

ONLINE SHOPPING

A Thesis submitted in partially fulfillment of therequirements for
the degree of

Master of Computer Application

by

KM. ANUPRIYA

(University Roll No: 1900290149054)

Under the Supervision of Mr. Ankit Verma

Assistant Professor

KIET Group of Institution, Ghaziabad.

**Affiliated to Dr. Abdul Kalam Technical University,
Lucknow**



July, 2021

DECLARATION

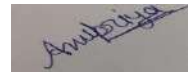
I hereby declare that the work presented in this report entitled “*ONLINE SHOPPING*”, was carried out by me. I have not submitted the matter embodied in this report for the award of any other degree or diploma of any other University or Institute.

I have given due credit to the original authors/sources for all the words, ideas, diagrams, graphics, computer programs, experiments, results, that are not my original contribution. I have used quotation marks to identify verbatim sentences and given credit to the original authors/sources.

I affirm that no portion of my work is plagiarized, and the experiments and results reported in the report are not manipulated. In the event of a complaint of plagiarism and the manipulation of the experiments and results, I shall be fully responsible and answerable

Name : KM. ANUPRIYA
Roll. No. : 1900290149054
Field : M.C.A. 6th Semester

KM. ANUPRIYA
(Candidate Sign)



CERTIFICATE

This is to certify that the thesis entitled “**Online Shopping**” submitted by Km. Anupriya Mishra (1900290149054) for the award of the degree of *Master of Computer Application* from Dr. APJ Abdul Kalam Technical University, Lucknow under my supervision as per the code of academic and research ethics.

The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree/diploma in this institute or any other institute/university. The thesis fulfills the requirements and regulations of the University and in my opinion meets the necessary standards for submission

Place : Ghaziabad

Date :

Signature of the Guide

The thesis is satisfactory / unsatisfactory

Signature of Internal Examiner

Signature of External Examiner

Approved by

Head of the Department

Mobiloitte Technologies (I) Pvt. Ltd.
D-115, Okhla Phase-1, New Delhi - 110020, India
Phone: +91-11-4649 9900
Email: info@mobiloitte.com
Web: www.mobiloitte.com
CIN# U72300DL2009PTC188611
GSTIN# 07AAGCM0385H1ZG

Dated: July 30, 2021

Certificate of Training

This is to certify that **Anu Priya** has successfully completed his training from **December 08, 2020** to **March 07, 2021** with Mobiloitte in **Software Development**.

During the period of training with us, he was found **punctual, hardworking, and inquisitive**.

We wish him good luck in future endeavors.

Sincerely,
Mobiloitte Technologies (I) Pvt. Ltd.



Pankaj Kumar
(For Mobiloitte Technologies)

ABSTRACT

Online shopping is also known as E-shopping; it is the process of buying and selling of goods and services through internet. It has become very popular in present days, due increasing the usage of internet and smart phone users, internet has become major platform for E-commerce and online shopping. Without internet it is impossible to imagine ECommerce. The consumers will buy various products like clothing, shoe, electronic items and services through online shopping according to their taste and preferences; it is a mode of zero channels of distribution means consumers will purchase the products directly from producers without any intermediaries or middlemen. It saves for lot of precious time, energy and money.

This study is conducted with the objectives of analyzing and understating the consumer's perception towards online shopping, to understand various problems faced by consumers at the time of online shopping and gives effective solutions to overcome such problems. It would make searching, viewing and selection of a product easier. It contains a sophisticated search engine for users to search for products specific to their needs. The search engine provides an easy and convenient way to search for products where a user can search for a product interactively and the search engine would refine the products available based on the user's input. The user can then view the complete specification of each product. They can also view the product reviews and write their own reviews. The application also provides a drag and drop feature so that a user can add a product to the shopping cart by dragging the item in to the shopping cart. The main emphasis lies in providing a user-friendly search engine for effectively showing the desired results and its drag and drop behavior.

This project is an attempt to provide the advantages of online shopping to customers of a real shop. It helps buying the products in the shop anywhere through internet by using an android device. Thus, the customer will get the service of online shopping and home delivery from his favorite shop. This system can be implemented to any shop in the locality or to multinational branded shops having retail outlet chains. If shops are providing an online portal where their customers can enjoy easy shopping from anywhere, the shops won't be losing any more customers to the trending online shops such as flip cart or eBay. Since the application is available in the Smartphone it is easily accessible and always available.

ACKNOWLEDGEMENTS

Success in life is never attained single handedly. My deepest gratitude goes to **Mr. ANKIT VERMA** for guidance, help and encouragement throughout my research work. Their enlightening ideas, comments, and suggestions. Words are not enough to express my gratitude to **Dr. Ajay Kumar Shrivastava, Professor and Head, Department of Computer Applications,** for his insightful comments and administrative help at various occasions. Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

Place : Ghaziabad

KM. ANUPRIYA

Date :

TABLE OF CONTENTS

Chapter No.	Contents	Page No.
	Title Page	1
	Declaration	2
	Certificate	3
	Abstract	4
	Acknowledgement	5
	Table of Contents	6
	List of Tables	8
	List of Figures	9
1	Introduction	10
	1.1 Existing System	11
	1.1.1 Drawback in the Existing System	11
2	Literature Review	12
3	System Analysis	14
	3.1 Purpose and Main Objective	14
	3.2 Scope	14
	3.3 Need for the Proposed System	14
	3.3.1 Feature and Benefits	15
	3.4 Feasibility	15
	3.4.1 Technical Feasibility	15
	3.4.2 Financial Feasibility	16
	3.4.3 Operational Feasibility	16
	3.4.4 Economic Feasibility	16
	3.5 Features of the proposed system	17
4	System Requirements Specifications	19
	4.1 User Class and Characterstics	19

4.2	Functional Requirements	19
4.3	Performance Requirement	20
4.4	Non-Functional Requirement	20
4.5	External Interface Requirement	20
4.5.1	User Interface	20
4.5.2	Hardware Interface	21
4.5.3	Software Interface	21
4.5.4	Communication Interface	21
4.6	General Constraints	21
4.7	Assumptions	21
4.8	Dependencies	21
5	System Design and its Specifications	23
5.1	Architectural Design	23
5.1.1	Data Flow Diagram	23
5.1.2	ER Diagram	26
6	Coding and Functionality (Screenshots)	26
	Functionality (Screenshots)	41
7	Testing	58
6.1	Unit Testing	58
6.2	Integration Testing	58
6.3	Validation Testing	59
8	Results and Challenges	60
9	Conclusion, Limitations and Future Scope	61
	Reference	62

LIST OF FIGURES

Figure	Title	Page
No.		No.
1	SYTEM DATA FLOW DIAGRAM	21
2	CONTEXT ANALYSIS DIAGRAM	22
3	E-R DIAGRAM	24
4	FUNCTIONALITY (SCREEN SHOTS)	41

CHAPTER 1

INTRODUCTION

Online shopping is the process whereby consumers directly buy goods, services etc. from a seller interactively in real-time without an intermediary service over the internet.

Online shopping is the process of buying goods and services from merchants who sell on the Internet. Since the emergence of the World Wide Web, merchants have sought to sell their products to people who surf the Internet. Shoppers can visit web stores from the comfort of their homes and shop as they sit in front of the computer. Consumers buy a variety of items from online stores. In fact, people can purchase just about anything from companies that provide their products online. Books, clothing, household appliances, toys, hardware, software, and health insurance are just some of the hundreds of products consumers can buy from an online store.

Many people choose to conduct shopping online because of the convenience. For example, when a person shops at a brick-and-mortar store, she has to drive to the store, find a parking place, and walk throughout the store until she locates the products she needs. After finding the items she wants to purchase, she may often need to stand in long lines at the cash register.

Despite the convenience of online shopping, not everyone chooses to purchase items and services online. Some people like the idea of physically going to a store and experiencing the shopping process. They like to touch the merchandise, try on clothing, and be around other people. Online shopping doesn't permit shoppers to touch products or have any social interaction. It also doesn't allow them to take the merchandise home the same day they buy it.

Online shopping allows you to browse through endless possibilities, and even offers merchandise that's unavailable in stores. If you're searching for a niche product that may not be distributed locally, you're sure to find what you're looking for on the internet. What's even more useful is the ability to compare items, similar or not, online. You can search through multiple stores at the same time, comparing material quality, sizes and pricing simultaneously. Shopping via the internet eliminates the need to sift through a store's products with potential buys like pants, shirts, belts and shoes all slung over one arm. Online shopping also eliminates the catchy, yet irritating music, as well as the hundreds, if not thousands, of other like-minded individuals who seem to have decided to shop on the same day.

Say 'goodbye' to the days when you stood in line waiting, and waiting, and waiting some more for a store clerk to finally check out your items. Online shopping transactions occur instantly- saving you time to get your other errands done! Additionally, unlike a store, online shopping has friendly customer service representatives available 24 hours a day, 7 days a week

to assist you with locating, purchasing and shipping your merchandise.

SYSTEM STUDY

Information systems projects“ originate from many reasons: to achieve greater speed in processing data, better accuracy and improved consistency, faster information retrieval, integration of business areas, reduced cost and better security. The sources also vary project proposals originate with department managers, senior executives and systems analysis.

Sometimes the real origin is an outside source, such as a government agency which stipulates a systems requirement the organization must meet. When the request is made, the first systems activity, the preliminary investigation, begins. The activity has three parts: request clarification, feasibility study and request approval.

1.1 Existing System:

The existing system was an automated system. But it was found to be inefficient in meeting the growing demands of population.

1.1.1 Drawbacks in the existing systems:

Disadvantage of the existing system:

Time Consuming

Expensive Needed an agent

We have to out for that.

CHAPTER 2

LITERATURE REVIEW

Online shopping is developing rapidly on the Internet today, but the amount of money involved remains very low. The total sales figures of Internet shopping represent only a small percentage of at-home shopping sales. Not only do problems of security and confidentiality constitute a real obstacle to its development, but the buyer inevitably has numerous questions concerning the delivery, exchange policy, and possible additional charges created by returning the product. All of these different risk forms inhibit the expansion of this new mode of purchase. According to Burke(1997), companies looking to develop their online sales need to search for ways to reduce the risk perceived by the consumer.[1]

The purpose of this study is to examine the consumer shopping channel extension focusing on attitude shift from offline to online store with a theoretical approach. Design/methodology/approach

– Two hundred and sixty two students in a large US midwestern university participated and provided usable survey responses.[2]

Consumers visit the shopping places not only for buying products and services, but they also use the places for meeting, socializing, and other entertaining activities. Therefore, mall developers and retailers must

attempt to make their shopping places most attractive in the views of shoppers.[4]

Forouhandeh Behnam et al., (2011) incontestable Warrant, Assurance, web site manoeuvrability and pleasure as factors that perceived because the on-line searching edges. Jush and Ling, (2012) additional that customers will relish on-line buying twenty four hour per day. Shoppers can purchase any merchandise and services anytime at everyplace. On-line searching is additional user friendly compare to future searching as a result of shoppers will simply accomplish his needs simply with a click of mouse while not effort their home.[5]

The online market for buying groceries and other consumable products is comparatively smaller but is starting to show promise. While durables are the starting point of adoption, consumables are attractive due to the frequency of purchase. Aside from online purchasing, digital is an increasingly important research and engagement platform. Consumable categories are not likely reach the same level of online prominence as non-consumable categories due to the hands-on buying nature and perishability of the products, but the market is wide open and an eager audience is at the ready. Master Card Worldwide Insights, (2008)

Studied that internet

penetration, income levels and cultural factors are key drivers of online shopping. In China and India huge growth in online shopping is expected as income and internet penetration rises. Credit cards are preferred payment mode in online shopping. The rising population of upper-middle income and increasing income level will probably boost the online shopping markets in China and India

Forrester Research, (2012) found that India's eCommerce market is at an early stage but is expected to see huge growth over the next four to five years. Retailers have a sizeable opportunity as the online population starts to spend more and buy more frequently online. Two key areas that companies must focus on in all markets are localized payment and fulfillment options. Over the past 12 months, venture capitalists have invested heavily in India's eCommerce market, new players have emerged, and the eCommerce ecosystem has developed, presenting a huge opportunity for companies willing to work through some of the logistics and payments challenges in India. Michal Pilik, (2012) examined that online buying behavior is affected by various factors like, economic factors, demographic factors, technical factors, social factors, cultural factors, psychological factors, marketing factors and legislative factors. Customers choose an online-shop mainly based on references, clarity and menu navigation, terms of delivery, graphic design and additional services. Complicated customers read discussions on the Internet before they spend their money on-line and when customers are unable to find the product

quickly and easily they leave online-shop. Dibb et al., (2001), Jobber, (2001), Kotler, (2003) described Consumer buying process as learning, information-processing and decision-making activity divided in several consequent steps: Problem identification, Information search, Alternatives evaluation, Purchasing decision, Post-purchase behavior. Efthymios Constantinides, (2004) identified the main constituents of the online experience as follows: the functionality of the Web site that includes the elements dealing with the site's usability and interactivity, the psychological elements intended for lowering the customer's uncertainty by communicating trust and credibility of the online vendor and Web site and the content elements including the aesthetic aspects of the online presentation and the marketing mix. Usability and trust are the issues more frequently found to influence the online consumer's behavior. Karayanni, (2003) examined that discriminating of potential determinants between web-shoppers and non shoppers. The most major discriminant variable between web shoppers and non shoppers was found to be web-shopping motives concerning time efficiency, availability of shopping on 24 hours basis and queues avoidance

CHAPTER 3

SYSTEM ANALYSIS

- a) This system is all about the converting the shopping system from manual to online.
- b) Customer can buy products online after login to the site.
- c) Administrator is adding product to database.
- d) Administrator can edit or delete the products from the database.
- e) After buying and making payment the products are send to customers address that he hasgiven.
- f) Customer can write feedback for the product or services.
- g) Admin can see daily sell and feedback given by customer.
- h) Administrator is adding the delivery report to the database.
- i) Both admin and customer can see the delivery report.

3.1 Purpose:

Online shopping tries to enhance access to care and improve the continuity and efficiency of services. Depending on the specific setting and locale, case managers are responsible for a variety of tasks, ranging from linking clients to services to actually providing intensive shopping and delivery services themselves

Main objective

- a) To shop wile in the comfort of your own home, without having to step out of the door.
- b) sell at lower rate due to less overhead.
- c) provide home delivery free of cost.
- d) No wait to see the products if someone else is taking that.

3.2 Scope:

This product has great future scope. Online shopping Internet software developed on and for the Windows and later versions environments and Linux OS. This project also provides security with the use of Login-id and Password, so that any unauthorized users can not use your account. The only Authorized that will have proper access authority can access the software.

3.3 Need for the proposed system:

The online shopping (HOME SHOP) is an easy to maintain, ready to run, scalable, affordable and reliable cost saving tool from Software Associates suited for small, medium, and large shopping complex and shopping malls.

3.3.1 Features and Benefits:

- Providing security
- Low cost
- Basic computer knowledge required
- Configurable and extensible application UI design
- The proposed system can be used even by the naïve users and it does not require any educational level, experience, and technical expertise in computer field but it will be of good use if the user has the good knowledge of how to operate a computer.

3.4 Feasibility study:

- A feasibility study is a short, focused study, which aims to answer a number of questions:
- Does the system contribute to the overall objectives of the organizations?
- Can the system be implemented using current technology and within given cost and schedule constraints?
- Can the system be integrated with systems which are already in place?

In preliminary investigation feasibility study has four aspects.

- Technical Feasibility
- Financial Feasibility
- Operational Feasibility
- Economic Feasibility

3.4.1 Technical Feasibility:

- Is the project feasibility within the limits of current technology?
- Does the technology exist at all?
- Is it available within given resource constraints (i.e., budget, schedule)?

Technical issues involved are the necessary technology existence, technical guarantees of accuracy, reliability, ease of access, data security, aspects of future expansion.

- Technology exists to develop a system.
- The proposed system is capable of holding data to be used.
- The proposed system is capable of providing adequate response and regardless of the number of users
- The proposed system being modular to the administrator, if he/she wants can add more features in the future and as well as be able to expand the system.
- As far as the hardware and software is concerned, the proposed system is completely liable with proper backup and security.
- Hence, we can say that the proposed system is technically feasible.

3.4.2 Financial Feasibility:

- Is the project possible, given resource constraints?
- Are the benefits that will accrue from the new system worth the costs?
- What are the savings that will result from the system, including tangible and intangible ones?
- What are the development and operational costs?

3.4.3 Operational Feasibility:

Define the urgency of the problem and the acceptability of any solution; if the system is developed, will it be used? Includes people-oriented and social issues: internal issues, such as manpower problems, labour objections, manager resistance, organizational conflicts and policies; also external issues, including social acceptability, legal aspects and government regulations.

If the system meets the requirements of the customers and the administrator we can say that the system is operationally feasible.

The proposed system will be beneficial only if it can be turned into a system which will meet the requirements of the store when it is developed and installed, and there is sufficient support from the users

The proposed system will improve the total performance.

- Customers here are the most important part of the system and the proposed system will provide them with a convenient mode of operation for them.
- The proposed system will be available to the customers throughout the globe.
- The proposed system will provide a better market for different dealers.

Hence, the proposed system is operationally feasible.

3.4.4 Economic Feasibility

Economic Feasibility is the most frequently used method for evaluating the effectiveness of the proposed system if the benefit of the proposed system outweighs the cost then the decision is made to design and implement the system.

- The cost of hardware and software is affordable.
- High increase in the amount of profit earned by going global.
- Easy and cheap maintenance of the system possible.
- Very cheap price for going global.

Hence, the proposed system is economically feasible.

3.5 FEATURES OF THE PROPOSED SYSTEM

- The proposed system is flexible both for the administrators and the customers visiting the website.
- The proposed system provides a unique platform for different silk vendors to interact using the same platform.
- The proposed system allows easy promotion of the site through emails and newsletters.
- The proposed system gives information about the delivery and present status of their orders.
- Management of data is easy.
- Security is provided wherever necessary.

PROPOSED SYSTEM

In the proposed website there are different parts or modules which are summarized as follows

CUSTOMER REGISTRATION:

Customers are required to register on the website before they can do the shopping. The website also provides several features for the non-registered user. Here they can choose their id and all the details regarding them are collected and a mail is sent to the email address for confirmation.

SHOPPING CART:

Shopping cart module tries to simulate the working of a store where user can view each design, color, size and price of the product available. The items they like can be added to the logical cart and can be removed if not required later. Billing and other payment related matters are handled here.

ADMINISTRATION:

This is the part of the website where the administrators can add delete or update the product information. Administrators are also responsible for adding and deleting the customers from the website. In addition, newsletter and promotions are also handled by the site administrator via e-mail.

SEARCH:

This facility is provided to both registered and unregistered user. User can search for the availability and type of products available on the website.

EMAILING:

Emailing module is concerned about promotions and newsletter and is handled by the administrator. This module is also concerned about sending activation and warning mails.

CHAPTER 4

SYSTEM REQUIREMENTS SPECIFICATIONS

System requirements are expressed in a software requirement document. The Software requirement specification (SRS) is the official statement of what is required of the system developers. This requirement document includes the requirements definition and the requirement specification. The software requirement document is not a design document. It should set out what the system should do without specifying how it should be done. The requirement set out in this document is complete and consistent.

The software specification document satisfies the following: -

- It specifies the external system behaviors.
- It specifies constraints on the implementation.
- It is easy to change.
- It serves as reference tool for system maintainers.
- It records forethought about the life cycle of the system.
- It characterizes acceptable response to undesired events.

4.1 User Class and Characteristics:

- General public
- Customers
- Administrator
- General public can use the system to see the product, their prices and quantity available.
- General user cannot buy the products.
- Customers are using for viewing and buying the products.
- Customer can also write feedbacks for products and services
- Administrators can add, edit & delete products. And provide services to the customer.
- Administrator can see the daily sell. Can also see the feedback given by the customer.
- Administrator maintaining the deliveries.

4.2 Functional Requirements:

The System must provide following functionalities—

- Keeping records of admission of customers.
- keeping the records of products.
- keeping the daily sell.
- Storing the feedback given by the customer.
- keeping details about the product it is delivered

or not. etc.

- Storing the items selected by the customer in the temporary storage.

4.3 Performance Requirements:

In order to maintain an acceptable speed at maximum number of uploads allowed from a particular customer will be any number of users can access the system at any time. Also, connections to the servers will be based on the criteria of attributes of the user like his location, and server will be working whole 24X 7 times.

4.4 Non-Functional Requirements:

Following Non-functional requirements will be there in the Insurance on internet:

- Secure access of confidential data (customer's details).
- 24 X 7 availability.
- Better component design to get better performance at peak time.

Flexible service based architecture will be highly desirable for future extension Nonfunctional requirements define system properties and constraints It arise through user needs, because of budget constraints or organizational policies, or due to the external factors such as safety regulations, privacy registration and so on.

Various other Non-functional requirements are:

1. Security
2. Reliability
3. Maintainability
4. Portability
5. Extensibility
6. Reusability
7. Application Affinity/Compatibility
8. Resource Utilization

4.5 External Interface Requirements:

4.5.1 User Interface:

User of the system will be provided with the Graphical user interface, there is no command line interface for any functions of the product. The user will get 2 pages: Login page followed by Password

4.5.2 Hardware Interface:

Hardware requirements for Insurance on internet will be same for both the parties which are follows:

Processor : - Pentium I or above.
RAM : - 128 MB or above.
HD : - 20 GB or above.
NIC : - For each party

4.5.3 Software Interface:-

Software required to make working of product are:-

- a) Front end- visual studio 2010
- b) Back end- Mongo DB

4.5.4 Communication Interfaces

The two parties should be connected through either by LAN or WAN for the communication. Communication channels



4.6 General Constraints

- The interface will be in English only.
- The system is working for single server.:Sender - Receiver
- There is no maintainability or backup so availability will get affected. The system is a single usersystem.
- GUI features available.

4.7 Assumptions

User must be trained for basic computer functionalities. User must have the basic knowledge of English

The system must be able to respond to database software within reasonable time

4.8 Dependencies

The product does require back-end database Mongo DB for storing the username and password for different types of user of the system as well as various databases regarding various insurance information.

CHAPTER 5

SYSTEM DESIGN SPECIFICATION

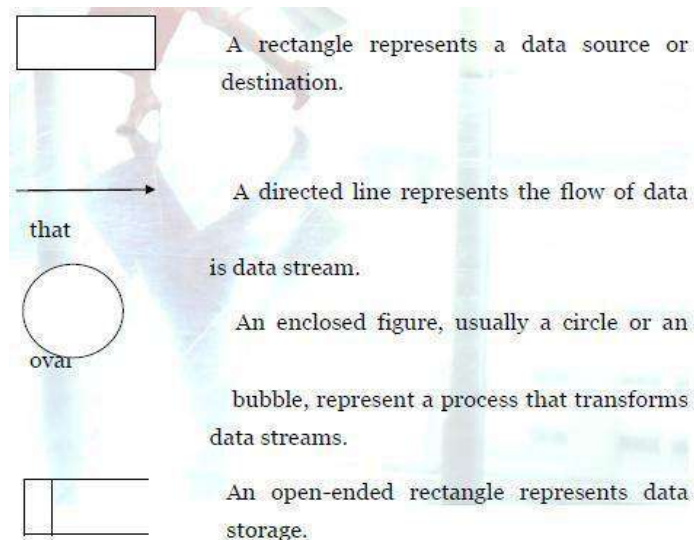
5.1 ARCHITECTURAL DESIGN

5.1.1 DATA FLOW DIAGRAMS:

Data flow diagrams (DFD) was first developed by LARRY CONSTANTINE as way representing system requirements in a graphical form, this lead to modular design. A DFD describes what data flow (logical) rather than how they are processed, so it does not depend on hardware, software, data structure or file organization. It is also known as “bubble chart”.

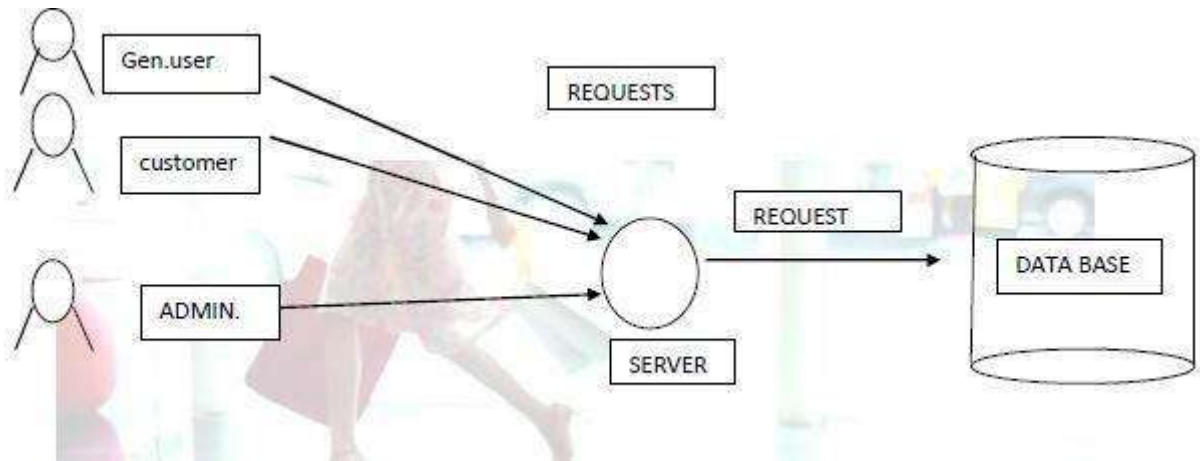
A Data Flow Diagrams is a structured analysis and design tool that can be used for flowcharting in place of, or in association with, information-oriented and process-oriented systems flowcharts. A DFD is a network that describes the flow of data and the processes that change, or transform, data throughout a system. This network is constructed by using a set of symbols that do not imply a physical implementation. It has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. So it is the starting point of the design phase that functionality decomposes the requirement specifications down to the lowest level of detail.

The symbols used to prepare DFD do not imply a physical implementation, a DFD can be considered to an abstract of the logic of an information-oriented or a process-oriented system flow-chart. For these reasons DFDs are often referred to as logical data flow diagrams. The four basic symbols used to construct data flow diagrams are shown below:



These are symbols that represent data flows, data sources, data transformations and data storage. The points at which data are transformed are represented by enclosed figures, usually circles, which are called nodes. The principal processes that take place at nodes are:

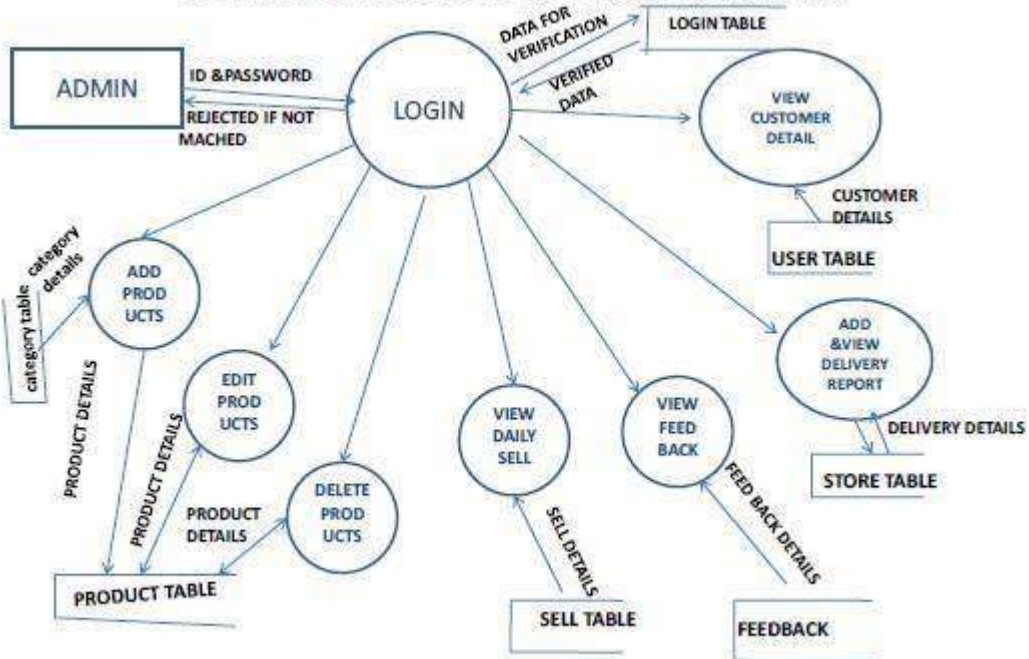
1. combining data streams
2. splitting data streams
3. modifying data streams



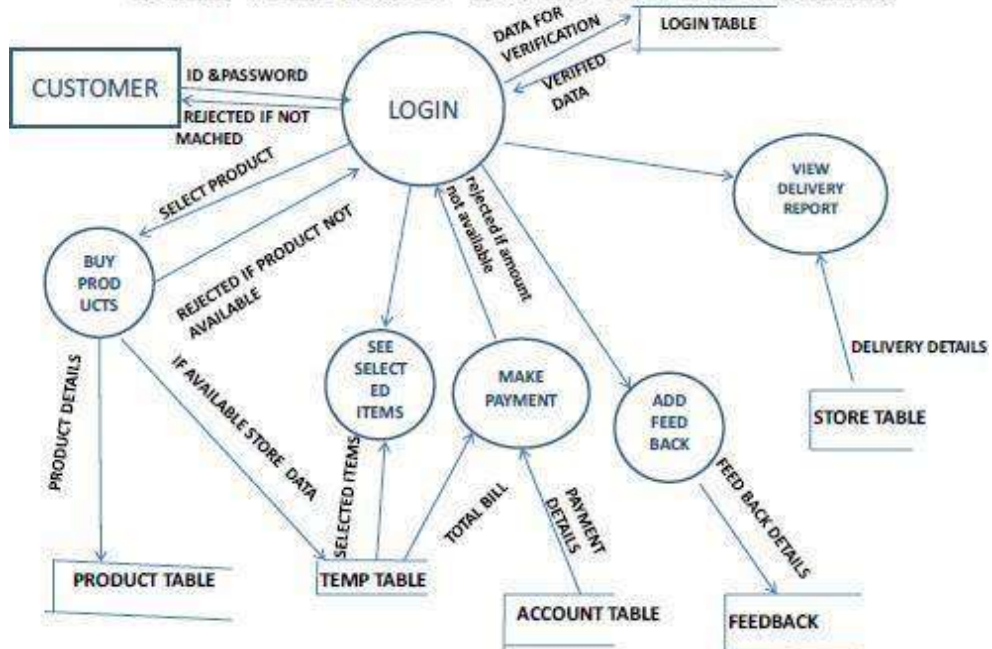
CAD(CONTEXT ANALYSIS DIAGRAM)



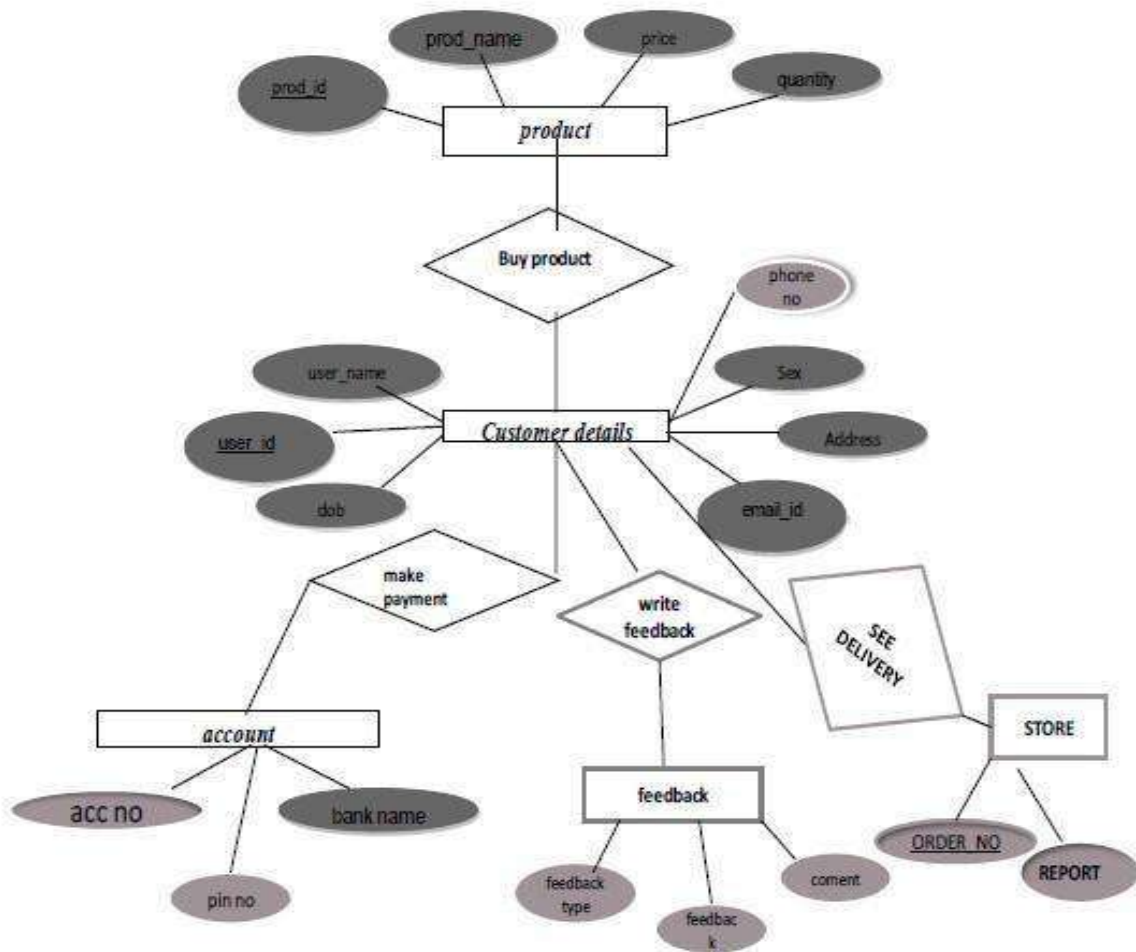
1 LEVEL DFD FOR ADMIN



1 LEVEL DFD FOR CUSTOMER



ER DIAGRAM



5.1.2 ER DIAGRAM

CHAPTER 6

Coding and Functionality

Add product

```
addProduct: async (req, res) => {
  try {
    userModel.findOne({ _id:req.userId, status: "ACTIVE", userType: "VENDOR" }, async (err, result) => {
      if (err) {
        response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, err)
      } else if (!result) {
        response(res, ErrorCode.NOT_FOUND, ErrorMessage.userNotFound, [])
      }
      else {
        req.body.productImage = await userFunction.uploadImage(req.body.productImage);
        req.body.productOwnerId = result._id;
        coordinates = [parseFloat(req.body.lng), parseFloat(req.body.lat)];
        req.body.location = { coordinates };
        req.body.productVideo = await userFunction.uploadVideo(req.body.productVideo);
        qr = [result.email, req.body.productName]
        req.body.productQRCode = await userFunction.qrCodeGenerator(qr);
        var productStartTime = new Date(req.body.productStartTime);
        var miliseconds = productStartTime.getTime() + (60 * 60 * 24 * 1000);
        req.body.productEndTime = new Date(miliseconds).toISOString(); new
        productModel(req.body).save((errSave, resultSave) => {
          if (errSave) {
            response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, errSave)
          } else {
            response(res, SuccessCode.SUCCESS, SuccessMessage.productAdded, resultSave)
          }
        }
      )
    }
  } catch (error) {
    response(res, ErrorCode.WENT_WRONG, ErrorMessage.wentWrong, error)
  }
}
```

List product

```
allProductList: (req, res) => {
  try {

    let query = { status: { $ne: "DELETE" } };
    if (req.body.search) {
      query.productName = new RegExp('^' + req.body.search, "i");
    }
    if (req.body.fromDate && !req.body.toDate) {
      query.$or = [{ productStartTime: { $gte: req.body.fromDate } }, { productEndTime: { $lte: req.body.fromDate } }]
    }
    if (!req.body.fromDate && req.body.toDate) {
      query.$or = [{ productStartTime: { $gte: req.body.toDate } }, { productEndTime: { $lte: req.body.toDate } }]
    }
    if (req.body.fromDate && req.body.toDate) {
      query.$or = [{ productStartTime: { $gte: req.body.fromDate } }, { productEndTime: { $lte: req.body.toDate } }]
    }
    if (req.body.categoryType) {
      query.categoryType = req.body.categoryType;
    }
    let option = {
      page: req.query.page || 1,
      limit: parseInt(req.query.limit) || 10,
      sort: { createdAt: -1 },
      populate: { path: 'productOwnerId', select: 'email mobileNumber' }
    };
    productModel.paginate(query, option, (err, result) => {
      if (err) {
        return res.send({
          statusCode: 500,
          responseMessage: "Internal server error.",
          error: err
        });
      } else if (result.docs.length == 0) {
        return res.send({
          statusCode: 404,
          responseMessage: "NO products."
        });
      } else {
        return res.send({
          statusCode: 200,
          responseMessage: "Products found successfully",
          result: result
        });
      }
    })
  } catch (error) {
    return res.send({

```

```
responseCode: 500,  
responseMessage: "Internal server error.",  
error: error })),
```

Product_Near_Me

```
productListNearMe: (req, res) => {
  try {
    productModel.aggregate(
      [
        {
          "$geoNear": {
            "near": {
              "type": "Point",
              "coordinates": [parseFloat(req.body.lng), parseFloat(req.body.lat)]
            },
            "spherical": true,
            "distanceField": "dis",
            "maxDistance": 5000,
          }
        },
        { "$skip": 0 },
        { "$limit": 2 }
      ],
      function (err, products) {
        if (err) throw err;
        console.log(products);

        products = products.map(function (x) {
          delete x.dis;
          return productModel(x);
        });

        productModel.populate(products, { path: "_id", select: 'productName' }, function (err, result) {
          if (err) {
            response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, err)
          }
          else if (result.length == 0) {
            response(res, ErrorCode.NOT_FOUND, ErrorMessage.productNotFound, [])
          }
          else {
            return res.send({
              statusCode: 200,
              responseMessage: "Nearby products.",
              result: result
            })
          }
        });
      }
    );
  } catch (error) {
    console.log(error)
    response(res, ErrorCode.WENT_WRONG, ErrorMessage.wentWrong, error)
  }
},
```

Sign Up page

```
signUp: (req, res) => { try
{
  console.log("i am here====>", req.body);
  userModel.findOne({ $or: [{ email: req.body.email }, { mobileNumber: req.body.mobileNumber }], {userName:req.body.userName}}, status: { $ne: "DELETE" } }, (err, result) => { //,represent and propertyconsole.log("fetch records====>", err, result);
    if (err) {
      response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, err)
    }
    else if (result) {
      if (req.body.email == result.email) {
        response(res, ErrorCode.ALREADY_EXIST, ErrorMessage.emailAlreadyCreadted, [])
      } else if (req.body.mobileNumber == result.mobileNumber) {
        response(res, ErrorCode.ALREADY_EXIST, ErrorMessage.phoneNumberAlreadyCreadted, [])
      }
    }
    else {
      response(res, ErrorCode.ALREADY_EXIST, ErrorMessage.userNameAlreadyCreadted, [])
    }
  }
  else {
    /*****password encryption*****/let password =
    bcrypt.hashSync(req.body.password);
    req.body.password = password;
    /*****otp*****/ req.body.otp =
    userfunction.generateOTP();
    /*****mail*****/ send*****/
    req.body.otpExpireTime = Date.now()
    userfunction.sendMail(req.body.email, req.body.otp, (emailErr, emailResult) => { if
    (emailErr) {
      response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, emailErr)
    }
    } else {
      new userModel(req.body).save((errSave, resultSave) => { if
      (errSave) {
        response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, errSave)
      } else {
        response(res, SuccessCode.SUCCESS, SuccessMessage.signUpSucces, resultSave)
      }
    }
  }
  }
  catch (error) {
    response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, error)
  }
},
```

Otp Verify

```
otpVerify: (req, res) => {
  try {
    console.log("i am here====>", req.body);
    userModel.findOne({ $or: [{ email: req.body.email }, { mobileNumber: req.body.email }], {userName:req.body.email}}, {status: "ACTIVE"}), (err, result) => {
      console.log("fetch records====>", err, result);
      if (err) {
        response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, err)
      }
      else if (!result) {
        response(res, ErrorCode.NOT_FOUND, ErrorMessage.userNotFound, [])
      }
      else {
        var time = new Date().getTime();
        var dbTime = result.otpExpireTime;
        var timeDiff = time - dbTime;
        console.log(timeDiff)
        if (timeDiff < (5 * 60 * 1000)) { if
          (req.body.otp == result.otp) {
            userModel.findOneAndUpdate(
              { _id: result._id },
              { $set: { emailVerify: true } },
              { new: true },
              (updateErr, updateResult) => {
                if (updateErr) {
                  response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, updateErr);
                }
                else {
                  response(res, SuccessCode.OTP_VERIFIED, SuccessMessage.otpVerified, updateResult);
                }
              })
          } else {
            response(res, ErrorCode.INVALID_OTP, ErrorMessage.invalidOtp, [])
          }
        }
      }
    }
  }
  } else {
    response(res, ErrorCode.OTP_EXPIRED, ErrorMessage.otpExpired, [])
  }
})
} catch (error) {
  response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, error)
},
},
```

Forgot password

```
forgotPassword: (req, res) => { try {
  userModel.findOne( { email: req.body.email, status: { $ne: "DELETE" } }, (err, result) => { console.log("fetch
    record=====>", err, result);
    if (err) {
      response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, err)
    }

    else if (!result) {
      response(res, ErrorCode.NOT_FOUND, ErrorMessage.userNotFound, []);
    }
    else {

      req.body.otp = userfunction.generateOTP(); var
      newOtp = req.body.otp;//to store send otpvar time =
      Date.now();//to store current time

      userfunction.sendMail(req.body.email, req.body.otp, (emailErr, emailResult) => { if
        (emailErr) {
          response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, emailErr);
        }
        else {
          /*****To update the time ,email verify,otp in
db*****/
          userModel.findByIdAndUpdate(
            { _id: result._id },
            { $set: { emailVerify: false, otp: newOtp, otpExpireTime: time } },
            { new: true }, (updateErr, updateResult) => { if
              (updateErr) {
                response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, updateErr)
              } else {
                response(res, SuccessCode.SUCCESS, SuccessMessage.otpSend, updateResult)
              }
            });
          });
        }
      } catch (error) {
        response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, err)
      }
    },
  },
```


Login page

```
login: async (req, res) => {
  try {
    console.log(req.body);
    userModel.findOne({ $or: [{ email: req.body.email }, { mobileNumber: req.body.email }, {userName:req.body.email}], status: { $ne:
"DELETE" } }, async (err, result) => {
      console.log("fetch records====>", err, result);
      if (err) {
        response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, err)
      }
      /*****CHECK EMAIL IN DB*****/
      else if (!result) {
        response(res, ErrorCode.NOT_FOUND, ErrorMessage.userNotFound, []);
      }
      else {
        /*****COMPARE PASSWORD*****/
        if (result.emailVerify == true) {
          var secret = speakeasy.generateSecret();
          req.body.secret = secret.base32;
          var qrLink = await userfunction.qrCodeGenerator(secret.otppath_url);
          let value = bcrypt.compareSync(req.body.password, result.password);

          if (value == true) {

            var token = jwt.sign({ _id: result._id, email: result.email }, 'mobilloite', { expiresIn: '2h' });
            userModel.findOneAndUpdate(
              { _id: result._id },
              { $set: { secret: req.body.secret } },
              { new: true },
              (updateErr, updateResult) => {
                if (updateErr) {
                  response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, updateErr)
                }
                else {
                  response(res, SuccessCode.SUCCESS, SuccessMessage.loginSuccess, { updateResult, token, qrLink })
                }
              }
            )
          }
          else {
            response(res, ErrorCode.INVALID_CREDENTIALS, ErrorMessage.invalidPassword, [])
          }
        }
        else {
          response(res, ErrorCode.INVALID_CREDENTIALS, ErrorMessage.notVerified)
        }
      }
    })
  } catch (error) {
    response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, error)
  }
}
```

Two way authentication on login

```
twoFA: (req, res) => {
  try {
    userModel.findOne({ $or: [{ email: req.body.email }, { mobileNumber: req.body.email }], status: { $ne: "DELETE" } }, (err, result) => {
      console.log("fetch records====>", err, result);
      if (err) {
        response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, err)
      }
      else if (!result) {
        response(res, ErrorCode.NOT_FOUND, ErrorMessage.userNotFound, []);
      }
      else {
        var verified = speakeasy.totp.verify({
          secret: result.secret,
          encoding: 'base32',
          token: req.body.Token
        });
        if (verified == true) {
          response(res, SuccessCode.SUCCESS, SuccessMessage.verified, result)
        }
        else {
          response(res, ErrorCode.INVALID_CREDENTIALS, ErrorMessage.tokenExpireWrong, [])
        }
      }
    })
  } catch (error) {
    response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, error)
  }
},
```

Get Profile

```
getProfile: (req, res) => {
  console.log("i am here=====>", req.body);
  try {
    userModel.findOne({ _id: req.userId, status: { $ne: "DELETE" } }).select('-password').exec((err, result) => {
      console.log("fetch records====>", err, result);
      if (err) {
        response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, err)
      }
      else if (!result) {
        response(res, ErrorCode.NOT_FOUND, ErrorMessage.userNotFound, []);
      }
      else {
        response(res, SuccessCode.SUCCESS, SuccessMessage.getProfile, result)
      }
    })
  }
  catch {
    response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, error)
  }
},
```

Product add to cart

```
addToCart: (req, res) => {
  try {
    userModel.findOne({ _id: req.userId, userType: "CUSTOMER", status: { $ne: "DELETE" } }, (err, result)
=> {
      if (err) {
        response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, err)
      }
      else if (!result) {
        response(res, ErrorCode.NOT_FOUND, ErrorMessage.userNotFound, [])
      }
      else {
        productModel.findOne({ productName: req.body.productName, status: "ACTIVE" }, (err1, result1) => {
          if (err1) {
            response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, err1)
          } else if (!result1) {
            response(res, ErrorCode.NOT_FOUND, ErrorMessage.productNotFound, [])
          } else {
            req.body.userId = result._id;
            req.body.productId = result1._id;
            req.body.cost = result1.cost;
            cartModel.findOne({ userId: req.body.userId, productId: req.body.productId }, (err2, result2) => {
              if (err2) {
                response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, err1)
              }
              else if (result2) {
                response(res, ErrorCode.ALREADY_EXIST, ErrorMessage.gotoCart, result2)
              } else {
                new cartModel(req.body).save((SaveErr, SaveResult) => {
                  if (SaveErr) {
                    response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, SaveErr)
                  }
                  else {
                    response(res, SuccessCode.SUCCESS, SuccessMessage.productAdded, SaveResult)
                  }
                })
              }
            })
          }
        })
      }
    })
  } catch (error) { response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, error)
  },
},
```

Booking

```
Booking: async (req, res) => {
  try {
    cartModel.find({ userId: req.body.userId }, async (err, result) => {if
      (err) {
        response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, err)
      } else if (result.length == 0) {
        response(res, ErrorCode.NOT_FOUND, ErrorMessage.cartEmpty, []) }
      else {
        const sum = result.map(o => o.cost);
        console.log(sum)
        const total = sum.reduce((initial, value) => {
          return initial + value
        }, 0);
        console.log(total)
        const ids = result.map(obj => obj.productId)
        req.body.productsIds = ids;
        console.log(req.body.productsIds);

const cardToken = await userfunction.createStripeToken(req.body.cardNumber, req.body.expMnth,
req.body.expYear, req.body.cvc);
        const customerId = await userfunction.createCustomer(cardToken);
        const transactionId = await userfunction.createCharge(total, req.body.currencyType,
customerId.id);

        if (transactionId.paid == true) {
          req.body.cardToken = cardToken.id;
          req.body.UserId = req.body.userId;
          req.body.transactionId = transactionId.id;
          req.body.transactionStatus = "SUCCESS";
          new transactionModel(req.body).save((saveErr, saveResult) => {
            console.log("====>", saveErr, saveResult)
            if (saveErr) {
              response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError, saveErr)
            } else {
              cartModel.deleteMany({ userId: req.body.userId }, ((deleteErr, deleteResult) => {
                if (deleteErr) {
                  response(res, ErrorCode.INTERNAL_ERROR, ErrorMessage.serverError,
deleteErr);
                } else {
                  response(res, SuccessCode.SUCCESS, SuccessMessage.transactionSuccess,
saveResult);
                }
              }) })
            } else { response(res, ErrorCode.REQUEST_FAILED, ErrorMessage.transactionfailed, [])
          }
        }

      } catch (error) {
        response(res, ErrorCode.WENT_WRONG, ErrorMessage.wentWrong, error)}}}
```

Default Static Page(About us, Contact us)

```
mongoose.model("static", staticSchema).findOne({}, (err, result) => {

  if (err) {
    console.log("Static error :", err);
  }
  else if (result) {
    console.log("Default Static values");
  } else {
    let obj1 = {
      Type: "ContactUs",
      title: "Hello Customers",
      description: "Welcome to our Ecommerce Site. If you have any problem or queries regarding our product.\n
You can contact us on :\n 007123,9087654343."

    };
    let obj2 = {
      Type: "AboutUs",
      title: "ABOUT US",
      description: "Hello customers we Ecommerece site. We offer our customers over 5000 quality products, and
our list of product categories and product offerings is growing every day.PresentBazaar customers know
they're getting the best prices and exclusive offerson a huge range of computer technology products like
desktops, notebooks, printers, mobile phones, networking, digital cameras, software, storage and more. Plus, we offer
other interesting products such as LCD TVs, MP3 players, gaming and home electronics."
    }

    mongoose.model("static", staticSchema).create(obj1, obj2, (err1, result1) => {if
      (err1) {
        console.log("Default Static creation error :", err1)
      } else {
        console.log("Default Static content created", result1);
      }};)))
```

Chat Module

One to one chat

```

oneToOneChat: (req) => {
  try {
    let response = {};
    console.log("====>" + req.senderId);

    return new Promise((resolve, reject) => {
      userModel.findOne({ _id: req.senderId, status: "ACTIVE" }, (err, result) => {
        console.log(result, err)
        if (err) {
          response = { responseCode: 500, responseMessage: "Internal server error.", err };
          resolve(response)
        }
        else if (!result) {
          response = { responseCode: 404, responseMessage: "Sender not found." }
          resolve(response)
        }
        else {
          userModel.findOne({ _id: req.receiverId, status: "ACTIVE", userType: { $ne: result.userType } },
            (err1, result1) => {
              if (err1) {
                response = { responseCode: 500, responseMessage: "Internal server error.", err1 }
                resolve(response)
              }
              else if (!result1) {
                response = { responseCode: 404, responseCode: "Receiver not found." }
                resolve(response)
              }
              else {
                req.senderId = result._id
                req.receiverId = result1._id
                chatModel.findOne({ $or: [{ $and: [{ senderId: req.senderId }, { receiverId: req.receiverId }] },
                  { $and: [{ senderId: req.receiverId }, { receiverId: req.senderId }] }] }, (err2, result2) => { console.log(err2,
                  result2);
                  if (err2) {
                    response = { responseCode: 500, responseMessage: "Internal server error", err2 }
                    resolve(response)
                  }
                  else if (result2) {

                    Type = req.Type;
                    receiver = req.receiverId;
                    sender = req.senderId;
                    message = req.message;
                    time = Date.now()
                    req.messageDetail = [{ sender, receiver, Type, message, time }];
                    chatModel.findByIdAndUpdate({ _id: result2._id },
                      { $push: { messageDetail: req.messageDetail } }, { new: true }).populate([
                        { path: "senderId", select: "email name" }, { path: "receiverId", select: "email name" }
                      ]).exec((pushErr, pushSave) => {
                        if (pushErr) {

```

```
response = { responseCode: 500, responseMessage: "Internal server error", result:
```



```

pushErr }

                resolve(response)
            }
            else {
                response = { responseCode: 200, responseMessage: "Message send successfully.",
result: pushSave }
                resolve(response)
            }
        })
    }
    else {
        req.messageDetail = [{
            Type: req.Type,
            receiver: req.receiverId,
            sender: req.senderId,
            message: req.message,
            time: Date.now()
        }];
        chatModel(req).save((saveErr, saveResult) => {
            console.log("====>" + saveErr, saveResult);
            if (saveErr) {
                response = { responseCode: 500, responseMessage: "Internal server error", result:
saveErr }
                resolve(response)
            }
            else {
                response = { responseCode: 200, responseMessage: "Message send successfully.",
result: saveResult }
                resolve(response)
            }
        }
    )
} catch (error) {
    response = { responseCode: 501, responseMessage: "Something went wrong", error }
    console.log(response);
}
},

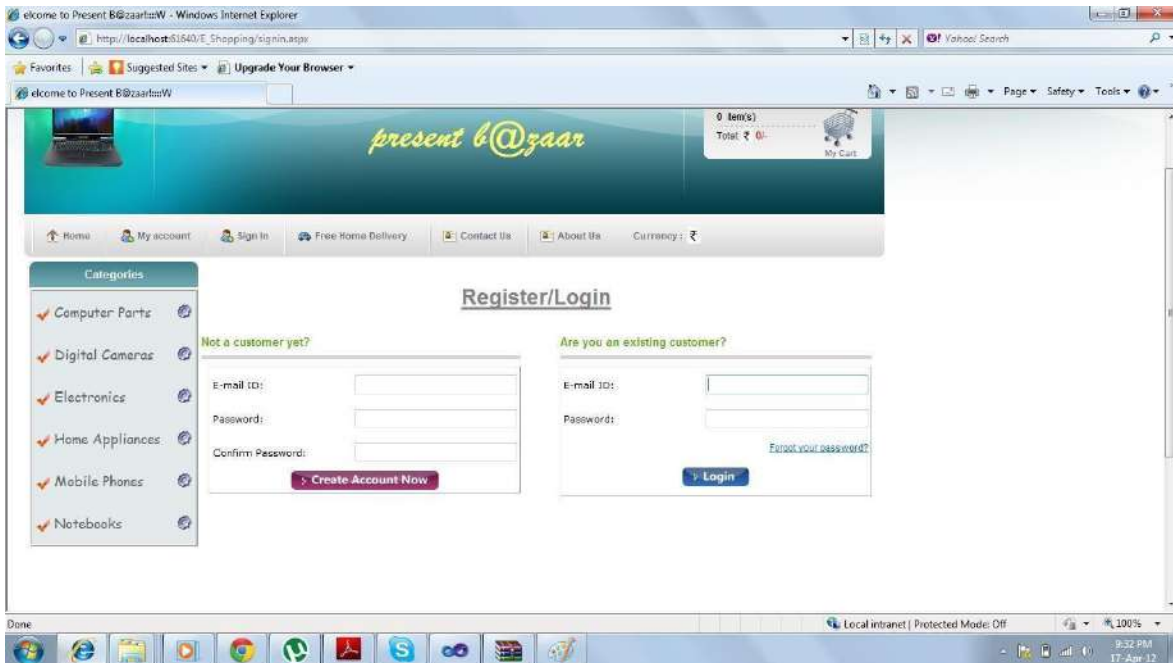
```

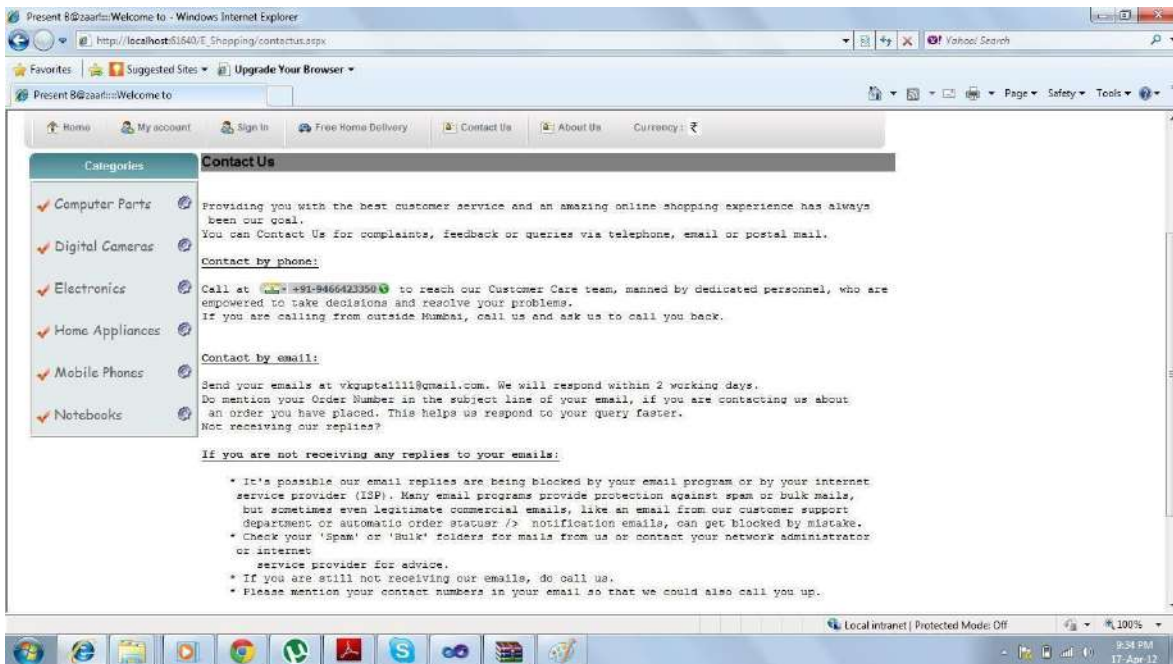
Chat History

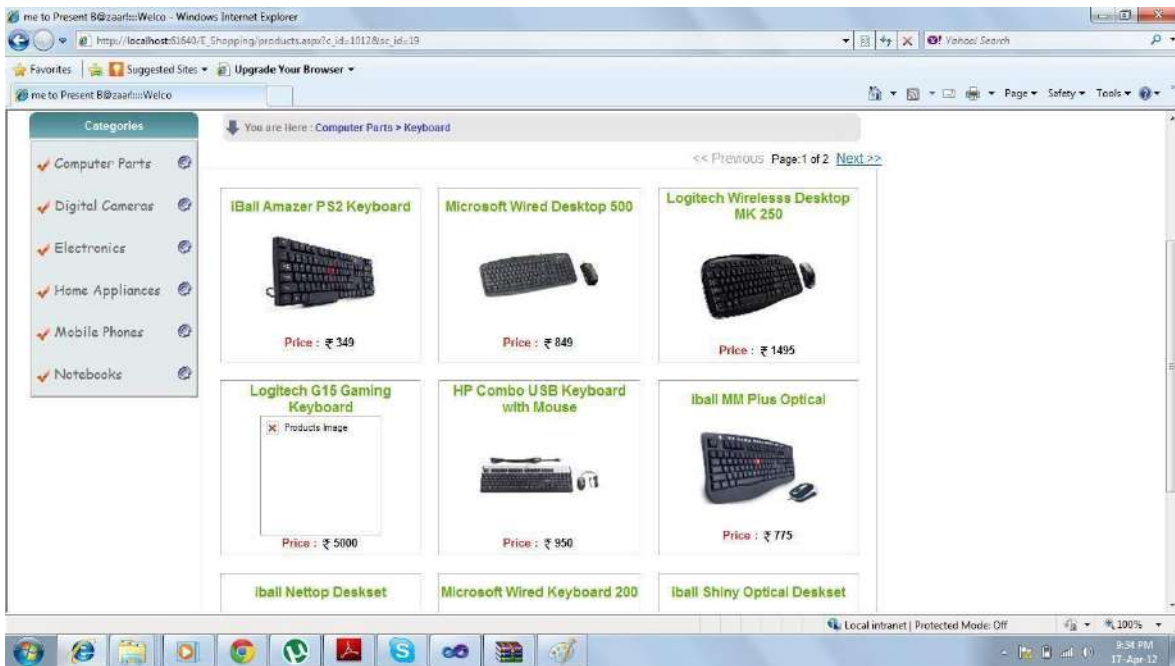
```
chatHistory: (req) => {
  try {
    let response = {};
    return new Promise((resolve, reject) => {
      chatModel.find({ $or: [{ senderId: req.userId }, { receiverId: req.userId }] }).populate([{ path: "senderId",
select: "email name" }, { path: "receiverId", select: "email name" }]).exec((err, result) => {
        if (err) {
          response = { responseCode: 500, responseMessage: "Internal server error.", result: err };
          resolve(response);
        } else if (result.length==0) {
          response = { responseCode: 404, responseMessage: "No Chat History is present." };
          resolve(response);
        } else {
          response = { responseCode: 200, responseMessage: "Chat history.", result: result }; resolve(response);
        }
      })
    })
  } catch (error) {
    console.log(error)
  }
}
```

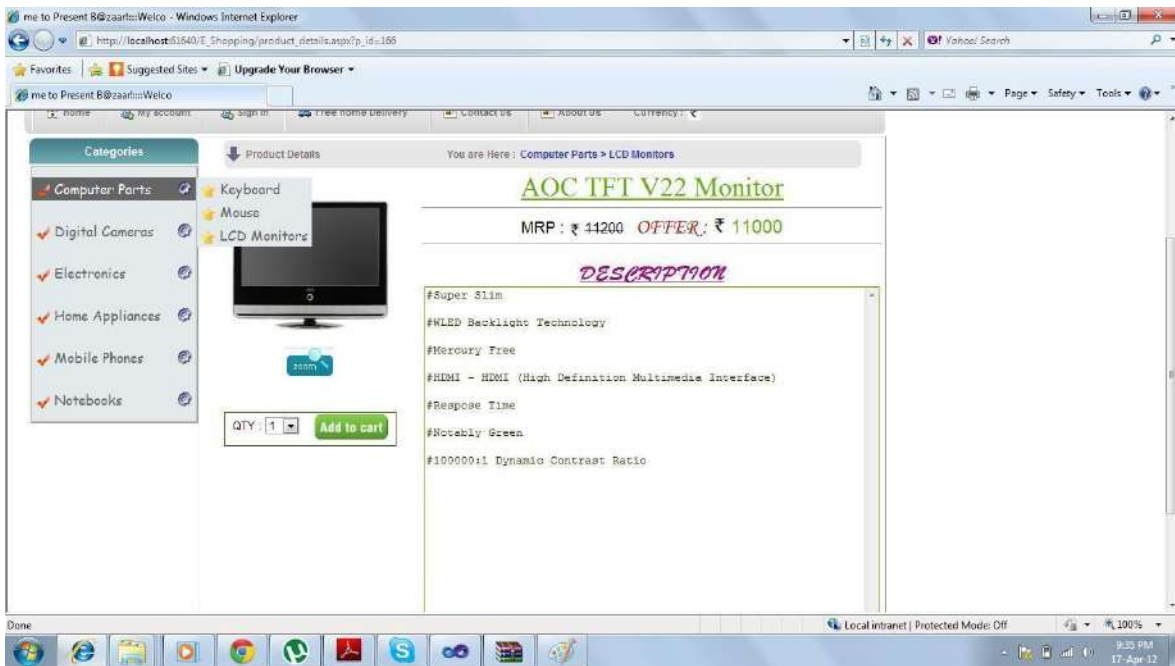
Functionality (Screen shots)

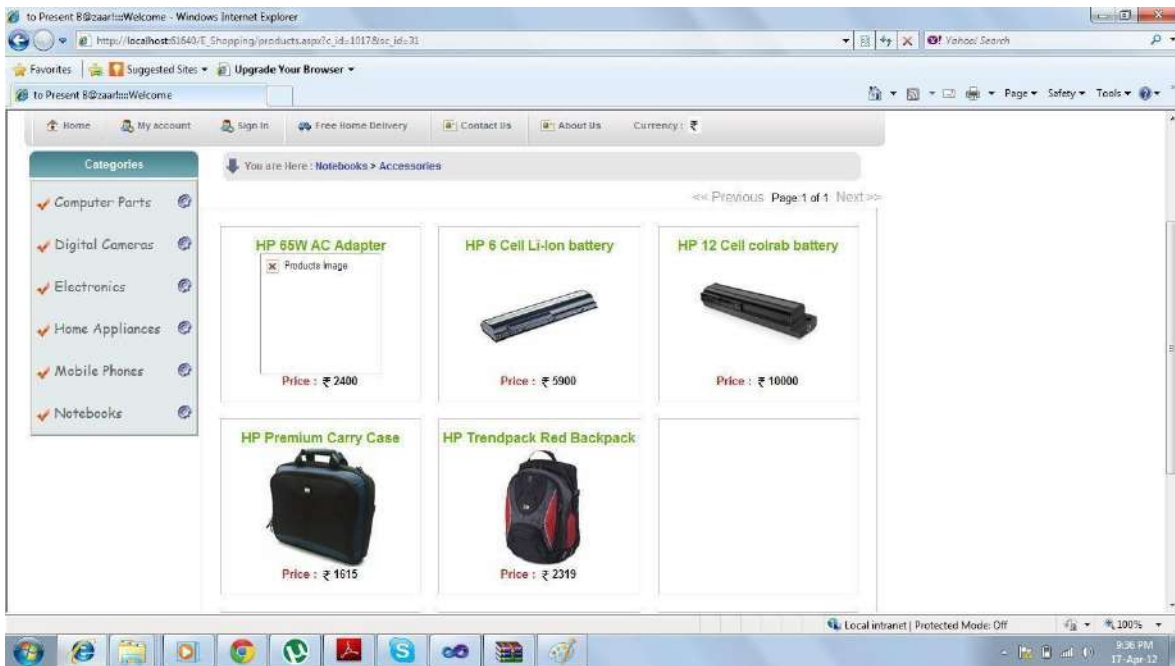












to Present 8@zaarfa>Welcome - Windows Internet Explorer

http://localhost:51640/E_Shopping/ship_info.aspx

to Present 8@zaarfa>Welcome

Note : Shipping Information is Mandatory

Shipping Information

First Name: Last Name:

City:

Address:

State:

Pin Code:

Mobile No: +91 -

Enter Security Question:

Select a Security Question (in case you forget your password):

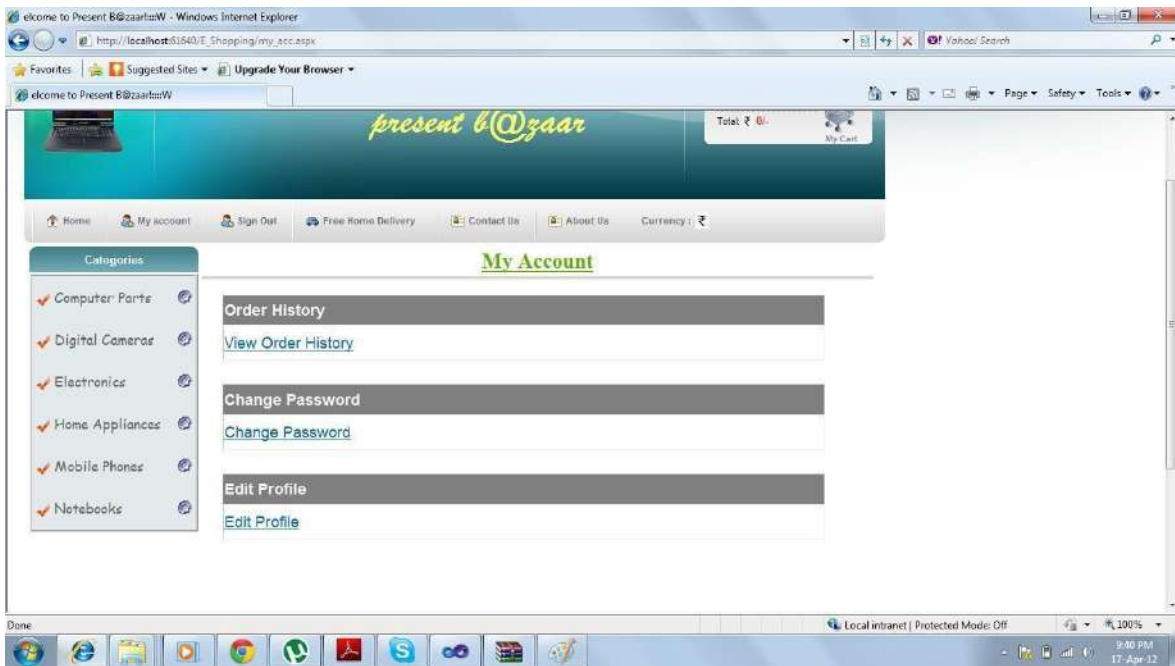
Enter Security Answer:

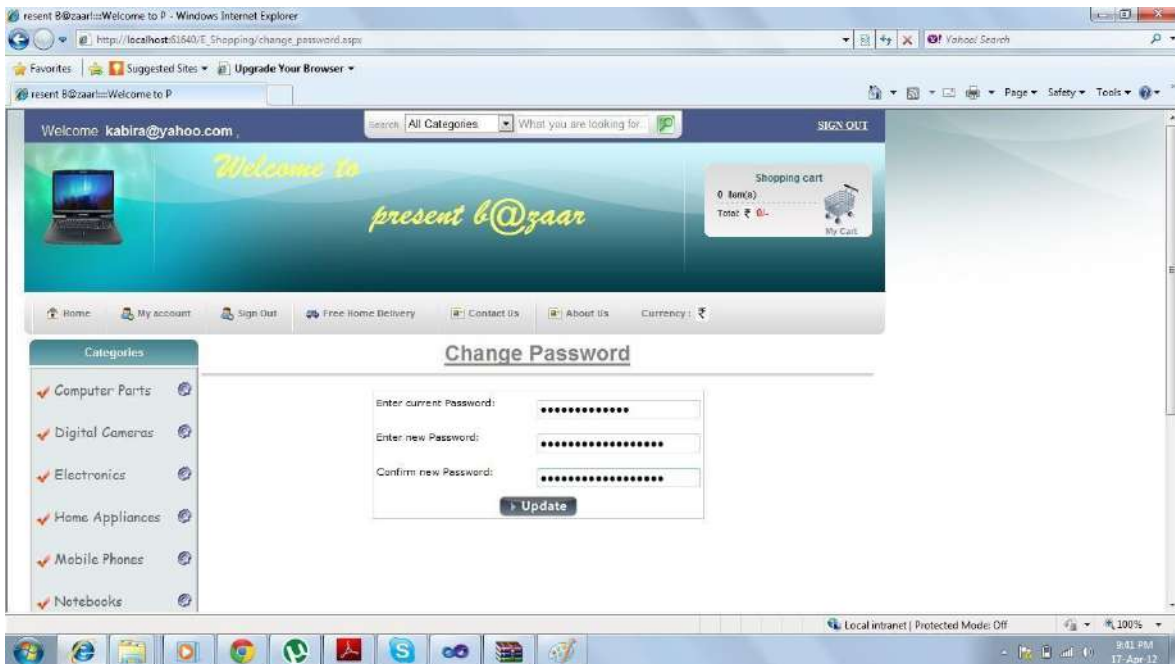
Optional Information

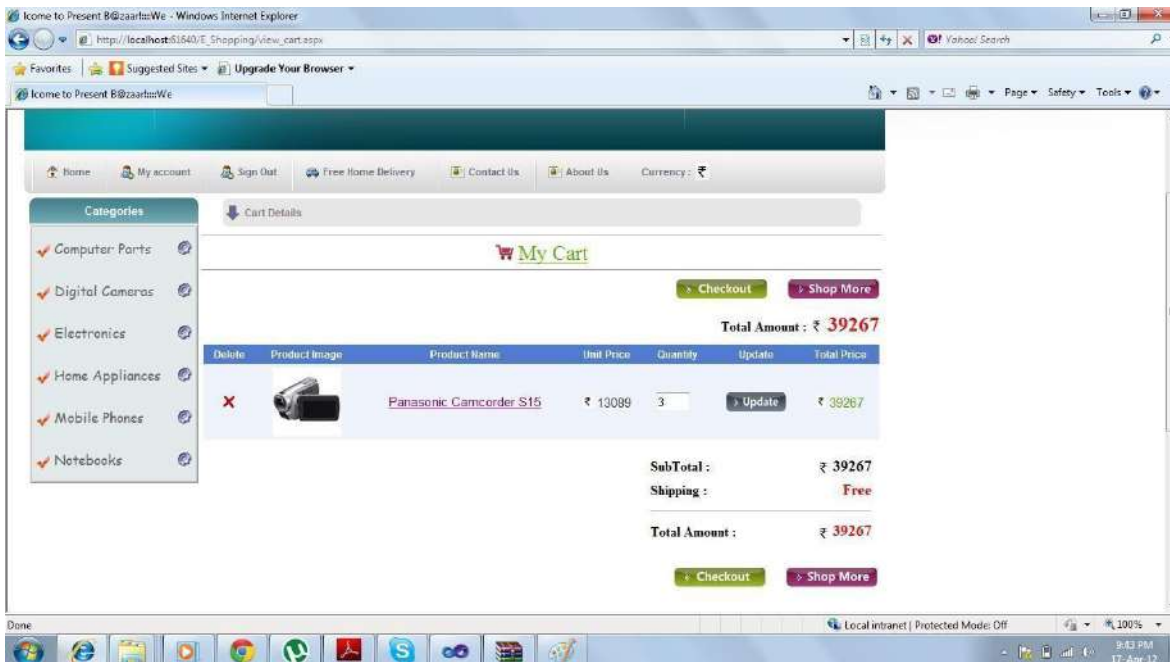
Date of Birth (MM/DD/YYYY):

Sex: ☐ Male ☐ Female

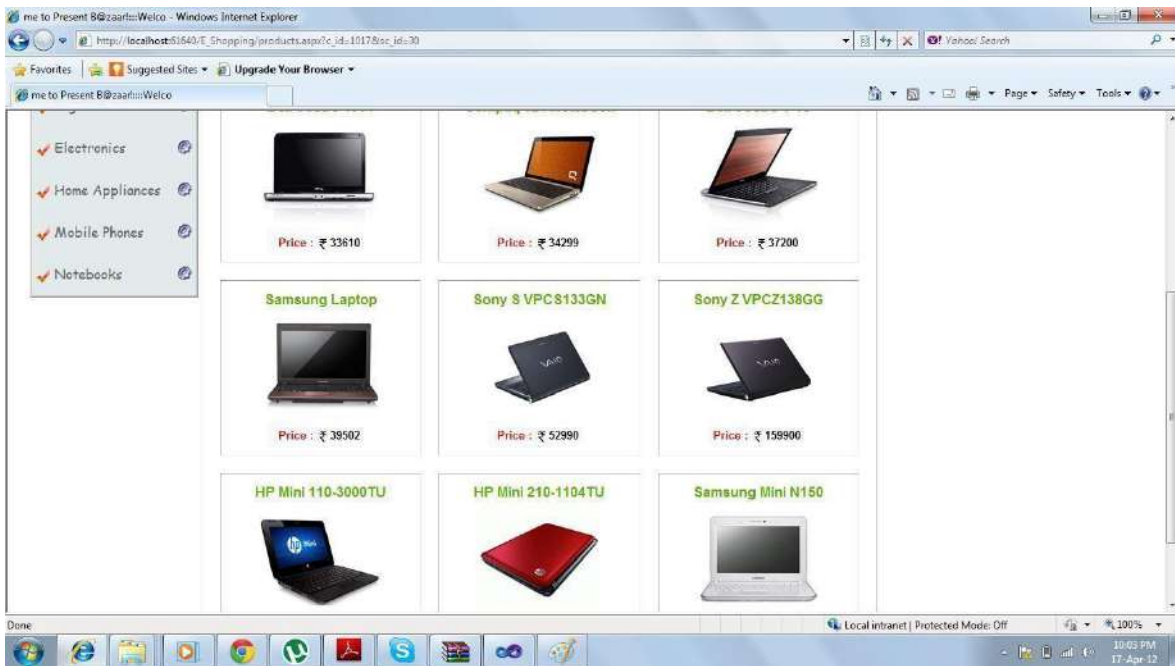
Marital Status: ☐ Single ☐ Married



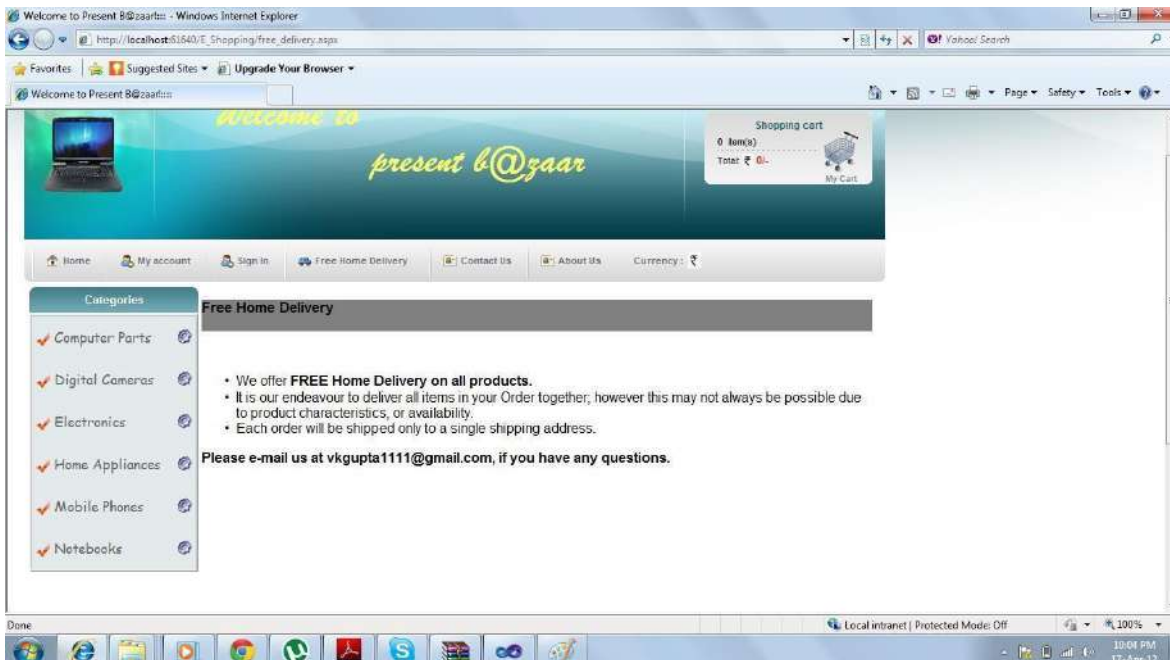


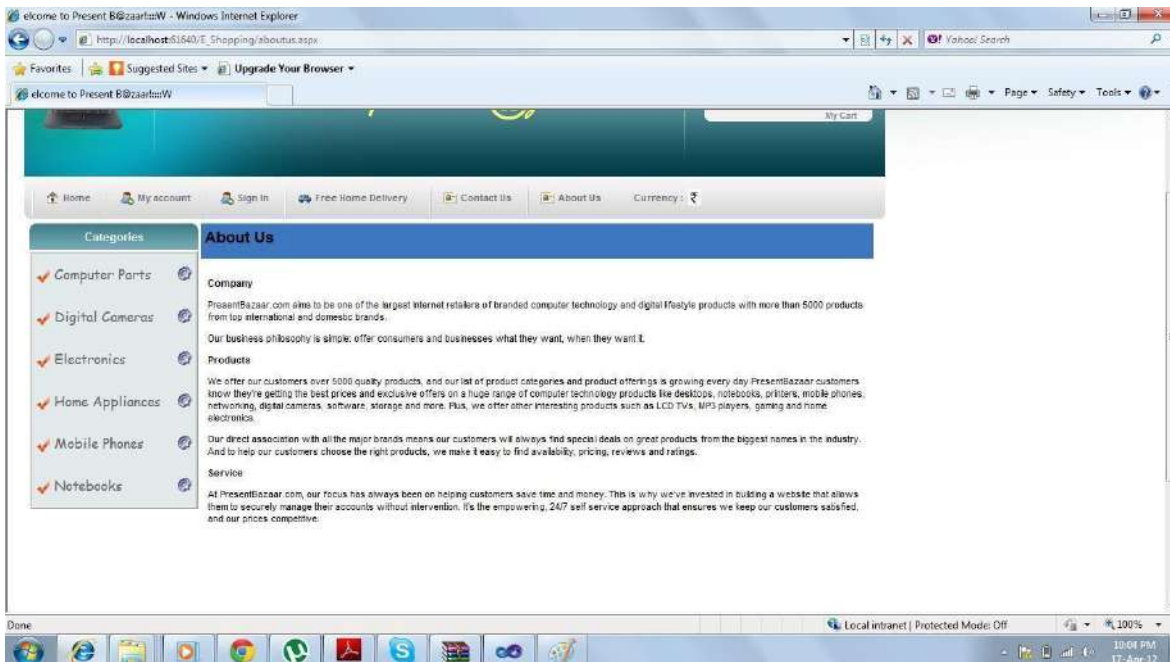


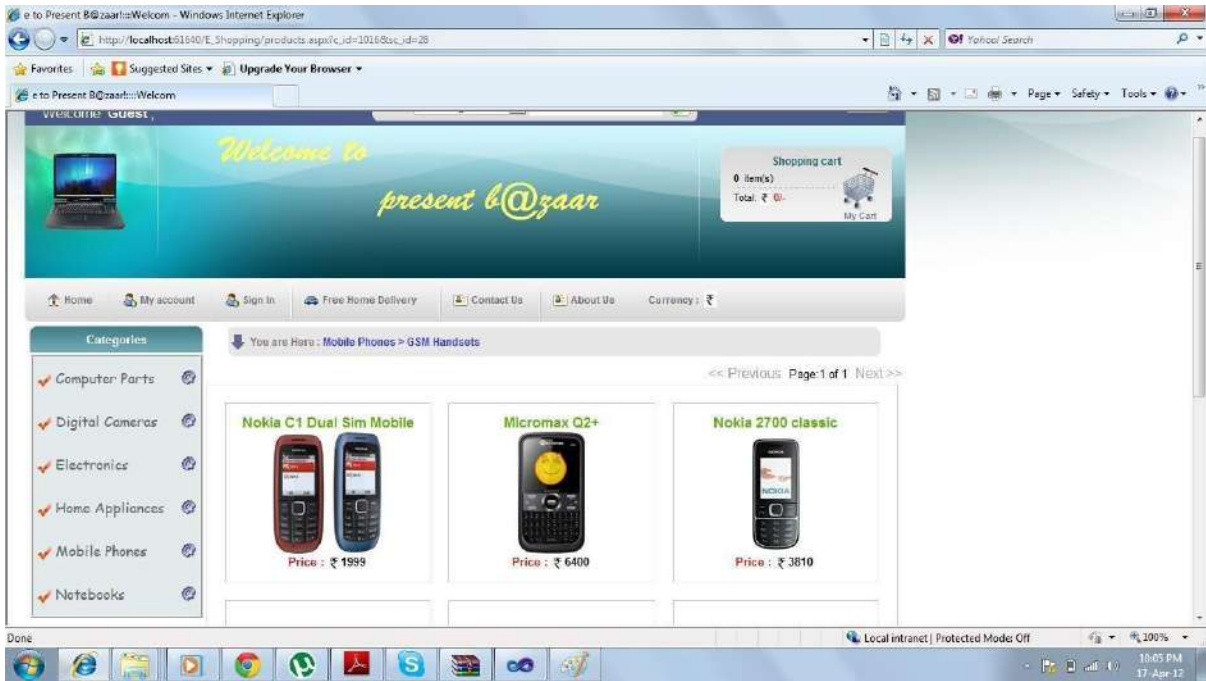












CHAPTER 7

TESTING

Software Testing is an empirical investigation conducted to provide stakeholders with information about the quality of the product or service under test, with respect to the context in which it is intended to operate. Software Testing also provides an objective, independent view of the software to allow the business to appreciate and understand the risks at implementation of the software. Test techniques include, but are not limited to, the process of executing a program or application with the intent of finding software bugs. It can also be stated as the process of validating and verifying that a software program/application/product meets the business and technical requirements that guided its design and development, so that it works as expected and can be implemented with the same characteristics.

Software Testing, depending on the testing method employed, can be implemented at any time in the development process, however the most test effort is employed after the requirements have been defined and coding process has been completed.

6.1 Unit Testing:

The primary goal of unit testing is to take the smallest piece of testable software in the application, isolate it from the remainder of the code, and determine whether it behaves exactly as you expect. Each unit is tested separately before integrating them into modules to test the interfaces between modules. Unit testing has proven its value in that a large percentage of defects are identified during its use.

Unit testing is a software verification and validation method where the programmer gains confidence that individual units of source code are fit for use. A unit is the smallest testable part of an application. In procedural programming a unit may be an individual program, function, procedure, etc., while in object-oriented programming, the smallest unit is a class, which may belong to a base/super class, abstract class or derived/child class.

Ideally, each test case is independent from the others: substitutes like method stubs, mock objects, fakes and test harnesses can be used to assist testing a module in isolation. Unit tests are typically written and run by software developers to ensure that code meets its design and behaves as intended. Its implementation can vary from being very manual (pencil and paper) to being formalized as part of build automation.

6.2 Integration Testing

Integration testing, also known as integration and testing (I&T), is a software development process which program units are combined and tested as groups in multiple ways. In this context, a unit is defined as the smallest testable part of an application. Integration testing can expose problems with the interfaces among program components before trouble occurs in real-world program execution. Integration testing is a component of Extreme Programming (XP), a

pragmatic method of software development that takes a meticulous approach to building a product by means of continual testing and revision.

There are two major ways of carrying out an integration test, called the bottom-up method and the top-down method. Bottom-up integration testing begins with unit testing, followed by tests of progressively higher-level combinations of units called modules or builds. In top-down integration testing, the highest-level modules are tested first and progressively lower-level modules are tested after that. In a comprehensive software development environment, bottom-up testing is usually done first, followed by top-down testing.

6.3 Validation testing

At the validation level, testing focuses on user visible actions and user recognizable output from the system. Validation testing is said to be successful when software functions in a manner that can be reasonably expected by the customer. Three types of validation testing

1. **Alpha testing** is simulated or actual operational testing by potential users/customers or an independent test team at the developers' site. Alpha testing is often employed for off-the-shelf software as a form of internal acceptance testing, before the software goes to beta testing.
2. **Beta testing** comes after alpha testing. Versions of the software, known as beta version, are released to a limited audience outside of the programming team. The software is released to groups of people so that further testing can ensure the product has few faults or bugs. Sometimes, beta versions are made available to the open public to increase the feedback field to a maximal number of future users
3. **Gray box testing** Gray box testing is the combination of black box and white box testing. Intention of this testing is to find out defects related to bad design or bad implementation of the system. It is used for web application

CHAPTER 8

Results & Challenges

The application can be used for any Ecommerce application. It is easy to use, since it uses the GUI provided in the user dialog. User friendly screens are provided. The application is easy to use and interactive making online shopping a recreational activity for users. It has been thoroughly tested and implemented.

7.1 Challenges

- Compatibility with browsers like Mozilla Firefox, Internet explorer etc
- Using a layered approach in developing the application which would make the application maintainable.
- Learning new technologies like using JavaScript for drag and drop behavior and Ajax toolkit controls with little guidance.

The overall idea of doing this project is to get a real time experience. Learn new technologies.

CHAPTER 9

Conclusions, Limitations & Future Scope

Conclusion

The 'Online Shopping' is designed to provide a web based application that would make searching, viewing and selection of a product easier. The search engine provides an easy and convenient way to search for products where a user can Search for a product interactively and the search engine would refine the products available based on the user's input. The user can then view the complete specification of each product. They can also view the product reviews and also write their own reviews. Use of Ajax components would make the application interactive and prevents annoying post backs. Its drag and drop feature would make it easy to use.

Limitations

This application does not have a built in check out process. An external checkout package has to be integrated in to this application. Also users cannot save the shopping carts so that they can access later i.e. they cannot create wish lists which they can access later. This application does not have features by which user can set price ranges for products and receive alerts once the price reaches the particular range.

Scope for Future Work

The following things can be done in future.

- The current system can be extended to allow the users to create accounts and save products in to wish list.
- The users could subscribe for price alerts which would enable them to receive messages when price for products fall below a particular level.
- The current system is confined only to the shopping cart process. It can be extended to have an easy to use check out process.
- Users can have multiple shipping and billing information saved. During checkout they can use the drag and drop feature to select shipping and billing information.

REFERNECES

- [1] [Kim J., Park J.,\(2005\), " A Consumer Shopping Channel Extension Model: AttitudeShift Toward the Online Store", Journal of Fashion Marketing and Management:, Vol.9, No.1, 106-121. \(Pubitemid 40777788\)](#)
- [2] [Case, Anne-Sophie \(2002\), "Perceived Risk and Risk-Reduction Strategies in Internet Shopping," International Review of Retail, Distribution and Consumer Research. 12 \(4\), 375-394.](#)
- [3] [Shih H. P., \(2004\), "An Empirical study on Predicting User Acceptance of e-shopping on the Web", Information & Management, Vol.41, 351-368.](#)
- [4] [Eastlick, M. A. and Feinberg, R. A., "Shopping motives for mail catalog shopping,"Journal of Business Research. 45\(3\), 1999.](#)
- [5] [Forouhandeh, B., Nejatian, H. and Ramanathan, K., "The online shopping adoption:barriers and advantages," In Proceeding of the 2nd International Conference on Business and Economic Research, 2149-2171, 2011.](#)
- [6] [Michal, P., 'On-line Shopping on B2C Markets in the Czech Republic," Journal ofCompetitiveness, 4\(4\), 2012](#)
- [7] [Lee, N. and Zhang, P., "Consumer Online Shopping Attitudes and Behavior: An Assessment of Research," In Eighth Americas Conference on Information System, 508-517, 2002.](#)
- [8] [PEW Internet and American Life Project, "Online Shopping Internet users like theconvenience but worry about the security of their financial information," 2008, Retrieved on April 23, 2013 from http://www.pewinternet.org/~media/Files/Reports/2008/PIP_Onl ine% 20Shoppin.pdf.pdf.](#)
- [9] [Suresh, A., M. and Shashikala, R., "Identifying Factors of Consumer Perceived Risktowards Online Shopping in India," In 3rd International Conference on Information and FinancialEngineering, 12, 2011,](#)
- [10] [Singh, M., "E-services and their role in B2C e-commerce," ManagingService Quality. 12\(6\), 2002](#)
- [11] [Master Card Worldwide Insights, "Online shopping in Asia/Pacific-Patterns,trends and future growth," 2008, Retrieved on April 29, 2013 from http://www.mastercard.com/us/company/en/insights/pdfs/2008/AsiaPacificOnlineSh op.pdf.](#)

- [12] Mcfee Report Shop Online with Confidence,” 2009, Retrieved on April 20,2013 from http://promos.mcafee.com/en-US/PDF/shoponlinewithconfidence_us.pdf.
- [13] 39. McKnight, D. H., Choudhury, V. and Kacmar, C., “Developing and validating trust measures for e-commerce: an integrative typology,” Information Systems Research, 13(3), 2002.
- [14] Gurleen, K., “Consumer’s Perception towards Online Shopping- The case of Punjab,” International Journal of Management & Information Technology, 1 (1), 2012.
- [15] Forrester Research Report, “Trends in India’s Ecommerce Market,” 2012, Retrieved on April 18, 2013 from <http://www.assoam.org/arb/general/ForresterTrendsInIndiasCommerce.pdf>.