**SYNOPSIS**

**1. SYNOPSIS**

The objective of this project is to propose a real time capturing system for consumer supplies using Quick Response (QR) code in a 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. Using Multiplexing and Demultiplexing process encode and decode the information from single QR code with special symbols and split the data back to their QR Code pattern where these QR Code pattern can be read by Android smart phones. Standard image codes like one-dimensional barcodes and two-dimensional codes with black and white patterns identifies a product for its value and basic features but does not authenticate it, moreover not every product that is identified, is used for authenticating manufacturer's warranty. So QR code verifies products by capturing it through the smart phone, then decodes and sends it to the server for authentication. In particular, we concentrate on the cases where the memory entries and their associations form a binary Hamming space or an infinite square grid. Particularly, we focus on minimizing the number of input clues needed to retrieve information with small uncertainty and present good constructions some of which are optimal. The customer forwards the selected product list to the server that enables the consumer to decide based on the products authenticity.

This project contains following modules:

**MODULES**

1. **Admin Module**

* **Add Staff**

In this function used to add staffs into the database through the user intreface

* [**View Staff**](http://localhost/Project%20Official/StaffView.php)

In this function used to view the stored details of the staffs

* [**Add Products**](http://localhost/Project%20Official/ProRegister.php)

In this function used to add products into the database through the user intreface

* [**View Products**](http://localhost/Project%20Official/ProductRegView.php)

In this function used to view the stored details of the products

* [**View Bill Report**](http://localhost/Project%20Official/ViewBillReport.php)

In this function used to view the billing details of the purchased items

* [**View Rating**](http://localhost/Project%20Official/ViewRating.php)

Admin can see the details of the rating details of given by the user

* [**Provide Offers**](http://localhost/Project%20Official/Offers.php)

It is used to add offers that provided by the shop by the admin

* [**View Offers**](http://localhost/Project%20Official/Offersview.php)

In this function used to view the billing details of the purchased items

* [**View Stock**](http://localhost/Project%20Official/StockAdmin.php)

In this function can see the stock details of products in the store

* [**Customers**](http://localhost/Project%20Official/Customerview.php)

Through this function can see the customer details that stored in the database

1. **Billing module**

* [**Verify Purchase Code Entry**](http://localhost/Project%20Official/Verifycodeentry.php)

In this function verifies the code that sends to the database by the users mobile

* [**View Reports**](http://localhost/Project%20Official/ViewReports.php)

It shows the purchase details of the products based on their date of purchase

* **View Stock Items**

In this function can see the stock details of products in the store

1. **Store Module**

* [**Stock Update**](http://localhost/Project%20Official/storestock.php)

This function is used to update the stock details from the store

* [**Stock View**](http://localhost/Project%20Official/StockView.php)

This function is used to view the stock details

* [**Out Of Stock Products**](http://localhost/Project%20Official/AlertView.php)

This function which shows the out of stock details that less than a fixed number of the products

* [**QR-Code Generation**](http://localhost/Project%20Official/QR-CodeGeneration.php)

In this function is used to create the QR-codes for each products

* [**View QR-Code With Details**](http://localhost/Project%20Official/viewQR.php)

Created QR-Codes shown in the section that stored in the database

1. **User Module**

* **Start Purchase**

In this function is used to start the purchase in android section

* **View Products**

User can view the products in android app

* **Offers**

User can view the offers in android app

* **Contact Us**

It shows the details of the corresponding shop that uses the software

* **Rate Us**

Providing this facility for the users to give their feedback through the rating

* **Recent Purchases**

Through this facility user can see the recently purchased items in the cart

**INTRODUCTION**

**2. INTRODUCTION**

The objective of 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. QR code verifies products by capturing it through the smart phone, then decodes and sends it to the server for authentication. The customer forwards the selected product list to the server that enables the consumer to decide based on the products authenticity.

* 1. **Purpose**

In this system, when we purchasing a product from an outlet shop, we need to wait a time to the billing. SO in this software the customer does not wait to bill. It will make the shopping easy and in this system customer have a better role. Other features of this software are the faster transaction and less man power consumption. So the system will more helpful for the customers.

**2.2 Scope**

Shopping with QR-Codes. It can be used in shopping markets. Helpful for human offline shopping. Fact that it is a stable way to represent software that works well for varying work load in the shops. We can buy the products even if small difference between the price of the shop products and out shop products.

**2.3 Overview**

The online based product purchasing is an important task in the modern world. Where the purchase using the QR-Code is a branch modern purchasing. That is by processing the printed QR-Codes by scanner. We can by the products by the QR-codes .Using QR-Codes and several other bar-codes processing techniques. This software simplifies the shopping time and man power consumption.

**SYSTEM DESCRIPTION**

**3. SYSTEM DESCRIPTION**

This paper presents a novel method of collaborating ease in offline shopping and the sense of security money wise as well as for customer satisfaction while doing shopping offline. This is implemented using an Android application. 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. The application mentioned here would read the barcode(s) of the product(s) & add it to the shopping cart in the application. 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.

**3.1 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.
* 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

Disadvantages

* Time consuming
* No role for client
* Waiting problem for customer

**3.2 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

Advantages

* Make easy shopping
* Better role for customer
* Faster transaction
* Less man power consumption

**3.3 Feasibility Study**

System analysis is the most important phase in the lifecycle of the development. The initial investigation points to the question whether the project is feasible. Feasibility is conducted to identify the best system that meets all the requirements. This includes an identification description, a valuation of the proposed systems and selection of the best system for the job. The requirements of the system are specified with a set of constraints such as system objectives and the description of the outputs. It is then duty of the analyst to evaluate the feasibility of the proposed system to generate the above results. Three key factors are to be considered during the feasibility study.

**3.4 System Environments**

**3.4.1 Hardware Requirements**

* The decision to acquire computer hardware or software must be handled in the same way as any other business decision. The variety of sizes and types of computing resources available puts a burden on the analyst who must select suitable hardware, software or services and advice the top management accordingly.
* Today, selecting a system is a serious and time consuming business. The time spent on the selection process is a function of the application and whether the system is basic microcomputer or a mainframe. In either case, planning system selection and acquiring experienced help were necessary payoff in the long run. The hardware needed depends upon the software’s design and complexity.
  + INPUT DEVICE : MOUSE , KEYBOARD
  + OUTPUT DEVICE : MONITOR
  + MEMORY : 2 GB RAM(Minimum)
  + PROCESSOR : PENTIUM 4 above

**3.4.2 Software Requirements**

The selection of the software is the important aspect in the development of the system. It should have all the facilities of meeting the objectives of the system.

* Operating System :  Windows/Linux/Android
* Front End :  PHP(For web application) and Android
* Back End : MySQL server 2008
* Web browser : Internet Explorer/Google Chrome/Firefox
* Webserver : Apache (for web applications)

**LANGUAGE DESCRIPTION**

**1 PHP**

PHP is a server side scripting language designed 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 Group. PHP originally stood for Personal Home Page, but it now stands for the recursive back acronym PHP: Hypertext Pre-processor.

PHP code may be embedded into HTML code, or it can be used in combination with various web template systems, web content management system 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.

**Is PHP the right choice for you?**

PHP is a great option for many reasons, so if you’re interviewing a company throwing around the PHP slag, here are some reasons why the language may be right for you or your project:

1. Fast Load Time – PHP results in faster site loading speeds. PHP codes runs much faster than ASP because it runs in its own memory space while ASP uses an overhead server and a COM based architecture.
2. Less Expensive Software – In working with PHP, most tools associated with the program are open source software, such as
3. Word Press, so you need not pay for them. As for ASP, you might need to buy additional tools to work with its programs.
4. Less Expensive Hosting – ASP programs need to run on Windows servers with IIS installed. Hosting companies need to purchase both of these components in order to work ASP; this often results in a more expensive cost for monthly hosting services. On the other hand, a PHP would only require running on a Linux server, which is available through a hosting provider at no additional cost.
5. Database Flexibility – PHP is flexible for database connectivity. It can connect to several databases the most commonly used is the MySQL. MySQL can be used for free. If ASP is used, MS-SQL, a Microsoft product must be purchased.
6. Increased Available Programming Talent – PHP is used more often creating a larger pool of talent to choose from for modifications and building and lowering the cost per hour for those services. And making it easier to find someone to update your site in the future if you choose to hire a staff member for the task or work with an alternate provider than the one who built your site.

**The main advantages of PHP programming include the following:**

**Open Source**

PHP is freely available for use. The community of open source PHP developers provides technical support and is constantly improving updating the core PHP functionalities. PHP is available at free of cost under PHP General Public License and most of its associative required software’s like MySQL, Text Editors and Apache Server are also freely available, so it proves very cost effective for the developers.

**Cross-Platform**

PHP provides high compatibility with leading operating systems and web servers such as thereby enabling it to be easily deployed across several different platforms. PHP scripts can run across operating systems such as Linux, Windows, Solaris, OpenBSD, Mac OSX etc and also provide support for all major web servers such as Apache, IIS, iPlanet etc.

**Power**

Several web tasks can now be easily performed using PHP. For example now we can develop from small websites to giant business and organizational websites, informative forums, chatting platforms, CRM solutions, e-commerce shopping carts, community websites, e-business, shopping carts and gigantic database driven sites.

**User Friendly**

Designed in a user friendly manner, PHP gives more flexibility than C, C++ and ASP and overall helps in increasing traffic to the site.

**Quick**

PHP is designed to work well with the web, and so things like accessing the GET and POST and working with HTML and URLs are built-ins in the PHP language. This makes it really concise and straightforward to make a website.

**Extensions**

Being an open source language, a large number of libraries and extensions, to extend its core functionalities, are available for download. The source code of PHP can be modified to include custom created extensions and components thereby increasing its extensibility.

**Easy Deployment**

There are many hosting companies that will, for a few dollars a month, give you a server running PHP so you can make a website really easily.

**Automatically Refreshes**

Nowadays developing dynamic websites are in the huge demand due to its specific characteristics like it automatically refreshes and does not need to make much changes manually.

**Community Support**

A huge advantage that PHP offers is its community. If you are looking for a particular script, chances are another user has already created something similar. Check within the PHP community for availability. Likewise, if you have created a function that others might enjoy, be sure to post the code for others.

**Other Tools**

If you need to access other web based tools like Google maps (which is always advisable for a business website), or any other, PHP makes it easy to access.

**Security**

PHP offers security as well that helps prevent malicious attacks. These security levels can be adjusted in the .ini file.

**Talent Availability**

You can hire PHP programmers more easily than any other language programmers since so many people know the language.

**2 MYSQL**

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation.

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

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

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, network together.

**3 Eclipse IDE**

Eclipse is an Integrated Development Environment (IDE) that is open source and maintained by many software development leaders like IBM and Borland. The idea behind the Eclipse platform is to provide a common environment that companies can modify and customize .by creating plug-ins. Plug-ins are created to do some specific tasks that a work group might need while developing a software product. The plug-ins and modification of the Eclipse platform can later be distributed royalty free.

In Computer programming, Eclipse is a multi-language IDE comprising a base work space and an extensible plug-in system for customizing the environment. It is written mostly in java. It can be used to develop applications in Java and, by means of various plug-ins, other programming language including Ada, C, C++, COBOL, Fortran, Haskell Java Script etc. It can also be used to develop packages for the software Mathematical Development environments include the Eclipse

Java development tools (JDT) for Java and Scala, Eclipse CDT for C/C++ and Eclipse PDT for PHP, among others.

The initial codebase originated from IBM visual age The Eclipse (SDK), which includes the Java development tools, is meant for Java developers. Users can extend its abilities by installing plug-ins written for the Eclipse Platform, such as development toolkits for other programming languages, and can write and contribute their own plug-in modules.

Android Fundamentals and Components

Important building blocks / components of android are as below.

* Activities
* Services
* Broadcast Receivers
* Content Providers

When we are developing application, always we will have some requirements like, passing messages with in an application (one screen to another) or between different applications.

For example: Let’s say a new message came to your phone, so we update the notification bar for that new message. So we will have to pass the information from message application to notification bar. Moral of the story is, the means of communication between the above mentioned components is through the below components.

* Intents
* Intent Filters

Let’s come to the GUI part. The User Interface elements are made up the below components, which we called:

* Views
* Notifications

**Android Activity**

Activity is the basic building block of every visible android application. It provides the means to render a GUI. Every screen in an application is an activity by itself. We can call each visible component as an activity in android. Though more than one activities work together to present an application sequence, each activity is an independent entity.

**Android Services**

Service is another building block of android applications which does not provide a UI. It is a program that can run in the background for an indefinite period. In windows platform, we call it a thread.

**Android Broadcast Receivers**

Broadcast Receiver is yet another type of component that can receive and respond to any broadcast announcements.

Let’s take an example: when a new message came to your inbox, this information will be broadcast to all applications. If any application wants to do something while a new message came, then it can receive the broadcasted message details (Like: sender’s number, content etc.) and process accordingly.

**Android Content Providers**

Content Providers are a separate league of components that expose a specific set of data to applications. Let’s take an example: If you want to search a contact in your contact database (Like: name, number etc), then you can use Content Provider for this purpose. You can say Content provider is the pointer in your application to a specific data base from other application. The android platform offers a new concept of communication through intents and intent filters.

**Android Intents**

Intents are messages that are passed between components. So we have a question in our mind. Is it equivalent to parameters passed to API calls? Yes, it is close to that. However, the

Fundamental differences between API calls and intents is’ the way of invoking it. Have a look the difference between these two as below.

* + - API calls are synchronous while intent-based invocation is asynchronous (mostly)
    - API calls are bound at compile time while intent-based calls are run-time bound (mostly)

It is these two differences that take Android platform to a different league.

In simple word, the core android components of an application — activities, services, and broadcast receivers — are activated through messages, called intents.

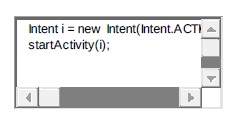
For example an activity can send intent to the Android system which starts another activity. So Intent is just a way to send message in android.

Android intents are basically 2 types in nature.

* **Implicit Intents**

Implicit intents specify the action which should be performed by other components or applications.

For Example: If you want to open an URL in a web browser from your application code, then following code tells the Android system to view a webpage. Typically the web browser is registered to this Intent but other component could also register them self to this to that intent. That means if you have installed web browsers like IE, Mozilla Firefox and Google Chrome, then all browsers might be registered to the intent (Intent.ACTION\_VIEW) to show a web page as per you request.



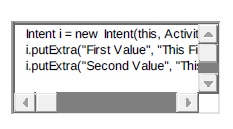


If only one component (here web browser in our example) is found, Android starts this component directly. If several components are identifier by the Android system, the user will get a selection dialog and can decide which component should be used for that Intent.

* **Explicit Intents**

Explicit intents are explicitly defines the exact component which should be called by the Android system, by using the Java class as identifier.

The following code shows how to create an explicit intent and send it to the Android system. That means Android system will directly execute your intent request as you requested. Explicit intents are typically used within an application as the classes in an application are controlled by the application developer. If you want to open an Android Activity from another activity, then below is the code using intent. Also you can send some data to that activity if required.



|  |  |
| --- | --- |
| 1  2  3 | Intent i = new Intent(this, ActivityTwo.class);  i.putExtra("First Value", "This First Value for ActivityTwo");  i.putExtra("Second Value", "This Second Value ActivityTwo"); |

* **Android Intent Filters**

To inform the system which implicit intents they can handle, activities, services, and broadcast receivers can have one or more intent filters. Each filter describes a capability of the component, a set of intents that the component is willing to receive. Let’s take an Example: The ‘Notepad Editor’ activity of the sample Note Pad application has two filters — one for starting up with a specific note that the user can view or edit, and another for starting with a new, blank note that the user can fill in and save.

Basically Intent Filters are defined in the AndroidManifest.xml file. For Broadcast Receiver it is possible to define in coding. An Intent Filters is defined by its category, action and data filters. It can also contain additional metadata. If a component does not define an Intent filter, then it can only be called by explicit Intents.

Android Layers

Basically android has the following layers:

* Application: (written in java, executing in Dalvik). All your custom applications will be available in this layer. This is the top layer in android OS.
* Framework services and libraries: (written mostly in java). Internally all of your custom application APIs will call this layer. So whatever API we will call from the java layer, all of them will go through this layer.
* Android Runtime: All applications and most framework code executes in a virtual machine. It means all the Java and framework code will be converted into executable code in this layer. The Android virtual machine is also called Dalvik virtual machine.
* Native libraries, daemons and services: (written in C or C++). Native libraries are nothing but part of internal Android OS. These libraries are used when we called any API from the application layer. Basically when we call any API from Java layer or from the upper layer, then it calls the API from the native layer which is mostly written in C/C++, SO we can say here the Java API converted into C/C++ API.
* The Linux kernel, which includes drivers for hardware, networking, file system access and inter-process-communication. This is the most complex layer in the Android OS. Whatever we will do on the upper layer, everything will pass through these layers, which give us the final output.

**3.4.3 Functional Requirements**

* Processor : Dual Core, 1.60GHz
* RAM : 2 GB
* Hard Disk Drive : 500GB
* LCD Color Monitor
* DVD Write For Backup

**SYSTEM ARCHITECTURE**

**4. SYSTEM ARCHITECTURE**

**4.1 ARCHITECTURAL DESIGN**

**DESIGNING OF SYSTEM**

Systems design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering.

## Overview

If the broader topic of product development "blends the perspective of marketing, design, and manufacturing into a single approach to product development," then design is the act of taking the marketing information and creating the design of the product to be manufactured. Systems design is therefore the process of defining and developing systems to satisfy specified requirements of the user.

Until the 1990s, systems design had a crucial and respected role in the data processing industry. In the 1990s, standardization of hardware and software resulted in the ability to build modular systems. The increasing importance of software running on generic platforms has enhanced the discipline of software engineering.

Object-oriented analysis and design methods are becoming the most widely used methods for computer systems design. The UML has become the standard language in object-oriented analysis and design. It is widely used for modelling software systems and is increasingly used for high designing non-software systems and organizations.

### Architectural design

The architectural design of a system emphasizes on the design of the systems architecture which describes the structure, behaviour, and more views of that system and analysis.

### Logical design

The logical design of a system pertains to an abstract representation of the data flows, inputs and outputs of the system. This is often conducted via modelling, using an over-abstract (and sometimes graphical) model of the actual system. In the context of systems, designs are included. Logical design includes entity-relationship diagrams (ER diagrams).

### Physical design

The physical design relates to the actual input and output processes of the system. This is explained in terms of how data is input into a system, how it is verified or authenticated, how it is processed, and how it is displayed. In physical design, the following requirements about the system are decided.

1. Input requirement,
2. Output requirements,
3. Storage requirements,
4. Processing requirements,
5. System control and backup or recovery.

Put another way, the physical portion of systems design can generally be broken down into three sub-tasks:

1. User Interface Design
2. Data Design
3. Process Design

User Interface Design is concerned with how users add information to the system and with how the system presents information back to them. Data Design is concerned with how the data is represented and stored within the system. Finally, Process Design is concerned with how data moves through the system, and with how and where it is validated, secured and/or transformed as it flows into, through and out of the system. At the end of the systems design phase, documentation describing the three sub-tasks is produced and made available for use in the next phase.

Physical design, in this context, does not refer to the tangible physical design of an information system. To use an analogy, a personal computer's physical design involves input via a keyboard, processing within the CPU, and output via a monitor, printer, etc. It would not concern the actual layout of the tangible hardware, which for a PC would be a monitor, CPU, motherboard, hard drive, modems, video/graphics cards, USB slots, etc. It involves a detailed design of a user and a product database structure processor and a control processor. The H/S personal specification is developed for the proposed system.

**DATABASE DESIGN**

A Database it collection of inter related data stored with minimum redundancy to serve many application. In database environment, common data are available and used by several users. Instead of each or user managing its own data, authorized users share data across applications with the database software managing the data as entity.

Database design is one of the important parts in developing software. It is a process of conceptual design of data It minimizes the artificially of using separate files, it is a definition of the entire information content of the organization and it specifics flier relationship between the data's. A National database consists of collection of tables having attributes are assigned a unique name. A row in the table represents a relationship among a set of values

The database design is sick up of two levels

* Conceptual Levels
* Normalization

**Conceptual Level**

The conceptual level represents lire major and relationships between them. Conceptual level describes the essential features of system data. It uses symbols from a modelling, method is called entity relationship analysis.

**Normalization**

The normalization of data refers to the way data item are grouped into record structures. Normalization is used to overcome the drawbacks like repetition of data, lots

of information and in consistency. In other words normalization is a technique of separating redundant fields and breaking up large tables into smaller one. After the conceptual level the next level of process of process of database design to organize the database structure into a good shape is called normalization.

In our design all tables have been normalized up to the third normal form. The different normal forms applied during the database design are given below.

* First Normal Form(1NF)
* Second Normal Form(2NF)
* Third Normal Form(3NF)

**First Normal Form**

First normal form (1NF) is a property of a relation in a relational database. A relation is in first normal form if and only if the domain of each attribute contains only atomic values, and the value of each attribute contains only a single value from that domain.

**Second Normal Form**

A relation that is in first normal form must meet additional criteria if it is to qualify for second normal form. Specifically: a relation is in 2NF if it is in 1NF and no non-prime attribute is dependent on any proper subset of any candidate key of the relation.

**Third Normal Form**

Third normal form is a normal form that is used in normalizing a database design to reduce the duplication of data and ensure referential integrity by ensuring that (1) the entity is in second normal form, and (2) all the attributes in the table are determined only by the candidate keys of that table and not by any non-prime attributes. 3NF was designed to improve database processing while minimizing storage costs.

**TABLE DESING**

Table is a collection of complete details about a particular subject. These data are saved in rows and columns. The data of each row are different units. Hence, rows are called records and columns of each row are called fields.

Data is stored in tables, which is available in the backend. The items and data, which are entered in the input, form id directly stored in this table using linking of database. We can link more than one table to input forms. Wee can collect the details from the different tables to display on the output.

**Tables**

**Table Name: Login**

**Primary Key: Login Id**

**Description : Store the login details**

|  |  |  |
| --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **DESCRIPTION** |
| **login\_id** | **int** | **Login id of user** |
| **Usertype** | **varchar** | **Specifies the type of user** |
| **username** | **varchar** | **User name of the user** |
| **password** | **varchar** | **Password of the user** |

**Table Name: Producttable**

**Primary Key: P\_ID**

**Description :Store the product details**

|  |  |  |
| --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **DESCRIPTION** |
| **P\_ID** | **int** | **Product id of the products** |
| **P\_Name** | **varchar** | **Product Name of the products in the database** |
| **P\_Quantity** | **varchar** | **Quantity of the product** |
| **UsagePeriodTime** | **varchar** | **The usage period time the** |
| **Price** | **varchar** | **Price of the product** |
| **Mfd** | **varchar** | **Date of manufacturing the product** |
| **Exp\_Date** | **varchar** | **Date of expiry the product** |
| **Type** | **varchar** | **It specify the product is veg or others** |
| **ProductBrand** | **varchar** | **Which company manyfacture the product** |

**Table Name: offertable**

**Primary Key:cap\_id**

**Description :Store new offers**

|  |  |  |
| --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **DESCRIPTION** |
| **P\_ID** | **int** | **Product id of the products** |
| **Pro\_id** | **varchar** | **Product id contain the products which contain offer** |
| **P\_Offer** | **varchar** | **It is the offer provided by the Manufacturer** |
| **Sp\_Offer** | **varchar** | **It is the offer provided by the shop** |
| **Discription** | **varchar** | **Discription about the offer** |
| **From\_Date** | **varchar** | **Date of offer which starts** |
| **End\_Date** | **varchar** | **Date of offer which ends** |
| **Total\_Price** | **varchar** | **Total price that is reduced by the offer** |

**Table Name: billmain**

**Primary Key: B\_ID**

**Description: Store the details of entire billing section**

|  |  |  |
| --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **DESCRIPTION** |
| **B\_ID** | **int** | **Bill id for each purchase** |
| **Uid** | **varchar** | **User id for identify the user** |
| **total\_amount** | **varchar** | **Total amount of purchase** |
| **gen\_code** | **varchar** | **One time generated code for passing the values user to database** |
| **Date** | **varchar** | **Date of purchase** |
| **Status** | **varchar** | **It is the purchasing status** |

**Table Name: billsub**

**Primary Key: B\_ID**

**Description: Store bills of each product**

|  |  |  |
| --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **DESCRIPTION** |
| **B\_ID** | **int** | **Bill id for each purchase** |
| **P\_ID** | **varchar** | **Product id of the products** |
| **Price** | **varchar** | **Total amount of purchase** |

**Table Name: cart**

**Primary Key: Ct\_id**

**Description:Cart table is used to store the recent purchases of a customer as individual**

|  |  |  |
| --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **DESCRIPTION** |
| **Ct\_id** | **int** | **Crt id for products added in the cart** |
| **P\_ID** | **varchar** | **Product id of the products** |
| **Cus\_id** | **varchar** | **Customer id for the users** |

**Table Name: customertable**

**Primary Key: Cus\_ID**

**Description : Store the details of customers**

|  |  |  |
| --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **DESCRIPTION** |
| **Cus\_ID** | **int** | **Customer id for users** |
| **FirstName** | **varchar** | **Firstname of the users** |
| **SecondName** | **varchar** | **SecondName of the users** |
| **Place** | **varchar** | **Place of the user** |
| **Email** | **varchar** | **Email of the user** |
| **Phone** | **varchar** | **Phone number of the user** |
| **UserName** | **varchar** | **User name of the user** |

**Table Name: dummybank**

**Primary Key: Cus\_ID**

**Description :Dummybank is used to store the bank details of the customer**

|  |  |  |
| --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **DESCRIPTION** |
| **Cus\_ID** | **int** | **Customer id for users** |
| **Acc\_No** | **varchar** | **Registered account number of the user** |
| **Security** | **varchar** | **Secuirity code of the account** |
| **Amount** | **varchar** | **Bank balance of the user** |

**Table Name: purchasetable**

**Primary Key: Cus\_ID**

**Description : Store the details of purchase**

|  |  |  |
| --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **DESCRIPTION** |
| **Cus\_ID** | **int** | **Customer id for users** |
| **P\_ID** | **varchar** | **Product id of the products** |
| **Pur\_Qty** | **varchar** | **Quantity of purchased items** |
| **Price** | **varchar** | **Price of the products** |
| **Pur\_Date** | **varchar** | **Date of purchase** |

**Table Name: ratingtable**

**Primary Key: P\_ID**

**Description: Rating table is used to store the store the customers ratings over the products and services of the shop**

|  |  |  |
| --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **DESCRIPTION** |
| **P\_ID** | **int** | **Product id of the products** |
| **Rating** | **varchar** | **Rating is used to rate the product by the rating stars** |
| **Uid** | **varchar** | **User id for identify the user** |
| **Date** | **varchar** | **Date of rating entered** |
| **Fdb** | **varchar** | **It is the feedback in text** |

**Table Name : reportstable**

**Primary Key : P\_ID**

**Description : Reports table is used to store the sale details of the shop**

|  |  |  |
| --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **DESCRIPTION** |
| **P\_ID** | **int** | **Id of report** |
| **P\_Quantity** | **varchar** | **Quantity of product** |
| **Pur\_Date** | **varchar** | **Purchase date** |

**Table Name: stocktable**

**Primary Key: stockid**

**Description : Store the details of available stocks**

|  |  |  |
| --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **DESCRIPTION** |
| **stockid** | **int** | **Serial number** |
| **P\_id** | **varchar** | **Product id** |
| **Stk\_qty** | **varchar** | **Quantity of stock** |

**Table Name: stafftable**

**Primary Key: sid**

**Description : Store the details of current staffs in the shop**

|  |  |  |
| --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **DESCRIPTION** |
| **Sid** | **int** | **Id of staff** |
| **Name** | **varchar** | **Name of staff** |
| **Place** | **varchar** | **Place** |
| **DOB** | **varchar** | **Date of birth** |
| **Qualification** | **varchar** | **Qualification** |
| **Email** | **varchar** | **e-mail id** |
| **Gender** | **varchar** | **Gender of staff** |
| **Designation** | **varchar** | **Designation of staff** |

**DATA FLOW DIAGRAM (DFD)**

A data flow diagram (DFD) or a bubble chart is a graphical tool for structured analysis. DFD models a system by using external entities from which data flows to a process, which transforms the data and creates output-data-flows which go to other processes or external entities or files. Data in files may also flow to processes as inputs. DFDs can be hierarchically organized, which help in partitioning and analyzing large systems. As a first step, one dataflow diagram can depict an entire system which gives the system overview. It is called context diagram of level 0 DFD. The context diagram can be further expanded.The successive expansion of a DFD from the context diagram to those giving more details is known as leveling of DFD .Thus a top down approach is used, starting with an overview and then working out the details. The main merit of DFD is that it can provide an overview of what data a system would process, what transformation of data are done, what files are used, and where the results flow. The data flow has been represented as a hierarchical DFD .Context Level DFD was drawn first. Then the processes were decomposed into several elementary levels and were represented in the order of importance.

**BASIC DFD SYMBOLS**

An arrow line depicts the flow with arrow head pointing in the direction of the flow.

A circle depicts certain Process

A rectangle depicts source or destination of data within the system

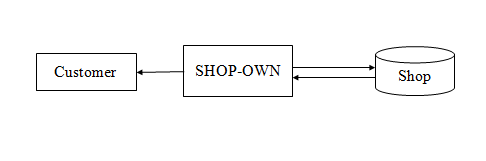


Indicates data storage

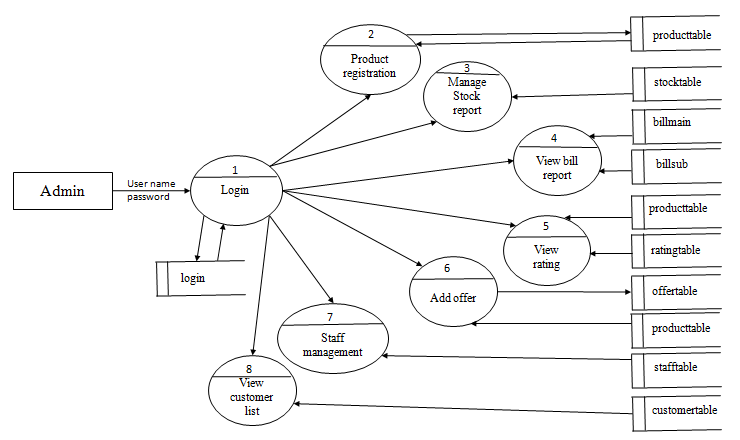
**DATA FLOW DIGRAM (DFD)**

**CONTEXT LEVEL DIAGRAM FOR SHOP-OWN**

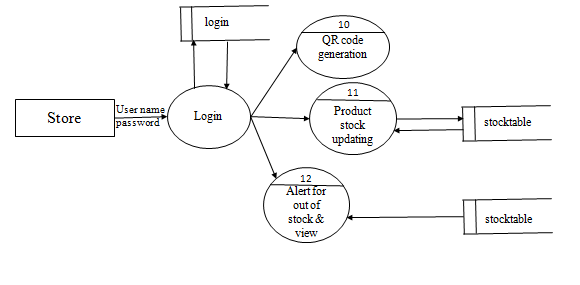
**Level – 0**

****

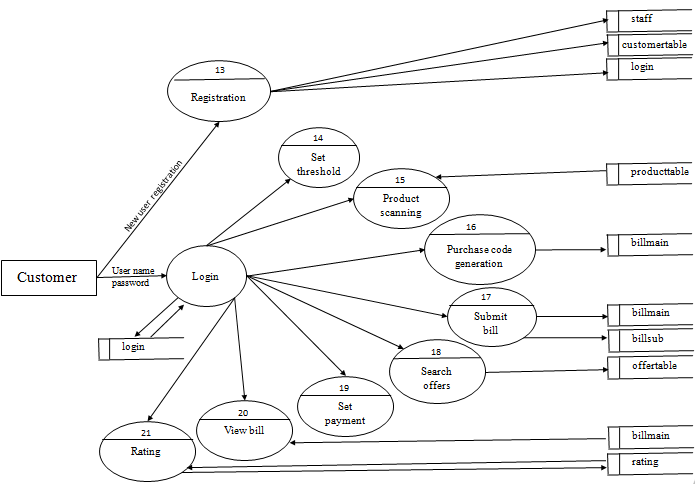
**Level-1 for Admin**

****

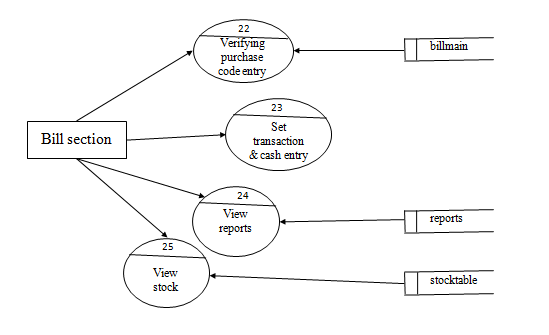
**Level-1 for Store**

****

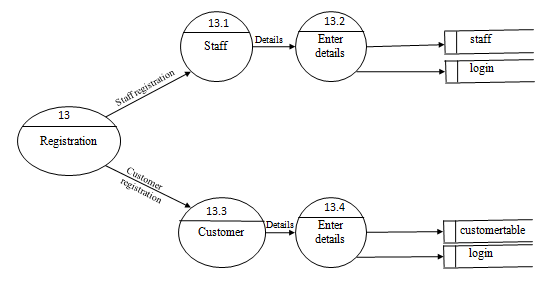
**Level-1 for Customer**



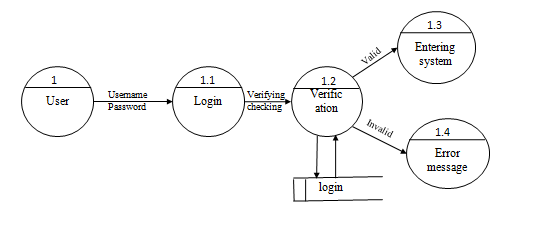
**Level-1 for Bill**

****

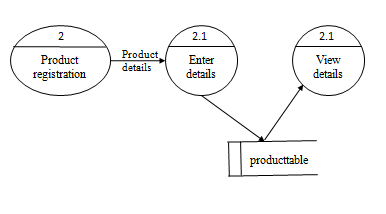
**Level-2 for Registration**

****

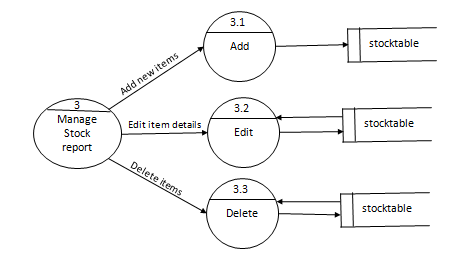
**Level-2 for Customer**

****

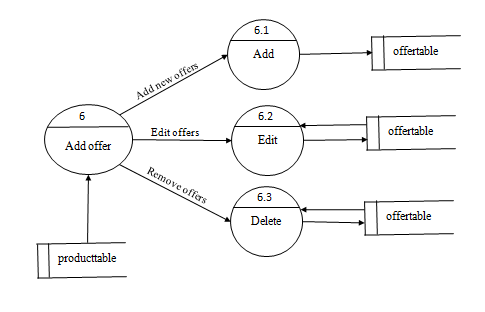
**Level-2 for Product registration**

****

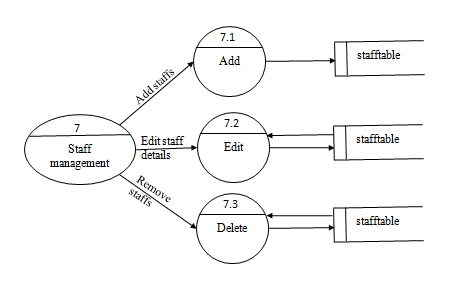
**Level-2 for Stock management**

****

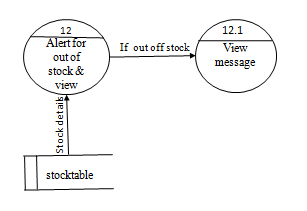
**Level-2 for Add offer**

****

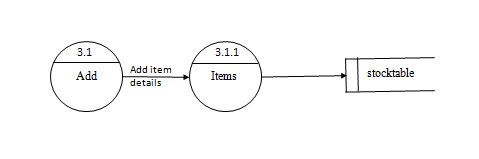
**Level-2 for Staff management**

****

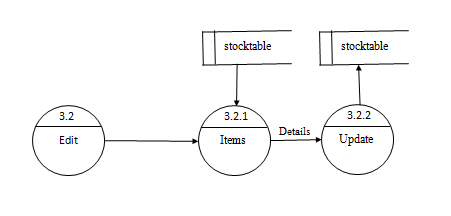
**Level-2 for Alert**

****

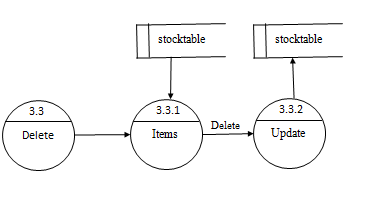
**Level-3 for Add items**

****

**Level-3 for Edit item details**

****

**Level-3 for Delete**

****

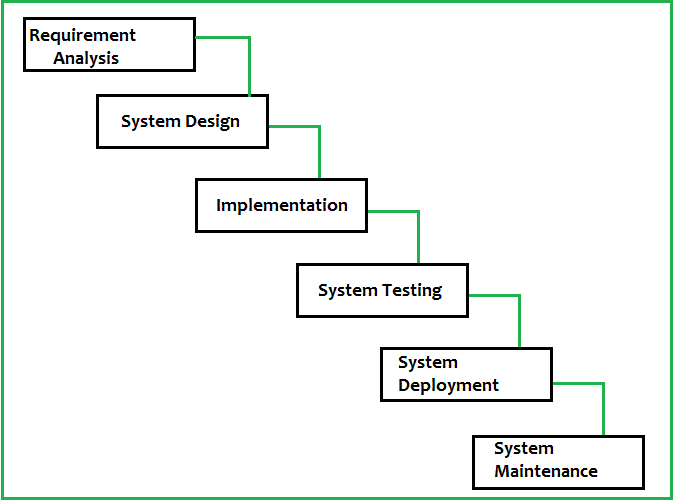
**Systems development life cycle (SDLC)**

**Waterfall model**

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

The waterfall model is a sequential design process, used in software development processes, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of conception, initiation, analysis, design, construction, testing, production/ implementation and maintenance.

The waterfall development model originates in the manufacturing and construction industries: highly structured physical environments in which after-the-fact changes are prohibitively costly, if not impossible. Because no formal software development methodologies existed at the time, this hardware-oriented model was simply adapted for software development.



The advantage of waterfall development is that it allows for departmentalization and managerial control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process like a car in a carwash, and theoretically, be delivered on time. Development moves from concept, through design, implementation, testing, installation, troubleshooting, and ends up at operation and maintenance. Each phase of development proceeds in strict order, without any overlapping or iterative steps.

The disadvantage of waterfall development is that it does not allow for much reflection or revision. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-thought out in the concept stage. Alternatives to the waterfall model include joint application development (JAD), rapid application development (RAD), synch and stabilize, build and fix, and the spiral model.

**SYSTEM TESTING**

**5. SYSTEM TESTING**

**5.1 INTRODUCTION**

Software testing is the process of executing a program with intension of finding errors in the code. Testing is the fundamental process of software success. Testing is not a distinct phase in system development life cycle but should be applicable throughout all phases. Testing enhances the integrity of a system by identifying deviations in design and development of the expected end product. It should focus more on the error-prone areas of the application. This help in the prevention of errors in a system and builds confidence that the system will work without error after testing. It is the process of executing a program with the intent of finding an error. Testing also adds value to the product by conforming to the user requirements. Testing verifies that software deliverable conforms precisely and design phases. A good test case is one that has a high probability of finding an as yet undiscovered error.

Testing involves a series of operation of a system of application under controlled conditions and subsequently evaluating the result. The controlled condition should include both normal and abnormal conditions. It is planned and monitor for each testing level (e.g., unit, integration, system and acceptance)

Testing is the major quality measure employed during software development. After the coding phase computer programs are available that can be executed for testing purpose. Testing not only has to uncover errors introduced during coding, but also locates error committed during the previous phase. Thus the aim of testing is to uncover requirements, design or coding errors in the programs

System testing of software is testing conducted on a complete, integrated system to evaluate the system’s compliances with its specified requirements. System testing falls within the scope of black box testing and such should require no knowledge of the inner design of the code. As a rule, testing takes as its input, all of the integrated software components that have successfully passed integration testing.

Testing is a process of executing a program with the interest of finding an error. A good test is one that has high probability of finding the yet undiscovered errors. The primary objective for the test case design is to drive a set of tests that has the highest likelihood for systematically uncovering different classes of errors in the software. Testing begins at the level and works outward to words the interaction of the entire software. A series of testing are performed for this project before the system is ready for acceptance. Some of the testing strategies applied for the systems are listed here.

**5.2 TESTING STRATEGIES**

A test strategy is an outline that describes the testing approach of the [software development cycle](https://en.wikipedia.org/wiki/Software_development_process). It is created to inform [project managers](https://en.wikipedia.org/wiki/Project_manager), testers, and developers about some key issues of the testing process. This includes the testing objective, methods of testing new functions, total time and resources required for the project, and the testing environment.

Test strategies describe how the product risks of the stakeholders are mitigated at the test-level, which types of testing are to be performed, and which entry and exit criteria apply. They are created based on development design documents. System design documents are primarily used and occasionally, conceptual design documents may be referred to. Design documents describe the functionality of the software to be enabled in the upcoming release. For every stage of development design, a corresponding test strategy should be created to test the new feature sets.

**Test Levels**

The test strategy describes the test level to be performed. There are primarily three levels of testing: [unit testing](https://en.wikipedia.org/wiki/Unit_testing), [integration testing](https://en.wikipedia.org/wiki/Integration_testing), and [system testing](https://en.wikipedia.org/wiki/System_testing). In most software development organizations, the developers are responsible for unit testing. Individual testers or test teams are responsible for integration and system testing.

**Roles and Responsibilities**

The roles and responsibilities of test leader, individual testers, project manager are to be clearly defined at a project level in this section. This may not have names associated: but the role has to be very clearly defined.

Testing strategies should be reviewed by the developers. They should also be reviewed by test leads for all levels of testing to make sure the coverage is complete yet not overlapping. Both the testing manager and the development managers should approve the test strategy before testing can begin.

**Environment Requirements**

Environment requirements are an important part of the test strategy. It describes what operating systems are used for testing. It also clearly informs the necessary OS patch levels and security updates required. For example, a certain test plan may require Windows XP Service Pack 3 to be installed as a prerequisite for testing.

**Testing Tools**

There are two methods used in executing test cases: manual and automated. Depending on the nature of the testing, it is usually the case that a combination of manual and automated testing is the best testing method.

**Risks and Mitigation**

Any risks that will affect the testing process must be listed along with the mitigation. By documenting a risk, its occurrence can be anticipated well ahead of time. Proactive action may be taken to prevent it from occurring, or to mitigate its damage. Sample risks are dependency of completion of coding done by sub-contractors, or capability of testing tools.

**Test Schedule**

A test plan should make an estimation of how long it will take to complete the testing phase. There are many requirements to complete testing phases. First, testers have to execute all test cases at least once. Furthermore, if a defect was found, the developers will need to fix the problem. The testers should then re-test the failed test case until it is functioning correctly. Last but not the least, the tester need to conduct regression testing towards the end of the cycle to make sure the developers did not accidentally break parts of the software while fixing another part. This can occur on test cases that were previously functioning properly.

The test schedule should also document the number of testers available for testing. If possible, assign test cases to each tester.

It is often difficult to make an accurate estimate of the test schedule since the testing phase involves many uncertainties. Planners should take into account the extra time needed to accommodate contingent issues. One way to make this approximation is to look at the time needed by the previous releases of the software. If the software is new, multiplying the initial testing schedule approximation by two is a good way to start.

**Regression test approach**

When a particular problem is identified, the programs will be debugged and the fix will be done to the program. To make sure that the fix works, the program will be tested again for that criterion. Regression tests will make sure that one fix does not create some other problems in that program or in any other interface. So, a set of related test cases may have to be repeated again, to make sure that nothing else is affected by a particular fix. How this is going to be carried out must be elaborated in this section. In some companies, whenever there is a fix in one unit, all unit test cases for that unit will be repeated, to achieve a higher level of quality.

**Test Groups**

From the list of requirements, we can identify related areas, whose functionality is similar. These areas are the test groups. For example, in a railway reservation system, anything related to ticket booking is a functional group; anything related with report generation is a functional group. Same way, we have to identify the test groups based on the functionality aspect.

**Test Priorities**

Among test cases, we need to establish priorities. While testing software projects, certain test cases will be treated as the most important ones and if they fail, the product cannot be released. Some other test cases may be treated like cosmetic and if they fail, we can release the product without much compromise on the functionality. This priority levels must be clearly stated. These may be mapped to the test groups also.

**Test Status Collections and Reporting**

When test cases are executed, the test leader and the project manager must know, where exactly the project stands in terms of testing activities. To know where the project stands, the inputs from the individual testers must come to the test leader. This will include, what test cases are executed, how long it took, how many test cases passed, how many failed, and how many are not executable. Also, how often the project collects the status is to be clearly stated. Some projects will have a practice of collecting the status on a daily basis or weekly basis.

**Test Records Maintenance**

When the test cases are executed, we need to keep track of the execution details like when it is executed, who did it, how long it took, what is the result etc. This data must be available to the test leader and the project manager, along with all the team members, in a central location. This may be stored in a specific directory in a central server and the document must say clearly about the locations and the directories. The naming convention for the documents and files must also be mentioned.

**Requirements traceability matrix**

Ideally, the software must completely satisfy the set of requirements. From design, each requirement must be addressed in every single document in the software process. The documents include the HLD, LLD, source codes, unit test cases, integration test cases and the system test cases. In a requirements traceability matrix, the rows will have the requirements. The columns represent each document. Intersecting cells are marked when a document addresses a particular requirement with information related to the requirement ID in the document. Ideally, if every requirement is addressed in every single document, all the individual cells have valid section ids or names filled in. Then we know that every requirement is addressed. If any cells are empty, it represents that a requirement has not been correctly addressed.

**Test Summary**

The senior management may like to have test summary on a weekly or monthly basis. If the project is very critical, they may need it even on daily basis. This section must address what kind of test summary reports will be produced for the senior management along with the frequency.

The test strategy must give a clear vision of what the testing team will do for the whole project for the entire duration. This document can be presented to the client, if needed. The person, who prepares this document, must be functionally strong in the product domain, with very good experience, as this is the document that is going to drive the entire team for the testing activities. Test strategy must be clearly explained to the testing team members right at the beginning of the project.

**5.3 UNIT TESTING**

Unit test comprises the set of tests are performed by an individual programmer prior to integration of unit to large systems.

Coding & debugging unit testing integration testing

Unit testing is done to testing the modules (classes) one by one in order to make sure that work by themselves before it was put together with other modules. The tests are very simple, at least for small modules with small interfaces to the out world. What was done to test the classes is to use the different methods that are defined and make sure they return the result that should be expected. Each and every screen was put into testing by giving random values as input. Breakage testing was done with the boundary conditions too. The modules where checked to see that the methods return the expected result and that the classes handle the wrong input in a correct way. For example, error messages whenever needed and can handle exceptions effectively.

After coding each dialogue is tested and run individually. All unnecessary coding were removed and it was ensured that all the modules worked, as the programmer would expect. Logical errors found were corrected. So by working all the modules independently and verifying the output of each module in the presence of staff was concluded that the program was functioning as expected.

This testing focuses on each module and individual software unit ensuring that they work properly. Unit testing checks for the changes made in the new system or any program in it. Unit testing includes white box testing.

This is the first level of testing. In this different modules are tested against the specification produced during the design of the modules. Unit testing is done for the verification of the code produced during the coding of single program modules in an isolated environment. Unit testing first focuses on the modules independently of one another to locale errors.

The goal of unit testing is to isolate each part of the program and show that the individual parts are correct. A unit test provides a strict, written contract that the piece of code must satisfy. As a result, it affords several benefits. Unit tests find problems early in the development cycle. I Init testing allows the programmer to retractor code at a later date, and make sure the module still works correctly (i.e. regression testing). The procedure is to write test cases for all functions and methods so that whenever a change causes a fault, it can he quickly identified and fixed.

All modules in “Desktop Utility with Word and OCR” are completed with unit testing. Every content system is validated with corresponding data type. If the password in the login is not correct, the validations such as password is not valid are occurred. If we are not opening word module in a correct format it shows error messages. Navigation from one page to another page is possible. So the unit testing is an important testing method. In this project Desktop Utility with Word and OCR, I first developed the interfaces for logging, voice command, print screen, word, sticky notes. OCR Connections are established between the interfaces and they are tested for proper communication. Then each application is tested individually for proper functioning.

**5.4 INTEGRATION TESTING**

Integration testing is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before system testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing.

The purpose of integration testing is to verify functional, performance, and reliability requirements placed on major design items. These "design items”, i.e. assemblages (or groups of units), are exercised through their interfaces using Black box testing, success and error cases being simulated via appropriate parameter and data inputs. Simulated usage of shared data areas and inter-process communication is tested and individual subsystems arc exercised through their input interface. Test cases are constructed to test that all components within assemblages interact correctly, for example across procedure calls or process activations, and this is done after testing individual modules, i.e. unit testing. The overall idea is a "building block" approach, in which verified assemblages are added to a verified base which is then used to support the integration testing of further assemblages.

Data can be lost access an interface, one module can have as adverse effort on another sub function when combined, may not produce the desired major functions. Integration testing is a systematic testing for constructing the program structure, while at the same time conducting tests to uncover errors associated within the interface. I he objective are to take unit tested as a whole. Here correction is difficult because the vast expenses of the entire program complicate the isolation of causes. Thus in the integration testing step all the errors uncovered are corrected for the next testing steps. Some different types of integration testing are big bang, top-down, and bottom-up.

In this project all the modules are tested for proper communication between them. Then analysis the each and every page of the system and checks the transaction is correctly updating in the database.

**5.5 VALIDATION TESTING**

This provides final assurance that the software meets all the functional behavioral and performance requirements. The software is completely assembled as package. Validation succeeds when the software functions in a manner in which user wishes. Validation refers to the process of using software in live environment in order to find errors.

System validation checks the quality of the software in both simulation and live put a lot of validation testing before finally implementing it. Thus the feedback from the validation phase generally produces changes in the software. The system objective, the functional performance, requirements were looked into see whether all these criteria are satisfying the system needs.

The system is then presented before the manager along with the reports generated the system then undergoes a testing phase with the sample test data provided by him. System testing in this manner would verify that all the modules works together and generate the intended results. All individual modules should be working in tandem so that the overall system function or performance is achieved

**BLACK BOX TESTING**:

Black box testing takes an external perspective of the test object to derive test cases. These tests can be functional or non-functional, through usually functional. The test designer selects valid and invalid inputs and determines the correct output. There is no knowledge of the test object’s internal structure.

This method of test design is applicable to all levels of software testing:

Unit integration functional testing system and acceptance the higher the level and hence the bigger and more complex the box, the more one is forced to use black box testing tosimplify. While this method can uncover unimplemented parts of the specification, one cannot be sure that all existing paths are tested.

**ADVANTAGES OF BLACK BOX TESTING**

* More effective on large unit of code than glass box testing
* Tester needs no knowledge of implementation, including specific programming language
* Tester and programmer are independent of each other
* Tests are done from a user’s point of view
* Will help to expose any ambiguities or inconsistencies in the specifications
* Test cases can be designed as soon as the specifications are complete
* Incorrect or missing functions.
* Interface errors.
* Error in data structures or external database access.
* Behavior or performance errors
* Initialization and termination errors.

**WHITE BOX TESTING**

White box testing, sometimes called glass-box testing is a test case design method that uses the control structure of the procedural design to derive test cases. Using white box testing method, the software engineer can derive test cases that

1. Guarantee that all independent paths within a module have been exercised at least one.
2. Exercise all logical decisions on their true and false sides,
3. Execute all loops at their boundaries and within their operational bounds, and
4. Exercise internal data structures to ensure their validity.

**5.6 TEST CASES**

A test case, in software engineering, is a set of conditions under which a tester will determine whether an application, software system or one of its features is working as it was originally established for it to do. The mechanism for determining whether a software program or system has passed or failed such a test is known as a test oracle. In some settings, an oracle could be a requirement or use case, while in others it could be a heuristic. It may take many test cases to determine that a software program or system is considered sufficiently scrutinized to be released. Test cases are often referred to as test scripts, particularly when written - when they are usually collected into test suites.

**Formal test cases**

In order to fully test that all the requirements of an application are met, there must be at least two test cases for each requirement: one positive test and one negative test. If a requirement has sub-requirements, each sub-requirement must have at least two test cases. Keeping track of the link between the requirement and the test is frequently done using a [traceability matrix](https://en.wikipedia.org/wiki/Traceability_matrix). Written test cases should include a description of the functionality to be tested, and the preparation required to ensure that the test can be conducted.

A formal written test-case is characterized by a known input and by an expected output, which is worked out before the test is executed. The known input should test a precondition and the expected output should test a post condition.

**Informal test cases**

For applications or systems without formal requirements, test cases can be written based on the accepted normal operation of programs of a similar class. In some schools of testing, test cases are not written at all but the activities and results are reported after the tests have been run.

In [scenario testing](https://en.wikipedia.org/wiki/Scenario_testing), hypothetical stories are used to help the tester think through a complex problem or system. These scenarios are usually not written down in any detail. They can be as simple as a diagram for a testing environment or they could be a description written in prose. The ideal scenario test is a story that is motivating, credible, complex, and easy to evaluate. They are usually different from test cases in that test cases are single steps while scenarios cover a number of steps of the key.

**Typical written test case format**

A test case is usually a single step, or occasionally a sequence of steps, to test the correct behaviour/functionality, features of an application. An expected result or expected outcome is usually given.

Additional information that may be included:

* test case ID
* test case description
* test step or order of execution number
* related requirement(s)
* depth
* test category
* author
* check boxes for whether the test can be or has been automated
* pass/fail
* remarks

Larger test cases may also contain prerequisite states or steps, and descriptions.

A written test case should also contain a place for the actual result.

These steps can be stored in a word processor document, spreadsheet, database or other common repository.

In a database system, you may also be able to see past test results and who generated the results and the system configuration used to generate those results. These past results would usually be stored in a separate table.

[Test suites](https://en.wikipedia.org/wiki/Test_suite) often also contain

* Test summary
* Configuration

Besides a description of the functionality to be tested, and the preparation required to ensure that the test can be conducted, the most time consuming part in the test case is creating the tests and modifying them when the system changes.

Under special circumstances, there could be a need to run the test, produce results, and then a team of experts would evaluate if the results can be considered as a pass. This happens often on new products' performance number determination. The first test is taken as the base line for subsequent test / product release cycles.

[Acceptance tests](https://en.wikipedia.org/wiki/Acceptance_test), which use a variation of a written test case, are commonly performed by a group of [end-users](https://en.wikipedia.org/wiki/End-user) or clients of the system to ensure the developed system meets the requirements specified or the contract. User acceptance tests are differentiated by the inclusion of [happy path](https://en.wikipedia.org/wiki/Happy_path) or positive test cases to the almost complete exclusion of negative test cases.

**SYSTEM IMPLEMENTATION**

**6. SYSTEM IMPLEMENTATION**

System Implementation uses the structure created during architectural design and the results of system analysis to construct system elements that meet the stakeholder requirements and system requirements developed in the early life cycle phases. These system elements are then integrated to form intermediate aggregates and finally the complete system-of-interest (SOI).

## Definition and Purpose

Implementation is the process that actually yields the lowest-level system elements in the system hierarchy (system breakdown structure). System elements are made, bought, or reused. Production involves the hardware fabrication processes of forming, removing, joining, and finishing, the software realization processes of coding and testing, or the operational procedures development processes for operators' roles. If implementation involves a production process, a manufacturing system which uses the established technical and management processes may be required.

The purpose of the implementation process is to design and create (or fabricate) a system element conforming to that element’s design properties and/or requirements. The element is constructed employing appropriate technologies and industry practices. This process bridges the system definition processes and the integration process. Figure 1 portrays how the outputs of system definition relate to system implementation, which produces the implemented (system) elements required to produce aggregates and the SoI.

### Purpose and Principle of the Approach

During the implementation process, engineers apply the design properties and/or requirements allocated to a system element to design and produce a detailed description. They then fabricate, code, or build each individual element using specified materials, processes, physical or logical arrangements, standards, technologies, and/or information flows outlined in detailed descriptions (drawings or other design documentation). A system element will be verified against the detailed description of properties and validated against its requirements.

If subsequent verification and validation (V&V) actions or configuration audits reveal discrepancies, recursive interactions occur, which includes predecessor activities or processes, as required, to mitigate those discrepancies and to modify, repair, or correct the system element in question. Figure 2 provides the context for the implementation process from the perspective of the U.S. Defence Acquisition University (DAU).

**FUTURE ENHANCEMENT**

**7. FUTURE ENHANCEMENT**

In feature we would like to introduce some new feature in our website like

* Access permissions for unregistered users.
* Chatting with admin.
* Online Purchasing
* Etc…

**SUMMARY AND CONCLUSION**

**8. SUMMARY AND CONCLUSION**

SHOP-OWN is the project done in partial fulfilment of the requirements for the award of the degree BSc Computer Science. The project is aimed at producing a product which will become a great possession for all the Consumers, while the shopping. The project is named SHOP-OWN, the abbreviation of Smart Shopping App with QR-Code Scanning, because it indeed supports the users in their purchases.

**COST ESTIMATION OF PROJECT**

The cost of our project depends upon the time consumed in developing the software and the initial cost that we have spent on it. Approximately our project will cost Rs 20,000.

**BIBLIOGRAPHY**

**9. BIBLIOGRAPHY**

**Books :**

* Database Management System - T P Abbas
* System analysis and design - Elias M Award
* Web Programming – Rajneesh Agarawal
* Principles of software engineering – Rohit Khurana
* Software engineering – Ian Summerville
* Php mysql bible – Tim converse, Joyce park
* Mysql. The complete reference – Vikram V.A Swami
* Php . The complete reference – Steven Holzer

**Web Sites:**

* www.apache.com
* www.w3school.com
* www.wikipedia.com
* www.stackoverflow.com
* www.codeproject.com
* www.undemy.com

**APPENDIX**

**10. APPENDIX**

**10.1 Appendix A**

**10.1.1 Code for QR-Code Generation**

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

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<title>Untitled Document</title>

</head>

<body>

<form id="form1" name="form1" method="post" action="">

<div align="center">

<table width="564" height="384" border="1">

<tr>

<td colspan="2"><div align="center">QR-Code Generation</div></td>

</tr>

<tr>

<td width="218"><div align="center">Product Name</div></td>

<td width="330"><label for="textfield"></label>

<label for="select"></label>

<div align="center">

<select name="select" size="1" id="select">

<option>SelectProduct</option>

<?php

include("connect.php");

$str="select \* from producttable";

$res=mysql\_query($str);

while($res1=mysql\_fetch\_array($res))

{

?>

<option value="<?php echo $res1[0] ?>"><?php echo $res1[1] ?></option>

<?php

}

?>

</select>

</div></td>

</tr>

<tr>

<td height="99" colspan="2"><div align="center">

<input type="submit" name="button" id="button" value="QR-Code Generate" />

</div></td>

</tr>

</table>

</div>

</form>

<?php

if(isset($\_POST['button'])){

$id=$\_POST['select'];

include('libs/qrlib.php');

QRcode::png('hai');

$tempDir = "qrCode/";

$codeContents =$id;

$fileName = $id.'.png';

$pngAbsoluteFilePath = $tempDir.$fileName;

$urlRelativeFilePath = EXAMPLE\_TMP\_URLRELPATH.$fileName;

QRcode::png($codeContents, $pngAbsoluteFilePath);

header("location:QR-CodeGeneration.php");

}

?>

</body>

</html>

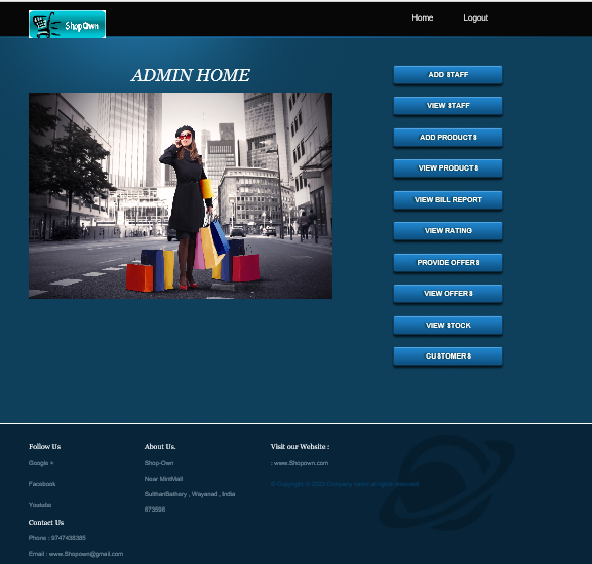
**10.2 Appendix B**

**10.2.1 Screen Shots**

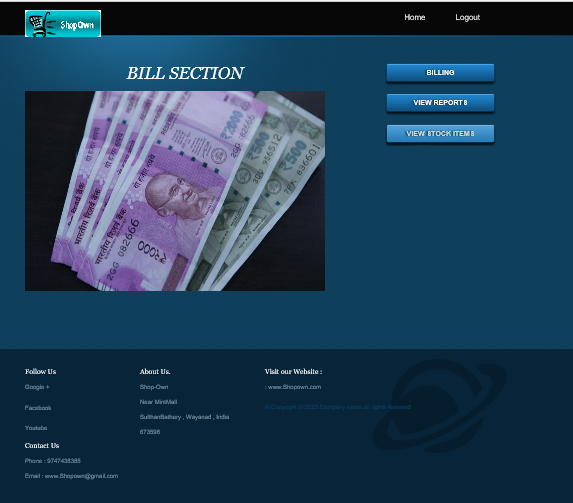
**LoginPage**

****

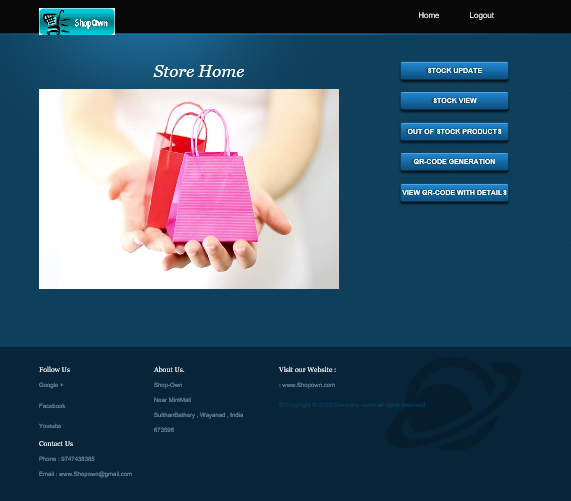
**Admin HomePage**

****

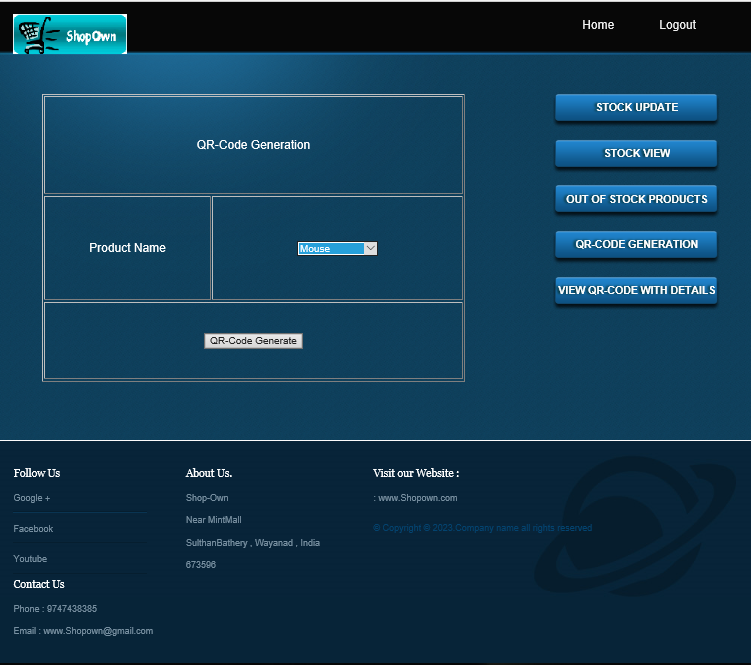
**Bill Section Home Page**

****

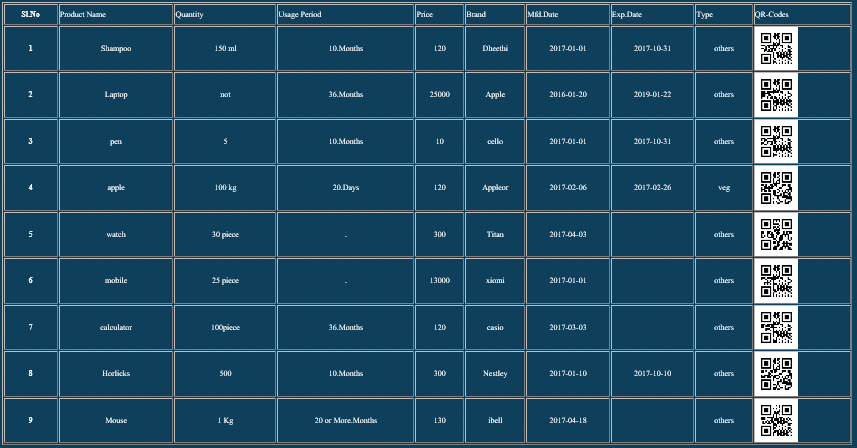
**Store Home Page**

****

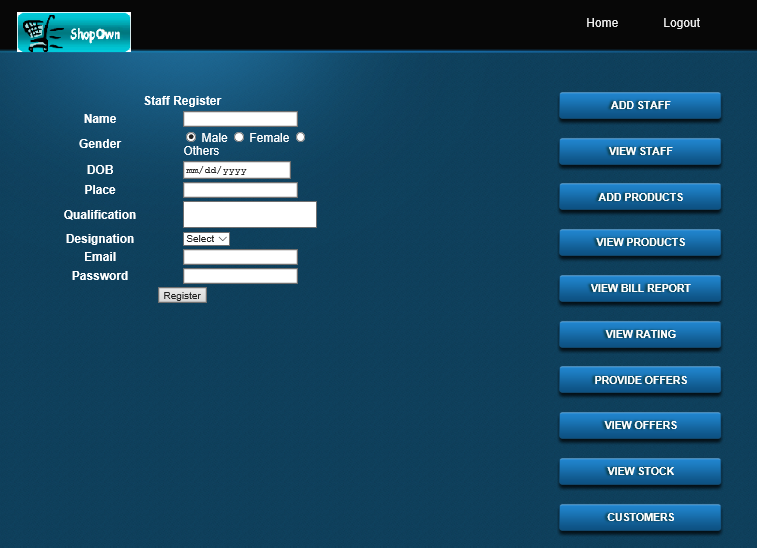
**QR-Code Generation Page**

****

**View Products With QR-Code Details**

****

**Staff Registration**

****