Project Report

ON

"BiharKart! - A Ecommerce Website"

Submitted to

Department of CSE

Sitamarhi Institute of Technology

Aryabhatta Knowledge University, Patna

Submitted by

Prashant Kumar
16105127027

Shahbaz Alam
16105127017

Pankaj Kumar
16105127015

Rajesh Ram
16105127016

Under the Guidance of

Mr. Neeraj Kumar

Assistant Professor
Department of CSE
SIT, Sitamarhi

Declaration

We hereby declare that, the project being submitted by us entitled "***BIHARKART! - A ECOMMERCE WEBSITE***" for the partial fulfilment for the award of the "***BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE & ENGINEERING***" is an authenticated record of work carried by us in the Fourth year of B.Tech submitted to *Department of CSE, Sitamarhi Institute of Technology, Sitamarhi*, which is affiliated to AICTE under the guidance of Prof. Neeraj Kumar.

Prashant Kumar
Shahbaz Alam
Pankaj Kumar
Rajesh Ram

Certificate

*This is to certify that the project work entitled "**BIHARKART! - A ECOMMERCE WEBSITE**" which is being submitted by **Prashant Kumar, Shahbaz Alam, Pankaj Kumar** and **Rajesh Ram** in partial fulfilment of the requirements for the 8th Semester Sessional Examination of Bachelor of Technology in **Computer Science & Engineering** during the academic year 2016-20. They have worked under the guidance of **Prof. Neeraj Kumar** (Asst. Prof. Department of CSE). This work is submitted to the department as a part of evaluation of 8th Semester Project.*

.....
Prof. Neeraj Kumar
Project Guide

.....
Prof. Sadique Nayeem
H.O.D.
Department of CSE

.....
External Examiner

PREFACE

Programming languages, paradigms and practices don't stand still very long. It often seems that the methods and techniques we applied yesterday are out of date today of course this rapid rate of change is also one of the things that keep programming existing.

There is always something new on the horizon. One characteristic that is constant in software industry today is the "change". Change is one of the most critical aspects of s/w development and management. New tools and new approaches are announced almost every day. The impact of these developments is often very extensive. Most important among them is maintaining ability, reusability, portability, security, and integrity and user friendliness. To build today's complex s/w we need to wound construction techniques and program structures that are easy to comprehend, implement and modify in wide variety of situations.

ACKNOWLEDGEMENT

First of all, We would like to thank the almighty God for listening our prayer and giving us strength to complete the project work.

We would like to express a deep sense of gratitude and thanks profusely to Mr. Neeraj Kumar, Assistant Professor, Department of CSE, Sitamarhi Institute of Technology, Sitamarhi (Bihar), who guided us throughout the project. Without his willing disposition, spirit of accommodation, frankness, timely clarification and above of all faith in us, this project could not have been completed in due time.

We would also like to thank whole of the faculty of the college for their cooperation and important support.

We must also express our deep regards and thanks to our parents for supporting and boosting our morale.

We finally pray that Almighty fulfils the aspirations of all of the who have been a part of this journey and those who will be a part of future journeys.

CONTENTS

TABLE OF DIAGRAMS

TABLE OF IMAGES

INTRODUCTION & OBJECTIVE

An E-Commerce portal which will allow formal and informal merchants in developing countries to advertise and sell their goods on the internet. This would permit rural communities to make their wares available to the rest of the world via the World Wide Web.

The objective of this project is to create an e-commerce web portal with a content management system which would allow product information to be updated securely using a mobile device. The web portal will have an online interface in the form of an e-commerce website that will allow users to buy goods from the merchants.

This project will be divided into following separate components:

- The Content Management System (CMS)
- The E-Commerce Website/Portal
- The product, merchant and customer database
- Reporting of the sales, orders, shipment, etc.
- The online transaction security system
- The data security system

Content Management System (CMS):

The CMS will be responsible for managing the product, merchant and customer database. The CMS will also handle any changes that must be made to the database as a result of transaction on the e-commerce website. This information would be processed and the database would be updated accordingly using SQL queries.

The E-Commerce Website/Portal:

The E-Commerce website/portal will provide merchants with a medium through which they will be able to sell their merchandise. It will provide online shoppers with an interface through which they will be able to purchase merchandise from formal and informal merchants. This e-commerce website component will be written in python, which is very popular and versatile e-commerce programming language. The website will provide shoppers with information about the various products that are for sale. The information will include prices, product descriptions, stock availability as well as photographs of the products.

Product, Merchant and Customer Database:

The product, merchant and customer database will store all information about the products that will be sold on the e-commerce portal (prices, product descriptions, photos of products). It will also store merchant information (names, banking details, contact details) as well as customer details (credit card information, shipping address). This database will use the MySQL architecture and will be manipulated using SQL queries via the content management system.

The e-commerce portal will have the following key features:

- An online shop that will allow online shoppers to buy wares from formal and informal merchants.
- A search engine on the website to allow customers to find specific types of merchandise.
- A secure online transaction system that will allow shoppers to purchase goods safely using their credit cards.
- A database of merchandise with photos, product descriptions and stock information. This database will also contain all relevant merchant and customer information.
- A data security system that will ensure that all data that is transmitted between the various system

Technologies used to develop this Web Application

1. Django (Web framework):-

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It's free and open source.

1.1 History of Django

- **2003** – Started by Adrian Holovaty and Simon Willison as an internal project at the Lawrence Journal-World newspaper.
- **2005** – Released July 2005 and named it Django, after the jazz guitarist Django Reinhardt.
- **2005** - Mature enough to handle several high-traffic sites.
- **2008** - In June 2008, it was announced that a newly formed Django Software Foundation (DSF) would maintain Django in the future.
- **Current** – Django is now an open source project with contributors across the world.

1.2 Django – Design Philosophies

Django comes with the following design philosophies –

- **Loosely Coupled** – Django aims to make each element of its stack independent of the others.
- **Less Coding** – Less code so in turn a quick development.
- **Don't Repeat Yourself (DRY)** – Everything should be developed only in exactly one place instead of repeating it again and again.
- **Fast Development** – Django's philosophy is to do all it can to facilitate hyper-fast development.
- **Clean Design** – Django strictly maintains a clean design throughout its own code and makes it easy to follow best web-development practices.

1.3 Advantages of Django

Here are few advantages of using Django which can be listed out here −

- **Object-Relational Mapping (ORM) Support** − Django provides a bridge between the data model and the database engine, and supports a large set of database systems including MySQL, Oracle, Postgres, etc. Django also supports NoSQL database through Django-nonrel fork. For now, the only NoSQL databases supported are MongoDB and google app engine.
- **Multilingual Support** − Django supports multilingual websites through its built-in internationalization system. So you can develop your website, which would support multiple languages.
- **Framework Support** − Django has built-in support for Ajax, RSS, Caching and various other frameworks.
- **Administration GUI** − Django provides a nice ready-to-use user interface for administrative activities.
- **Development Environment** − Django comes with a lightweight web server to facilitate end-to-end application development and testing.

2. Bootstrap (front-end framework):-

Bootstrap is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. It contains CSS- and (optionally) JavaScript-based design templates for typography, forms, buttons, navigation, and other interface components.

Bootstrap is the sixth-most-starred project on GitHub, with more than 135,000 stars, behind freeCodeCamp (almost 307,000 stars) and marginally behind Vue.js framework.[2] According to Alexa Rank, Bootstrap is in the top-2000 in the USA while vuejs.org is in the top-7000 in the USA.[3]

2.1 History

Bootstrap, originally named Twitter Blueprint, was developed by Mark Otto and Jacob Thornton at Twitter as a framework to encourage consistency across internal tools. Before Bootstrap, various libraries were used for interface development, which led to inconsistencies and a high maintenance burden. According to Twitter developer Mark Otto:

A super small group of developers and I got together to design and build a new internal tool and saw an opportunity to do something more. Through that process, we saw ourselves build something much more substantial than another internal tool. Months later, we ended up with an early version of Bootstrap as a way to document and share common design patterns and assets within the company.

After a few months of development by a small group, many developers at Twitter began to contribute to the project as a part of Hack Week, a hackathon-style week for the Twitter development team. It was renamed from Twitter Blueprint to Bootstrap, and released as an open source project on August 19, 2011. It has continued to be maintained by Mark Otto, Jacob Thornton, and a small group of core developers, as well as a large community of contributors.

2.2 Features

Bootstrap is a web framework that focuses on simplifying the development of informative web pages (as opposed to web apps). The primary purpose of adding it to a web project is to apply Bootstrap's choices of color, size, font and layout to that project. As such, the primary factor is whether the developers in charge find those choices to their liking. Once added to a project, Bootstrap provides basic style definitions for all HTML elements. The result is a uniform appearance for prose, tables and form elements across web browsers. In addition, developers can

take advantage of CSS classes defined in Bootstrap to further customize the appearance of their contents. For example, Bootstrap has provisioned for light- and dark-colored tables, page headings, more prominent pull quotes, and text with a highlight.

Bootstrap also comes with several JavaScript components in the form of jQuery plugins. They provide additional user interface elements such as dialog boxes, tooltips, and carousels. Each Bootstrap component consists of an HTML structure, CSS declarations, and in some cases accompanying JavaScript code. They also extend the functionality of some existing interface elements, including for example an auto-complete function for input fields.

The most prominent components of Bootstrap are its layout components, as they affect an entire web page. The basic layout component is called "Container", as every other element in the page is placed in it. Developers can choose between a fixed-width container and a fluid-width container. While the latter always fills the width of the web page, the former uses one of the four predefined fixed widths, depending on the size of the screen showing the page:

- Smaller than 576 pixels
- 576–768 pixels
- 768–992 pixels
- 992–1200 pixels
- Larger than 1200 pixels

Once a container is in place, other Bootstrap layout components implement a CSS Flexbox layout through defining rows and columns.

A precompiled version of Bootstrap is available in the form of one CSS file and three JavaScript files that can be readily added to any project. The raw form of Bootstrap, however, enables developers to implement further customization and size optimizations. This raw form is modular, meaning that the developer can remove unneeded components, apply a theme and modify the uncompiled Sass files.

3. Python:-

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

Python was conceived in the late 1980s as a successor to the ABC language. Python 2.0, released in 2000, introduced features like list comprehensions and a garbage collection system capable of collecting reference cycles. Python 3.0, released in 2008, was a major revision of the language that is not completely backward-compatible, and much Python 2 code does not run unmodified on Python 3.

The biggest strength of the Python is large library which can be used for the following:-

- Machine Learning
- GUI Applications (like Kivy, Tkinter, PyQt etc.)
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)
- Image processing (like OpenCV, Pillow)
- Web scraping (like Scrapy, BeautifulSoup, Selenium)
- Test frameworks
- Multimedia
- Scientific computing
- Text processing and many more..

3.1 Features of Python

1) ***Easy to Learn and Use***

   Python is easy to learn and use. It is developer-friendly and high level programming language.

2) ***Expressive Language***

   Python language is more expressive means that it is more understandable and readable.

3) ***Interpreted Language***

   Python is an interpreted language i.e. interpreter executes the code line by line at a time. This makes debugging easy and thus suitable for beginners.

4) ***Cross-platform Language***

   Python can run equally on different platforms such as Windows, Linux, Unix and Macintosh etc. So, we can say that Python is a portable language.

5) ***Free and Open Source***

   Python language is freely available at offical web address.The source-code is also available. Therefore it is open source.

6) ***Object-Oriented Language***

   Python supports object oriented language and concepts of classes and objects come into existence.

7) ***Extensible***

   It implies that other languages such as C/C++ can be used to compile the code and thus it can be used further in our python code.

8) ***Large Standard Library***

   Python has a large and broad library and prvides rich set of module and functions for rapid application development.

9) ***GUI Programming Support***

   Graphical user interfaces can be developed using Python.

10) ***Integrated***

    It can be easily integrated with languages like C, C++, JAVA etc.

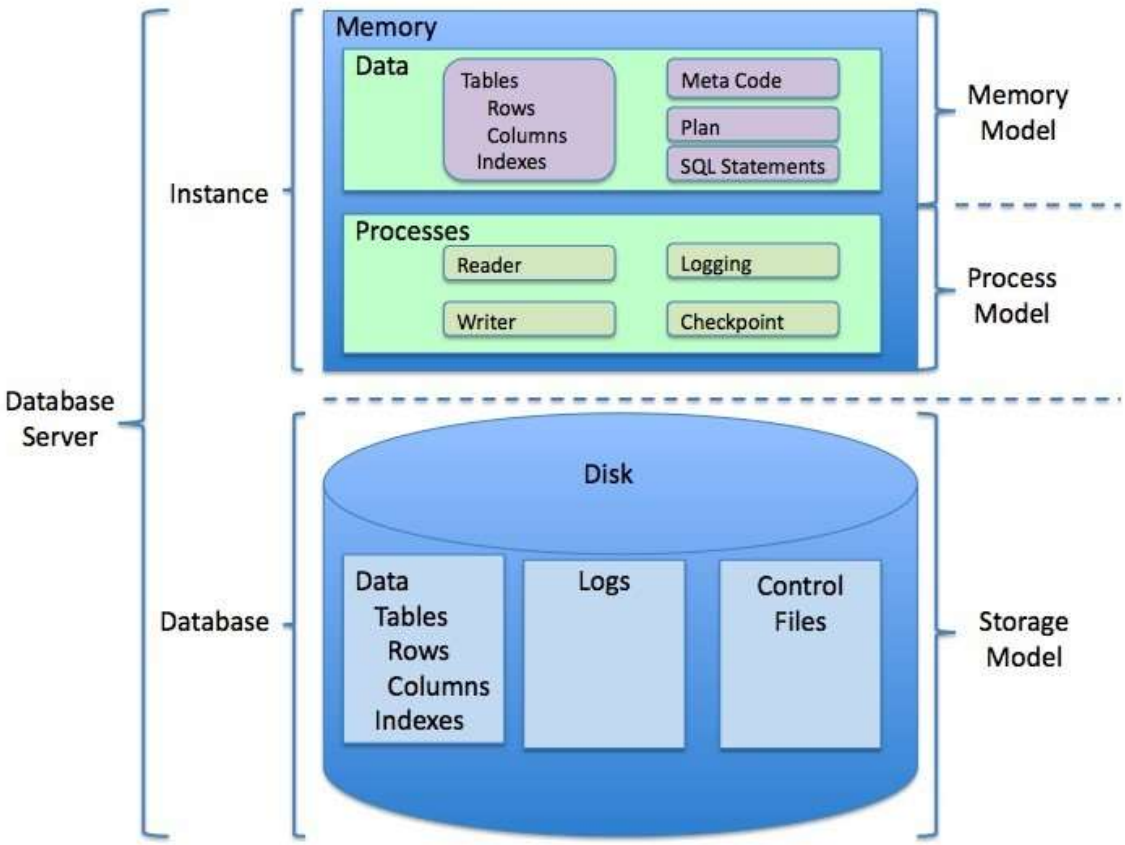
4. SQLite:-

SQLite is a relational database management system (RDBMS) contained in a C library. In contrast to many other database management systems, SQLite is not a client–server database engine. Rather, it is embedded into the end program.

Brief Introduction about RDBSM:-

A ***relational database management system (RDBMS)*** is a database management system (DBMS) that is based on the relational model as invented by E. F. Codd, of IBM's San Jose Research Laboratory. Many popular databases currently in use are based on the relational database model.

RDBMSs have become a predominant choice for the storage of information in new databases used for financial records, manufacturing and logistical information, personnel data, and much more since the 1980s. Relational databases have often replaced legacy hierarchical databases and network databases because they are easier to understand and use. However, relational databases have been challenged by object databases, which were introduced in an attempt to address the object-relational impedance mismatch in relational database, and XML databases.



Diag. 1: RDBMS Architecture Diagram

SQLite is *ACID-compliant* and implements most of the SQL standard, generally following PostgreSQL syntax. However, SQLite uses a dynamically and weakly typed SQL syntax that does not guarantee the domain integrity. This means that one can, for example, insert a string into a column defined as an integer. SQLite will attempt to convert data between formats where appropriate, the string "123" into an integer in this case, but does not guarantee such conversions, and will store the data as-is if such a conversion is not possible.

SQLite is a popular choice as embedded database software for local/client storage in application software such as web browsers. It is arguably the most widely deployed database engine, as it is used today by several widespread browsers, operating systems, and embedded systems (such as mobile phones), among others. SQLite has bindings to many programming languages.

4.1 Design

Unlike client–server database management systems, the SQLite engine has no standalone processes with which the application program communicates. Instead, the SQLite library is linked in and thus becomes an integral part of the application program. Linking may be static or dynamic. The application program uses SQLite's functionality through simple function calls, which reduce latency in database access: function calls within a single process are more efficient than inter-process communication.

SQLite stores the entire database (definitions, tables, indices, and the data itself) as a single cross-platform file on a host machine. It implements this simple design by locking the entire database file during writing. SQLite read operations can be multitasked, though writes can only be performed sequentially.

Due to the server-less design, SQLite applications require less configuration than client-server databases. SQLite is called zero-conf because it does not require service management (such as startup scripts) or access control based on GRANT and passwords. Access control is handled by means of file system permissions given to the database file itself. Databases in client-server systems use file system permissions which give access to the database files only to the daemon process.

4.2 Features

SQLite implements most of the SQL-92 standard for SQL but it lacks some features. For example, it partially provides triggers, and it cannot write to views (however it provides INSTEAD OF triggers that provide this functionality). While it provides complex queries, it still has limited ALTER TABLE function, as it cannot modify or delete columns.

SQLite uses an unusual type system for a SQL-compatible DBMS; instead of assigning a type to a column as in most SQL database systems, types are assigned to individual values; in language terms it is dynamically typed. Moreover, it is weakly typed in some of the same ways that Perl is: one can insert a string into an integer column (although SQLite will try to convert the string to an integer first, if the column's preferred type is integer). This adds flexibility to columns, especially when bound to a dynamically typed scripting language. However, the technique is not portable to other SQL products. A common criticism is that SQLite's type system lacks the data integrity mechanism provided by statically typed columns in other products. The SQLite web site describes a "strict affinity" mode, but this feature has not yet been added. However, it can be implemented with constraints like CHECK(typeof(x)='integer').

Tables normally include a hidden rowid index column which gives faster access. If a database includes an Integer Primary Key column SQLite will typically optimize it by treating it as an alias for rowid, causing the contents to be stored as a strictly typed 64-bit signed integer and changing its behavior to be somewhat like an auto-incrementing column. Future versions of SQLite may include a command to introspect whether a column has behavior like that of rowid to differentiate these columns from weakly-typed, non-autoincrementing Integer Primary Keys.

SQLite with full Unicode function is optional.

Several computer processes or threads may access the same database concurrently. Several read accesses can be satisfied in parallel. A write access can only be satisfied if no other accesses are currently being serviced. Otherwise, the write access fails with an error code (or can automatically be retried until a configurable timeout expires). This concurrent access situation would change when dealing with temporary tables. This restriction is relaxed in version 3.7 when write-ahead logging (WAL) is turned on enabling concurrent reads and writes.

SQLite version 3.7.4 first saw the addition of the FTS4 (full text search) module, which features enhancements over the older FTS3 module. FTS4 allows users to perform full text searches on documents similar to how search engines search webpages. Version 3.8.2 added support for creating tables without row-id, which may provide space and performance improvements. Common table expressions support was added to SQLite in version 3.8.3.

In 2015, with the json1 extension and new subtype interfaces, SQLite version 3.9 introduced JSON content managing.

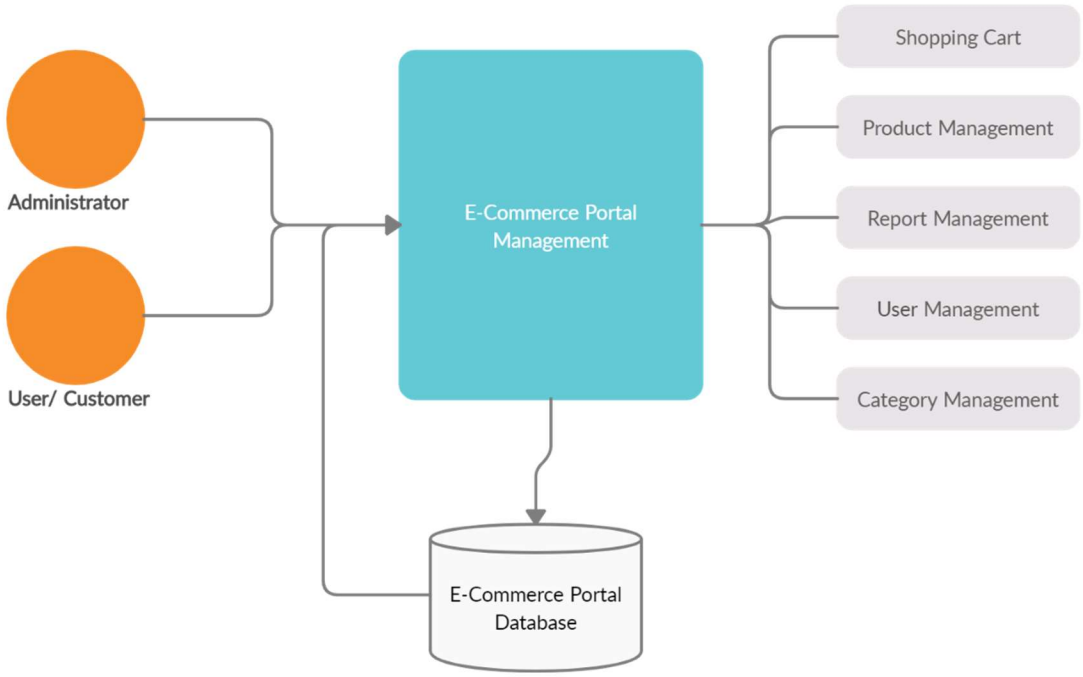
DATA FLOW DIAGRAM

Data flow diagram is a graphic tool. It is used to describe and analyze the movement of data through a system manual or computerize. They focus on the data flowing into the system, between processes in and out of the data stores.

E-Commerce website Data Flow Diagram is often used as a preliminary step to create an overview of the e-commerce shopping system without going into great detail, which can later be elaborated. It normally consists of overall application dataflow and processes of the e-commerce shopping system process. It contains all of the user flow and their entities such all the flow of Shopping, Shopping Cart, Order, Payment, Product, Delivery, Confirm Order. All of the below diagrams have been used for the visualization of data processing and structured design of the e-commerce shopping system process and working flow.

Level 0 DFD:-

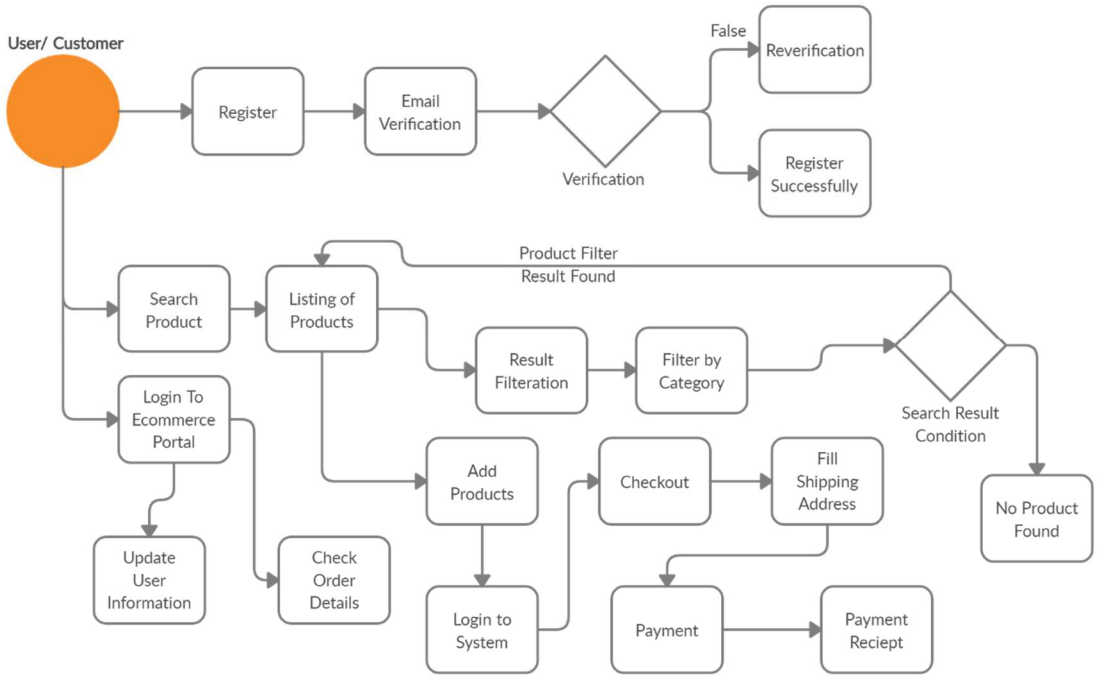
This is the Level 0 DFD of E-Commerce website, where we have elaborated the high-level process of e-commerce shopping system. It's a basic overview of the whole E-Commerce website or process being analyzed or modelled. It's designed to be an at-a-glance view of Product, Delivery and Confirm Order showing the system as a single high-level process, with its relationship to external entities of Shopping, Shopping Cart and Order. It should be easily understood by a wide audience, including Shopping, Order and Product.



Diag. 2: Level 0 DFD for E-Commerce Website

Level 1 DFD:-

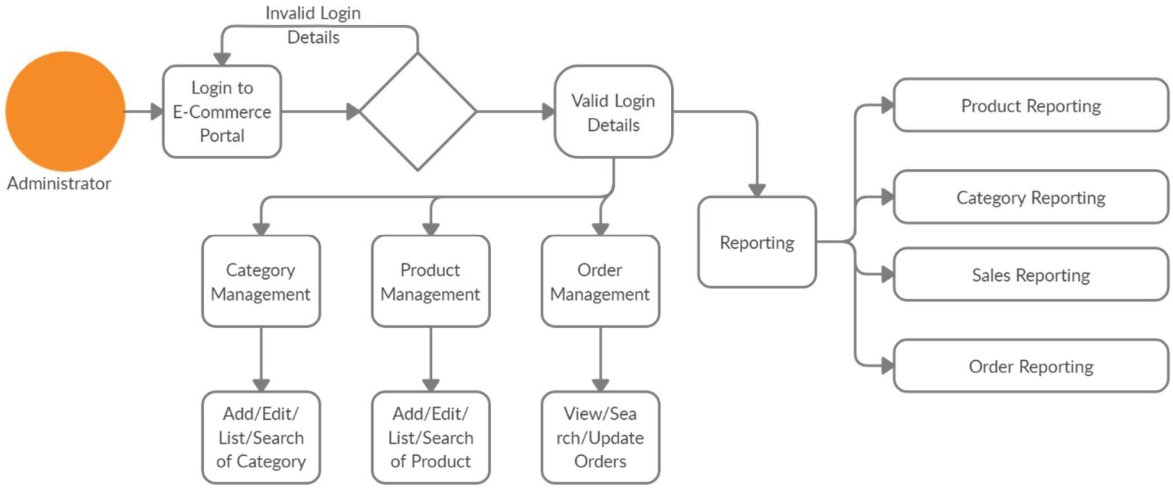
Level 1 DFD of E-Commerce website shows how the system is divided into sub-systems (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provide all of the functionality of the E-Commerce website as a whole. It also identifies internal data stores of Confirm Order, Delivery, Product, Payment, Order that must be present in order for the e-commerce shopping system to do its job, and shows the flow of data between the various parts of Shopping, Order, Delivery, Confirm Order, Product of the system.



Diag. 3: Level 1 DFD for E-Commerce Website

Level 2 DFD:-

Level 2 DFD then goes one step deeper into parts of Level 1 of Shopping System. It may require more functionalities of e-commerce shopping system to reach the necessary level of detail about the e-commerce website functioning. Level 1 DFD of E-Commerce website shows how the system is divided into sub-systems (processes). The 2nd Level DFD contains more details of Confirm Order, Delivery, Product, Payment, Order, Shopping Cart, Shopping.



Diag. 4: Level 2 DFD for E-Commerce Website

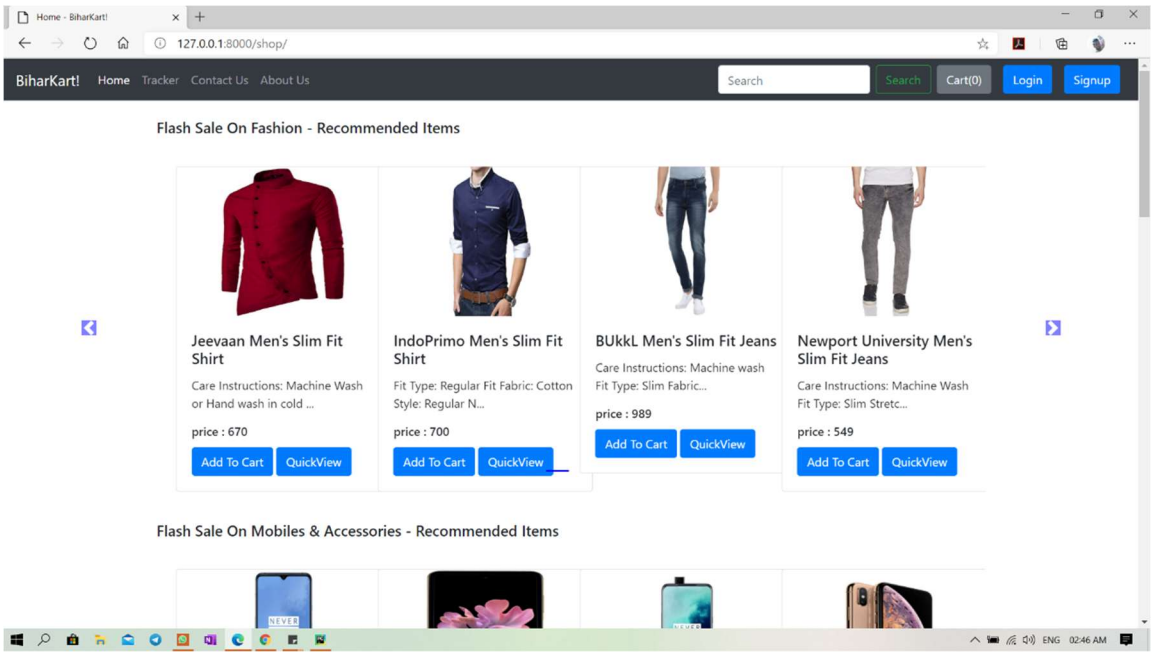
PROJECT LAYOUT

Modular Description:-

- Home Page
- Sign Up Page
- Login Page
- Search Page
- Product View Page
- Add to cart & Cart view
- CheckOut Page
- Order Tracker Page
- Contact Us Page
- About Us Page

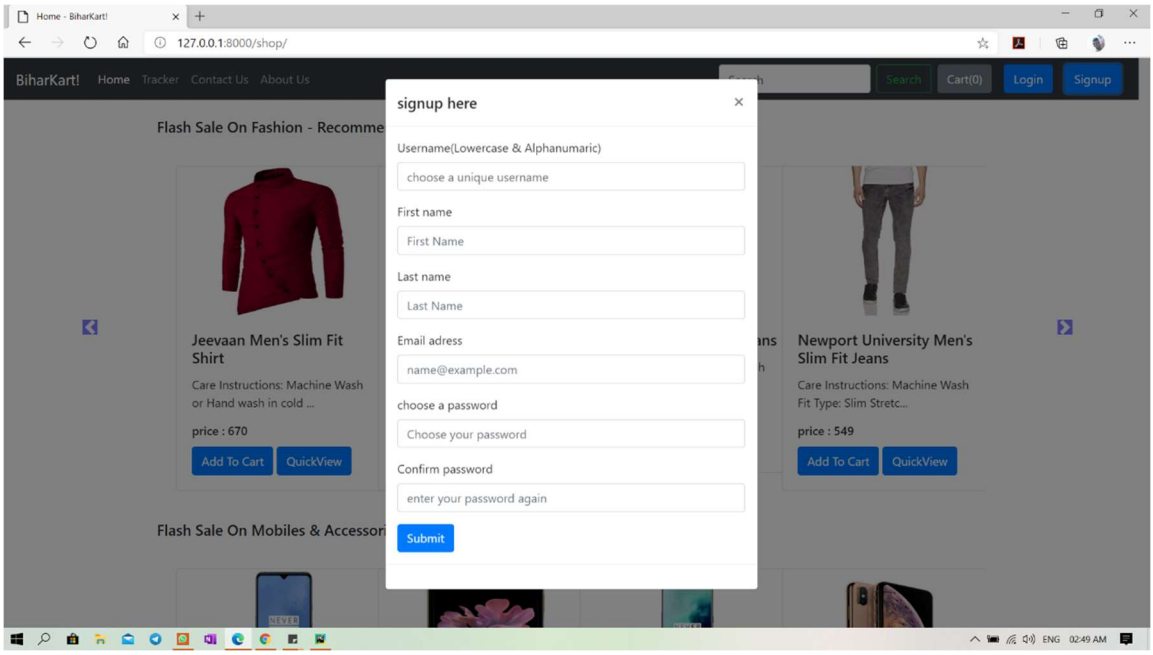
Screenshots:-

Home Page



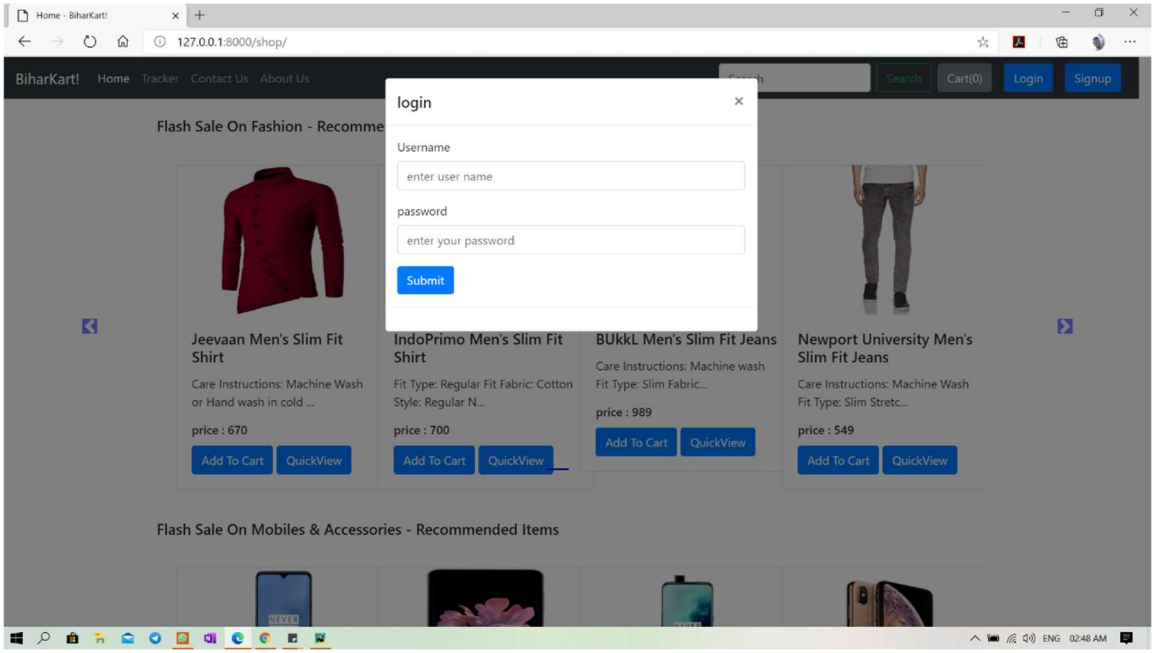
Img. 1: Home Page

Sign Up Page



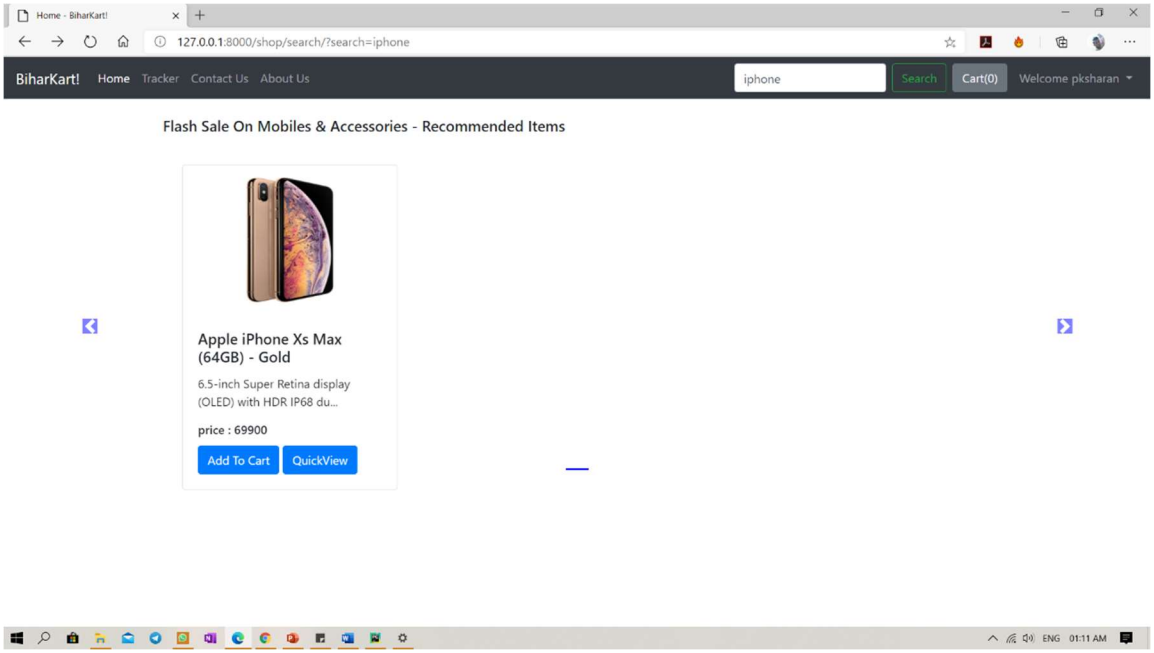
Img. 2: Sign Up Page

Login Page



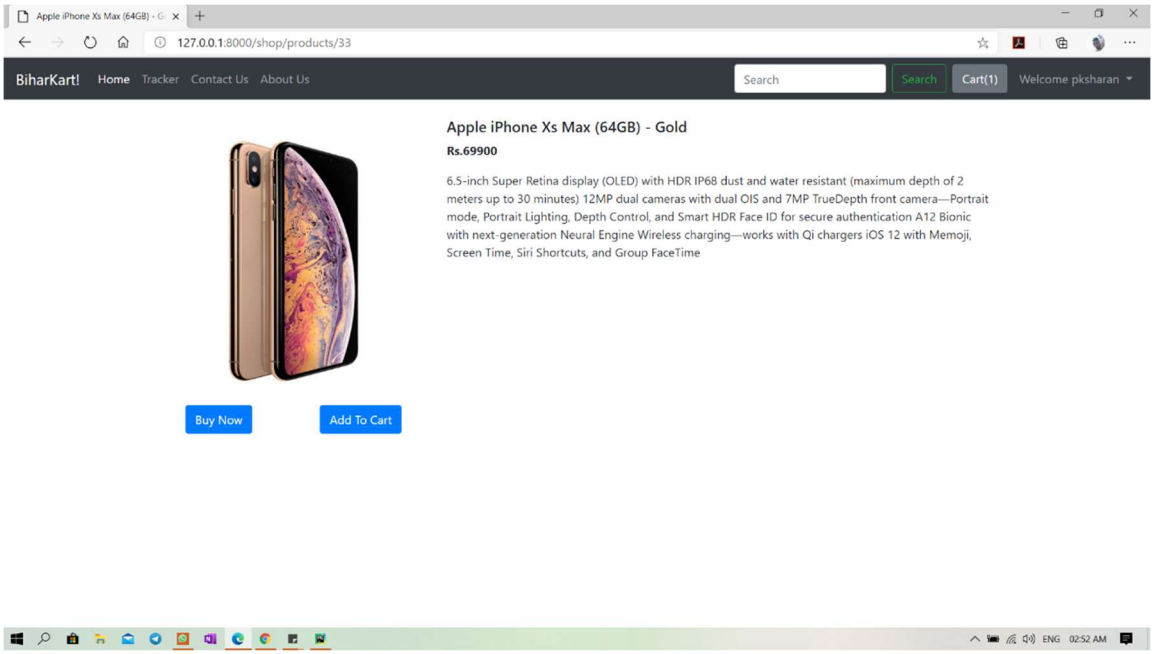
Img. 3: Login Page

Search Page



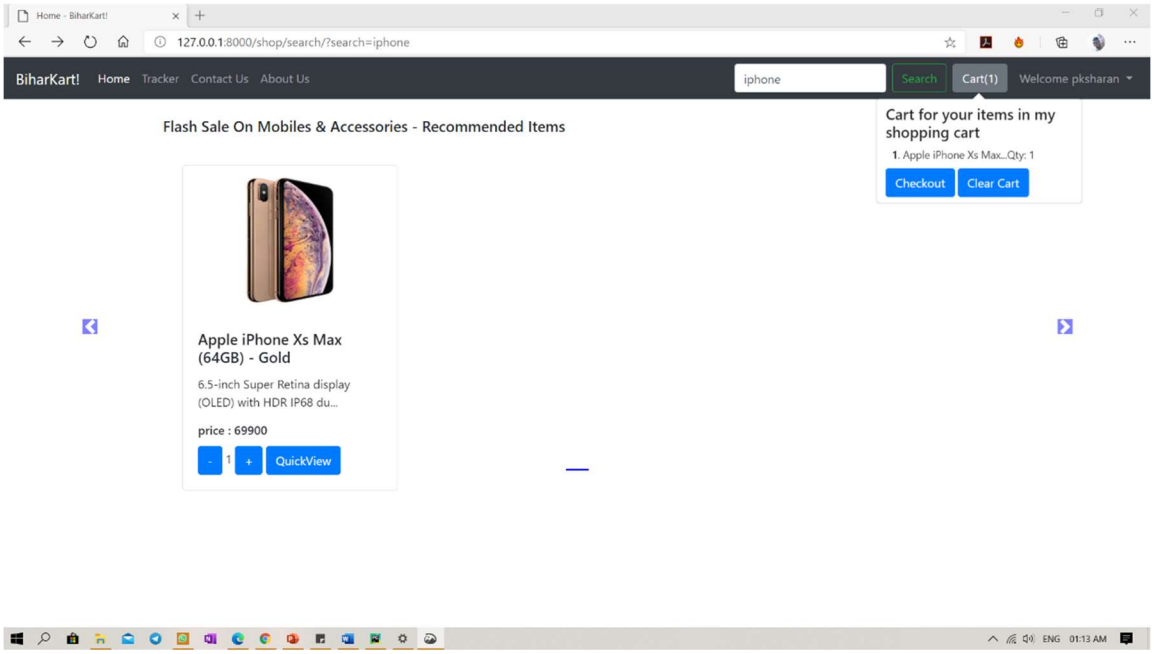
Img. 4: Search Page

Product View Page



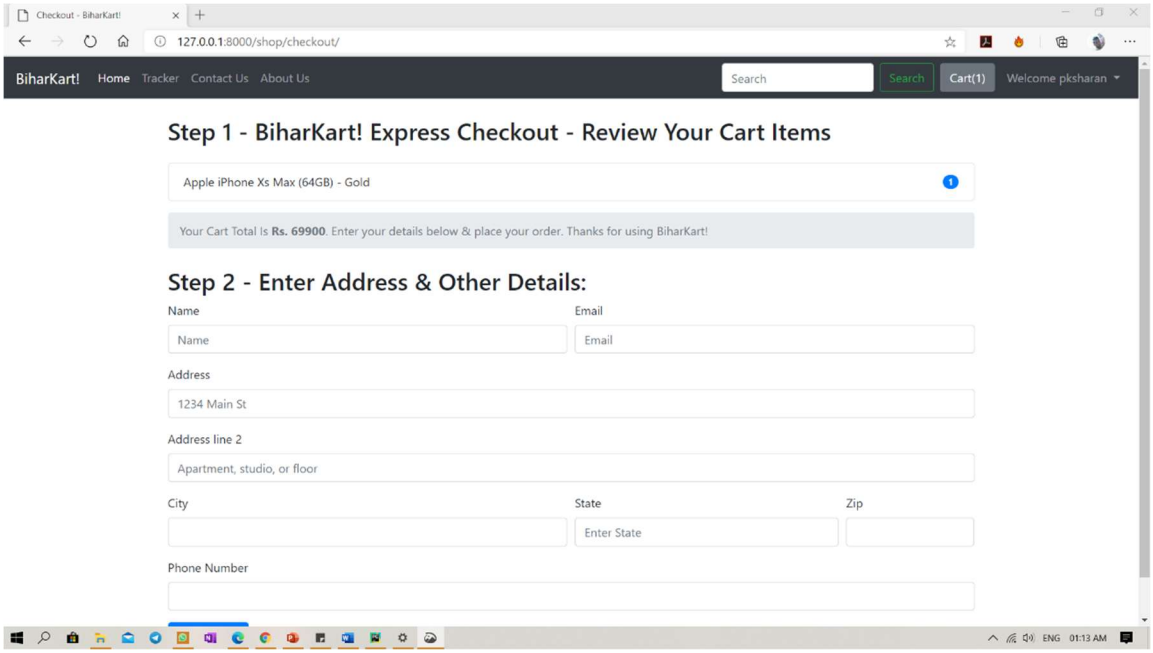
Img. 5: Product View Page

Add to cart & Cart view



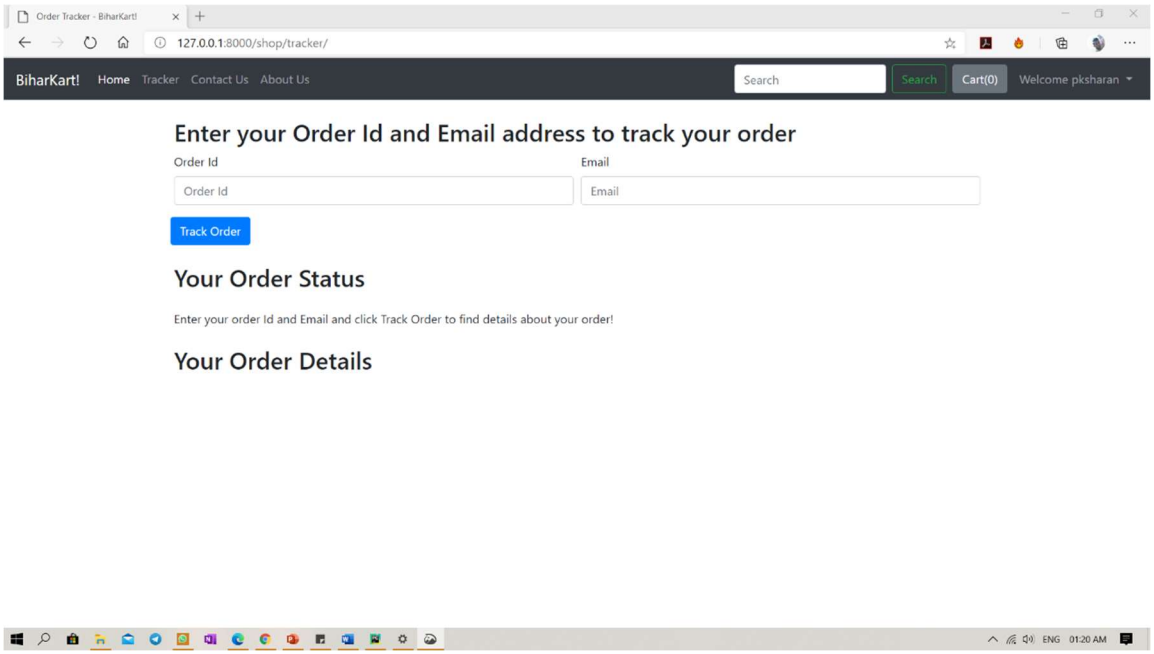
Img. 6: Add to cart & Cart view

CheckOut Page



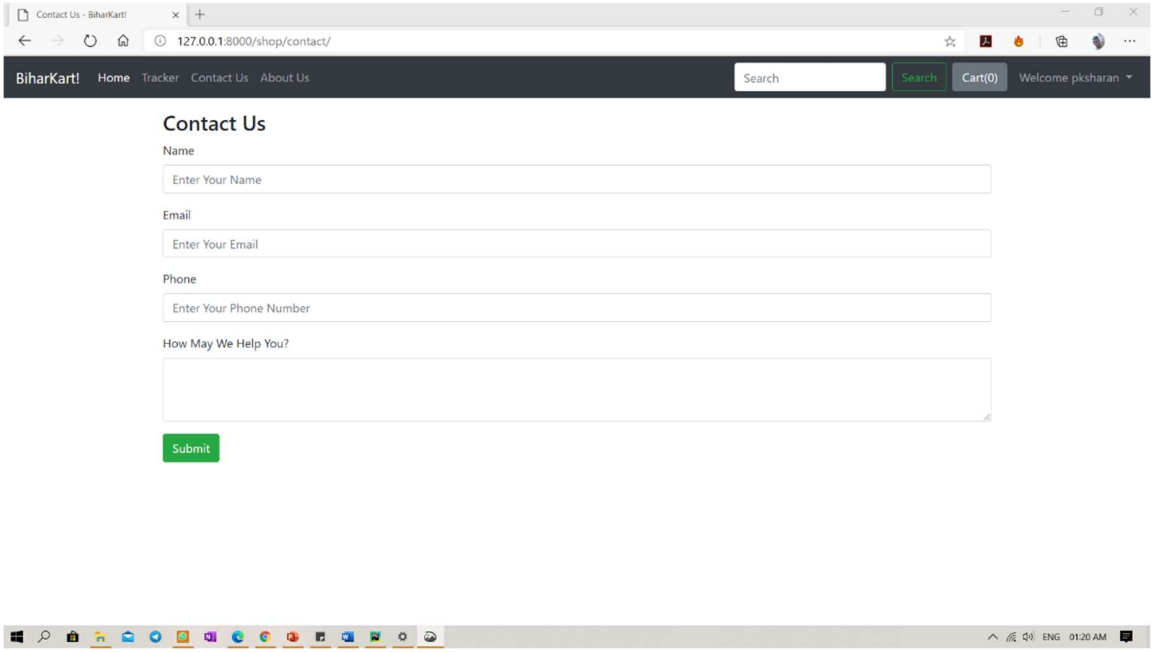
Img. 7: CheckOut Page

Order Tracker Page



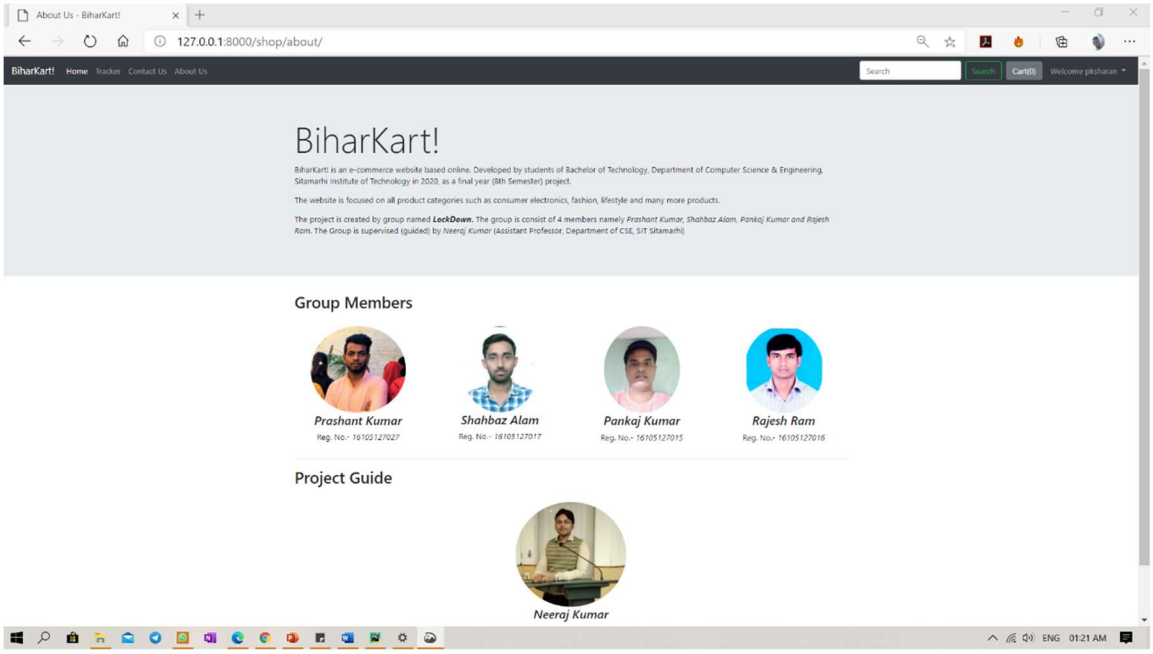
Img. 8: Order Tracker Page

Contact Us Page



Img. 9: Contact Us Page

About Us Page



Img. 10: About Us Page

BENEFITS

The following are the benefits of using the E-Commerce website:

For Customers

1. Customers can order any product anytime (website provides 27X7 service).
2. Customers locate their desired product quicker.
3. Customers will get abundant information about product.
4. Customers will have not to travel long distances to reach their preferred physical store. They can visit the same store virtually, with just a few clicks.
5. Customers can make their analysis of expenditure on marketing easily.

For Sellers

1. Sellers will gain new customers with search engine visibility.
2. Sellers will be able to get detailed report anytime.
3. Sellers can make analysis of all the sales.
4. It will automatically save sale reports digitally, so sellers will have not to keep old and bulky registers.

SAMPLE CODE

Home Page (index.html):-

```
{% extends 'shop/basic.html' %}
{% block title%}Home - BiharKart!{% endblock %}
{% block css %}
.col-md-3
{
display: inline-block;
margin-left:-4px;
}
.carousel-indicators .active {
background-color: blue;
}
.col-md-3 img{
width: 170px;
height: 200px;
}
body .carousel-indicator li{
}
body .carousel-indicators{
bottom: 0;
}
body .carousel-control-prev-icon,
body .carousel-control-next-icon{
background-color: blue;
}
.carousel-control-prev,
.carousel-control-next{
top: auto;
bottom: auto;
padding-top: 222px;
}
body .no-padding{
padding-left: 0,
padding-right: 0;
}
{% endblock %}
{% block body %}
{% load static %}
<div class="container">
    <!--Slideshow starts here -->
    {% for product, range, nSlides in allProds %}
    <h5 class="my-4">Flash Sale On {{product.0.catogary}} -
Recommended Items</h5>
    <div class="row">
        <div id="demo{{forloop.counter}}" class="col carousel
slide my-3" data-ride="carousel">
```

```
            <ul class="carousel-indicators">
                <li data-target="#demo{{forloop.counter}}" data-
slide-to="0" class="active"></li>
                {% for i in range %}
                <li data-
target="#demo{{forloop.parentloop.counter}}" data-slide-
to="{{i}}"></li>
                {% endfor %}
            </ul>
            <div class="container carousel-inner no-padding">
                <div class="carousel-item active">
                    {% for i in product %}
                    <div class="col-xs-3 col-sm-3 col-md-3">
                        <div class="card align-items-center"
style="width: 18rem;">
                            <img src='/media/{{i.image}}'
class="card-img-top" alt="...">
                            <div class="card-body">
                                <h5 class="card-title"
id="namepr{{i.id}}">{{i.product_name}}</h5>
                                <p class="card-
text">{{i.desc|slice:"0:53"}}...</p>
                                <h6 class="card-title" >price :
<span id="pricepr{{i.id}}">{{i.price}}</span></h6>

                                <span id="divpr{{i.id}}"
class="divpr">
                                    <button id="pr{{i.id}}"
class="btn btn-primary cart">Add To Cart</button>
                                </span>
                                <a
href="/shop/products/{{i.id}}"><button id="qv{{i.id}}" class="btn
btn-primary cart">QuickView</button></a>
                            </div>
                        </div>
                    </div>
                    {% if forloop.counter|divisibleby:4 and
forloop.counter > 0 and not forloop.last %}
                </div>
                <div class="carousel-item">
                    {% endif %}
                    {% endfor %}
                </div>
            </div>
        </div>
        <!-- left and right controls for the slide -->
        <a class="carousel-control-prev"
href="#demo{{forloop.counter}}" data-slide="prev">
            <span class="carousel-control-prev-icon"></span>
```

```
        </a>
        <a class="carousel-control-next"
href="#demo{{forloop.counter}}" data-slide="next">
            <span class="carousel-control-next-icon"></span>
        </a>
    </div>
    {% endfor %}
</div>
{% endblock %}
{% block js %}
<script>
// Find out the cart items from localStorage
if (localStorage.getItem('cart') == null) {
    var cart = {};
} else {
    cart = JSON.parse(localStorage.getItem('cart'));
    document.getElementById('cart').innerHTML =
Object.keys(cart).length;
    updateCart(cart);




}
// If the add to cart button is clicked, add/increment the item
//$('.cart').click(function() {
$('.divpr').on('click', 'button.cart', function(){
    var idstr = this.id.toString();
    if (cart[idstr] != undefined) {
        qty = cart[idstr][0] + 1;

    } else {
        qty = 1;
        name = document.getElementById('name'+idstr).innerHTML;
        price = document.getElementById('price'+idstr).innerHTML;
        cart[idstr] = [qty, name, parseInt(price)];
    }
    updateCart(cart);
});


//Add Popover to cart
$('#popcart').popover();


updatePopover(cart);
function updatePopover(cart)
{
    console.log('we are inside updatePopover');
```

```
    var popStr = "";
    popStr = popStr +"<h5>  Cart for your items in my shopping
cart</h5><div class=' mx-2 my-2'> ";
    var i = 1;
    for(var item in cart){

        popStr = popStr + "<b>" + i + "</b>. ";
        popStr = popStr + document.getElementById('name' + item)
.innerHTML.slice(0, 19) + "...Qty:   "  + cart[item][0]
        + '<br>';
        i = i+1;


    }

   // popStr = popStr + "</div>"

    popStr = popStr + "</div> <a href='/shop/checkout'><button
class='btn btn-primary' id ='checkout'>Checkout</button></a>
<button class='btn btn-primary' onclick='clearCart()' id
='clearCart'>Clear Cart</button>     "

    console.log(popStr);
    document.getElementById("popcart").setAttribute('data-
content', popStr);
    $('#popcart').popover();







}



function clearCart() {
    cart = JSON.parse(localStorage.getItem('cart'));
    for (var item in cart) {
        document.getElementById('div' + item).innerHTML =
'<button id="' + item + '" class="btn btn-primary cart">Add To
Cart</button>'
    }
    localStorage.clear();
    cart = {};
    updateCart(cart);
}

function updateCart(cart) {
```

```
    var sum =0;
    for (var item in cart) {
        sum = sum + cart[item][0];
        document.getElementById('div' + item).innerHTML =
"<button id='minus" + item + "' class='btn btn-primary minus'>-
</button> <span id='val" + item + "'>" + cart[item][0] +
"</span> <button id='plus" + item + "' class='btn btn-primary
plus'> + </button>";
    }

    localStorage.setItem('cart', JSON.stringify(cart));
    document.getElementById('cart').innerHTML = sum;
    console.log(cart)

    updatePopover(cart);
}

// If plus or minus button is clicked, change the cart as well as
the display value
$('.divpr').on("click", "button.minus", function() {
    a = this.id.slice(7, );
    cart['pr' + a][0] = cart['pr' + a][0] - 1;
    cart['pr' + a][0] = Math.max(0, cart['pr' + a][0]);


    document.getElementById('valpr' + a).innerHTML = cart['pr' +
a][0];
    updateCart(cart);


});


$('.divpr').on("click", "button.plus", function() {

    a = this.id.slice(6, );
    cart['pr' + a][0] = cart['pr' + a][0] + 1;

    document.getElementById('valpr' + a).innerHTML = cart['pr' +
a][0];
    updateCart(cart);



});

</script>
{% endblock %}
```

Nav Bar (basic.html):-

```
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1, shrink-to-fit=no">

    <!-- Bootstrap CSS -->
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.2.1/css/boot
strap.min.css" integrity="sha384-
GJzZqFGwb1QTTN6wy59ffF1BuGJpLSa9DkKMp0DgiMDm4iYMj70gZWKYbI706tWS"
crossorigin="anonymous">

    <title>{% block title%} {% endblock %}</title>
     <style>
       {% block css %} {% endblock %}
  </style>
  </head>
  <body>

    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
  <a class="navbar-brand" href="/shop">BiharKart!</a>
  <button class="navbar-toggler" type="button" data-
toggle="collapse" data-target="#navbarSupportedContent" aria-
controls="navbarSupportedContent" aria-expanded="false" aria-
label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>



   </div>

  <div class="collapse navbar-collapse"
id="navbarSupportedContent">
    <ul class="navbar-nav mr-auto">
      <li class="nav-item active">
        <a class="nav-link" href="/shop">Home <span class="sr-
only">(current)</span></a>
      </li>

      <li class="nav-item">
        <a class="nav-link" href="/shop/tracker">Tracker</a>
      </li>
```

```
      <li class="nav-item">
        <a class="nav-link" href="/shop/contact">Contact Us</a>
      </li>

      <li class="nav-item">
        <a class="nav-link" href="/shop/about">About Us</a>
      </li>


    </ul>
     <form method='get' action='/shop/search/' class="form-inline
my-2 my-lg-0">
      <input class="form-control mr-sm-2" type="search"
placeholder="Search" aria-label="Search" name="search"
id="search">
      <button class="btn btn-outline-success my-2 my-sm-0"
type="submit">Search</button>

    </form>
    <button type="button" class="btn btn-secondary mx-2"
id="popcart" data-container="body" data-toggle="popover" data-
placement="bottom" data-html="true" data-content="Vivamus
sagittis lacus vel augue laoreet rutrum faucibus.">


  Cart(<span id="cart">0</span>)
</button>
  </div>

 {% if user.is_authenticated %}
  <ul class="navbar-nav ml-auto">

          <li class="nav-item dropdown">
        <a class="nav-link dropdown-toggle" href="#"
id="navbarDropdown" role="button" data-toggle="dropdown" aria-
haspopup="true" aria-expanded="false">
           Welcome {{request.user}}

        </a>
        <div class="dropdown-menu" aria-
labelledby="navbarDropdown">
          <a class="dropdown-item"
href="/shop/logout/">Logout</a>
          <div class="form-group">
        {% if messages %}
        <ul class="messages">
          {% for message in messages %}

          <li>
```

```
            <div class="alert alert-succes">{{  message }}</div>
          </li>
          {% endfor %}
        </ul>
        {% endif %}




          </div>
      </li>
  </ul>


 {% else %}



<!-- signup trigger modal -->
<button type="button" class="btn btn-primary mx-2" data-
toggle="modal" data-target="#loginModal">
  Login
</button>

<!-- signup trigger modal -->
<button type="button" class="btn btn-primary mx-2" data-
toggle="modal" data-target="#signupModal">
  Signup
</button>


 {% endif%}

</nav>

<!--signup Modal -->
<div class="modal fade" id="signupModal" tabindex="-1"
role="dialog" aria-labelledby="signupModal" aria-hidden="true">

  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title" id="signupModalTitle">signup
here</h5>
        <button type="button" class="close" data-dismiss="modal"
aria-label="Close">
          <span aria-hidden="true">&times;</span>
        </button>
```

```
      </div>
      <div class="modal-body">
        <form action="/shop/signup/"  method="post">
       <div class="form-group">
    <label for="username">Username(Lowercase &
Alphanumaric)</label>
    <input type="text" class="form-control" id="username"
name="username" placeholder="choose a unique username" required>
  </div>
  <div class="form-group">
    <label for="fname">First name</label>
    <input type="text" class="form-control" id="fname"
name="fname" placeholder="First Name" required>
  </div>
    <div class="form-group">
    <label for="lname">Last name</label>
    <input type="text" class="form-control" id="lname"
name="lname" placeholder="Last Name" required>
  </div>
    <div class="form-group">
    <label for="email">Email adress</label>
    <input type="email" class="form-control" id="email"
name="email" placeholder="name@example.com" required>
  </div>
    <div class="form-group">
    <label for="pass1">choose a password</label>
    <input type="password" class="form-control" id="pass1"
name="pass1" placeholder="Choose your password" required>
  </div>
    <div class="form-group">
    <label for="pass2">Confirm password</label>
    <input type="password" class="form-control" id="pass2"
name="pass2" placeholder="enter your password again" required>
  </div>
    {%csrf_token %}
    <button type="submit"  class="btn btn-primary"  class="nav-
link" href="/shop/login/">Submit</button>
</form>
      </div>
      <div class="modal-footer">

      </div>
    </div>
  </div>
</div>
```

```
<!--login Modal -->
<div class="modal fade" id="loginModal" tabindex="-1"
role="dialog" aria-labelledby="loginModal" aria-hidden="true">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title" id="loginModalTitle">login</h5>
        <button type="button" class="close" data-dismiss="modal"
aria-label="Close">
          <span aria-hidden="true">&times;</span>
        </button>
      </div>
      <div class="modal-body">
  <form action="/shop/login/" method="post">
   {% csrf_token %}
  <div class="form-group">
    <label for="username">Username</label>
    <input type="text" class="form-control" id="loginusername"
name="loginusername" placeholder="enter user name" required>
  </div>
 <div class="form-group">
    <label for="pass"> password</label>
    <input type="password" class="form-control"
id="loginpassword" name="loginpassword" placeholder="enter your
password " required>
  </div>


    <button type="submit" class="btn btn-primary"
href="mm.html">Submit</button>
</form>
      </div>
      <div class="modal-footer">

      </div>
    </div>
  </div>
</div>


  {% block body %} {% endblock %}

    <!-- Optional JavaScript -->
    <!-- jQuery first, then Popper.js, then Bootstrap JS -->
    <script src="https://code.jquery.com/jquery-3.3.1.js"
  integrity="sha256-2Kok7MbOyxpgUVvAk/HJ2jigOSYS2auK4Pfzbm7uH60="
  crossorigin="anonymous"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.6/umd/
```

```
popper.min.js" integrity="sha384-
wHAiFfRlMFy6i5SRaxvfOCifBUQy1xHdJ/yoi7FRNXMRBu5WHdZYu1hA6ZOblgut"
crossorigin="anonymous"></script>
    <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.2.1/js/bootst
rap.min.js" integrity="sha384-
B0UglyR+jN6CkvvICOB2joaf5I4l3gm9GU6Hc1og6Ls7i6U/mkkaduKaBhlAXv9k"
crossorigin="anonymous"></script>
    {% block js %} {% endblock %}
  </body>
</html>
```

Backend Functions (views.py):-

```
from django.shortcuts import render, redirect
from .models import product, Contact, Orders, OrderUpdate
from math import ceil
import json

# Create your views here.
from django.http import HttpResponse
from django.contrib.auth.models import User
from django.contrib import messages
from  django.contrib.auth import authenticate, login, logout


def index(request):
    allProds = []
    catprods = product.objects.values('catogary', 'id')
    cats = {item['catogary'] for item in catprods}
    for cat in cats:
        prod = product.objects.filter(catogary=cat)
        n = len(prod)
        nSlides = n // 4 + ceil((n / 4) - (n // 4))
        allProds.append([prod, range(1, nSlides), nSlides])
    params = {'allProds':allProds}
    return render(request, 'shop/index.html', params)


def about(request):
    return render(request, 'shop/about.html')


def contact(request):
    if request.method=="POST":
        name = request.POST.get('name', '')
        email = request.POST.get('email', '')
        phone = request.POST.get('phone', '')
```

```
        desc = request.POST.get('desc', '')
        contact = Contact(name=name, email=email, phone=phone,
desc=desc)
        contact.save()
    return render(request, 'shop/contact.html')


def tracker(request):
    if request.method=="POST":
        orderId = request.POST.get('orderId', '')
        email = request.POST.get('email', '')
        try:
            order = Orders.objects.filter(order_id=orderId,
email=email)
            if len(order)>0:
                update =
OrderUpdate.objects.filter(order_id=orderId)
                updates = []
                for item in update:
                    updates.append({'text': item.update_desc,
'time': item.timestamp})
                    response = json.dumps({"status":"success",
"updates": updates, "itemsJson": order[0].items_json},
default=str)
                return HttpResponse(response)
            else:
                return HttpResponse('{"status":"noitem"}')
        except Exception as e:
            return HttpResponse('{"status":"error"}')

    return render(request, 'shop/tracker.html')



def searchMatch(query, item):
    '''return true only if query matches the item'''
    if query in item.desc.lower() or query in
item.product_name.lower() or query in item.catogary.lower():
        return True
    else:
        return False


def search(request):
    query = request.GET.get('search')
    allProds = []
    catprods = product.objects.values('catogary', 'id')
    cats = {item['catogary'] for item in catprods}
    for cat in cats:
```

```
        prodtemp = product.objects.filter(catogary=cat)
        prod = [item for item in prodtemp if searchMatch(query,
item)]

        n = len(prod)
        nSlides = n // 4 + ceil((n / 4) - (n // 4))
        if len(prod) != 0:
            allProds.append([prod, range(1, nSlides), nSlides])
    params = {'allProds': allProds, "msg": ""}
    if len(allProds) == 0 or len(query) < 4:
        params = {'msg': "Please make sure to enter relevant
search query"}
    return render(request, 'shop/index.html', params)


def productView(request, myid):

    # Fetch the product using the id
    Product = product.objects.filter(id=myid)
    return render(request, 'shop/prodView.html',
{'product':Product[0]})


def checkout(request):
    if request.method=="POST":
        items_json = request.POST.get('itemsJson', '')
        amount = request.POST.get('amount', '')
        name = request.POST.get('name', '')
        email = request.POST.get('email', '')
        address = request.POST.get('address1', '') + " " +
request.POST.get('address2', '')
        city = request.POST.get('city', '')
        state = request.POST.get('state', '')
        zip_code = request.POST.get('zip_code', '')
        phone = request.POST.get('phone', '')
        order = Orders(items_json=items_json, name=name,
email=email, address=address, city=city,
                       state=state, zip_code=zip_code,
phone=phone, amount=amount)
        order.save()
        update = OrderUpdate(order_id=order.order_id,
update_desc="The order has been placed")
        update.save()
        thank = True
        id = order.order_id
        return render(request, 'shop/checkout.html',
{'thank':thank, 'id': id})
    return render(request, 'shop/checkout.html')
```

```
def handleSignup(request):

    if request.method == 'POST':

        username = request.POST['username']
        fname = request.POST['fname']
        lname = request.POST['lname']
        email = request.POST['email']
        pass1 = request.POST['pass1']
        pass2 = request.POST['pass2']


        if len(username)>10:
            messages.error(request, "username must be under 10
character")
            return redirect('shophome')

        if not username.isalnum():
            messages.error(request, "username must be only
contain letter and number ")
            return redirect('shophome')


        if pass1 != pass2:
            messages.error(request, "password do not match")
            return redirect('shophome')


        myuser = User.objects.create_user(username, email, pass1)
        myuser.first_name = fname
        myuser.last_name = lname
        #myuser.confirm_password = pass2
        myuser.save()
        messages.success(request, "your acccount has been
created")
        return redirect('shophome')
    else:

        return HttpResponse('404 not found')



def handleLogin(request):
    if request.method == "POST":
        loginusername = request.POST['loginusername']
        loginpassword = request.POST['loginpassword']


        user = authenticate(username=loginusername,
```

```
password=loginpassword)

        if user is not None:
            login(request, user)
            messages.success(request, "succecsfull login")
            return redirect('shophome')

        else:
            messages.error(request, "invalid credition , please
try again")
            return redirect('shophome')


    return HttpResponse('handleLogin')


def handleLogout(request):

    logout(request)
    messages.success(request, " succsessfull logout")
    return redirect('shophome')
```

CONCLUSION

After careful observation, it has come to my conclusion that e-commerce has undeniably become an important part of our society. The world wide web is and will have a large part in our daily lives. It is therefore critical that small businesses have their own to keep in competition with the larger websites. Since web developers have lowered down the prices for their services, it has become more affordable for small businesses to use the world wide web to sell their products. Although there are negative aspects of e-commerce, small businesses have tried to accommodate to the needs of the consumers. For example, one of the negative aspects of e-commerce is that consumers lack the advice and guidance of sellers, to accommodate that, they have customer service through the phone of online to answer any questions. It is also important to note that e-commerce does not benefit all small companies equally. How much revenue a business gets from e-commerce depends on what kind of service it gives. For example, most people would like to try on clothes before they buy them, so it probably would not benefit a small business that sells clothes as much as a small business that sells home supplies or specialty books. Nevertheless, e-commerce does benefit any business even in small ways. This is why it is crucial to understand how e-commerce affects small businesses because it is becoming such a huge part of how society functions that it effects the economy greatly and whatever happens to the economy affects us. This is why is it important to understand this subject because in the long run, it will affect all of us.

FUTURE ENHANCEMENT

Today, the market place is flooded with several e-commerce options for shoppers to choose from. A variety of innovative products and services are being offered spoiling customers for choice. Online shopping is no more a privilege enjoyed by your friends and family living in US or UK. Today, it is a reality in India. In the last couple of years. The growth of e-commerce industry in India has been phenomenal as more shoppers have started discovering the benefits of using this platform. There is enough scope for online businesses in the future if they understand the Indian shopper's psyche and cater to their needs.

BIBLIOGRAPHY

- https://www.python.org/
- https://www.djangoproject.com/
- https://getbootstrap.com/
- Dr. Shahid Amin, Prof. Keshav Kansana, Jenifur Majid. "*A Review Paper on E-Commerce.*" TIMS 2016-International Conference. Gwalior. 2016
- Nitika goyal, Deepam Goyal. "*Impact of E-Commerce in India: Issues & Challenges.*" International Journal of Advanced Research in Computer Science. Volume 7, No. 6(Special Issue), November 2016
- Ashok Panigrahi, Ranjan Upadhyaya, Dr. P. P. Raichurkar. "*E-Commerce Services in India: Prospects and Problems.*" International Journal on Textile Engineering and Processes. Vol 2, Issue 1, January 2016
- Nisha Chanana, Sangeeta Goele. "*Future of E-Commerce in India.*" International Journal of Computing & Business Research. 2012
- Raj Kumar Singh. "*E-Commerce in India: Opportunities and Challenges.*" Proceedings of 10th International Conference on Digital Strategies for Organizational Success. January 6, 2019
- Dr. B Karunakar, Bisheswar Sinha. "*E-Commerce in India.*" International Journal of Management Research and Business Strategy. Volume 5, Issue 3, July 2016