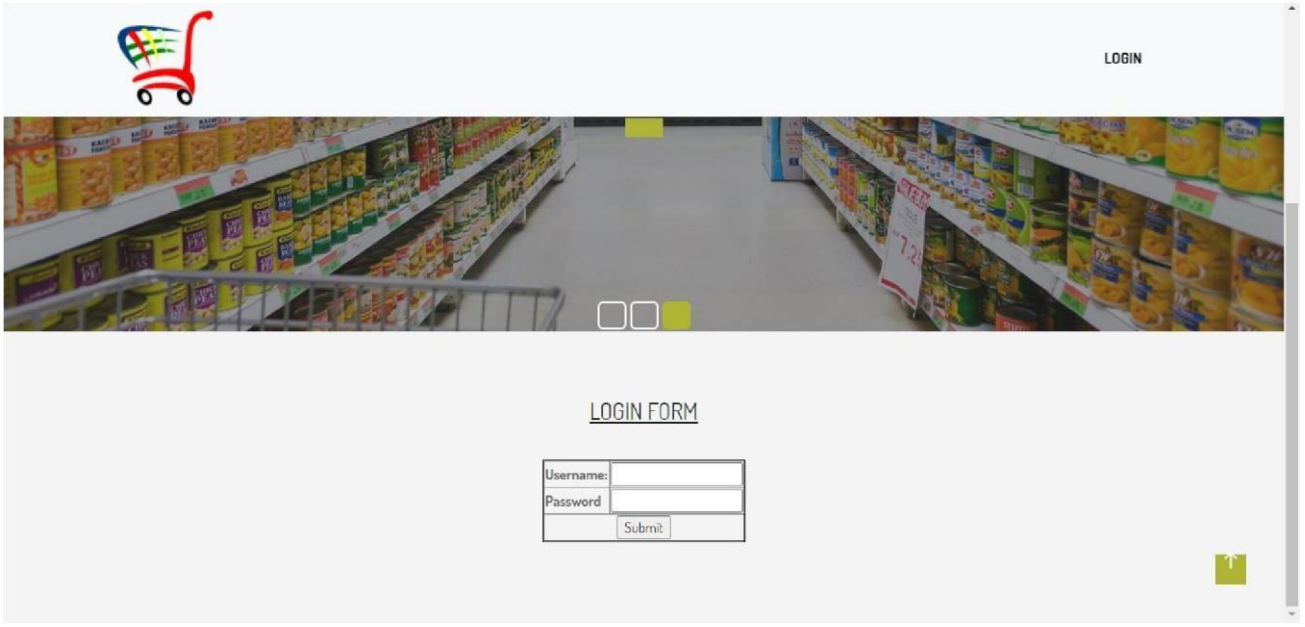


Fig 7.1.2.1 Home Page

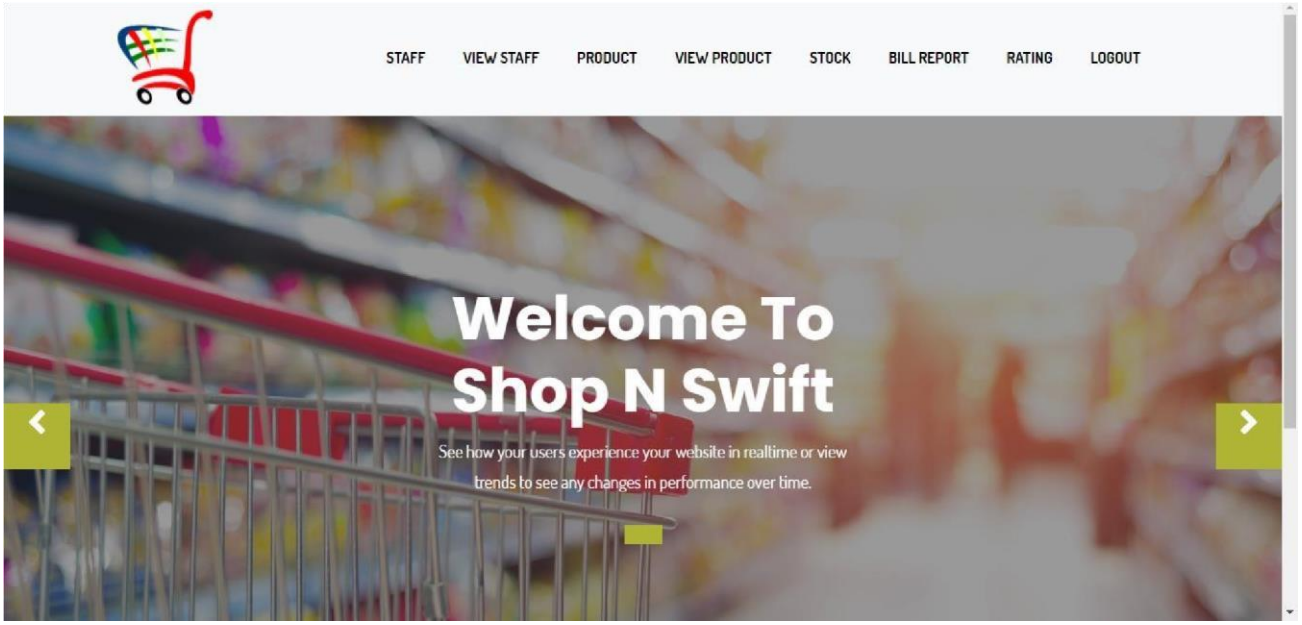


Fig 7.1.2.2 Admin Home Page

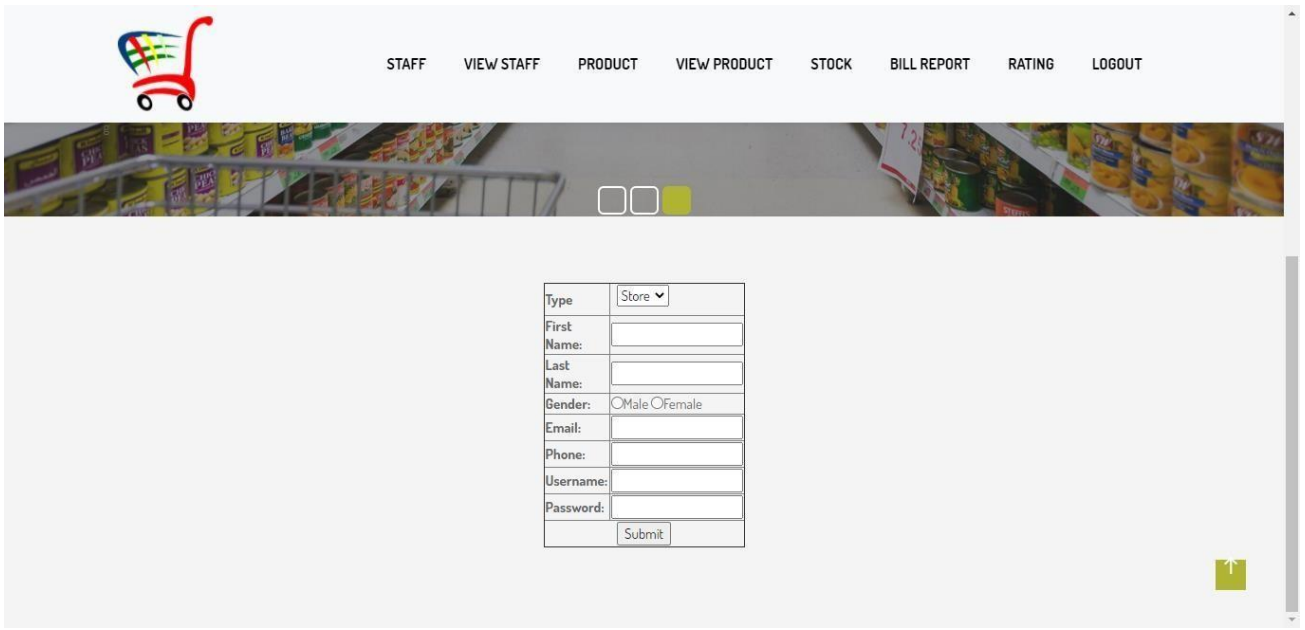


Fig 7.1.2.3 Staff Registration Form

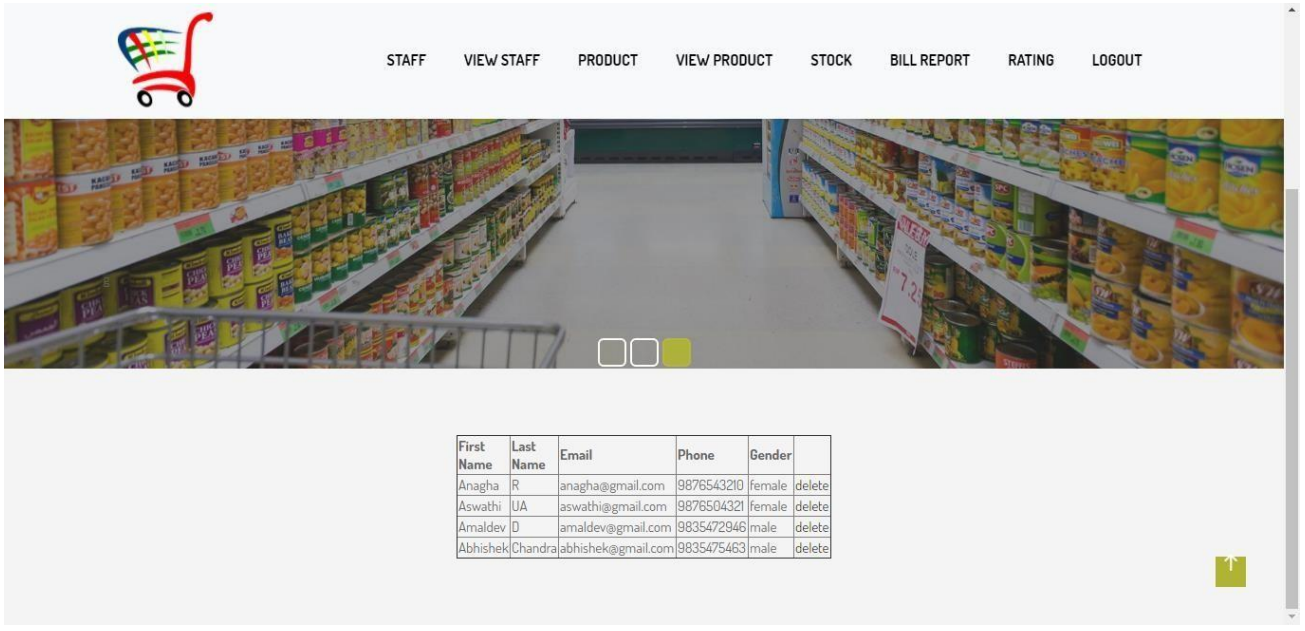


Fig 7.2.1.4 View Staff Form

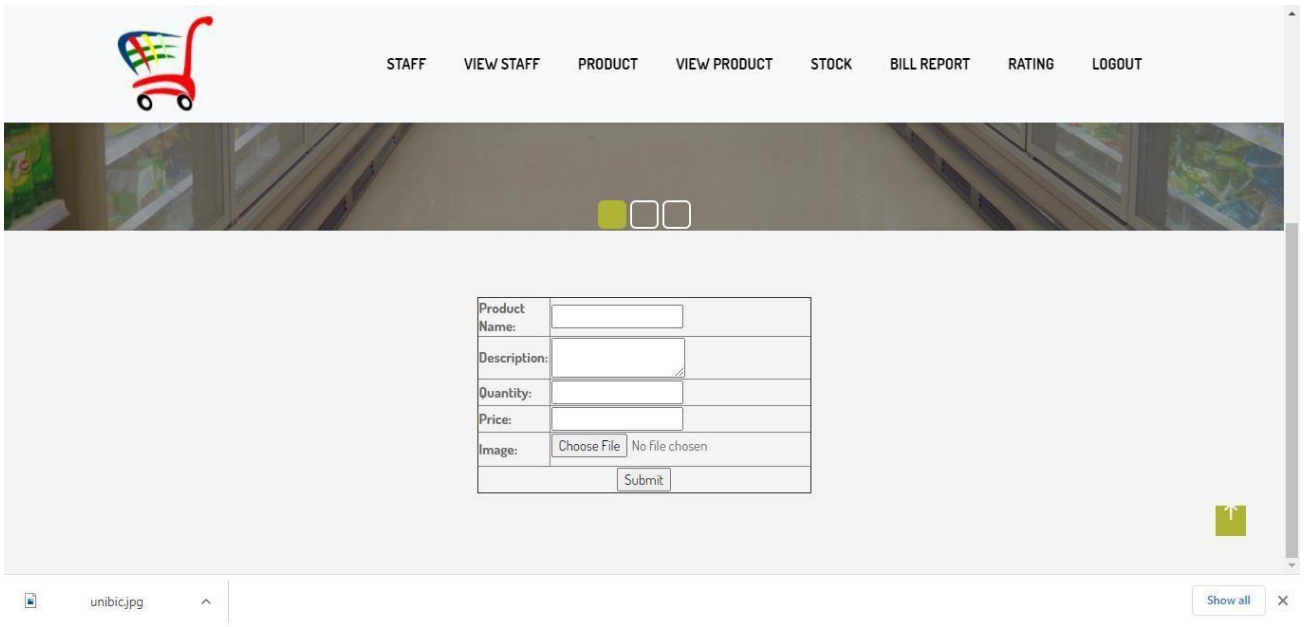


Fig 7.1.2.5 Add Product Form

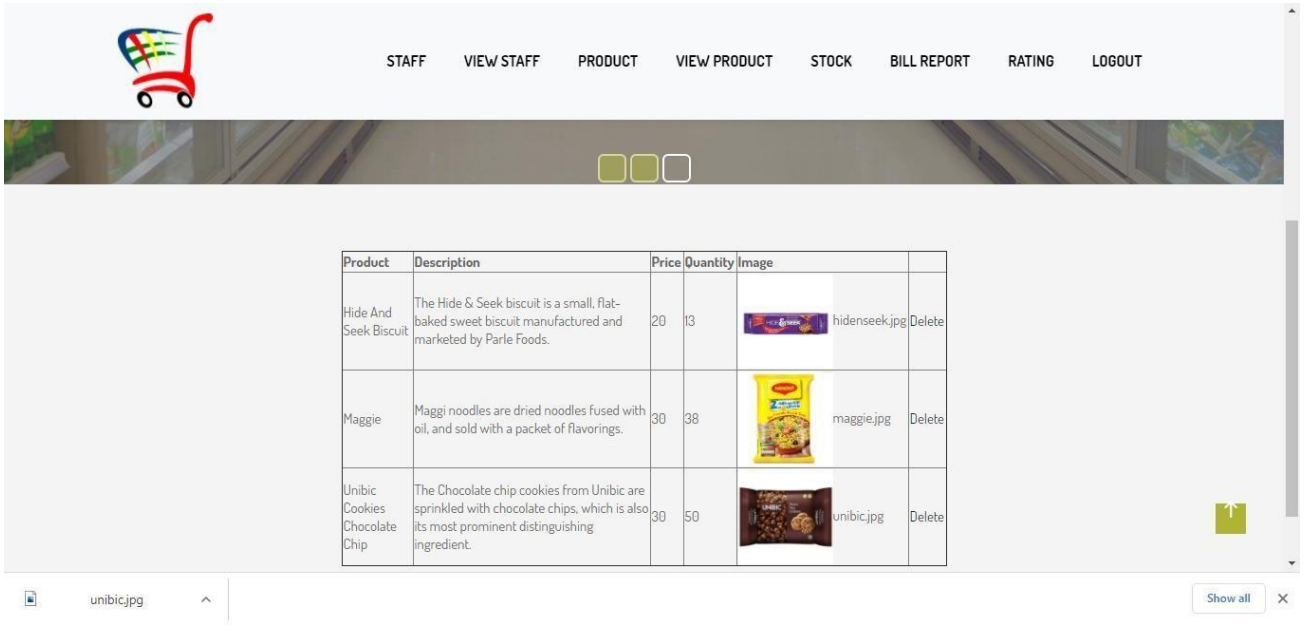


Fig 7.2.1.6 View Product Form

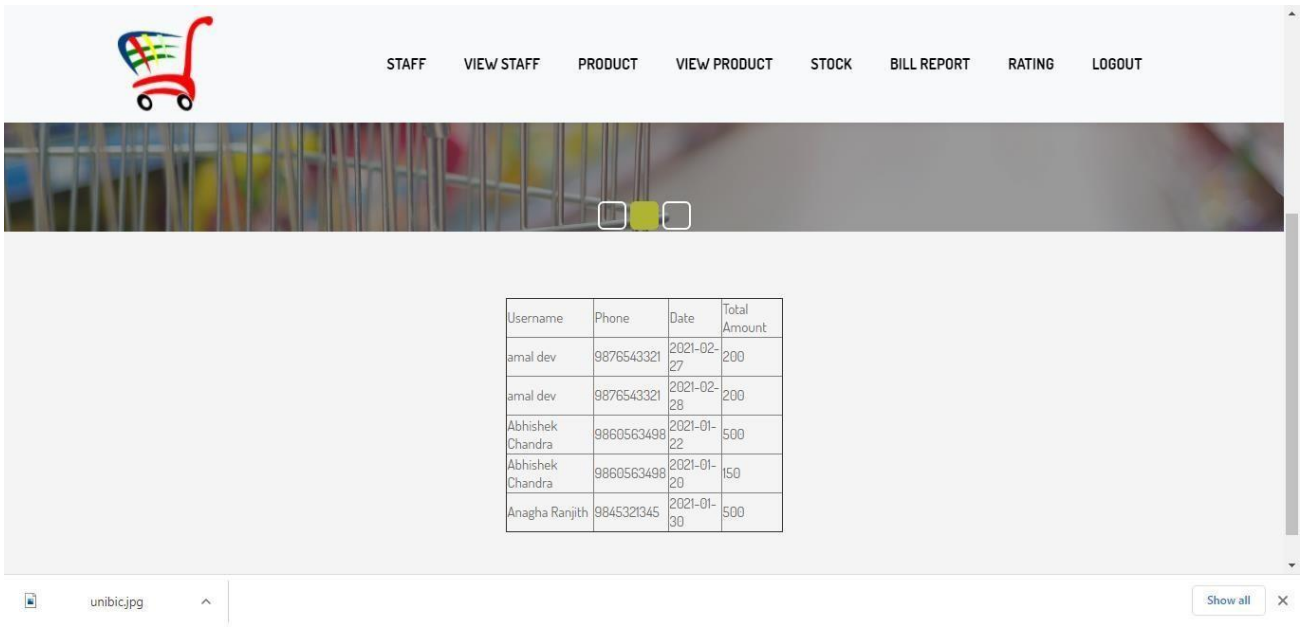


Fig 7.1.2.7 View Bill Report From

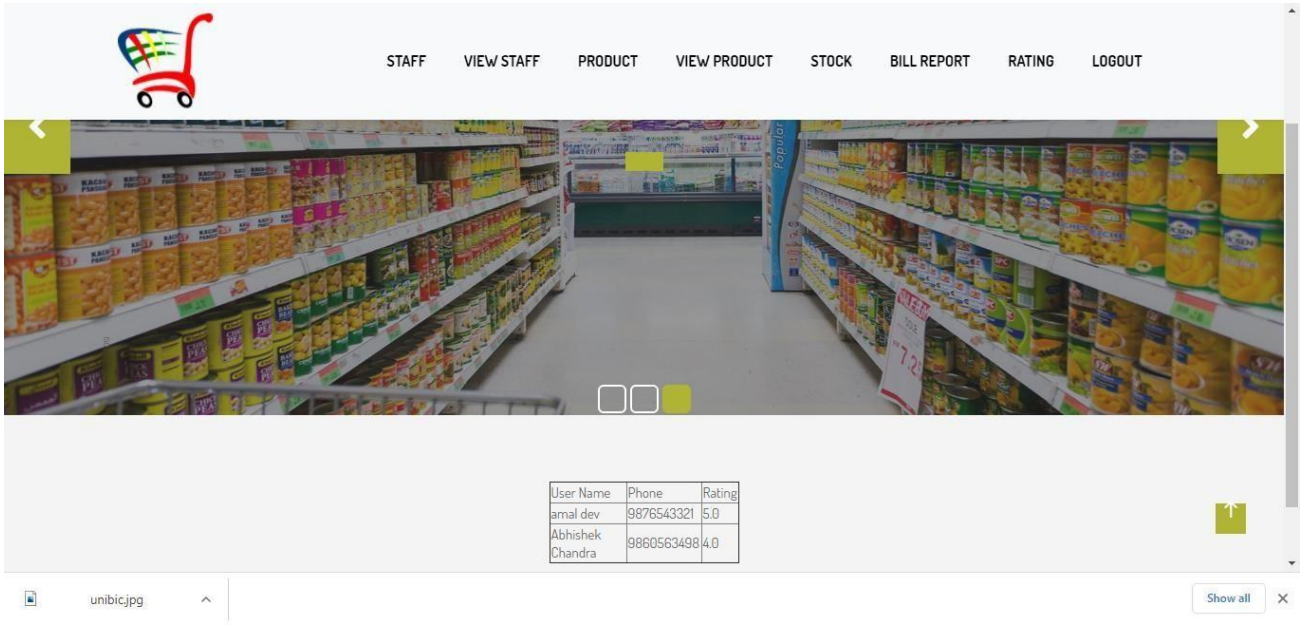


Fig 7.1.2.8 View Rating From

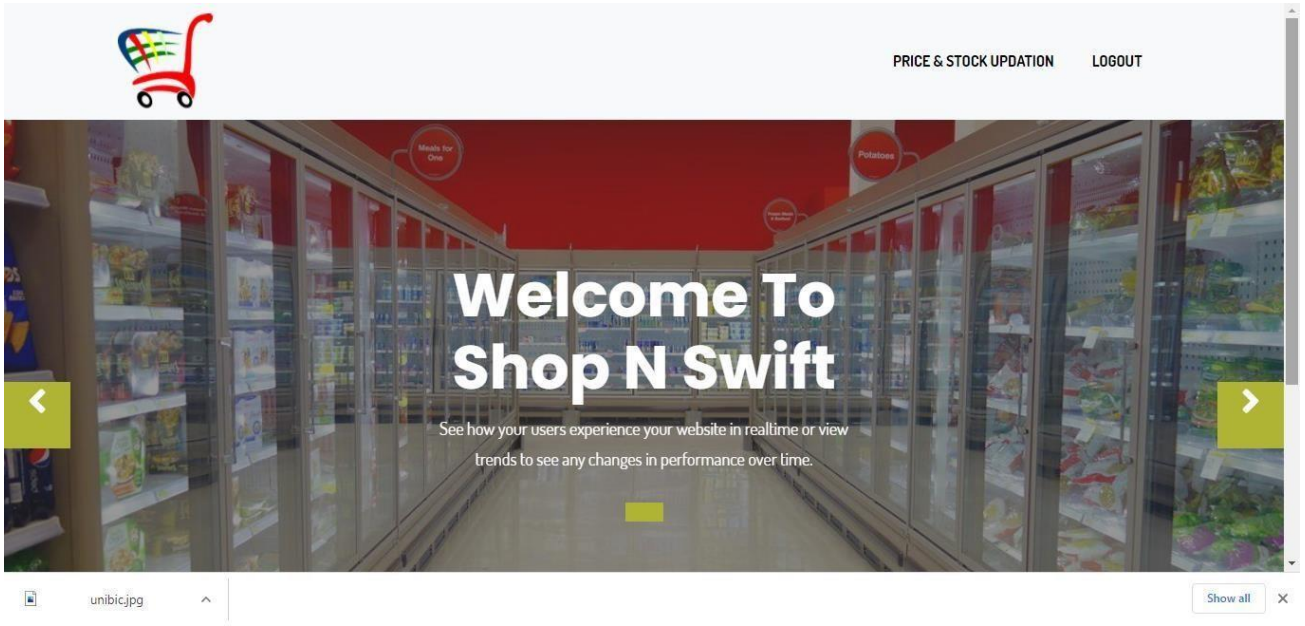


Fig 7.1.2.9 Sales Home

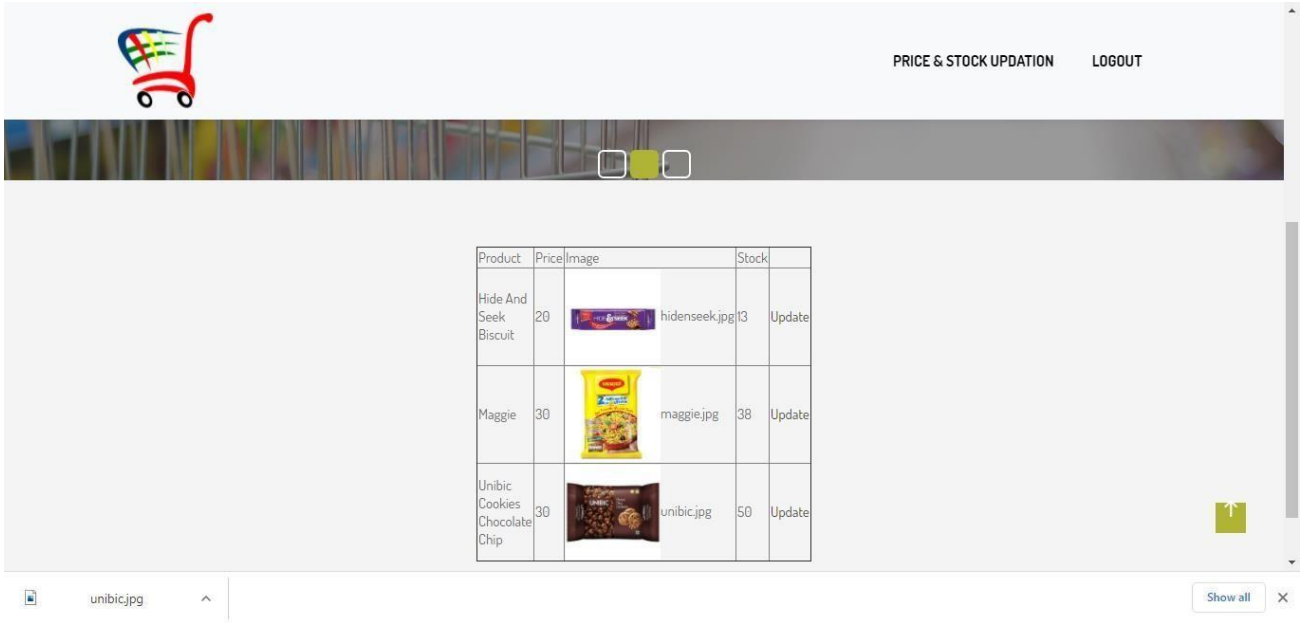


Fig 7.1.2.10 Price Stock And Updation From

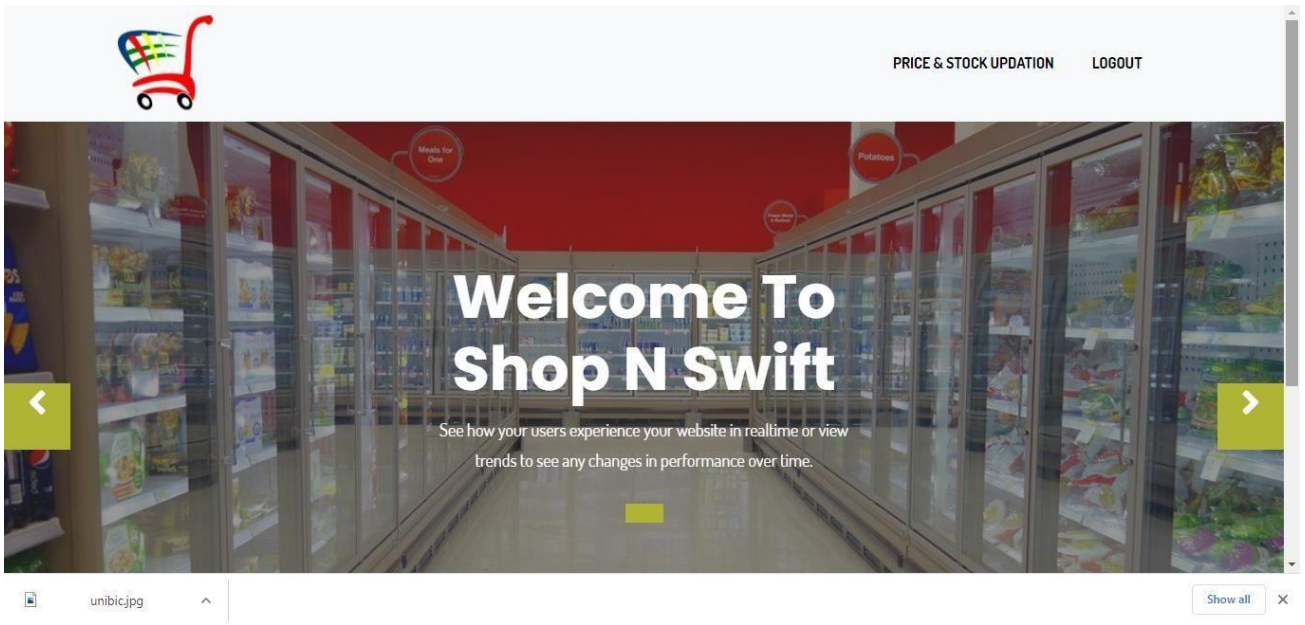


Fig 7.1.2.11 Store Home

Fig 7.1.2.11 User Interface

8

REFERENCE

8.1 REFERENCE BOOKS

Java: The Complete Reference Book by Herbert Schildt.

Java Programming: A comprehensive introduction Book by Dale Skrien and Herbert Schildt.

MySQL Reference Manual: Documentation from the Sourse Book by David Axmark, Kaj Arno, and Michael Widenius.

MySQL: The Complete Reference Book by Vikram Vaswani.

Python: The Complete Reference Book by Martin Brown and Martin C. Brown

Roger S, "Software Engineering – A Practitioner's Approach", Seventh Edition

Pearson Education, "Software Engineering by Ian Sommerville", 9th Edition,

8.2 REFERENCE WEBSITES

www.geeksforgeeks.org

www.ieee.org

www.sciencedirect.com

www.stackoverflow.com

Richard Fairley, "Software Engineering Concepts", 2008

9

CONCLUSION

AND

FUTURE ENHANCEMENT

9.1 FUTURE ENHANCEMENT

- Support for other platforms like IOS
- Improve the sequrity system with RFID (Radio-Frequency Identification) tags and embedded sensors

- Adding products which can be bought in kilogram / gram
- Distributes to multiple supermarkets

9.2 CONCLUSION

Technology has made significant progress over the years to provide consumers a better online shopping experience and will continue to do so for years to come. With the rapid growth of products and brands, people have speculated that online shopping will overtake in-store shopping. While this has been the case in some areas, there is still demand for brick and mortar stores in market areas where the consumer feels more comfortable seeing and touching the product being bought. However, the availability of online shopping has produced a more educated consumer that can shop around with relative ease without having to spend a large amount of time. In exchange, online shopping has opened up doors to many small retailers that would never be in business if they had to incur the high cost of owning a brick and mortar store. This project is to propose a real time capturing system for consumer supplies using Quick Response (QR) code in an Android smart phone. In recent years, extensive research has been carried out on vision-based automatic identification technology that recognizes image codes using smart phones to provide various services that can recognize the authenticity of any product.