Table of Contents

❖ ABSTRACT:

The objective of this project is to develop a website which would act as an online shop, where multiple products would be available and customers would be able to create their accounts and purchase their desired products. To accomplish this task; we would be using a python library named as Django as a backend and HTML, CSS etc. as frontend.

❖ METHODOLOGY:

• Django:

It is a high level python web framework. Django includes various helping task modules and libraries which can be used to handle common Web development tasks makes. This makes web development extremely rapid, secure, scalable and versatile. It is an open source framework with a wide and strong supportive community. Moreover, its detailed excellent documentation is extremely supportive.

Following classes, models and apps of Django framework have been used in this project:

1. models.Model:

A model is a Python class that subclasses django.db.models.Model and maps to a single database table. It is a single source of information about a specific data. It contains the essential fields and behaviors of the data stored. Each attribute of the model represents a database field. Database fields can be of many types. Following data field types have been used in this project:

- **IntegerField**: It can store values from -2147483648 to 2147483647.
- **CharField**: A string field, for small- to large-sized strings.
- **EmailField**: A CharField that checks that the value is a valid email address using Email Validator.
- **TextField**: It is an extension of CharField and can save large texts.
- **ImageField**: For saving images in the database. It Inherits all its attributes and methods from FileField, but also validates that the uploaded object is a valid image.

Apart of this, Django provides an automatically-generated database-access API in the admin app to view, access, modify and manipulate the database with a user-friendly interface.

2. UserCreationForm:

Django authentication framework provides a form named UserCreationForm (which inherits from ModelForm class). It has three fields namely username, password1 and password2 (for confirmation of password1).

3. TemplateView:

HTML interfaces in Django are handled by the Django views. This can be done in multiple forms i.e. through functions, classes, etc. Classes which are used to render html interfaces, they inherit TemplateView class. It is basically a sub-class of the View Class that renders a Django template. TemplateView is used to render a basic html interface to the front end of the website. TemplateView class is one of the simplest generic class-based views that help developers create a view for a specific template.

4. CrispyForms:

Crispy forms have been used to display different forms. It provides a full developed template for forms, ready to use with basic edition in its implementation. It takes care of many back-end checks such as password validation, password count etc.

MTV ARCHITECTURAL PATTERN OF DJANGO:

Django framework follows the **model template view** architectural pattern i.e MTV. Django also provides an optional administrative create, read, update and delete interface that is configured via admin models.

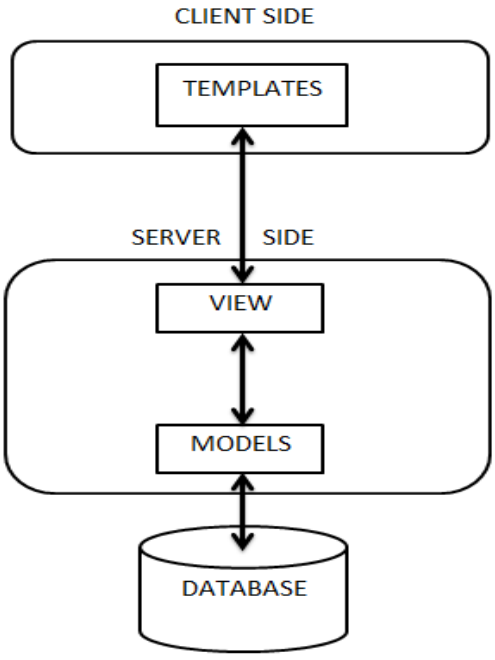


Figure 1MTV Graphical Representation

- **Pillow:**

Pillow is a python imaging library (PIL, newer version is named Pillow) used for opening, saving, manipulating and displaying different format files of images. This project employees the use of this library to save the product images in database and display them.

- **HTML, CSS, JAVASCRIPT:**

Front end of the project is developed using HTML, CSS and JAVASCRIPT.

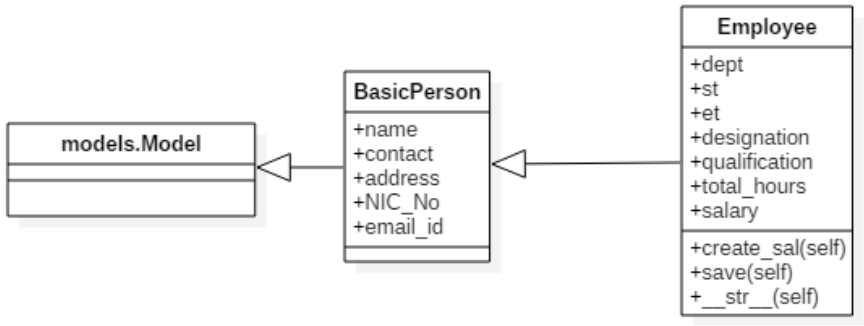
❖ OOP IMPLEMETATIONS:

- **Inheritance:**

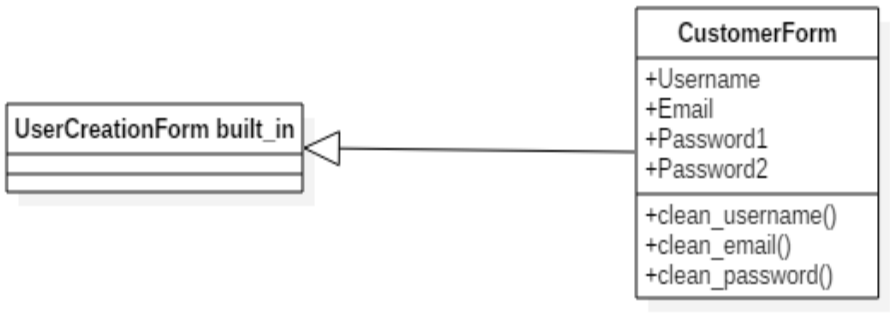
Inheritance is applied between two classes of **BasicPerson** and **Employee** class in which the former acts as the parent class. The parent class contains common attributes. The reason for this inheritance was that if one wanted to extend the code and add further classes such as supplier or any other staff member, would inherit from the same parent class.

Apart of this, following built-in classes have been inherited:

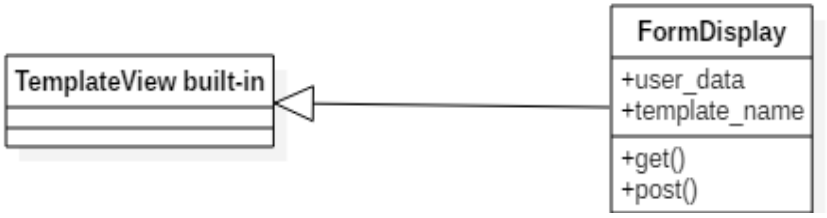
1. **Models.Model:** parent of every database class



2. **UserCreationForm:** parent of CustomerForm class

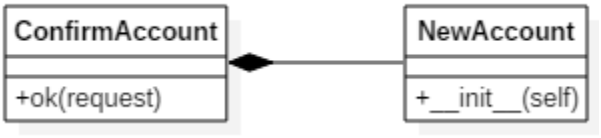


3. **TemplateView:** parent of FormDisplay class



- ## Composition:

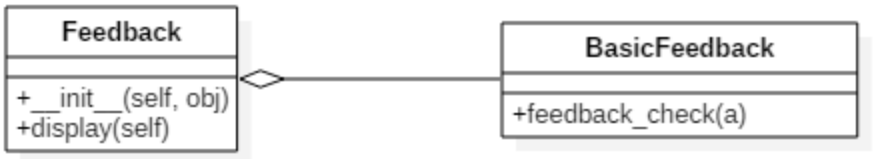
Composition is applied between **NewAccount** and **ConfirmAccount** class in which the former acts as content and holds the information of user, passes this information to the container (Confirm account) which displays it onto the html page.



- ## Aggregation:

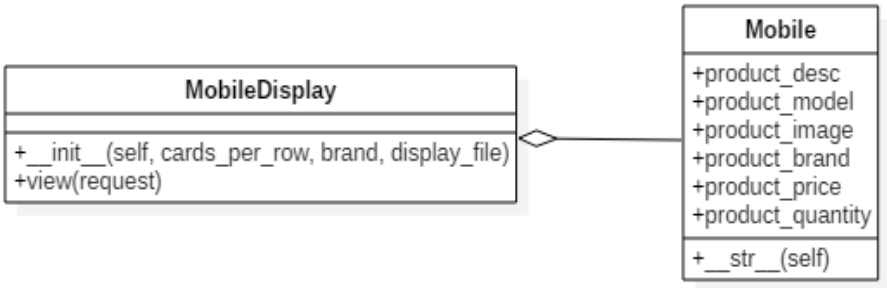
1. FeedbackDisplay –Feedback:

Aggregation is applied between **FeedbackDisplay** and **Feedback** in which the former acts as content to hold the feedback given by user and the container class renders it onto the html page.



2. MobileDisplay-Mobile:

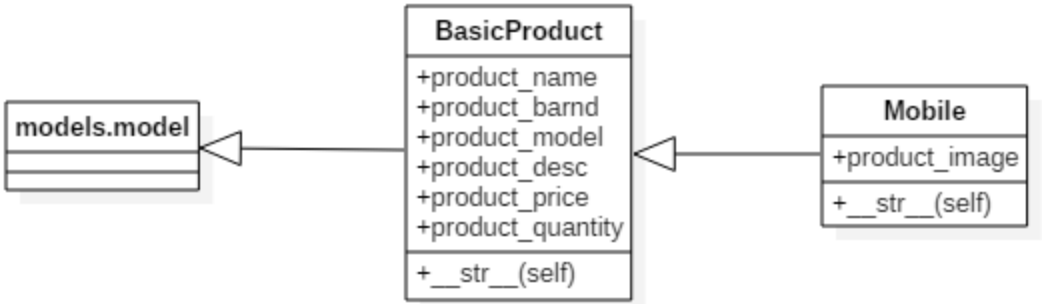
The products of Mobile (content) class are displayed by fetching their objects through aggregation in the MobileDisplay (container) class.



- **Polymorphism:**

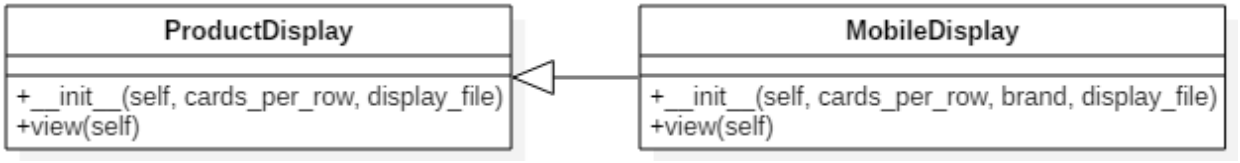
1. BasicProduct-Mobile:

The BasicProduct class is defined as an abstract class. It is inherited from the models. Models but it does not create any database of its own since it's an abstract class. In Django, model classes are made abstract by setting abstract =true in the Meta class defined within the abstract class. This class contains the basic fields (attributes) associated to all kinds of products. Different model classes for products of different categories will inherit this class.



2. ProductDisplay-MobileDisplay:

Product display class is a basic abstract class for displaying products of different categories. Hence the method of view is defined as an abstract method and is implemented in the Mobile Display (concrete class).



If the project is further developed for selling products of more categories other than mobiles, than those products will also inherit these two abstract classes.

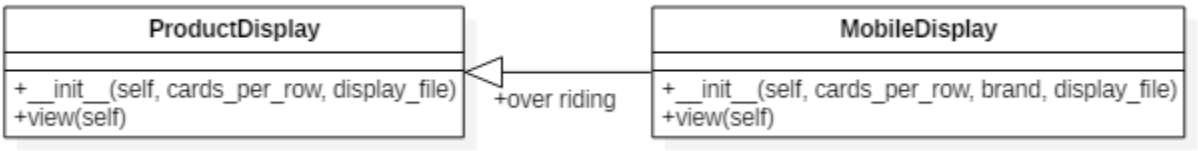
- **Overloading:**

The idea of overloading has been established in the BasicFeedback class to handle the feedback page if the user submits blank feedback.

- **Over-Riding:**

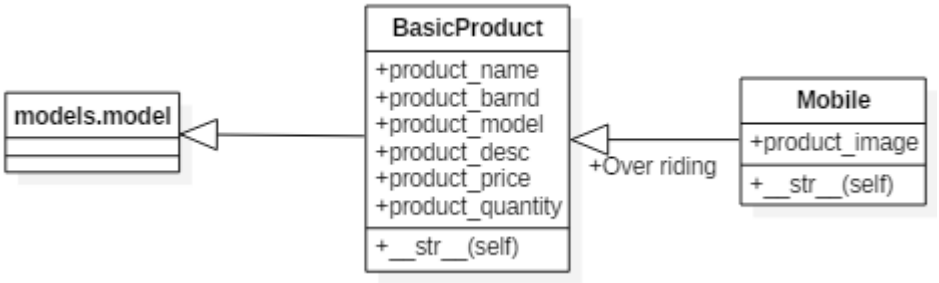
1. def views() :

The views function defined in ProductDisplay is over-ridden in MobileDisplay class.



2. def __str__():

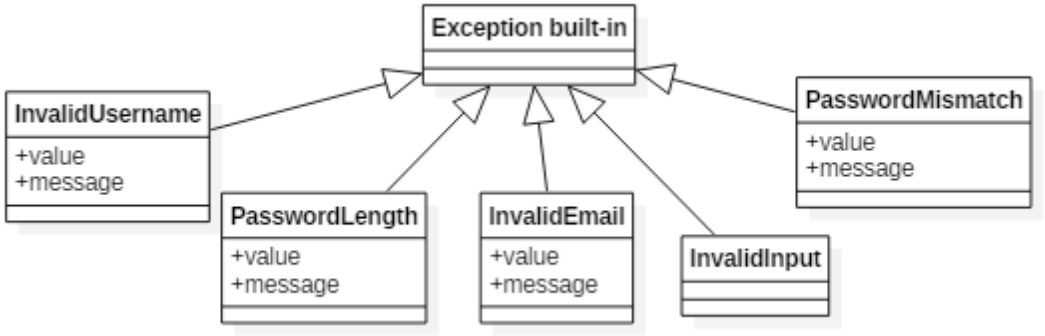
This is a builtin function to return name of objects created as object1,object2, etc. it is over ridden to return name of product an product model in BasicProduct and ProductDisplay class respectively.



- **Exception Handling:**

While a user signs up, the following custom exceptions have been made which are raised when the following exceptions arise.

- **Password Length :** A password containing less than eight characters
- **Password Mismatch :** Password cannot be confirmed
- **Invalid Username :** An account with the same username already exists
- **Invalid Email :** An account with the same email already exists



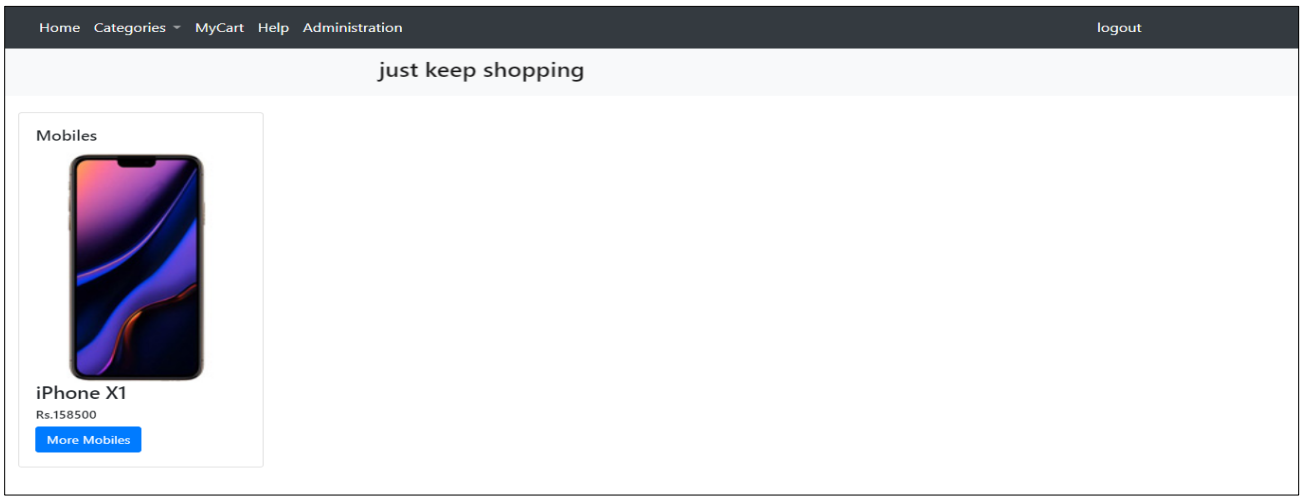
Other than the above mentioned exceptions, an exception **'Invalid input'** has been customarily defined which deals with the incorrect format of employee's data.

❖CONCLUSION:

The main product of this shopping site is mobile phone of three brands i.e. Nokia, Samsung and Apple. When the user first visits the website, he/she will be prompted to the home page. A navigation bar which contains the following features:

- **Home :**

  It sends the user to the home page

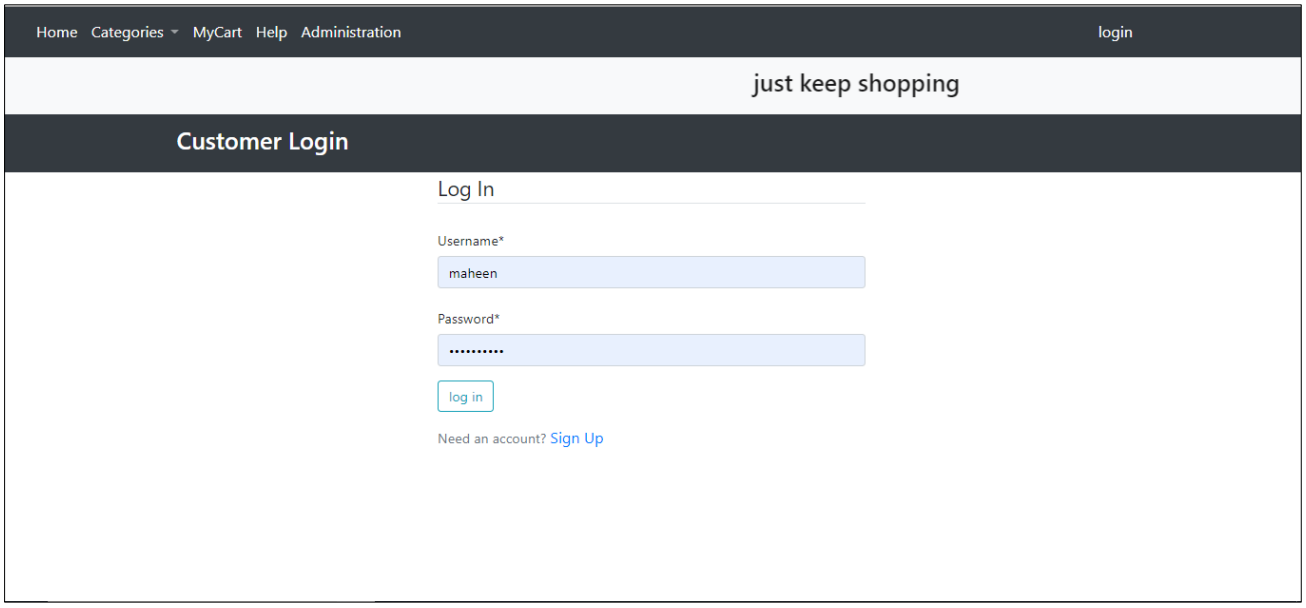


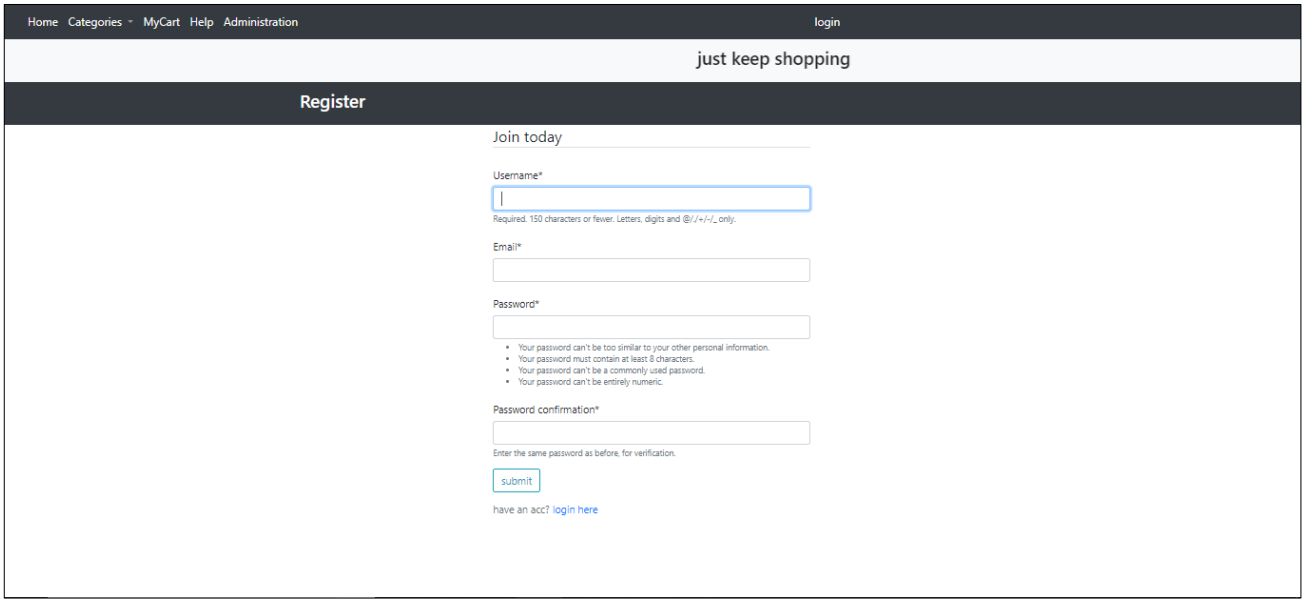
- **Help :**

  It allows the user to send a query

- **Login and Sign up :**

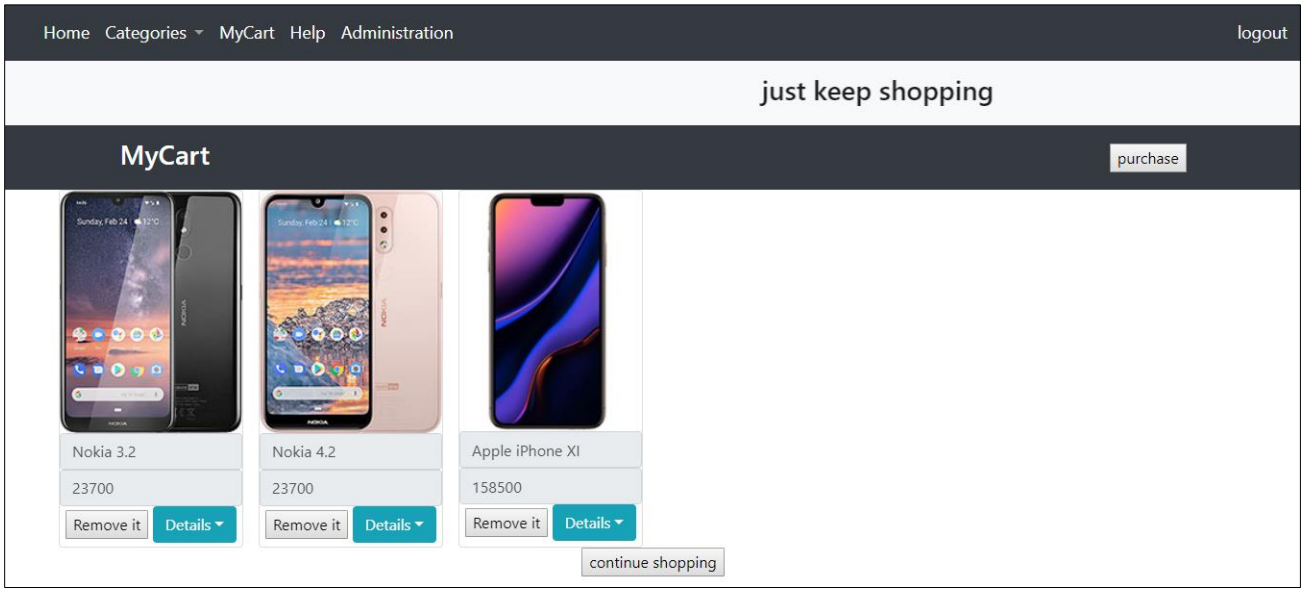
Once the user wants to purchase any product, he/she must sign up and then he can login.



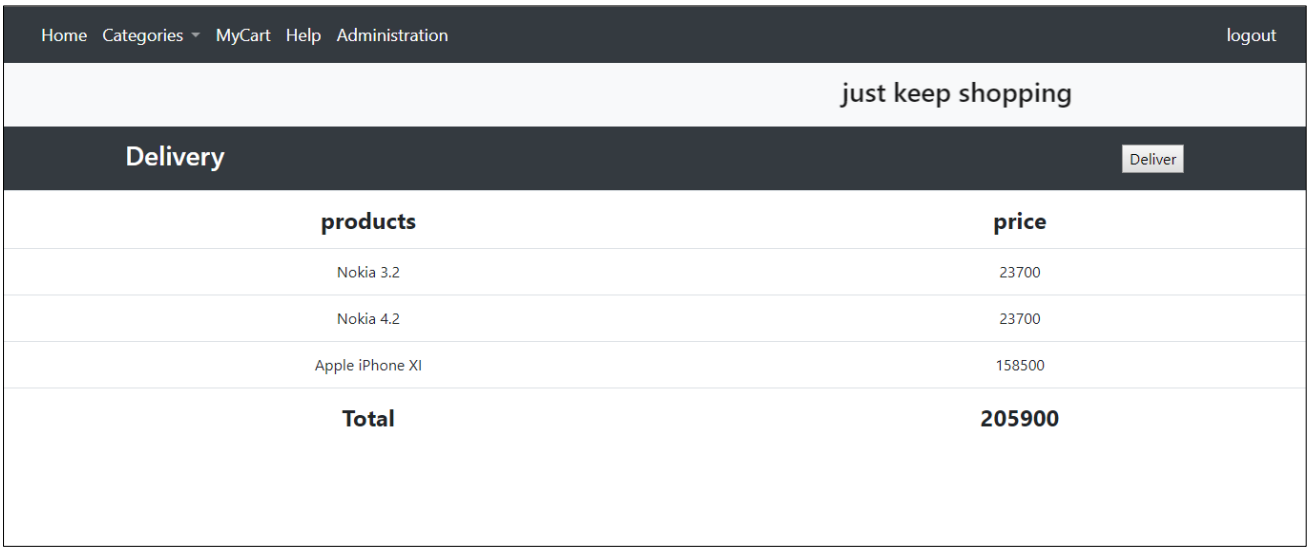


- **My Cart :**

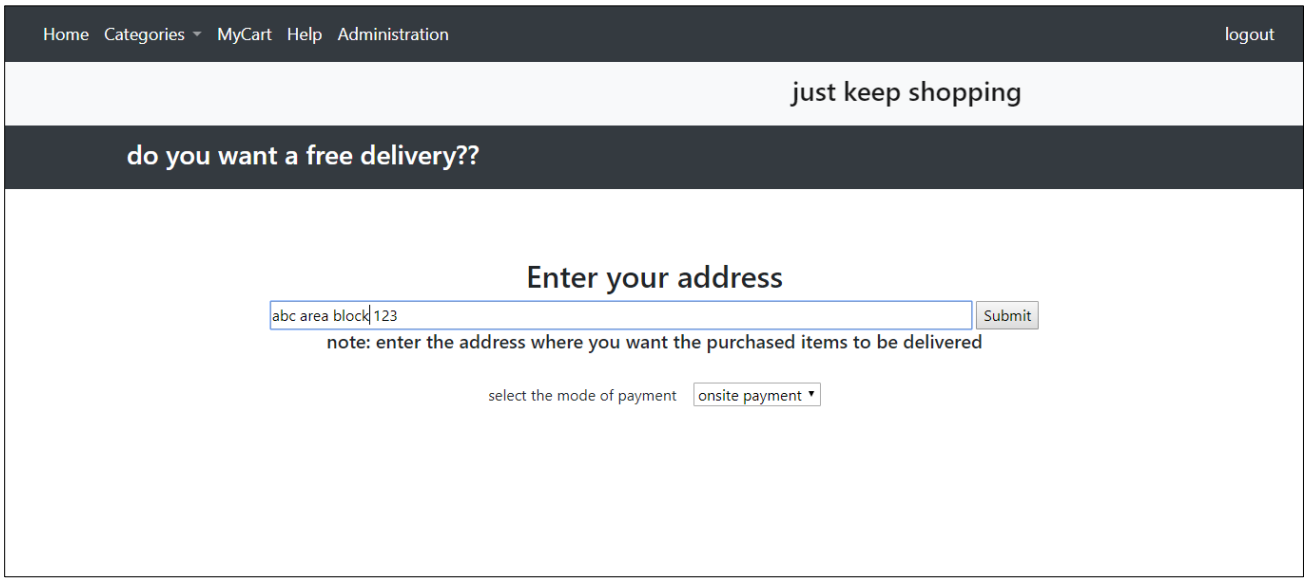
  A user can view the items which he/she wants to buy.



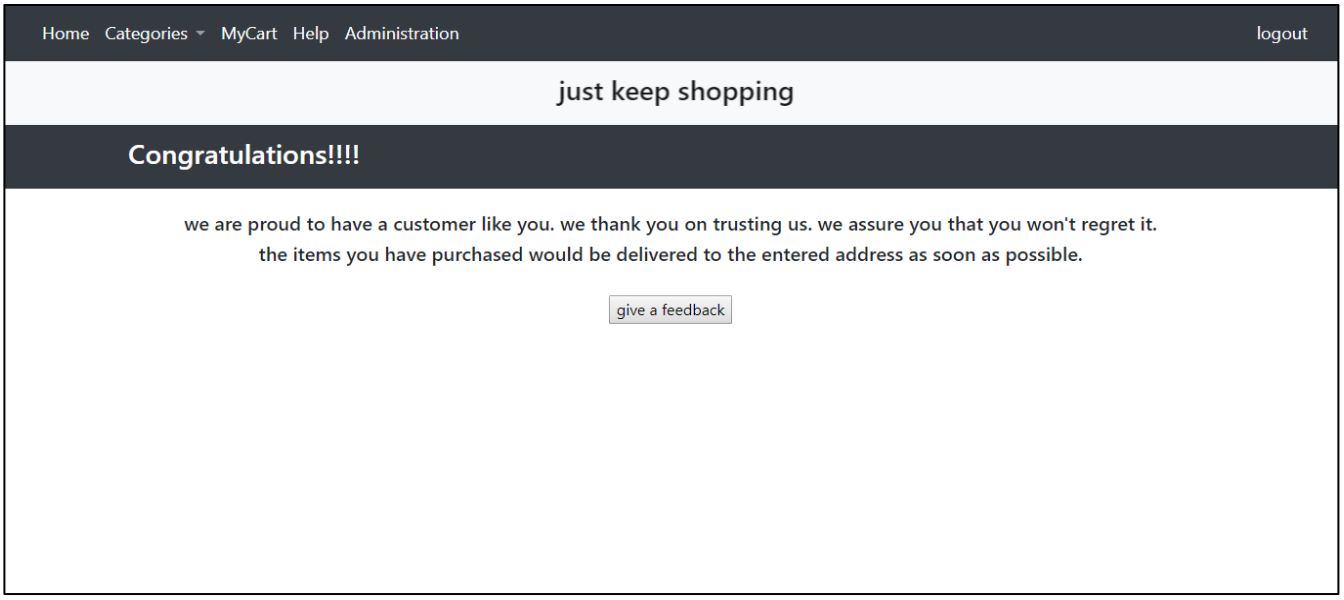
- **Bill Display:**
  Total amount of products in the cart will be displayed when **purchase** is clicked.



- **Product Delivery:**
  The user will provide the address and select the mode of payment of his goods.



- **Feedback:**
  After the users have purchased their products, they can also submit their feedback about the website.



So far, this shopping site deals with a single category of product i.e Mobiles but it can be easily extended for products of many different categories due to its object oriented structured programming.