Online Shopping System

A Major Project report for the evaluation and partial fulfillment of the requirement for the award of the degree

B. TECH. (IT)

Submitted By –

Names	Roll Number
Aditya Aggarwal	(18/BIT/004)
Mohammad Anas	(18/BIT/032)
Mohit Gautam	(18/BIT/034)
Prasoon Srivastava	(18/BIT/041)

UNDER THE SUPERVISION OF
Dr. Aarti Gautam Dinker

AND THE CO-SUPERVISION OF
Mr. Ankur Singhal

UNIVERSITY SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

GAUTAM BUDDHA UNIVERSITY

GREATER NOIDA –201312, GAUTAM BUDDHA NAGAR UTTAR PRADESH INDIA

31st May, 2022

TABLE OF CONTENTS

ACKNOWLEDGMENT

I would like to express our special thanks to our Mentors Dr. Aarti Gautam Dinker and Mr. Ankur Singla as well as our HOD Dr. Neeta Singh who gave us the opportunity to do this wonderful project on the topic "Online Shopping System" which also helped us in doing a lot of Research and also came to know about so many new things.
Secondly we would also like to thank all our friends and teachers who helped us a lot in finalizing this project within the time frame.

By:- Aditya Aggarwal
Mohammad Anas
Mohit Gautam
Prasoon Srivastava

ABOUT THE PROJECT

"**Online Shopping System**" is a web-based project which is made for remote-home shopping or local Home services through Internet.

As the technology is advancing the way of life is changing accordingly. Now-a-day's we can place the order for anything from our home. There is no need to go a physical store for services we want. The order can be placed online through Internet.

Now we think that how the days have been changed with time i.e. people had to stand in rows and to wait for serviceman for their turn to get billed for a particular thing from a shop. But now-a-days, instead, we can get those items on the door-step in few hours. People no longer had to suffer the rush of the shop & service shop. Due to the advancement of technology, online shopping is the way for fast and hassle-free shopping. The way of shopping was completely changed with the coming of Internet Technology. People have to fill a simple form on the internet to place their order on any popular local service, shop or service centre for the thing they want to purchase. Now they can place their order from the home.

LANGUAGE AND TOOLS TO BE USED

Hardware Requirement:-

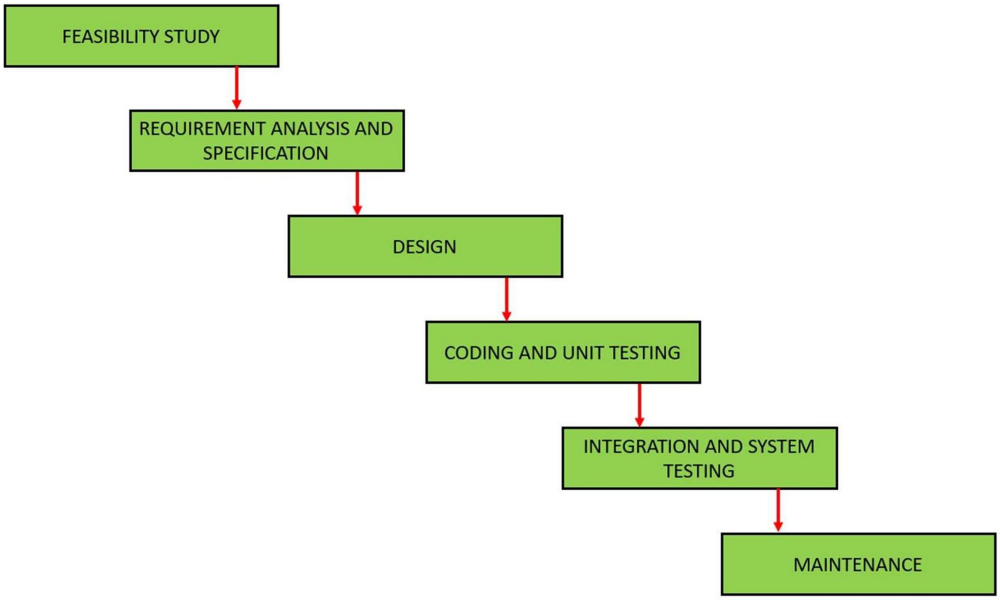
- Processor : CORE i3 Processor
- Secondary Storage: 20 GB free space on HDD
- Memory: 2 GB RAM
- Network Adapter: Ethernet Adapter
- Others: Color Monitor, Keyboard, Mouse.

Software Requirement:-

- Platform: Windows
- Operating System: Windows 7,8,8.1,10.
- IDE: Visual Studio Code
- Front End Tool: Bootstrap
- Back- End Tool: Django
- Scripting Tool: html , js script, json
- Server: Firebase
- For designing Front End, we are using Bootstrap. And for Database, we are using Django

THE SOFTWARE DEVELOPMENT LIFE CYCLE

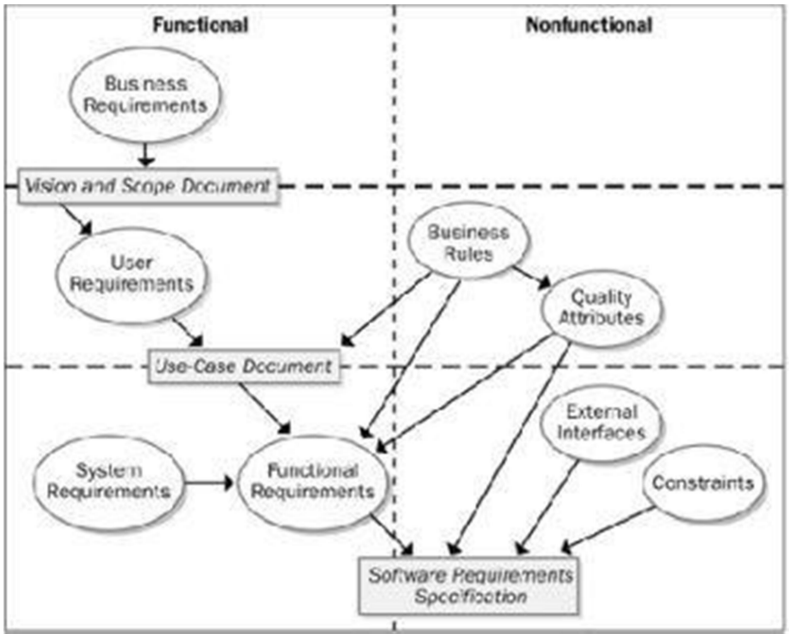
Classical waterfall model is the basic **software development life cycle** model. It is very simple but idealistic. Earlier this model was very popular but nowadays it is not used. But it is very important because all the other software development life cycle models are based on the classical waterfall model. Classical waterfall model divides the life cycle into a set of phases. This model considers that one phase can be started after completion of the previous phase. That is the output of one phase will be the input to the next phase. Thus, the development process can be considered as a sequential flow in the waterfall. Here the phases do not overlap with each other. The different sequential phases of the classical waterfall model are given below:



REQUIREMENT ANALYSIS

The broad spectrum of tasks and techniques that lead to an understanding of requirements is called requirements engineering. Requirements engineering, which is a major process in software engineering, provides the appropriate mechanism for understanding what the customer wants, analyzing need, assessing feasibility, specifying the solution unambiguously, and so on. It encompasses seven distinct tasks, which are inception, elicitation, elaboration, negotiation, specification, validation, and management. Some of these tasks occur in parallel and all are adapted to the needs of the project. At project inception, among other things, system analysts establish a basic understanding of the problem, the people who want a solution and the nature of the solution that is desired. The activity includes establishing product vision and project scope.

Requirements analysis is the activity of elaborating basic requirements established during the inception, elicitation and negotiation tasks. Requirements analysis results in the software specification detailing the operational characteristics, interface with other system elements, and constraints that the software must meet.



DESIGN CONSTRAINTS

Retrieve sensors, actuators, scenarios and constraints:-

The display of service icons as well as the management window needs information about available services as well as saved scenarios and constraints. This information is provided to the GUI by the HomeBlock system in JSON format through a simple HTTP API. It ensures the decoupling between the verification engine and the user interface, making the back-end reusable easily with different GUI.

Specify scenarios and constraints:-

The GUI allows specifying scenarios and constraints by building ECA rules presented in Sect. 2.3. Based on previous experiences, it was decided to disallow users to directly specify conditions inside scenarios. This is because users often confuse between the event and condition parts. Conditions are instead proposed by the system when a scenario under specification violates a constraint, as shown in Fig. 10c (the careful reader will have noticed that the scenario in Fig. 9a is inconsistent with the constraint in Fig. 9b). If the user accepts the condition, a column is added to the right of the actions, containing a list of conditions that restrict the execution of the scenario. The updated scenario is shown in Fig. 10d.

Upload scenarios and constraints:-

Every time a valid scenario—composed of at least one event and one action—is modified or saved, an updated version is sent to HomeBlock for verifying its consistency against active constraints. The first step taken by the system is to transform the scenario, received in the form of an ECA rule, to TA, as shown in Sect. 2.3. If it contains a service that is not part of the system model yet, it is also added to it as presented in Sect. 3.2. Constraints are uploaded upon saving, and are also transformed to their TA representation as shown in Sect. 2.3. At the moment, running scenarios are not checked against newly added constraints, but it would be a technical matter to enable this function.

SYSTEM ANALYSIS

Hardware Requirements:

- Processor : i3
- RAM : 2 gb
- Hard Disk : 20 gb
- Communication Channel : Internet

SOFTWARE REQUIREMENT SPECIFICATION

The Software Requirements Specification is produced at the culmination of the analysis task. The function and performance allocated to software as part of system engineering are refined by establishing a complete information description, a detailed functional and behavioural description, an indication of performance requirements and design constraints, appropriate validation criteria, and other data pertinent to requirements.

 Functional Requirement Specification (FRS):

Functional Requirement Specification is termed as FRS document. This document serves as a detailed illustration of all low-level granular specification of system that is to present into the fulfilment of software. Following are some features of FRS:

- This document elaborates on the functions to the user.
- Helps user to understand interactions of software and its behaviour.
- It helps designing testing parameters for software.
- Provides better aid for wire-frames and conceptual diagrams.

 Non-functional requirements: These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to other. They are also called non-behavioral requirements. They basically deal with issues like:

- Portability
- Security
- Maintainability
- Reliability
- Scalability
- Performance
- Reusability
- Flexibility

PROJECT PLAN

Scope management:

Automated attendance management is a very active topic of research. A lot of work has been done in this field and there is a lot to improve. Being one of the most successful applications of the image processing, face recognition has a vital role in technical field especially in the field of security purpose. Human face recognition is an important field for verification purpose especially in the case of student's attendance. This project is aimed at implementing a digitized system for attendance recording. Current attendance marking methods are monotonous & time consuming. Manually recorded attendance can be easily manipulated. Hence this method is proposed to tackle all these issues.

As part of the futurework, we would like to develop an application that would allow theuser to add or delete face classes in the training set. This would give usersthe freedom to define their own user groups rather than a pre-defined set on the server. We would also like to explore better algorithms for face detection and face recognition to increase the number of studentsto be detected and recognize.

People management:

S. NO.	Activity	Planned StartDate	Planned End Date	Actual StartDate	Actual End Date	Remarks
1	Identify Project					
2	Making Project Plan					
3	System Study					
4	System Design					
5	System Testing					
6	Implementation					

Risk Management Plan:

One of the problems in real-time face recognition is the difficulty to obtain sufficient and suitable images for training and testing purpose. It is hard to obtain in real-time databases with a variety of variables, and it is hard to obtain publicly available databases. Yale face database is one of the databases that could be downloaded by the public. Hence, Yale face database is adopted and used in this proposed approach. However, Yale face database consists of only grayscale images without any background. Hence, our own database consists of colour images which is categorized to high- quality images and low quality-images are also used. Besides, it is very difficult to obtain an open source or the free face recognition software in order to make comparisons. In this proposed approach, Face SDK window demo version software is downloaded and implemented in the laptop. By using laptop built in webcam to recognize faces, the proposed algorithm and Face SDK demo able to be compared. From the Face recognition website (Luxand.com, 2018), they explained that the Face SDK is a high performance, multi-platform face recognition, identification and facial feature detection solution. For Face Recognition software, the self-learning Al enables video-based identification and the enrolment can be done at any time as simple as putting a name tag in a video, the system will identify that subject in all past, present and future videos. As a video-based identification software, it is believed to work better than key-frame based identification.

Communication Plan:

In this method the camera is fixed in the classroom and it will capture the image, the **faces** are detected and then it is recognized with the database and finally the **attendance** is marked. Eigen **faces** is set of Eigen vectors which are used in computer vision problem of **face recognition**.

Time management Plan:

S. NO.	Activity	Planned StartDate	Planned End Date	Actual StartDate	Actual End Date	Remarks
1	Identify Project					
2	Making Project Plan					
3	System Study					
4	System Design					
5	System Testing					
6	Implementation					

FEASIBILITY ANALYSIS

After doing the project study and analysing all the existing or required functionalities of the system, the next task is to do the feasibility study for the project. All projects are feasible given unlimited resources and infinite time. Feasibility study includes consideration of all the possible ways to provide a solution to the given problem. The proposed solution should satisfy all the user requirements and should be flexible enough so that future changes can be easily done based on the future upcoming requirements.

ECONOMICAL FEASIBILITY

This is a very important aspect to be considered while developing a project. We decided the technology based on minimum possible cost factor.

- All hardware and software cost has to be borne by the organization.
- Overall, we have estimated that the benefits the organization is going to receive from the proposed system will surely overcome the initial costs and the later on running cost for system.

TECHNICAL FEASIBILITY

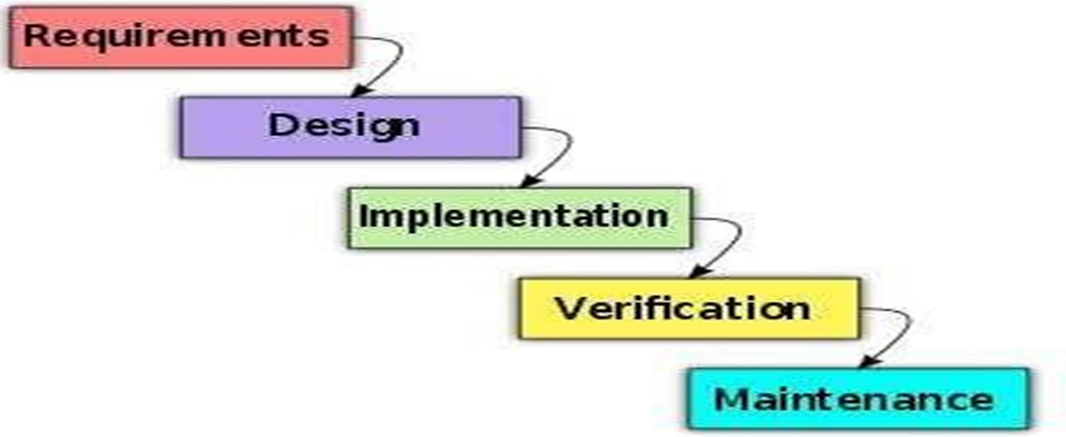
This included the study of function, performance and constraints that may affect the ability to achieve an acceptable system. For this feasibility study. we studied complete functionality to be provided in the system, as described in the System Requirement Specification (SRS), and checked if everything was possible using different type of frontend and backend platform.

OPERATIONAL FEASIBILITY

No doubt the proposed system is fully GUI based that is very user friendly and all inputs to be taken all self-explanatory even to a layman. Besides, a proper training has been conducted to let know the essence of the system to the users so that they feel comfortable with new system. As far our study is concerned the clients are comfortable and happy as the system has cut down their loads and doing.

METHODOLOGY ADOPTED

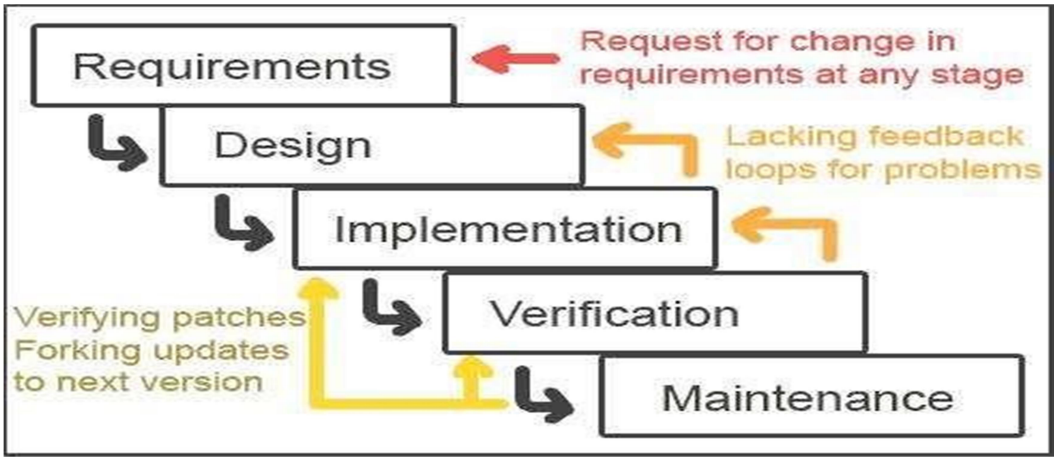
For developing the project, the methodology used here is waterfall model of SDLC. The **waterfall model** is a sequential design process, often used in software development processes, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of Conception, Initiation, Analysis, Design, Construction, Testing and Maintenance.



The unmodified "waterfall model". Progress flows from the top to the bottom, like a waterfall. The waterfall development model originates in the manufacturing and construction industries: highly structured physical environments in which after-the-fact changes are prohibitively costly, if not impossible. Since no formal software development methodologies existed at the time, this hardware-oriented model was simply adapted for software development.

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

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



SYSTEM DESIGN

System design is the solution for the creation of a new system. This phase focuses on the detailed implementation of the feasible system. It emphasis on translating design. Specifications to performance specification. System design has two phases of development:

- Logical Design
- Physical Design

During logical design phase the analyst describes inputs (sources), output s(destinations), databases (data sores) and procedures (data flows) all in a format that meets the user requirements. The analyst also specifies the needs of the user at a level that virtually determines the information flow in and out of the system and the data resources. Here the logical design is done through data flow diagrams and database design. The physical design is followed by physical design or coding. Physical design produces the working system by defining the design specifications, which specify exactly what the candidate system must do. The programmers write the necessary programs that accept input from the user, perform necessary processing on accepted data and produce the required report on a hard copy or display it on the screen.

Input Design

Input design is the link that ties the information system into the world of its users. The input design involves determining the inputs, validating the data, minimizing the data entry and provides a multi-user facility. Inaccurate inputs are the most common cause of errors in data processing. Errors entered by the data entry operators can be controlled by input design. The user-originated inputs are converted to a computer-based format in the input design. Input data are collected and organized into groups of similar data. Once identified, the appropriate input media are selected for processing. All the input data are validated and if any data violates any conditions, the user is warned by a message.

Output Design

Computer output is the most important and direct source of information to the user. Output design is a very important phase since the output needs to be in an efficient manner. Efficient and intelligible output design improves the system relationship with the user and helps in decision making. Allowing the user to view the sample screen is important because the user is the ultimate judge of the quality of output. The output module of this system is the selected notifications.

TESTING

In the Testing portion of the website, we ensure several security measures in order to save our website from various Cyber Crime attacks by Hackers and Cyber Criminals. Following points are kept in mind to ensure the security practices for this Web Application.

- No Right Click Functionality is present. This has been done in order to enhance security.
- No Clear Text Transmission of credentials of User and Admin while processing. By this attacker will not get easily the passwords of Users which will empower privacy.
- The Passwords are using MD5 Encryption.
- We implemented security measures to prevent security attacks such as Sql Injection, Cross Site Scripting (XSS), Authentication Bypass etc.
- In future when this application will be hosted on Cloud platforms this will automatically get Cloudflare and SSL protection which will make this web application more secure.

Software testing can be divided into two steps:

1. **Verification:** it refers to the set of tasks that ensure that software correctly implements a specific function.

2. **Validation:** it refers to a different set of tasks that ensure that the software that has been built is traceable to customer requirements.

- **Verification:** "Are we building the product, right?"
- **Validation:** "Are we building the right product?"

MAINTENANCE

Software Maintenance is the process of modifying a software product after it has been delivered to the customer. The main purpose of software maintenance is to modify and update software application after delivery to correct faults and to improve performance.

Need for Maintenance:

Software Maintenance must be performed in order to:

- Correct faults.
- Improve the design.
- Implement enhancements.
- Interface with other systems.
- Accommodate programs so that different hardware, software, system features, and telecommunications facilities can be used.
- Migrate legacy software.
- Retire software.

Categories of Software Maintenance:

Maintenance can be divided into the following:

1. **Corrective maintenance:**

   Corrective maintenance of a software product may be essential either to rectify some bugs observed while the system is in use, or to enhance the performance of the system.

2. **Adaptive maintenance:**

   This includes modifications and updations when the customers need the product to run on new platforms, on new operating systems, or when they need the product to interface with new hardware and software.

3. **Perfective maintenance:**

   A software product needs maintenance to support the new features that the users want or to change different types of functionalities of the system according to the customer demands.

4. **Preventive maintenance:**

   This type of maintenance includes modifications and updations to prevent future problems of the software. It goals to attend problems, which are not significant at this moment but may cause serious issues in future.

MODULE DESCRIPTION

- **HOMEPAGE:**

  The very first page user will interact with website will be the homepage. Homepage of the website will be dynamically designed as per the current industry trends. Website and all of its pages will be user friendly in terms of screen optimization i.e., website will automatically detect the screen size of gadget and will be displayed accordingly, this will be done with help of bootstrap technology.

  Homepage will show basic working of the online home service booking system. How user can browse the products, add them to their cart, visit their cart to check what services they have added etc.

- **NEW USER REGISTRATION:**

  The user needs to have an account before they can add anything in their cart. If the user is not having an account then it's necessary to register on the website. The registration form will have certain entries which will ask for the user's basic personal information and identity.

- **ADMIN LOGIN:**

  The admin is the one who is the owner of the website. The admin will have all the access to the website and can make changes when and wherever necessary. The admin will can check all the details of the user/customer related to their personal details, order details and account security. Admin can also add the new services on the website and can delete the old one if it is out of stock or removed by the seller.

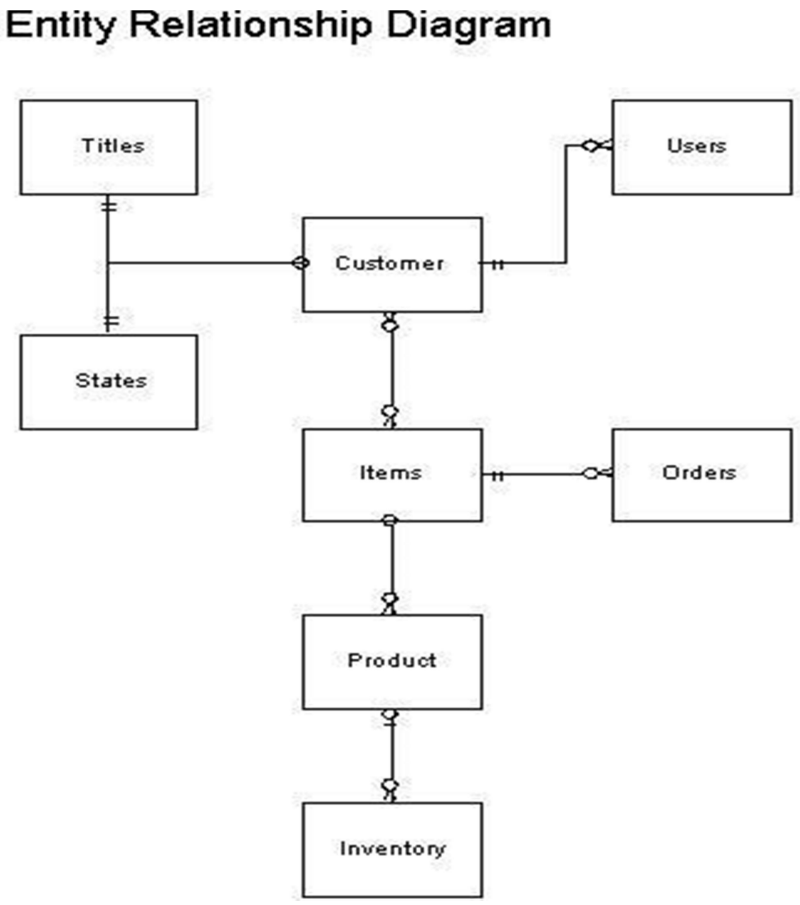
- **PRODUCTS:**

  The products page of the website will have all the products related to various categories and the customer can browse them and add them to their cart just by clicking on add to cart button. The customer can also add filter to have relevant products on the page.

- **CONTACT US:**

  In this page of the website user will get the information for contacting the website admin by just filling a form or by sending the query to the admin by the details given in the Contact us section for any UI/UX issues or functionality issues.

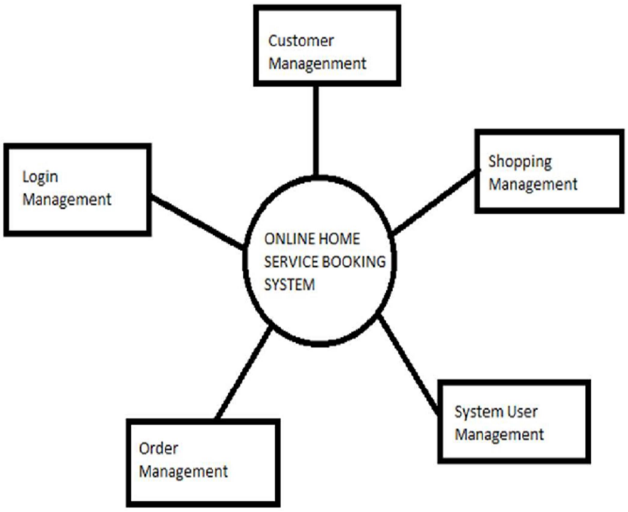
ER DIAGRAM


Entity Relationship Diagram
Titles
Users
Customer
States
Items
Orders
Product
Inventory

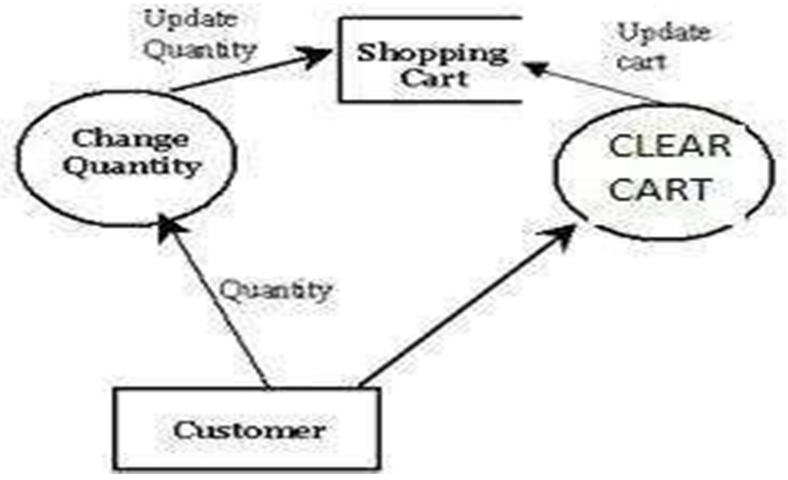
DATA FLOW DIAGRAM

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system. A data flow diagram can also be used for the visualization of data processing (structured design). Dataflow diagrams can be used to provide the end user with a physical idea of where the data they input, ultimately has an effect upon the structure of the whole system from order to dispatch to restock how any system is developed can be determined through a dataflow diagram.

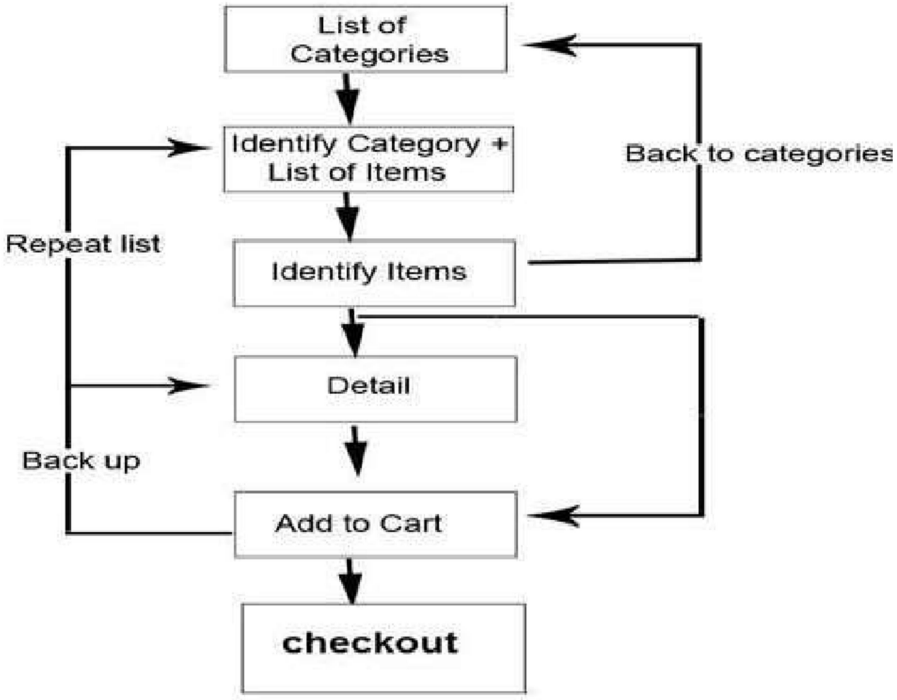
LEVEL ZERO DFD:



LEVEL 1 DFD:



LEVEL 2 DFD:


List of Categories
Identify Category + List of Items
Identify Items
Detail
Add to Cart
checkout
Back to categories
Repeat list
Back up

PROJECT FILES

base.html

```
<!doctype html>
{% load static %}
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <!-- Bootstrap CSS -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta1/dist/css/bootstrap.min.css" rel="stylesheet"
integrity="sha384-giJF6kkoqNQ00vy+HMDP7azOuL0xtbfIcaT9wjKHr8RbDVddVHyTfAAsrekwKmP1"
crossorigin="anonymous">

    <!--Owl Carousel CSS-->
    <link rel="stylesheet" href="{% static 'app/css/owl.carousel.min.css' %}">

    <!--FontAwesome CSS-->
    <link rel="stylesheet" href="{% static 'app/css/all.min.css' %}">

    <!--Custom CSS-->
    <link rel="stylesheet" href="{% static 'app/css/style.css' %}">


    <title>Achats | {% block title %} {% endblock title %} </title>
  </head>
  <body>
    <nav class="navbar navbar-expand-lg navbar-dark bg-primary">
     <div class="container">
       <a class="navbar-brand" href="/">Achats</a>
       <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-
label="Toggle navigation">
         <span class="navbar-toggler-icon"></span>
       </button>
       <div class="collapse navbar-collapse" id="navbarSupportedContent">
         <ul class="navbar-nav me-auto mb-2 mb-lg-0">
           <li class="nav-item">
             <a class="nav-link active" aria-current="page" href="/">Home</a>
           </li>
           <li class="nav-item dropdown">
             <a class="nav-link dropdown-toggle text-white" href="#" id="electronicsDropdown" role="button"
data-bs-toggle="dropdown" aria-expanded="false">
               Electronics
             </a>
             <ul class="dropdown-menu" aria-labelledby="electronicsDropdown">
               <li><a class="dropdown-item" href="{% url 'mobile' %}">Mobile</a></li>
               <li><a class="dropdown-item" href="#">Laptop</a></li>
             </ul>
           </li>
           <li class="nav-item dropdown">
```

```
          <a class="nav-link dropdown-toggle text-white" href="#" id="fashionDropdown" role="button" data-
bs-toggle="dropdown" aria-expanded="false">
            Fashion
          </a>
          <ul class="dropdown-menu" aria-labelledby="fashionDropdown">
            <li><a class="dropdown-item" href="#">Top Wear</a></li>
            <li><a class="dropdown-item" href="#">Bottom Wear</a></li>
          </ul>
        </li>
      </ul>
      <form class="d-flex">
        <input class="form-control me-2" type="search" placeholder="Search" aria-label="Search">
      </form>
      <div>
       <ul class="navbar-nav me-auto mb-2 mb-lg-0">
       {% if request.user.is_authenticated %}
        <li class="nav-item dropdown mx-2">
          <a class="nav-link dropdown-toggle text-white" href="#" id="profileDropdown" role="button" data-
bs-toggle="dropdown" aria-expanded="false">
            {{request.user.username|capfirst}}
          </a>
          <ul class="dropdown-menu" aria-labelledby="profileDropdown">
            <li><a class="dropdown-item" href="{% url 'profile' %}">Profile</a></li>
            <li><a class="dropdown-item" href="{% url 'orders' %}">Orders</a></li>
            <li><a class="dropdown-item" href="{% url 'passwordchange' %}">Change Password</a></li>
            <li><a class="dropdown-item" href="{% url 'logout' %}">Logout</a></li>
          </ul>
        </li>
        <li class="nav-item mx-2">
          <a href="{% url 'showcart' %}" class="nav-link text-white"><span class="badge bg-
danger">{{totalitem}}</span> Cart </a>
        </li>
        {% else %}
        <li class="nav-item mx-2">
         <a href="{% url 'login' %}" class="nav-link text-white">Login</a>
        </li>
        <li class="nav-item mx-2">
         <a href="{% url 'customerregistration' %}" class="nav-link text-white">Registration</a>
        </li>
        {% endif %}
      </ul>
     </div>
    </div>
   </div>
  </nav>
  {% block banner_slider %} {% endblock banner_slider %}
  {% block livesale %} {% endblock livesale %}
  {% block main-content %} {% endblock main-content %}
  {% block payment-gateway %} {% endblock payment-gateway %}

  <!-- Start Footer -->
  <footer class="container-fluid bg-dark text-center p-2 mt-5">
    <small class="text-white">Copyright &copy; 2022 || Designed By GBU Students || </small>
    <img src="{% static 'app/images/payment.png' %}" alt="" srcset="" class="img-fluid" height="2px">
  </footer> <!-- End Footer -->
```

```
    <!-- Jquery -->
    <script src="https://code.jquery.com/jquery-3.5.1.min.js" integrity="sha256-
9/aliU8dGd2tb6OSsuzixeV4y/faTqgFtohetphbbj0=" crossorigin="anonymous"></script>
    <!-- Bootstrap Bundle with Popper -->
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta1/dist/js/bootstrap.bundle.min.js"
integrity="sha384-ygbV9kiqUc6oa4msXn9868pTtWMgiQaeYH7/t7LECLbyPA2x65Kgf80OJFdroafW"
crossorigin="anonymous"></script>

    <script src="{% static 'app/js/owl.carousel.min.js' %}"></script>
    <script src="{% static 'app/js/all.min.js' %}"></script>
    <script src="{% static 'app/js/myscript.js' %}"></script>
  </body>
</html>
```

home.html

```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Home{% endblock title %}

{% block banner_slider %}
<!--Banner Slider-->
 <div id="carouselExampleControls" class="carousel slide" data-bs-ride="carousel">
  <div class="carousel-inner">
    <div class="carousel-item active">
      <img src="{% static 'app/images/banner/b1.png' %}" class="d-block w-100" alt="..." height=500 >
    </div>
    <div class="carousel-item">
      <img src="{% static 'app/images/banner/b2.avif' %}" class="d-block w-100" alt="..." height=500>
    </div>
    <div class="carousel-item">
      <img src="{% static 'app/images/banner/b3.avif' %}" class="d-block w-100" alt="..." height=500>
    </div>
    <div class="carousel-item">
      <img src="{% static 'app/images/banner/b4.avif' %}" class="d-block w-100" alt="..." height=500>
    </div>
    <div class="carousel-item">
      <img src="{% static 'app/images/banner/b5.avif' %}" class="d-block w-100" alt="..." height=500>
    </div>
    <div class="carousel-item">
      <img src="{% static 'app/images/banner/b6.jpg' %}" class="d-block w-100" alt="..." height=500>
    </div>
    <div class="carousel-item">
      <img src="{% static 'app/images/banner/b7.avif' %}" class="d-block w-100" alt="..." height=500>
    </div>
  </div>
  <a class="carousel-control-prev" href="#carouselExampleControls" role="button" data-bs-slide="prev">
    <span class="carousel-control-prev-icon" aria-hidden="true"></span>
    <span class="visually-hidden">Previous</span>
  </a>
  <a class="carousel-control-next" href="#carouselExampleControls" role="button" data-bs-slide="next">
    <span class="carousel-control-next-icon" aria-hidden="true"></span>
    <span class="visually-hidden">Next</span>
  </a>
</div>
<!-- End Banner Slider -->
{% endblock banner_slider %}

{% block livesale %}
<!-- Live Sale Section -->
<div class="container-fluid">
  <div class="row bg-dark text-center p-5 text-white border-bottom shadow">
    <h1>ENJOY</h1>
    <span>“Recreational shopping is the shortest distance between two points: you and broke.”</span>
    <small class="fw-lighter">– Victoria Moran.</small>
  </div>
```

```
</div>
<br><br>
<!-- End Live Sale Section -->
{% endblock livesale %}

{% block main-content %}

<!-- 1 Product Slider -->
<div class="mx-3">
  <h2>Mobile Phones</h2>
  <!-- Slider 2 -->
  <div class="owl-carousel" id="slider1">
   {% for b in mobiles %}
   <a href="{% url 'product-detail' b.id %}" class="btn"><div class="item"><img
src="{{b.product_image.url}}" alt="" height="200px"><span class="fw-bold">{{b.title}}</span><br><span
class="fs-5">Rs. {{b.discounted_price}}</span></div></a>
   {% endfor %}
  </div>
 </div>
 <!-- End 1 Product Slider -->

<!-- 2 Product Slider -->
<div class="m-3">
  <h2>Top Wears</h2>
  <!-- Slider 1 -->
  <div class="owl-carousel" id="slider2">
   {% for b in topwears %}
   <a href="{% url 'product-detail' b.id %}" class="btn"><div class="item"><img
src="{{b.product_image.url}}" alt="" height="200px"><span class="fw-bold">{{b.title}}</span><br><span
class="fs-5">Rs. {{b.discounted_price}}</span></div></a>
   {% endfor %}
  </div>
 </div>
 <!-- End 2 Product Slider -->

<!-- 3 Product Slider -->
<div class="m-3">
 <h2>Bottom Wears</h2>
 <!-- Slider 1 -->
 <div class="owl-carousel" id="slider3">
  {% for b in bottomwears %}
  <a href="{% url 'product-detail' b.id %}" class="btn"><div class="item"><img
src="{{b.product_image.url}}" alt="" height="200px"><span class="fw-bold">{{b.title}}</span><br><span
class="fs-5">Rs. {{b.discounted_price}}</span></div></a>
  {% endfor %}
 </div>
</div>
<!-- End 3 Product Slider -->

<!-- Payment Info Section -->
<div class="container my-5">
 <div class="row">
  <div class="col-sm-3">
   <div class="card mb-3">
    <div class="card-body">
```

```
    <img src="{% static 'app/images/payavail/cc.jpg' %}" alt="" srcset="" class="img-fluid">
    </div>
   </div>
  </div>
  <div class="col-sm-3">
   <div class="card mb-3">
    <div class="card-body">
     <img src="{% static 'app/images/payavail/upi.jpg' %}" alt="" srcset="" class="img-fluid">
    </div>
   </div>
  </div>
  <div class="col-sm-3">
   <div class="card mb-3">
    <div class="card-body">
     <img src="{% static 'app/images/payavail/nb.jpg' %}" alt="" srcset="" class="img-fluid">
    </div>
   </div>
  </div>
  <div class="col-sm-3">
   <div class="card mb-3">
    <div class="card-body">
     <img src="{% static 'app/images/payavail/bj.jpg' %}" alt="" srcset="" class="img-fluid">
    </div>
   </div>
  </div>
 </div>
</div>
<!-- End Payment Info Section -->

<!-- 4 Product Slider -->
<div class="mx-3">
 <h2>Laptops</h2>
 <!-- Slider 2 -->
 <div class="owl-carousel" id="slider4">
  {% for b in laptops %}
  <a href="{% url 'product-detail' b.id %}" class="btn"><div class="item"><img
src="{{b.product_image.url}}" alt="" height="200px"><span class="fw-bold">{{b.title}}</span><br><span
class="fs-5">Rs. {{b.discounted_price}}</span></div></a>
  {% endfor %}
 </div>
</div>
<!-- End 4 Product Slider -->
{% endblock main-content %}
```

login.html

```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Login{% endblock title %}
{% block main-content %}
<div class="container">
 <div class="row my-3">
  <div class="col-sm-6 offset-sm-3">
   <h3>Login</h3>
   <hr>
   <form action="" method="post" novalidate class="shadow p-5">
     {% csrf_token %}
     {% for fm in form %}
      <div class="form-group">
      {{fm.label_tag}} {{fm}} <small class="text-danger">{{fm.errors|striptags}}</small> <br>
      </div>
     {% endfor %}
     <small><a href="{% url 'password_reset' %}">Forgot Password ?</a></small> <br>
     <input type="submit" class="btn btn-primary mt-4" value="Login">
     <br>
     <div class="text-center text-primary fw-bold"><small>New to Achats ? <a href="{% url
'customerregistration' %}" class="text-danger">Create an Account</a> </small></div>

     {% if form.non_field_errors %}
      {% for error in form.non_field_errors %}
       <p class="alert alert-danger my-3">{{error}}</p>
      {% endfor %}
     {% endif %}
    </form>
  </div>
 </div>
</div>
{% endblock main-content %}
```

productdetail.html

```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Product Detail{% endblock title %}
{% block main-content %}
<div class="container my-5">
 <div class="row">
  <div class="col-sm-6 text-center align-self-center">
   <img src="{{product.product_image.url}}" alt="" class="img-fluid img-thumbnail">
  </div>
  <div class="col-sm-5 offset-sm-1">
   <h2>{{product.title}}</h2>
   <hr>
   <p>{{product.description}}</p> <br>
   <h4>Rs. {{product.discounted_price}} <small class="fw-light text-decoration-line-
through">{{product.selling_price}}</small></h4> <br>

   {% if user.is_authenticated  %}
    {% if item_already_in_cart  %}
     <a href="{% url 'showcart' %}" class="btn btn-warning shadow px-5 py-2 ms-4">Go to Cart</a>
    {% else %}
     <form action="/add-to-cart" class="d-inline">
      <input type="hidden" name="prod_id" value="{{product.id}}" id="prod_id">
      <button type="submit" class="btn btn-primary shadow px-5 py-2">Add to Cart</button>
     </form>
    {% endif %}
   {% endif %}

   {% if not request.user.is_authenticated %}
    <form action="/add-to-cart" class="d-inline">
     <input type="hidden" name="prod_id" value="{{product.id}}" id="prod_id">
     <button type="submit" class="btn btn-primary shadow px-5 py-2">Add to Cart</button>
    </form>
   {% endif %}

   <a href="{% url 'checkout' %}" class="btn btn-danger shadow px-5 py-2 ms-4">Buy Now</a>
   <h5 class="mt-5">Available Offers</h5>
   <ul>
    <li>Bank Offer 5% Unlimited Cashback on Flipkart Axis Bank Credit</li>
    <li>Special Price Get extra ₹3000 off (price inclusive of discount)</li>
    <li>No cost EMI ₹1,667/month. Standard EMI also available</li>
    <li>Partner Offer ₹2000 Flipkart Gift Card on Every 1000th Transaction with a new Visa Debit/Credit
Card</li>
   </ul>
  </div>
 </div>
</div>
{% endblock main-content %}
```

addtocart.html

```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Cart{% endblock title %}
{% block main-content %}
<div class="container my-5">
 {% comment %} Below Code will be used by add_to_cart view {% endcomment %}
 {% if messages %}
 {% for message in messages %}
  <p {% if message.tags %} class="alert alert-{{message.tags}} mb-5" {% endif %}>{{message}}</p>
 {% endfor %}
 {% endif %}
 <div class="row">
  <h1 class="text-center mb-5">Shopping Cart</h1>
  <div class="col-sm-8">
  <div class="card">
   <div class="card-body">
    <h3>Cart Items</h3>
    {% for cart in carts %}
    <hr>
    <div class="row">
     <div class="col-sm-3 text-center align-self-center"><img src="{{cart.product.product_image.url}}" alt=""
srcset="" class="img-fluid img-thumbnail shadow-sm" height="150" width="150"> </div>
     <div class="col-sm-9">
      <div>
       <h5>{{cart.product.title}}</h5>
       <p class="mb-2 text-muted small">{{cart.product.description}}</p>
       <div class="my-3">
        <label for="quantity">Quantity:</label>
        <a class="minus-cart btn" pid="{{cart.product.id}}"><i class="fas fa-minus-square fa-lg"></i></a>
          <span id="quantity">{{cart.quantity}}</span>
        <a class="plus-cart btn" pid="{{cart.product.id}}"><i class="fas fa-plus-square fa-lg"></i></a>
       </div>
       <div class="d-flex justify-content-between">
        <a class="btn btn-sm btn-secondary mr-3 remove-cart" pid={{cart.product.id}}>Remove item </a>
        <p class="mb-0"><span><strong>Rs. {{cart.product.discounted_price}}</strong></span></p>
       </div>
      </div>
     </div>
    </div>
    {% endfor %}
   </div>
   </div>
  </div>

  <div class="col-sm-4">
   <div class="card">
    <div class="card-body">
     <h3>The Total Amount of</h3>
     <ul class="list-group">
      <li class="list-group-item d-flex justify-content-between align-items-center border-0 px-0 pb-
0">Amount<span>Rs. <span id="amount">{{amount|floatformat:2}}</span> </span></li>
```

```
      <li class="list-group-item d-flex justify-content-between align-items-center border-0 px-
0">Shipping<span>Rs. 70.00</span></li>
      <hr>
      <li class="list-group-item d-flex justify-content-between align-items-center border-0 px-0 mb-3">
       <div>
         <strong>Total</strong> <small>(including VAT)</small>
        </div>
        <span><strong>Rs. </strong><strong id="totalamount">
{{totalamount|floatformat:2}}</strong></span>
      </li>
     </ul>
     <div class="d-grid"><a href="{% url 'checkout' %}" class="btn btn-danger fw-bold">Place
Order</a></div>
    </div>
   </div>
  </div>
 </div>
</div>
<div class="container">
 <div class="row">
  <div class="col-sm-8">
   <div class="card">
    <div class="card-body">
     <h5 class="mb-4">We accept</h5>
     <img src="{% static 'app/images/payment.png' %}" alt="" class="img-fluid">
    </div>
  </div>
  </div>
 </div>
</div>
{% endblock main-content %}
```

orders.html

```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Orders{% endblock title %}
{% block main-content %}
<div class="container my-5">
 <div class="row">
 <h3>Welcome Customer</h3>
  <div class="col-sm-2 border-end">
   <ul class="list-unstyled">
    <li class="d-grid"><a href="{% url 'orders' %}" class="btn btn-primary">Orders</a></li>
   </ul>
  </div>
  <div class="col-sm-9 offset-sm-1">
   <div class="row">
    {% for op in order_placed %}
     <div class="col-sm-12">
      <div class="row shadow-sm mb-3">
       <div class="col-sm-2"><img src="{{op.product.product_image.url}}" alt="" srcset="" class="img-fluid"
height="150" width="150"></div>
       <div class="col-sm-7">
        <p>Product: {{op.product.title}}</p>
        <p>Quantity: {{op.quantity}}</p>
        <p>Price: {{op.total_cost}}</p>
       </div>
       <div class="col-sm-3 fw-bold">
        <p>Order Status: {{op.status}}</p>
         {% if op.status == 'Accepted' %}
         <div class="progress">
          <div class="progress-bar" role="progressbar" style="width: 20%" aria-valuenow="20" aria-
valuemin="0" aria-valuemax="100"></div>
         </div>
         {% endif %}

         {% if op.status == 'Packed' %}
         <div class="progress">
          <div class="progress-bar bg-info" role="progressbar" style="width: 40%" aria-valuenow="40" aria-
valuemin="0" aria-valuemax="100"></div>
         </div>
         {% endif %}

         {% if op.status == 'On The Way' %}
         <div class="progress">
          <div class="progress-bar bg-warning" role="progressbar" style="width: 70%" aria-valuenow="70" aria-
valuemin="0" aria-valuemax="100"></div>
         </div>
         {% endif %}

         {% if op.status == 'Delivered' %}
         <div class="progress">
          <div class="progress-bar bg-success" role="progressbar" style="width: 100%" aria-valuenow="100"
aria-valuemin="0" aria-valuemax="100"></div>
```

```
          </div>
          {% endif %}
         </div>
        </div>
       </div>
      </div>
     {% endfor %}
    </div>
   </div>
  </div>
 </div>
{% endblock main-content %}
```

checkout.html

```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Checkout{% endblock title %}
{% block main-content %}
<div class="container">
 <div class="row mt-5">
  <div class="col-sm-6">
    <h4>Order Summary</h4>
    <hr>
    {% for item in cart_items %}
     <div class="card mb-2">
       <div class="card-body">
         <h5>Product: {{item.product.title}}</h5>
         <p>Quantity: {{item.quantity}}</p>
         <p class="fw-bold">Price: {{item.total_cost}}</p>
       </div>
     </div>
    {% endfor %}
    <small>Term and Condition: Lorem ipsum dolor sit amet consectetur adipisicing elit. Mollitia, ullam saepe! Iure optio repellat dolor velit, minus rem. Facilis cumque neque numquam laboriosam, accusantium adipisci nisi nihil in et quis?</small>
  </div>
  <div class="col-sm-4 offset-sm-1">
    <h4>Select Shipping Address</h4>
    <hr>
    <form action="/paymentdone">
    {% for ad in add %}
     <div class="card">
       <div class="card-body">
       <h5>{{ad.name}}</h5>
       <p>{{ad.locality}}, {{ad.city}}, {{ad.state}}- {{ad.zipcode}}</p>
       </div>
     </div>
       <div class="form-check mt-2 mb-5">
         <input class="form-check-input" type="radio" name="custid" id="custadd{{forloop.counter}}" value="{{ad.id}}">
         <label class="form-check-label fw-bold" for="custadd{{forloop.counter}}">
           Address: {{ forloop.counter }} </label>
       </div>
     {% endfor %}
       <div class="text-end">
         <!-- <button type="submit" class="btn btn-warning mt-3 px-5 fw-bold">Continue</button> -->
         <div id="paypal-button-container"></div>
       </div>
     </form>
    </div>
  </div>
</div>
{% endblock main-content %}

{% block payment-gateway %}
```

```
<script src="https://www.paypal.com/sdk/js?client-id=test&currency=USD"></script>

   <script>
      // Render the PayPal button into #paypal-button-container
      paypal.Buttons({

         // Set up the transaction
         createOrder: function(data, actions) {
            return actions.order.create({
               purchase_units: [{
                  amount: {
                     value: '88.44'
                  }
               }]
            });
         },

         // Finalize the transaction
         onApprove: function(data, actions) {
            return actions.order.capture().then(function(orderData) {
               // Successful capture! For demo purposes:
               console.log('Capture result', orderData, JSON.stringify(orderData, null, 2));
               var transaction = orderData.purchase_units[0].payments.captures[0];
               alert('Transaction '+ transaction.status + ': ' + transaction.id + '\n\nSee console for all available
details');

               // Replace the above to show a success message within this page, e.g.
               // const element = document.getElementById('paypal-button-container');
               // element.innerHTML = '';
               // element.innerHTML = '<h3>Thank you for your payment!</h3>';
               // Or go to another URL:  actions.redirect('thank_you.html');
            });
         }


      }).render('#paypal-button-container');
   </script>
{% endblock payment-gateway %}
```

myscript.js

```
$('#slider1, #slider2, #slider3').owlCarousel({
    loop: true,
    margin: 20,
    responsiveClass: true,
    responsive: {
        0: {
            items: 1,
            nav: false,
            autoplay: true,
        },
        600: {
            items: 3,
            nav: true,
            autoplay: true,
        },
        1000: {
            items: 5,
            nav: true,
            loop: true,
            autoplay: true,
        }
    }
})

$('.plus-cart').click(function () {
    var id = $(this).attr("pid").toString();
    var eml = this.parentNode.children[2];
    console.log(id);
    $.ajax(
        {
            type: "GET",
            url: "/pluscart",
            data: {
                prod_id: id
            },
            success: function (data) {
                console.log(data);
                eml.innerText = data.quantity;
                document.getElementById("amount").innerText = data.amount.toFixed(2);
                document.getElementById("totalamount").innerText = data.totalamount.toFixed(2);
            }
        })
});

$('.minus-cart').click(function () {
    var id = $(this).attr("pid").toString();
    var eml = this.parentNode.children[2];
    $.ajax(
        {
            type: "GET",
            url: "/minuscart",
```

```
        data: {
            prod_id: id
        },
        success: function (data) {
            console.log(data)
            eml.innerText = data.quantity;
            document.getElementById("amount").innerText = data.amount.toFixed(2);
            document.getElementById("totalamount").innerText = data.totalamount.toFixed(2);
        }
    })
});

$('.remove-cart').click(function () {
    var id = $(this).attr("pid").toString();
    var elm = this;
    $.ajax(
        {
        type: "GET",
        url: "/removecart",
        data: {
            prod_id: id
        },
        success: function (data) {
            console.log(data)
            document.getElementById("amount").innerText = data.amount;
            document.getElementById("totalamount").innerText = data.totalamount;
            elm.parentNode.parentNode.parentNode.parentNode.remove()
        }
    })
});
```

views.py

```
from django.shortcuts import render, redirect, HttpResponse
from .models import Customer, Product, Cart, OrderPlaced
from .forms import CustomerRegistrationForm, CustomerProfileForm
from django.views import View
from django.contrib import messages
from django.http import JsonResponse
from django.db.models import Q
from django.contrib.auth.decorators import login_required
from django.utils.decorators import method_decorator

class ProductView(View):
        def get(self, request):
                totalitem = 0
                topwears = Product.objects.filter(category='TW')
                bottomwears = Product.objects.filter(category='BW')
                mobiles = Product.objects.filter(category='M')
                if request.user.is_authenticated:
                        totalitem = len(Cart.objects.filter(user=request.user))
                return render(request, 'app/home.html', {'topwears':topwears, 'bottomwears':bottomwears,
'mobiles':mobiles, 'totalitem':totalitem})

class ProductDetailView(View):
        def get(self, request, pk):
                totalitem = 0
                product = Product.objects.get(pk=pk)
                print(product.id)
                item_already_in_cart=False
                if request.user.is_authenticated:
                        totalitem = len(Cart.objects.filter(user=request.user))
                        item_already_in_cart = Cart.objects.filter(Q(product=product.id) &
Q(user=request.user)).exists()
                return render(request, 'app/productdetail.html', {'product':product,
'item_already_in_cart':item_already_in_cart, 'totalitem':totalitem})

@login_required()
def add_to_cart(request):
        user = request.user
        item_already_in_cart1 = False
        product = request.GET.get('prod_id')
        item_already_in_cart1 = Cart.objects.filter(Q(product=product) & Q(user=request.user)).exists()
        if item_already_in_cart1 == False:
                product_title = Product.objects.get(id=product)
                Cart(user=user, product=product_title).save()
                messages.success(request, 'Product Added to Cart Successfully !!' )
                return redirect('/cart')
        else:
                return redirect('/cart')
  # Below Code is used to return to same page
  # return redirect(request.META['HTTP_REFERER'])

@login_required
```

```
def show_cart(request):
	totalitem = 0
	if request.user.is_authenticated:
		totalitem = len(Cart.objects.filter(user=request.user))
		user = request.user
		cart = Cart.objects.filter(user=user)
		amount = 0.0
		shipping_amount = 70.0
		totalamount=0.0
		cart_product = [p for p in Cart.objects.all() if p.user == request.user]
		print(cart_product)
		if cart_product:
			for p in cart_product:
				tempamount = (p.quantity * p.product.discounted_price)
				amount += tempamount
				totalamount = amount+shipping_amount
			return render(request, 'app/addtocart.html', {'carts':cart, 'amount':amount,
'totalamount':totalamount, 'totalitem':totalitem})
		else:
			return render(request, 'app/emptycart.html', {'totalitem':totalitem})
	else:
		return render(request, 'app/emptycart.html', {'totalitem':totalitem})

def plus_cart(request):
	if request.method == 'GET':
		prod_id = request.GET['prod_id']
		c = Cart.objects.get(Q(product=prod_id) & Q(user=request.user))
		c.quantity+=1
		c.save()
		amount = 0.0
		shipping_amount= 70.0
		cart_product = [p for p in Cart.objects.all() if p.user == request.user]
		for p in cart_product:
			tempamount = (p.quantity * p.product.discounted_price)
			# print("Quantity", p.quantity)
			# print("Selling Price", p.product.discounted_price)
			# print("Before", amount)
			amount += tempamount
			# print("After", amount)
		# print("Total", amount)
		data = {
			'quantity':c.quantity,
			'amount':amount,
			'totalamount':amount+shipping_amount
		}
		return JsonResponse(data)
	else:
		return HttpResponse("")

def minus_cart(request):
	if request.method == 'GET':
		prod_id = request.GET['prod_id']
		c = Cart.objects.get(Q(product=prod_id) & Q(user=request.user))
		c.quantity-=1
		c.save()
```

```
                amount = 0.0
                shipping_amount= 70.0
                cart_product = [p for p in Cart.objects.all() if p.user == request.user]
                for p in cart_product:
                        tempamount = (p.quantity * p.product.discounted_price)
                        # print("Quantity", p.quantity)
                        # print("Selling Price", p.product.discounted_price)
                        # print("Before", amount)
                        amount += tempamount
                        # print("After", amount)
                # print("Total", amount)
                data = {
                        'quantity':c.quantity,
                        'amount':amount,
                        'totalamount':amount+shipping_amount
                }
                return JsonResponse(data)
        else:
                return HttpResponse("")

@login_required
def checkout(request):
        user = request.user
        add = Customer.objects.filter(user=user)
        cart_items = Cart.objects.filter(user=request.user)
        return render(request, 'app/checkout.html', {'add':add, 'cart_items':cart_items})

@login_required
def payment_done(request):
        custid = request.GET.get('custid')
        print("Customer ID", custid)
        user = request.user
        cartid = Cart.objects.filter(user = user)
        customer = Customer.objects.get(id=custid)
        print(customer)
        for cid in cartid:
                OrderPlaced(user=user, customer=customer, product=cid.product,
quantity=cid.quantity).save()
                print("Order Saved")
                cid.delete()
                print("Cart Item Deleted")
        return redirect("orders")

def remove_cart(request):
        if request.method == 'GET':
                prod_id = request.GET['prod_id']
                c = Cart.objects.get(Q(product=prod_id) & Q(user=request.user))
                c.delete()
                amount = 0.0
                shipping_amount= 70.0
                cart_product = [p for p in Cart.objects.all() if p.user == request.user]
                for p in cart_product:
                        tempamount = (p.quantity * p.product.discounted_price)
                        # print("Quantity", p.quantity)
                        # print("Selling Price", p.product.discounted_price)
```

```
                        # print("Before", amount)
                        amount += tempamount
                        # print("After", amount)
                # print("Total", amount)
                data = {
                        'amount':amount,
                        'totalamount':amount+shipping_amount
                }
                return JsonResponse(data)
        else:
                return HttpResponse("")

@login_required
def address(request):
        totalitem = 0
        if request.user.is_authenticated:
                totalitem = len(Cart.objects.filter(user=request.user))
        add = Customer.objects.filter(user=request.user)
        return render(request, 'app/address.html', {'add':add, 'active':'btn-primary', 'totalitem':totalitem})

@login_required
def orders(request):
        op = OrderPlaced.objects.filter(user=request.user)
        return render(request, 'app/orders.html', {'order_placed':op})

def mobile(request, data=None):
        totalitem = 0
        if request.user.is_authenticated:
                totalitem = len(Cart.objects.filter(user=request.user))
        if data==None :
                        mobiles = Product.objects.filter(category='M')
        elif data == 'Redmi' or data == 'Samsung':
                        mobiles = Product.objects.filter(category='M').filter(brand=data)
        elif data == 'below':
                        mobiles = Product.objects.filter(category='M').filter(discounted_price__lt=10000)
        elif data == 'above':
                        mobiles = Product.objects.filter(category='M').filter(discounted_price__gt=10000)
        return render(request, 'app/mobile.html', {'mobiles':mobiles, 'totalitem':totalitem})


class CustomerRegistrationView(View):
 def get(self, request):
  form = CustomerRegistrationForm()
  return render(request, 'app/customerregistration.html', {'form':form})

 def post(self, request):
  form = CustomerRegistrationForm(request.POST)
  if form.is_valid():
   messages.success(request, 'Congratulations!! Registered Successfully.')
   form.save()
  return render(request, 'app/customerregistration.html', {'form':form})

@method_decorator(login_required, name='dispatch')
class ProfileView(View):
        def get(self, request):
```

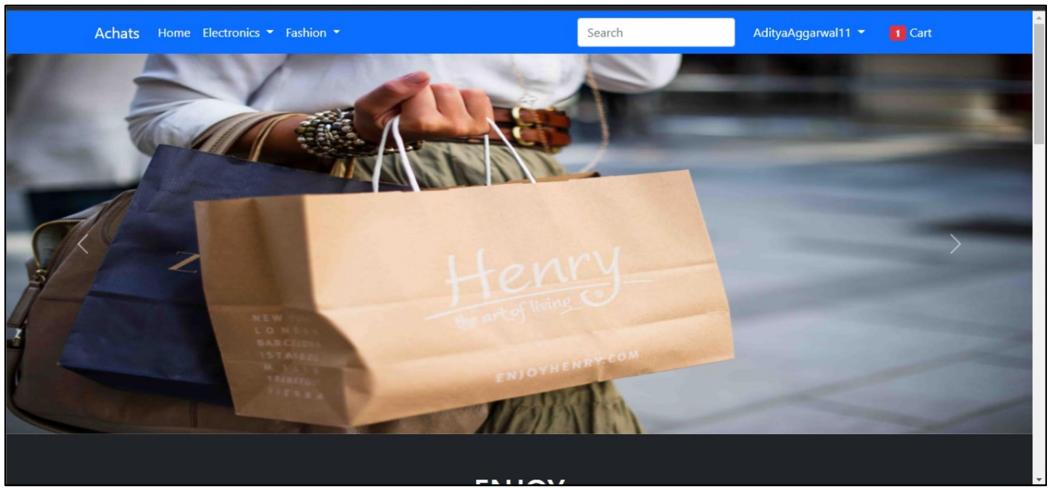
```
                totalitem = 0
                if request.user.is_authenticated:
                        totalitem = len(Cart.objects.filter(user=request.user))
                form = CustomerProfileForm()
                return render(request, 'app/profile.html', {'form':form, 'active':'btn-primary',
'totalitem':totalitem})

        def post(self, request):
                totalitem = 0
                if request.user.is_authenticated:
                        totalitem = len(Cart.objects.filter(user=request.user))
                form = CustomerProfileForm(request.POST)
                if form.is_valid():
                        usr = request.user
                        name  = form.cleaned_data['name']
                        locality = form.cleaned_data['locality']
                        city = form.cleaned_data['city']
                        state = form.cleaned_data['state']
                        zipcode = form.cleaned_data['zipcode']
                        reg = Customer(user=usr, name=name, locality=locality, city=city, state=state,
zipcode=zipcode)
                        reg.save()
                        messages.success(request, 'Congratulations!! Profile Updated Successfully.')
                return render(request, 'app/profile.html', {'form':form, 'active':'btn-primary',
'totalitem':totalitem})
```

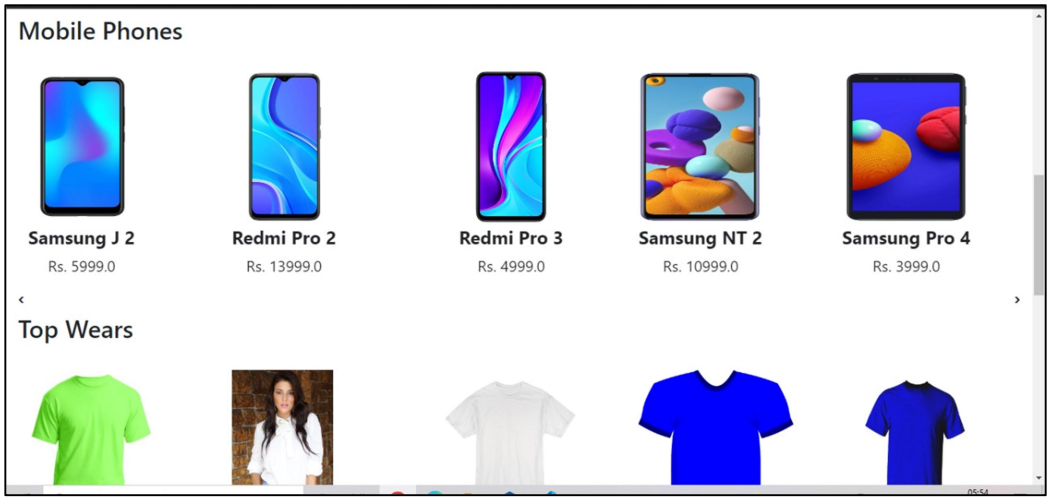
urls.py

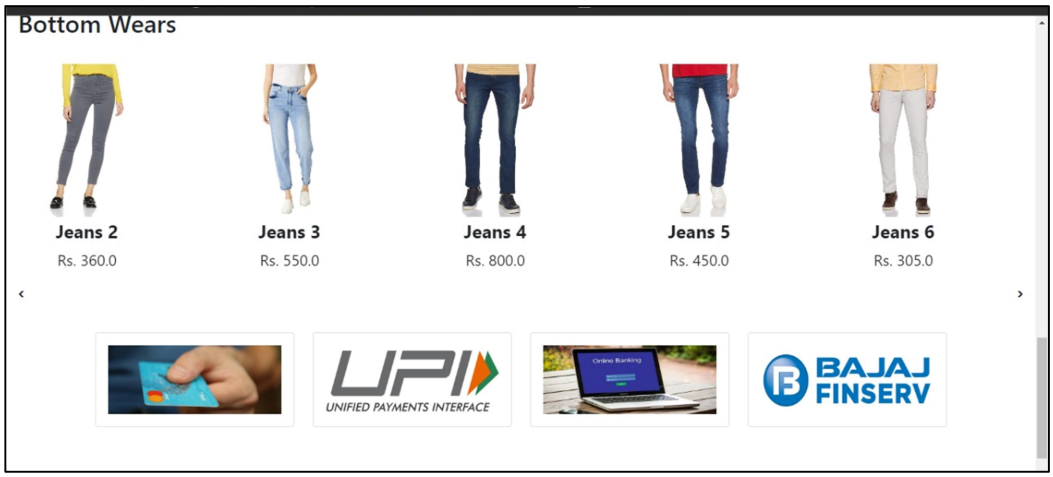
```
from django.urls import path
from app import views
from django.conf import settings
from django.conf.urls.static import static
from django.contrib.auth import views as auth_views
from .forms import LoginForm, MyPasswordChangeForm, MyPasswordResetForm, MySetPasswordForm
urlpatterns = [
    # path('', views.home),
    path('', views.ProductView.as_view(), name="home"),
    # path('product-detail', views.product_detail, name='product-detail'),
    path('product-detail/<int:pk>/', views.ProductDetailView.as_view(), name='product-detail'),
    path('add-to-cart/', views.add_to_cart, name='add-to-cart'),
    path('cart/', views.show_cart, name='showcart'),
    path('pluscart/', views.plus_cart),
    path('minuscart/', views.minus_cart),
    path('removecart/', views.remove_cart),
    path('checkout/', views.checkout, name='checkout'),
    path('address/', views.address, name='address'),
    path('orders/', views.orders, name='orders'),
    path('paymentdone/', views.payment_done, name='paymentdone'),
    path('mobile/', views.mobile, name='mobile'),
    path('mobile/<slug:data>', views.mobile, name='mobiledata'),
    path('accounts/login/', auth_views.LoginView.as_view(template_name='app/login.html',
authentication_form=LoginForm), name='login'),
    # path('profile/', views.profile, name='profile'),
    path('profile/', views.ProfileView.as_view(), name='profile'),
    path('logout/', auth_views.LogoutView.as_view(next_page='login'), name='logout'),
    path('passwordchange/',
auth_views.PasswordChangeView.as_view(template_name='app/passwordchange.html',
form_class=MyPasswordChangeForm, success_url='/passwordchangedone/'), name='passwordchange'),
    path('passwordchangedone/',
auth_views.PasswordChangeDoneView.as_view(template_name='app/passwordchangedone.html'),
name='passwordchangedone'),
    path("password-reset/",
auth_views.PasswordResetView.as_view(template_name='app/password_reset.html',
form_class=MyPasswordResetForm), name="password_reset"),
    path("password-reset/done/",
auth_views.PasswordResetDoneView.as_view(template_name='app/password_reset_done.html'),
name="password_reset_done"),
    path("password-reset-confirm/<uidb64>/<token>/",
auth_views.PasswordResetConfirmView.as_view(template_name='app/password_reset_confirm.html',
form_class=MySetPasswordForm), name="password_reset_confirm"),
    path("password-reset-complete/",
auth_views.PasswordResetCompleteView.as_view(template_name='app/password_reset_complete.html'),
name="password_reset_complete"),
    path('registration/', views.CustomerRegistrationView.as_view(), name='customerregistration')
] + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

SCREENSHORTS

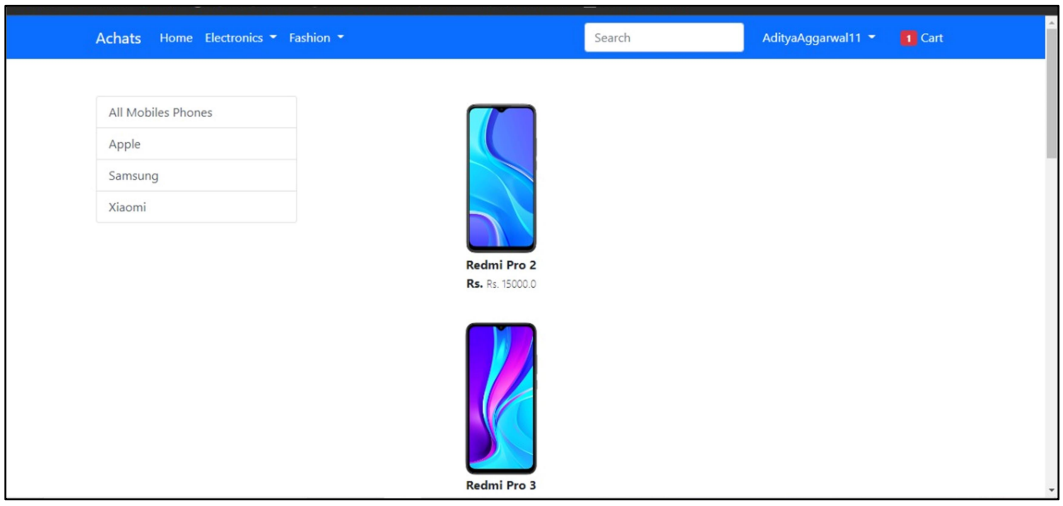
- **Homepage**



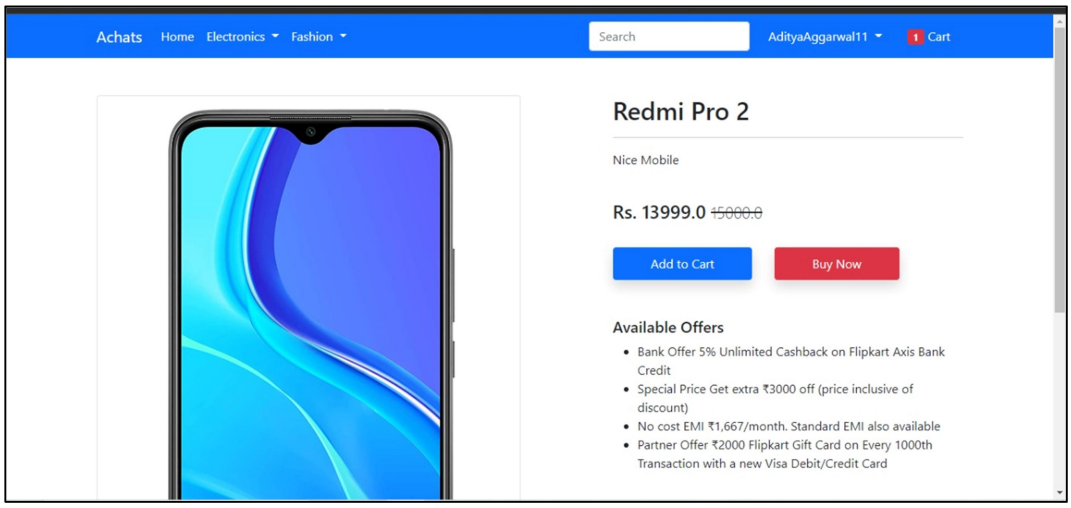




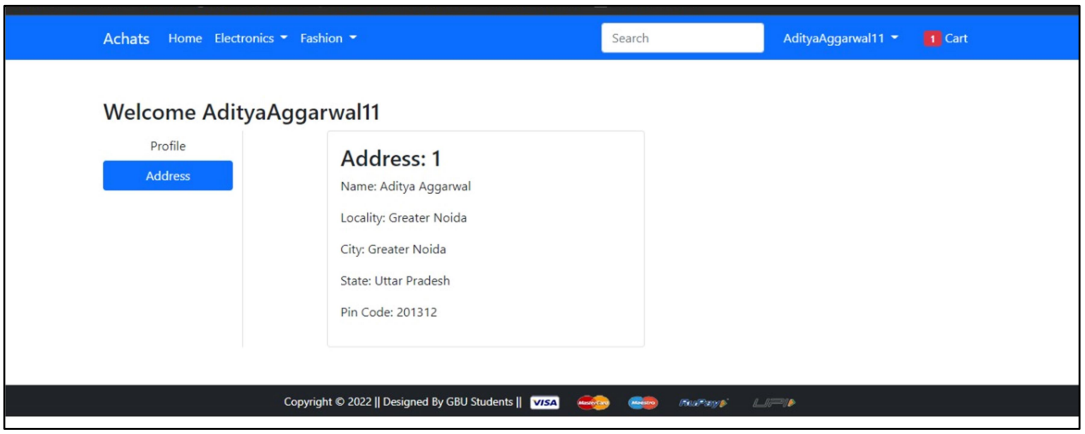
- **Category**



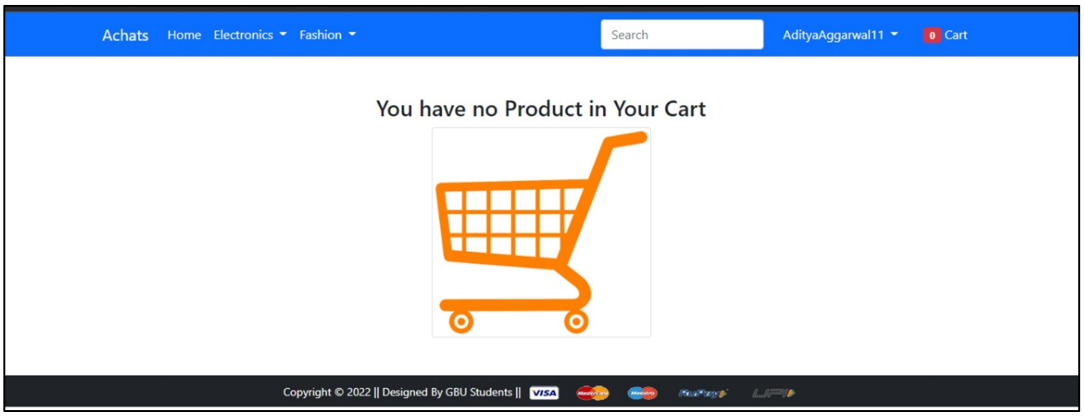
- **Item Page**



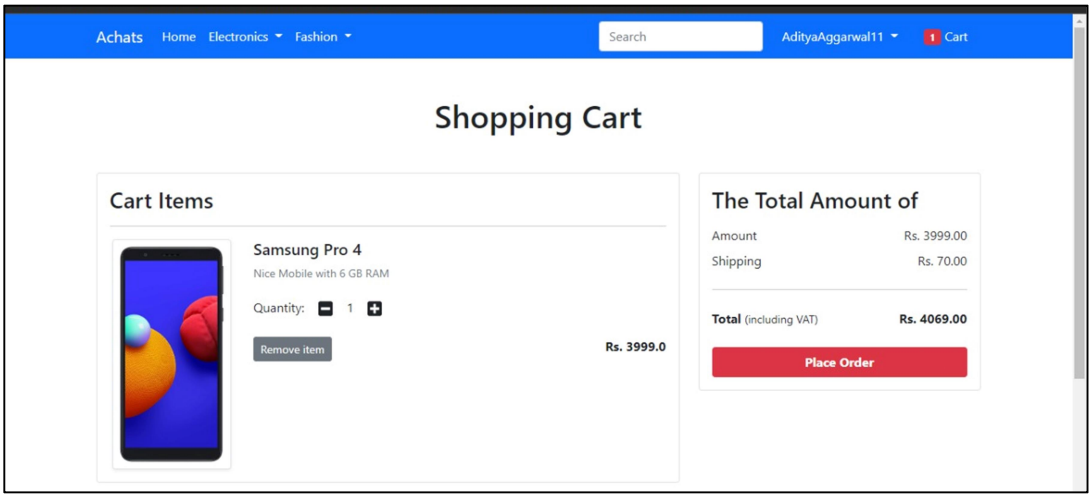
- **Address/Shipping details**



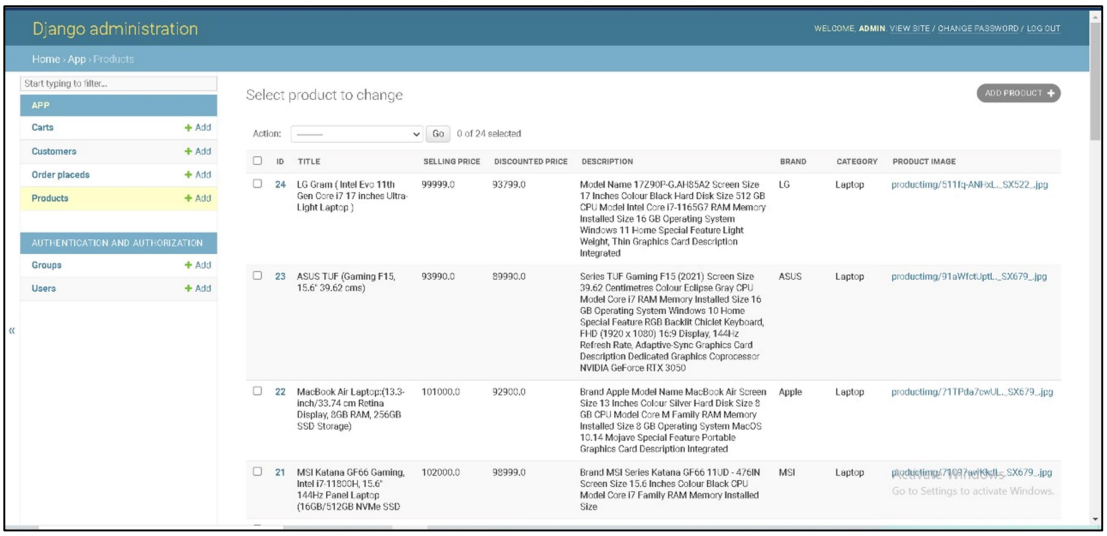
- **Cart**
  - **Empty**



  - **Occupied**



- **Database**


Django administration
WELCOME, ADMIN. VIEW SITE / CHANGE PASSWORD / LOG OUT
Home › App › Products
Start typing to filter...
APP
Carts + Add
Customers + Add
Order placeds + Add
Products + Add
AUTHENTICATION AND AUTHORIZATION
Groups + Add
Users + Add
Select product to change
ADD PRODUCT +
Action: ——— Go 0 of 24 selected
ID TITLE SELLING PRICE DISCOUNTED PRICE DESCRIPTION BRAND CATEGORY PRODUCT IMAGE
24 LG Gram (Intel Evo 11th Gen Core i7 17 inches Ultra-Light Laptop) 99999.0 93799.0 Model Name 17Z90P-G.AH85A2 Screen Size 17 Inches Colour Black Hard Disk Size 512 GB CPU Model Intel Core i7-1165G7 RAM Memory Installed Size 16 GB Operating System Windows 11 Home Special Feature Light Weight, Thin Graphics Card Description Integrated LG Laptop productimg/511fq-ANHxL._SX522_.jpg
23 ASUS TUF (Gaming F15, 15.6" 39.62 cms) 93990.0 89990.0 Series TUF Gaming F15 (2021) Screen Size 39.62 Centimetres Colour Eclipse Gray CPU Model Core i7 RAM Memory Installed Size 16 GB Operating System Windows 10 Home Special Feature RGB Backlit Chiclet Keyboard, FHD (1920 x 1080) 16:9 Display, 144Hz Refresh Rate, Adaptive-Sync Graphics Card Description Dedicated Graphics Coprocessor NVIDIA GeForce RTX 3050 ASUS Laptop productimg/91aWfctUptL._SX679_.jpg
22 MacBook Air Laptop(13.3-inch/33.74 cm Retina Display, 8GB RAM, 256GB SSD Storage) 101000.0 92900.0 Brand Apple Model Name MacBook Air Screen Size 13 Inches Colour Silver Hard Disk Size 8 GB CPU Model Core M Family RAM Memory Installed Size 8 GB Operating System MacOS 10.14 Mojave Special Feature Portable Graphics Card Description Integrated Apple Laptop productimg/71TPda7cwUL._SX679_.jpg
21 MSI Katana GF66 Gaming, Intel i7-11800H, 15.6" 144Hz Panel Laptop (16GB/512GB NVMe SSD 102000.0 98999.0 Brand MSI Series Katana GF66 11UD - 476IN Screen Size 15.6 Inches Colour Black CPU Model Core i7 Family RAM Memory Installed Size MSI Laptop productimg/71i87wlKkdL._SX679_.jpg
Activate Windows
Go to Settings to activate Windows.

FEATURES OF PROJECT

1- **Avoid crowds or to save time.**

2- **Affordable**

3- **Easy**

4- **Provide many Options for every services**

5- **Ease of Use:-** Simplicity is the mark of a good design You can implement this concept by: Creating shopping categories Adding filters Incorporating comparison capabilities

6- **Site Speed:-** Site speed is an indicator of an easy to use website and draws a lot of parallels with the ability to convert visitors.

7- **Mobile Friendly:-** Another of the major method to increase user base is to make your site easier to use is by optimizing for mobile.

8- **User Reviews:-** The major pitfall of online shopping is the fact that you cannot physically touch or experience a product. However, the reviews provided by previous users can help guide other users.

9- **High Resolution Photo and/or Video**

10- **Security Features:-** E-commerce sites are prime targets for cyber criminals, who can steal your sensitive information as well as your customers.

11- **Other must-have security features include:-**

Two-factor authentication: Adds an extra layer of security by requiring user name/password and a system-generated code sent via email or text.
Firewall: A wall that blocks malicious traffic and permits authorized traffic

SYSTEM SECURITY

Security is the most important part of any system. It can be either the security of system program functionalities or underlying database. We have very cautious
process of authentication of user that no one could change its contents in unauthorized manner. Security and integrity of database are very important for any software system because databases are the backbone of the system. Security need to be implemented at every level of the system so that only authorized user can access the system for update and other significance process. Entering correct password while opening the system or we can say that entering the system is the process of authentication. If anyone is entering the password is wrong then he/she cannot access the system for any change purpose. The main purpose of the security is to save system from accidentally changes or loss of information or also getting wrong information. The system administrator is the person that can change the information or update the information. He can also grant the permission that who has to enter the system and what can he do. So security is the most important topic to be concerned.

FUTURE SCOPE OF THE PROJECT

"**Online Shopping System**" is a web-based project which is made for remote-shopping or servicing through Internet. As the technology is being advanced the way of life is changing accordingly. Now-a-day's we can place the order for anything from our home. There is no need to go the shop of the things we want. The order can be placed online through Internet. The payment, the confirmation of purchasing; we can do everything online. Now we can think that how the days have been changed with time. People had to stand in rows to wait there terms to buy a particular thing from a popular shop. But what is happening now-a-day's, we can get those things on our door-step in few hours. In future we will try to make this website which work so flexible and beneficial for the customer and also try to make smooth service .

Some future scopes of the project are:

1. All home services are provided at reasonable price.
2. Improve quality and provide better services.
3. Connecting small businesses to the customers.
4. Effective communication between both ends.
5. Provide all locals shops at your fingers.

REFERENCES

- https://www.urban-company.com
- https://en.wikipedia.org/wiki/Viola%E2%80%93Jones mework
- https://homeservice.com