**Lecture 9. Веб-сайт электронной коммерции с использованием Django**

Этот проект касается разработки виртуального веб-сайта «Веб-сайт **электронной коммерции»** . Он предоставляет пользователю список различных продуктов, доступных для покупки в магазине. Для удобства совершения покупок в Интернете пользователю предоставляется корзина. После выбора товара он отправляется на процедуру подтверждения заказа. Система реализована с использованием веб-фреймворка Python Django. Чтобы разработать сайт электронной коммерции, необходимо изучить и понять многие технологии.

**Объем** : Объем проекта будет ограничен некоторыми функциями веб-сайта электронной коммерции. Он будет отображать продукты, клиенты могут выбирать каталоги и выбирать продукты, а также могут удалять продукты из своей корзины, указывая количество каждого товара. Выбранные товары будут собраны в корзину. При оформлении заказа товар на карточке будет представлен как заказ. Клиенты могут оплатить товары в корзине, чтобы завершить заказ. У этого проекта большое будущее. Проект также обеспечивает безопасность с использованием идентификатора входа и паролей, чтобы неавторизованные пользователи не могли получить доступ к вашей учетной записи. Только авторизованное лицо, имеющее соответствующие права доступа, может получить доступ к программному обеспечению.

**Технологии, используемые в проекте:**

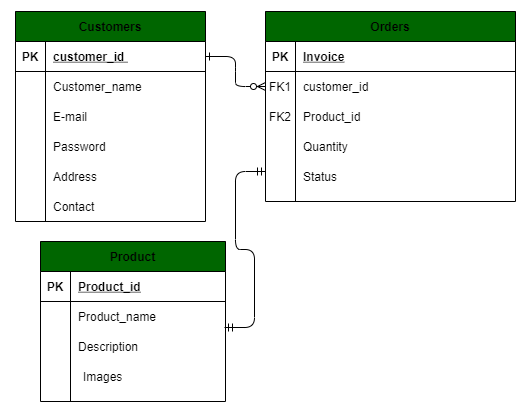
* Фреймворк Django и база данных SQLite, которые по умолчанию поставляются с Django.

**Необходимый набор навыков для создания проекта:**

* Знание Python и основ Django Framework.
* ER и диаграммы вариантов использования

**Клиентский интерфейс:**

1. Клиент покупает продукт
2. Клиент меняет количество
3. Покупатель добавляет товар в корзину
4. Клиент просматривает корзину
5. Клиент проверяет
6. Клиент отправляет заказ



ER-диаграмма для Заказчика

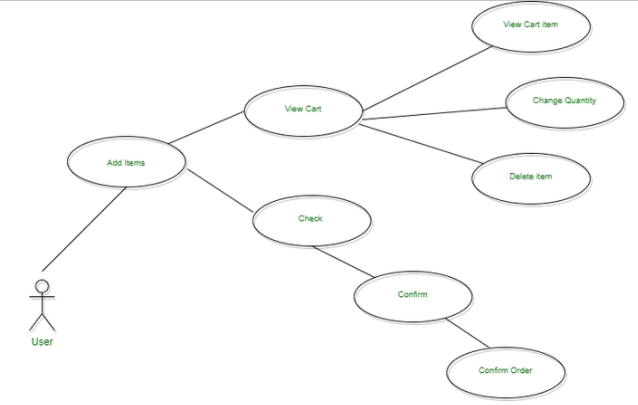
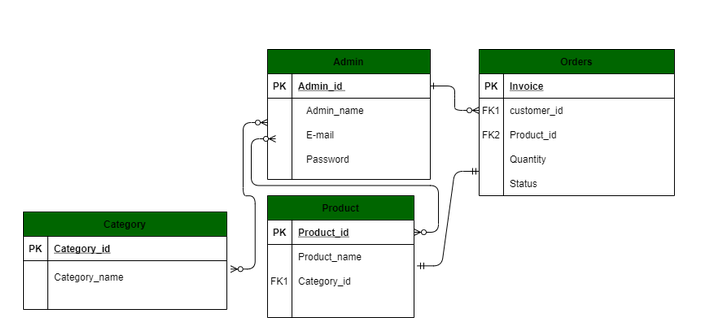


Диаграмма вариантов использования для клиента

**Интерфейс администратора :**

* Вход администратора
* Админ вставляет элемент
* Админ удаляет товар
* Админ изменяет элемент



ER-диаграмма для администратора

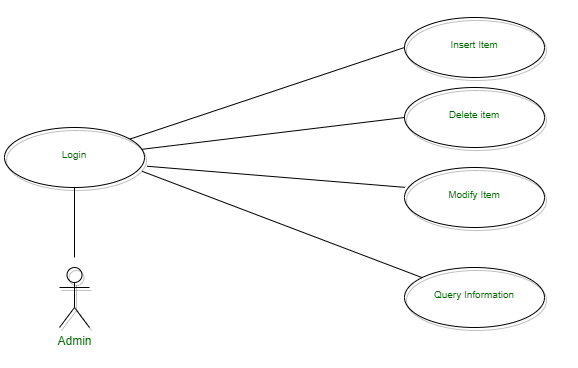
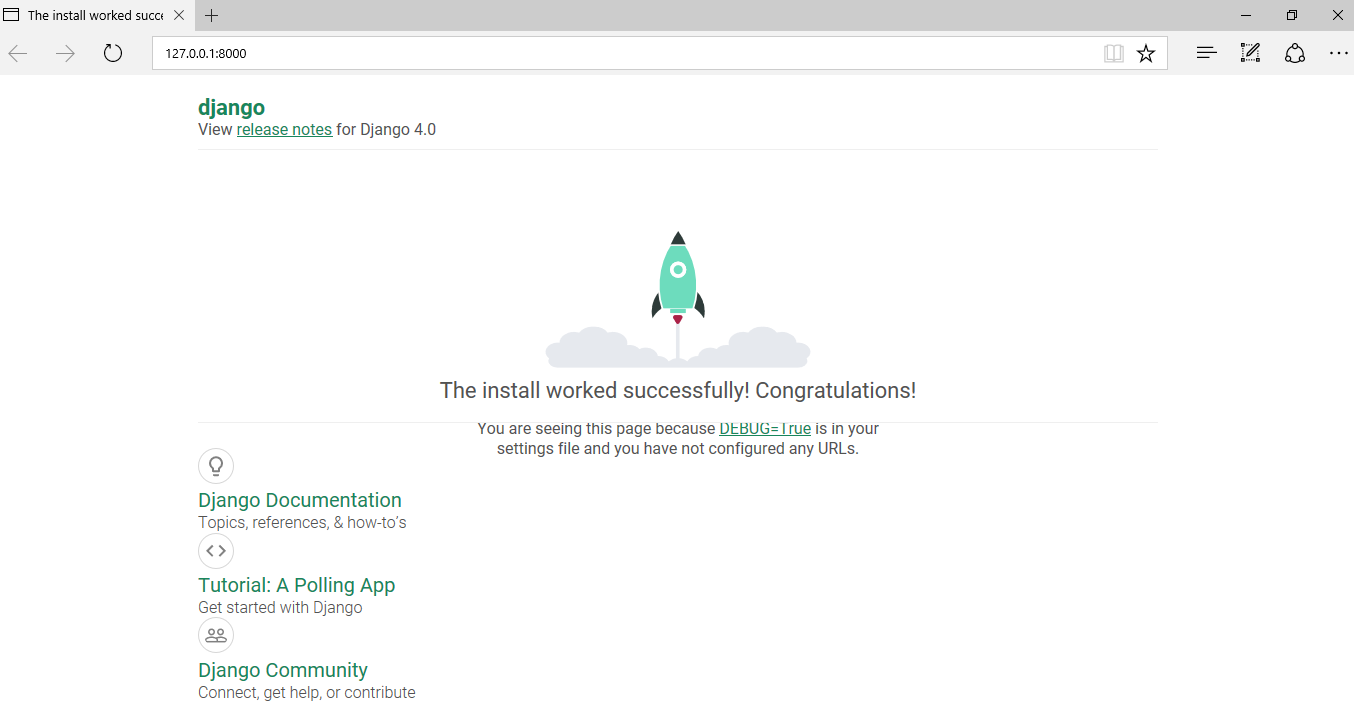


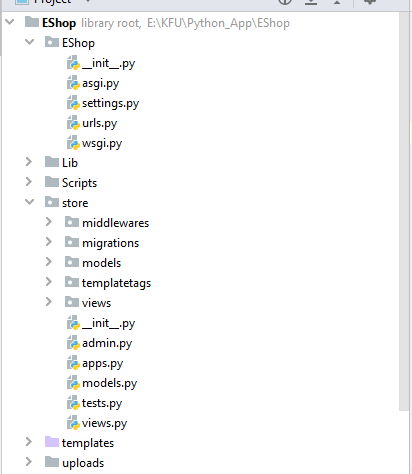
Диаграмма вариантов использования для администратора

Пошаговая реализация:

* Создать обычный проект. Откройте IDE и создайте обычный проект, **выбрав «Файл» -> «Новый проект».**
* Установите Django: Далее мы установим модуль Django из терминала. Для выполнения этой задачи мы будем использовать встроенный терминал PyCharm. Можно также использовать cmd в Windows для установки модуля, выполнив команду ***python -m pip install django .***
* Проверить установленную версию Django: чтобы проверить установленную версию Django, вы можете запустить команду ***python -m django -version*** , как показано ниже.
* Создать проект Django: Когда мы выполним команду ***django-admin startproject*** , она создаст проект Django внутри обычного проекта, который мы уже создали здесь. ***django-admin startproject ProjectName.***
* Проверьте версию Python3: ***python3 –version***
* Запуск веб-сервера Django по умолчанию: Django предоставляет веб-сервер по умолчанию, на котором мы можем запускать наши приложения. ***Команда запуска сервера python manage.py*** в терминале ***.***По умолчанию сервер работает на порту 8000. Получите доступ к веб-серверу по выделенному URL-адресу.



Откройте папку проекта с помощью текстового редактора. Структура каталогов должна выглядеть следующим образом:



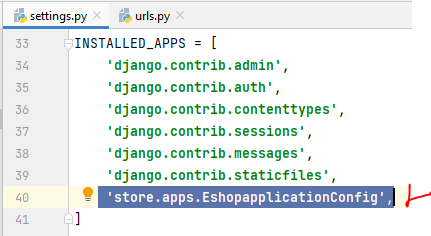
Прежде всего установим Pillow: python -m pip install Pillow

Python Imaging Library — библиотека языка Python, предназначенная для работы с растровой графикой.

Разработка библиотеки прекращена (последняя правка датируется 2011 годом). Однако проект под названием Pillow, являющийся [форком](https://ru.wikipedia.org/wiki/%D0%A4%D0%BE%D1%80%D0%BA" \o "Форк) PIL, развивается и включает, в том числе, поддержку Python 3.x. Этот форк был принят в качестве замены оригинальной библиотеки и включён в некоторые дистрибутивы [Linux](https://ru.wikipedia.org/wiki/Linux" \o "Linux), включая [Debian](https://ru.wikipedia.org/wiki/Debian_GNU/Linux" \o "Debian GNU/Linux) и [Ubuntu](https://ru.wikipedia.org/wiki/Ubuntu" \o "Ubuntu) (с [13.04](https://ru.wikipedia.org/w/index.php?title=Ubuntu_13.04_Raring_Ringtail&action=edit&redlink=1))

Структура проекта

Теперь добавьте приложение магазина на сайт электронной коммерции в **settings.py** .



**urls.py**

Этот файл содержит все шаблоны URL(urls.py), используемые веб-сайтом.

|  |
| --- |
| **from** django.contrib **import** admin **from** django.urls **import** path **from** django.conf **import** settings **from** django.conf.urls.static **import** static  urlpatterns = [  path(**'admin/'**, admin.site.urls), ]+static(settings.MEDIA\_URL, document\_root=settings.MEDIA\_ROOT) |

**Модели**

На приведенном ниже снимке экрана показаны необходимые модели, которые нам нужно будет создать. Эти модели представляют собой таблицы, которые будут храниться в базе данных SQLite.



Давайте посмотрим на каждую модель и поля, необходимые для каждой модели store/models.

1. Category. Данная модель имеет следующий вид в виде таблицы.

|  |  |
| --- | --- |
| Название | Тип |
| Id | Идентификатор создается автоматически |
| Name | CharField(max\_length=50) |
| CREATE TABLE "store\_category" (  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,  "name" varchar(50) NOT NULL) | |

store/models/category.py

|  |
| --- |
| **from** django.db **import** models  **class** Category(models.Model):  name = models.CharField(max\_length=50)  @staticmethod  **def** get\_all\_categories():  **return** Category.objects.all()   **def** \_\_str\_\_(self):  **return** self.name |

1. Customer. Данная модель имеет следующий вид в виде таблицы.

|  |  |
| --- | --- |
| Название | Тип |
| Id | Идентификатор создается автоматически |
| first\_name | CharField(max\_length=50) |
| last\_name | CharField(max\_length=50) |
| Phone | CharField(max\_length=12) |
| Email | EmailField() |
| Password | CharField(max\_length=100) |
| CREATE TABLE "store\_customer" (  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,  "phone" varchar(10) NOT NULL,  "email" varchar(254) NOT NULL,  "password" varchar(100) NOT NULL,  "last\_name" varchar(50) NOT NULL,  "first\_name" varchar(50) NOT NULL) | |

store/models/customer.py

|  |
| --- |
| **from** django.db **import** models  **class** Customer(models.Model):  first\_name = models.CharField(max\_length=50)  last\_name = models.CharField (max\_length=50)  phone = models.CharField(max\_length=12)  email=models.EmailField()  password = models.CharField(max\_length=100)   *#to save the data* **def** register(self):  self.save()   @staticmethod  **def** get\_customer\_by\_email(email):  **try**:  **return** Customer.objects.get(email= email)  **except**:  **return False   def** isExists(self):  **if** Customer.objects.filter(email = self.email):  **return True   return False** |

1. Products. Данная модель имеет следующий вид в виде таблицы.

|  |  |
| --- | --- |
| Название | Тип |
| Id | Идентификатор создается автоматически |
| Name | CharField(max\_length=60) |
| Price | IntegerField(default=0) |
| Category | ForeignKey(Category, on\_delete=models.CASCADE,default=1 ) |
| description | CharField(max\_length=250, default=**''**, blank=**True**, null= **True**) |
| Image | ImageField(upload\_to=**'uploads/products/'**) |
| CREATE TABLE "store\_products" (  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,  "name" varchar(60) NOT NULL,  "price" integer NOT NULL,  "description" varchar(250) NULL,  "category\_id" integer NOT NULL REFERENCES "store\_category" ("id") DEFERRABLE INITIALLY DEFERRED,  "image" varchar(100) NOT NULL) | |

Состояние DEFERRABLE или NOT DEFERRABLE задается при создании ограничения и изменить его можно только пересоздание ограничения.

store/models/product.py

|  |
| --- |
| **from** django.db **import** models **from** .category **import** Category **class** Products(models.Model):  name = models.CharField(max\_length=60)  price= models.IntegerField(default=0)  category= models.ForeignKey(Category,on\_delete=models.CASCADE,default=1 )  description= models.CharField(max\_length=250, default=**''**, blank=**True**, null= **True**)  image= models.ImageField(upload\_to=**'uploads/products/'**)   @staticmethod  **def** get\_products\_by\_id(ids):  **return** Products.objects.filter (id\_\_in=ids)  @staticmethod  **def** get\_all\_products():  **return** Products.objects.all()   @staticmethod  **def** get\_all\_products\_by\_categoryid(category\_id):  **if** category\_id:  **return** Products.objects.filter (category=category\_id)  **else**:  **return** Products.get\_all\_products(); |

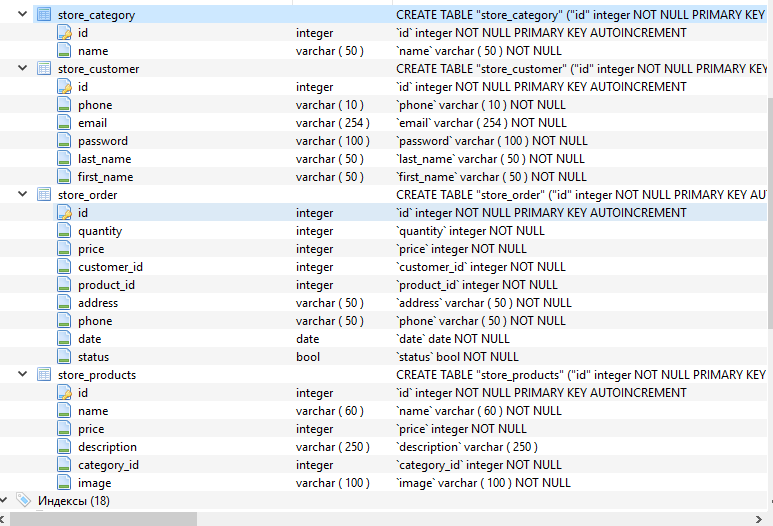
1. Orders. Данная модель имеет следующий вид в виде таблицы.

|  |  |
| --- | --- |
| Название | Тип |
| Id | Идентификатор создается автоматически |
| Product | ForeignKey(Products, on\_delete=models.CASCADE) |
| Customer | ForeignKey(Customer, on\_delete=models.CASCADE) |
| Quantity | IntegerField(default=1) |
| Price | IntegerField() |
| Address | CharField (max\_length=50, default=**''**, blank=**True**) |
| Phone | CharField (max\_length=50, default=**''**, blank=**True**) |
| Date | DateField (default=datetime.datetime.today) |
| Status | BooleanField (default=**False**) |
| CREATE TABLE "store\_order" (  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,  "quantity" integer NOT NULL,  "price" integer NOT NULL,  "customer\_id" integer NOT NULL REFERENCES  "store\_customer" ("id") DEFERRABLE INITIALLY DEFERRED,  "product\_id" integer NOT NULL REFERENCES  "store\_products" ("id") DEFERRABLE INITIALLY DEFERRED,  "address" varchar(50) NOT NULL,  "phone" varchar(50) NOT NULL,  "date" date NOT NULL,  "status" bool NOT NULL) | |

***quantity - количество***

store/models/orders.py

|  |
| --- |
| **from** django.db **import** models **from** .product **import** Products **from** .customer **import** Customer **import** datetime  **class** Order(models.Model):  product = models.ForeignKey(Products,  on\_delete=models.CASCADE)  customer = models.ForeignKey(Customer,  on\_delete=models.CASCADE)  quantity = models.IntegerField(default=1)  price = models.IntegerField()  address = models.CharField (max\_length=50, default=**''**, blank=**True**)  phone = models.CharField (max\_length=50, default=**''**, blank=**True**)  date = models.DateField (default=datetime.datetime.today)  status = models.BooleanField (default=**False**)   **def** placeOrder(self):  self.save()   @staticmethod  **def** get\_orders\_by\_customer(customer\_id):  **return** Order.objects.filter(customer=customer\_id).order\_by(**'-date'**) |



Скопируйте папку uploads и файл базы данных из архива и вставьте их в корневой каталог проекта.

### **Представление:**

1. В папке шаблонов (templates) мы создадим шаблон с именем base.html, содержимое которого будет следующим:

|  |
| --- |
| *<!--region Description-->* <!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://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"  integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"  crossorigin="anonymous"**>  <**title**>E-commerce Website</**title**>  <**style**>  **nav** {  **background-color**: **green**;  }  .**one-edge-shadow** {  **box-shadow**: 0 8**px** 2**px** -5**px rgb**(246, 245, 245);  }  .**display-8** {  **font-weight**: 200;  **font-size**: 30**px**;  **font-colour**: **teal-200**;  }  </**style**> </**head**> <**body**> *<!-- navbar -->* <**nav class="one-edge-shadow  navbar navbar-light  sticky-top navbar  navbar-expand-lg  text-white"**>  <**a class="navbar-brand ms-3 text-dark " href="/"**>  <**link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.0/css/all.css"  integrity="sha384-lZN37f5QGtY3VHgisS14W3ExzMWZxybE1SJSEsQp9S+oqd12jhcu+A56Ebc1zFSJ"  crossorigin="anonymous"**>  <**img src="https://www.freepnglogos.com/uploads/shopping-cart-png/shopping-cart-svg-png-icon-download-28.png"  style="width**:50**px**;**height**:50**px**;**color**:**text-light**;**"**>E-Commerce</**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 class="collapse navbar-collapse" id="navbarSupportedContent"**>  <**ul class="navbar-nav mr-auto"**>  <**li class="nav-item active"**>  <**a class="nav-link" href="/store"**>Store <**span class="sr-only"**>(current)</**span**></**a**>  </**li**>  </**ul**>  <**ul class=" navbar-nav my-2 my-lg-0"**>  <**li class="nav-item active"**>  <**a class="nav-link" href="/cart"**>Cart  <**span class="badge badge-success"**>{{ **request**.**session**.**cart**.**keys**|**length** }}</**span**>  <**span class="sr-only"**>(current)</**span**></**a**>  </**li**>  *{# {% if request.session.customer %}#}* <**li class="nav-item active "**>  *{# <a class="nav-link" href="/orders">Orders<span class="sr-only">(current)</span></a>#}* </**li**>  <**li class="nav-item active "**>  <**a class="nav-link" href="/logout"**>Logout<**span class="sr-only"**>(current)</**span**></**a**>  </**li**>  *{# {% else %}#}* <**li class="nav-item active"**>  <**a class="nav-link" href="/signup"**>Signup<**span class="sr-only"**>(current)</**span**></**a**>  </**li**>  <**li class="nav-item active"**>  <**a class="nav-link" href="/login"**>Login<**span class="sr-only"**>(current)</**span**></**a**>  </**li**>  *{# {% endif %}#}* </**ul**>  </**div**> </**nav**> *<!---->* {% **block content** %} {% **endblock** %} </**body**> </**html**> *<!--endregion-->* |

1. В папке templates создаем файл Index.html, содержимое которого следующее:

|  |
| --- |
| {% **extends 'base.html'** %} {% **block content** %}  *{# {% load cart %}#}  {# {% load custom\_filter %}#}  <!-- body -->* <**div class="container-fluid mt-3"**>  <**div class="row"**>  *<!-- all products -->* <**div id='products' class="col-lg-9 mx-auto"**>  <**div class="row mx-auto"**>  {% **for product in products** %}  <**div class="card mx-auto mb-3" id=**{{ **product**.**id** }} **style="width:18rem;"**>  <**img class="card-img-top" src="**{{ **product**.**image**.**url** }}**" alt="Card image cap"**>  <**div class="card-body"**>  <**p class="card-title"**>{{ **product**.**name** }}</**p**>  </**div**>  </**div**>  {% **endfor** %}  </**div**>  </**div**>  </**div**>  </**div**> {% **endblock** %} |

1. В папке Views создайте файл с именем home.py и внутри него создайте метод Index:

|  |
| --- |
| **from** django.shortcuts **import** render **from** django.http **import** HttpResponseRedirect **from** django.views **import** View **from** store.models.product **import** Products  *# Create your views here.* **class** Index(View):  **def** get(self, request):  *# print()* **return** HttpResponseRedirect(**f'/store{**request.get\_full\_path()[1:]**}'**)  **def** store(request):  products = Products.get\_all\_products();  data = {}  data[**'products'**] = products  **return** render(request, **'index.html'**, data) |

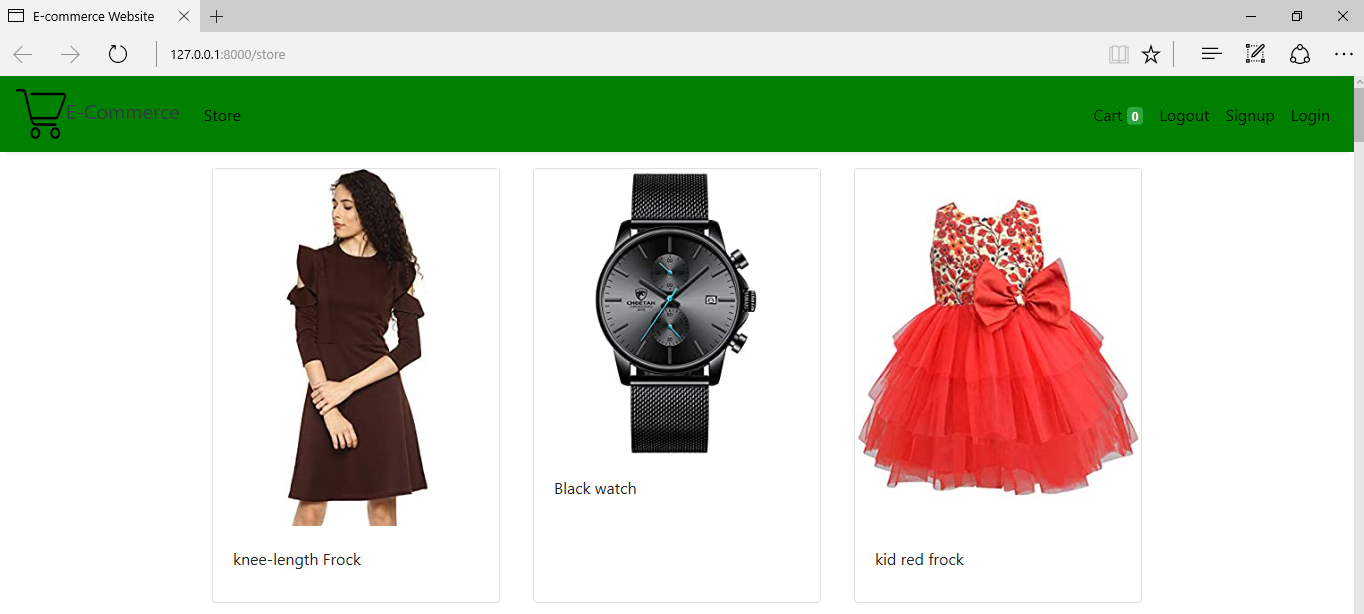
1. Измените файл urls следующим образом:

|  |
| --- |
| **from** django.conf **import** settings **from** django.conf.urls.static **import** static **from** django.urls **import** path **from** store.views.home **import** Index, store  urlpatterns = [  path(**''**, Index.as\_view(), name=**'homepage'**),  path(**'store'**, store, name=**'store'**), ]+static(settings.MEDIA\_URL, document\_root=settings.MEDIA\_ROOT) |

1. В конец файла settings.py мы добавляем следующий код для отображения изображений.

|  |
| --- |
| **import** os; MEDIA\_URL = **"/image/download/"** MEDIA\_ROOT = BASE\_DIR STATICFILES\_DIRS = [  os.path.join(BASE\_DIR, **"static"**), ] |

Запустим программу



**Фильтр**

1. 2) Чтобы показать категории справа, измените метод store в файле store/views/home.py следующим образом.

|  |
| --- |
| **def** store(request):  categories = Category.get\_all\_categories()  products = Products.get\_all\_products();  data = {}  data[**'products'**] = products  data[**'categories'**] = categories  *# print('you are : ', request.session.get('email'))* **return** render(request, **'index.html'**, data) |

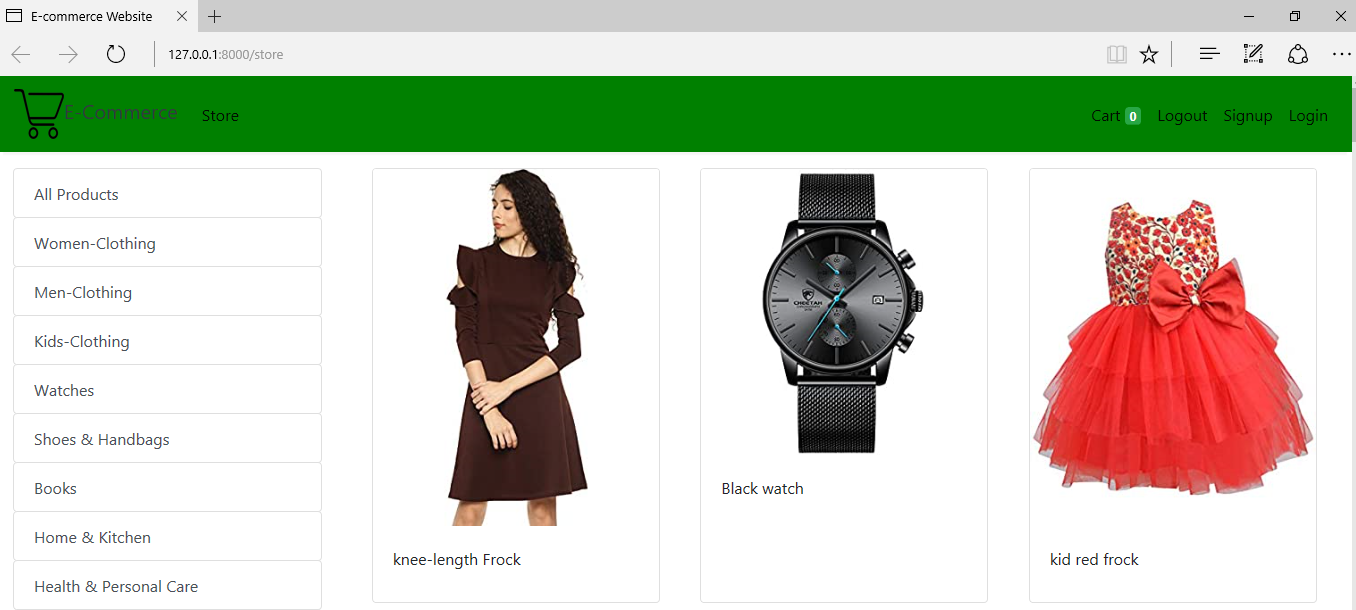
1. В итоге файл views/home.py принимает следующий вид:

|  |
| --- |
| **from** django.shortcuts **import** render **from** django.http **import** HttpResponseRedirect **from** django.views **import** View **from** store.models.product **import** Products,Category  *# Create your views here.* **class** Index(View):  **def** get(self, request):  *# print()* **return** HttpResponseRedirect(**f'/store{**request.get\_full\_path()[1:]**}'**)  **def** store(request):  categories = Category.get\_all\_categories()  products = Products.get\_all\_products();  data = {}  data[**'products'**] = products  data[**'categories'**] = categories  *# print('you are : ', request.session.get('email'))* **return** render(request, **'index.html'**, data) |

1. аходим в файл templates/Index.html в начале после тега <div class="row"> пишем следующие команды:

|  |
| --- |
| *<!-- filter -->* <**div class="col-lg-3 mx-auto"**>  <**div class="list-group"**>  <**a href="/" class="list-group-item list-group-item-action btn btn-outline-success"**>All Products</**a**>  {% **for category in categories** %}  <**a href="/?category=**{{**category**.**id**}}**"  class="list-group-item list-group-item-action btn btn-outline-success "**>{{**category**.**name**}}</**a**>  {% **endfor** %}  </**div**> </**div**> |

В результате, если мы запустим программу, она примет следующий вид:



1. Для того, чтобы активировать фильтр, т.е. при нажатии на категорию показать сопутствующие товары, перейдите в файл views/home.py и измените содержимое метода store следующим образом.

|  |
| --- |
| **def** store(request):  products = **None** categories = Category.get\_all\_categories()  categoryID = request.GET.get(**'category'**)  **if** categoryID:  products = Products.get\_all\_products\_by\_categoryid(categoryID)  **else**:  products = Products.get\_all\_products();  data = {}  data[**'products'**] = products  data[**'categories'**] = categories  print(**'you are : '**, request.session.get(**'email'**))  **return** render(request, **'index.html'**, data) |

В результате при нажатии на нужный элемент появится связанная с ним информация.

**Внизу карточки вывод количества и счетчик**

В папке магазина создайте каталог с именем templatetags, внутри которого мы создадим файл с именем custom\_filter.py, содержимое которого следующее

|  |
| --- |
| **from** django **import** template  register = template.Library()   @register.filter(name=**'currency'**) **def** currency(number):  **return "Rs "** + str(number)   @register.filter(name=**'multiply'**) **def** multiply(number, number1):  **return** number \* number1 |

1. Аналогично создадим файл templatetags/cart.py со следующим содержимым:

|  |
| --- |
| **from** django **import** template  register = template.Library () @register.filter (name=**'is\_in\_cart'**) **def** is\_in\_cart(product, cart):  keys = cart.keys ()  **for** id **in** keys:  **if** int (id) == product.id:  **return True  return False**;   @register.filter (name=**'cart\_quantity'**) **def** cart\_quantity(product, cart):  keys = cart.keys ()  **for** id **in** keys:  **if** int (id) == product.id:  **return** cart.get (id)  **return** 0; |

1. В классе Index (views/home) добавляем метод post следующим образом.

|  |
| --- |
| **def** post(self , request):  product = request.POST.get(**'product'**)  remove = request.POST.get(**'remove'**)  cart = request.session.get(**'cart'**)  **if** cart:  quantity = cart.get(product)  **if** quantity:  **if** remove:  **if** quantity<=1:  cart.pop(product)  **else**:  cart[product] = quantity-1  **else**:  cart[product] = quantity+1  **else**:  cart[product] = 1  **else**:  cart = {}  cart[product] = 1  request.session[**'cart'**] = cart  print(**'cart'** , request.session[**'cart'**])  **return** redirect(**'homepage'**) |

В результате файл принимает следующий вид:

|  |
| --- |
| **from** django.shortcuts **import** render, redirect **from** django.http **import** HttpResponseRedirect **from** django.views **import** View **from** store.models.product **import** Products,Category  *# Create your views here.* **class** Index(View):  **def** get(self, request):  *# print()* **return** HttpResponseRedirect(**f'/store{**request.get\_full\_path()[1:]**}'**)   **def** post(self , request):  product = request.POST.get(**'product'**)  remove = request.POST.get(**'remove'**)  cart = request.session.get(**'cart'**)  **if** cart:  quantity = cart.get(product)  **if** quantity:  **if** remove:  **if** quantity<=1:  cart.pop(product)  **else**:  cart[product] = quantity-1  **else**:  cart[product] = quantity+1  **else**:  cart[product] = 1  **else**:  cart = {}  cart[product] = 1  request.session[**'cart'**] = cart  print(**'cart'** , request.session[**'cart'**])  **return** redirect(**'homepage'**)   **def** store(request):  products = **None** categories = Category.get\_all\_categories()  categoryID = request.GET.get(**'category'**)  **if** categoryID:  products = Products.get\_all\_products\_by\_categoryid(categoryID)  **else**:  products = Products.get\_all\_products();  data = {}  data[**'products'**] = products  data[**'categories'**] = categories  print(**'you are : '**, request.session.get(**'email'**))  **return** render(request, **'index.html'**, data) |

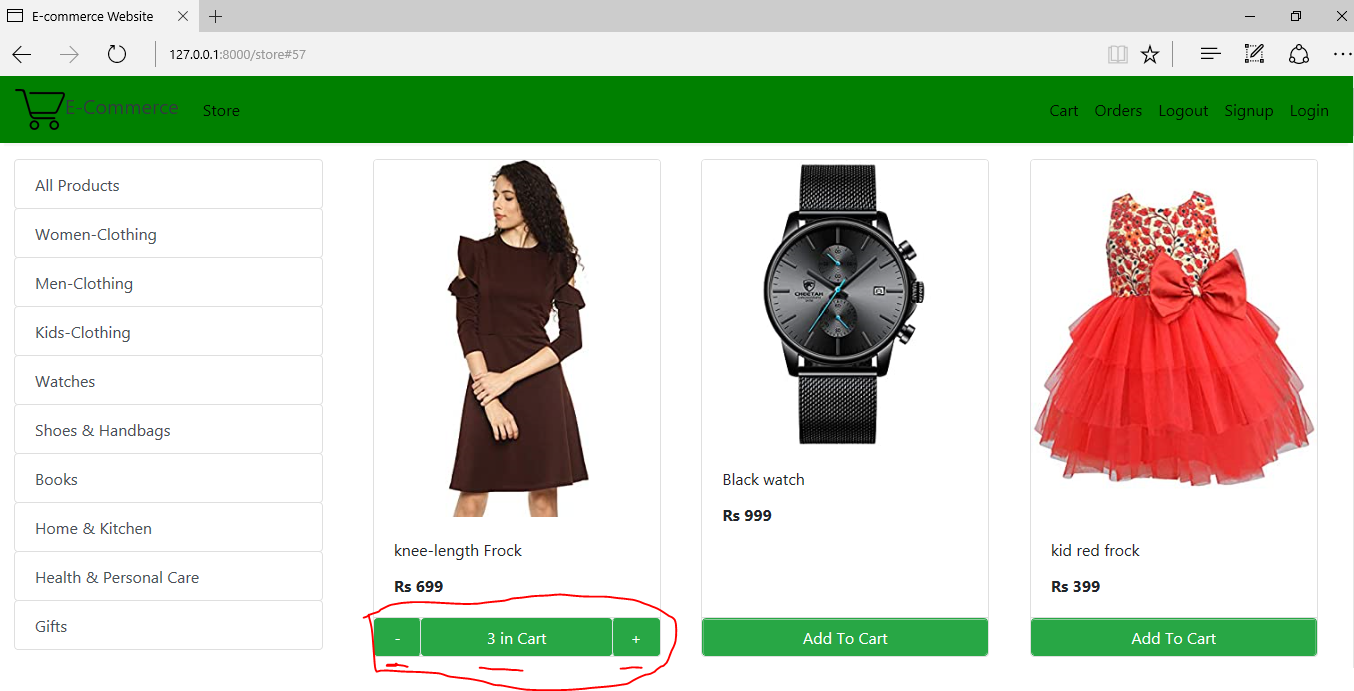
1. Измените начало файла Index.html следующим образом:

|  |
| --- |
| {% **extends 'base.html'** %} {% **block content** %} {% **load cart** %}  {% **load custom\_filter** %} {% **load static** %} |

1. Измените блок AllProducts на следующий вид.

|  |
| --- |
| *<!-- all products -->* <**div id='products' class="col-lg-9 mx-auto"**>  <**div class="row mx-auto"**>  {% **for product in products** %}  <**div class="card mx-auto mb-3" id=**{{ **product**.**id** }} **style="width:18rem;"**>  <**img class="card-img-top" src="**{{ **product**.**image**.**url** }}**" alt="Card image cap"**>  <**div class="card-body"**>  <**p class="card-title"**>{{ **product**.**name** }}</**p**>  <**p class="card-text"**><**b**>{{ **product**.**price**|**currency** }}</**b**></**p**>  *<!--* {{**product** | **is\_in\_cart**:**request**.**session**.**cart** }} *-->* </**div**>  <**div class="card-footer p-0 no-gutters"**>  {% **if product**|**is\_in\_cart**:**request**.**session**.**cart** %}  <**div class="row no-gutters"**>  <**form action="/#**{{ **product**.**id** }}**" class="col-2 " method="post"**>  {% **csrf\_token** %}  <**input hidden type="text" name='product' value='**{{ **product**.**id** }}**'**>  <**input hidden type="text" name='remove' value='True'**>  <**input type="submit" value=" - "  class="btn btn-block btn-success border-right"**>  </**form**>  <**div class="text-center col btn btn-success"**>{{ **product**|**cart\_quantity**:**request**.**session**.**cart** }}  in Cart  </**div**>  <**form action="/#**{{ **product**.**id** }}**" class="col-2 " method="post"**>  {% **csrf\_token** %}  <**input hidden type="text" name='product' value='**{{ **product**.**id** }}**'**>  <**input type="submit" value=" + "  class="btn btn-block btn-success border-left"**>  </**form**>  </**div**>  {% **else** %}  <**form action="/#**{{ **product**.**id** }}**" method="POST" class="btn-block"**>  {% **csrf\_token** %}  <**input hidden type="text" name='product' value='**{{ **product**.**id** }}**'**>  <**input type="submit" class="float-right btn btn-success form-control"  value="Add To Cart"**>  </**form**>  {% **endif** %}  </**div**>  </**div**>  {% **endfor** %}  </**div**> </**div**> |

В результате после выполнения программы появится следующая форма:



**Страница Входа**

1. В файле views создаем файл с именем signup, содержание которого следующее:

|  |
| --- |
| **from** django.shortcuts **import** render, redirect **from** django.contrib.auth.hashers **import** make\_password **from** store.models.customer **import** Customer **from** django.views **import** View  **class** Signup (View):  **def** get(self, request):  **return** render (request, **'signup.html'**)   **def** post(self, request):  postData = request.POST  first\_name = postData.get (**'firstname'**)  last\_name = postData.get (**'lastname'**)  phone = postData.get (**'phone'**)  email = postData.get (**'email'**)  password = postData.get (**'password'**)  *# validation* value = {  **'first\_name'**: first\_name,  **'last\_name'**: last\_name,  **'phone'**: phone,  **'email'**: email  }  error\_message = **None** customer = Customer (first\_name=first\_name,  last\_name=last\_name,  phone=phone,  email=email,  password=password)  error\_message = self.validateCustomer (customer)   **if not** error\_message:  print (first\_name, last\_name, phone, email, password)  customer.password = make\_password (customer.password)  customer.register ()  **return** redirect (**'homepage'**)  **else**:  data = {  **'error'**: error\_message,  **'values'**: value  }  **return** render (request, **'signup.html'**, data)   **def** validateCustomer(self, customer):  error\_message = **None  if** (**not** customer.first\_name):  error\_message = **"Please Enter your First Name !!"  elif** len (customer.first\_name) < 3:  error\_message = **'First Name must be 3 char long or more'  elif not** customer.last\_name:  error\_message = **'Please Enter your Last Name'  elif** len (customer.last\_name) < 3:  error\_message = **'Last Name must be 3 char long or more'  elif not** customer.phone:  error\_message = **'Enter your Phone Number'  elif** len (customer.phone) < 10:  error\_message = **'Phone Number must be 10 char Long'  elif** len (customer.password) < 5:  error\_message = **'Password must be 5 char long'  elif** len (customer.email) < 5:  error\_message = **'Email must be 5 char long'  elif** customer.isExists ():  error\_message = **'Email Address Already Registered..'** *# saving* **return** error\_message |

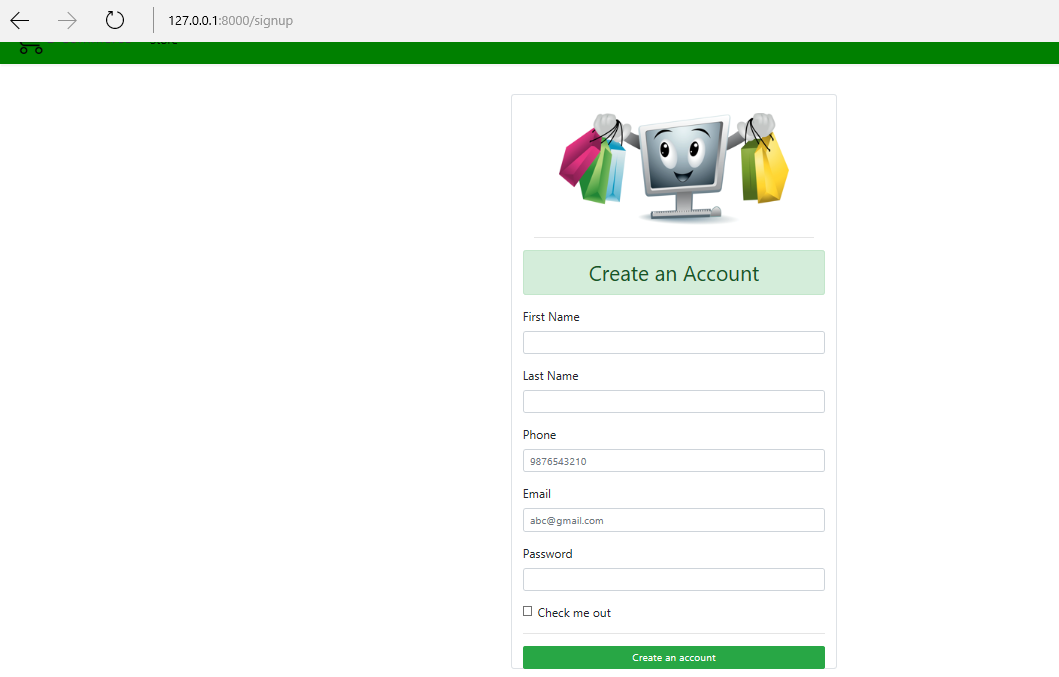
1. В папке шаблонов с именем signup.html мы создадим страницу со следующим содержимым:

|  |
| --- |
| {% **extends 'base.html'** %} {% **block content** %}  <**div class="container mt-4"**>  <**div class="p-3 m-3"**>  <**div class="col-lg-5 rounded mx-auto border pt-4"**>  <**div class="text-center col"**>  <**img  src="https://th.bing.com/th/id/Rd51d7c0fde6d96a31538a86cbd57f3c0?rik=%2fTJxRkU2V371QQ&riu=http%3a%2f%2ftechnologyend.com%2fwp-content%2fuploads%2f2015%2f11%2fwhy-shop-online.jpg&ehk=v1KiYT0i5U1ABCRueDSdLqnazh2BdlVD89Z4qXE0Kgs%3d&risl=&pid=ImgRaw"  alt=""  class="" style="height**: 150**px**;**"**>  *<!-- <hr> -->* <**hr**>  </**div**>  <**h3 class="alert alert-success rounded-pill" style="text-align**:**center"**>Create an Account</**h3**>  <**form action="/signup" method="POST"**>   {% **csrf\_token** %}   {% **if error** %}  <**div class="alert alert-danger" role="alert"**>  {{ **error** }}  </**div**>  {% **endif** %}  *<!-- firstname -->* <**div class="form-group"**>  <**label for=""**>First Name</**label**>  <**input type="text" name="firstname"  id="" value="**{{ **values**.**first\_name** }}**" class="form-control form-control-sm"  placeholder=""**>  </**div**>   *<!-- last name -->* <**div class="form-group"**>  <**label for=""**>Last Name</**label**>  <**input type="text" name="lastname"  id="" value="**{{ **values**.**last\_name** }}**" class="form-control form-control-sm"** >  </**div**>   *<!-- phone -->* <**div class="form-group"**>  <**label for=""**>Phone</**label**>  <**input type="text" name="phone"  id="" class="form-control form-control-sm"  value="**{{ **values**.**phone** }}**"  placeholder="9876543210"** >  </**div**>  *<!-- email -->* <**div class="form-group"**>  <**label for=""**>Email</**label**>  <**input required type="email" name="email" id=""  value="**{{ **values**.**email** }}**"  class=" form-control-sm form-control" placeholder="abc@gmail.com"**>  </**div**>  *<!-- password -->* <**div class="form-group"**>  <**label for=""**>Password</**label**>  <**input type="password"  name="password"  id=""  class="form-control form-control-sm"**>  </**div**>   *<!--Checkme button-->* <**div class="mb-3 form-check"**>  <**input type="checkbox" class="form-check-input" id="exampleCheck1"**>  <**label class="form-check-label d-grid gap-2" for="exampleCheck1"**>Check me out</**label**>  </**div**>  <**hr**>  <**button type="submit" class="btn btn-sm btn-success col-lg-12"**>Create an account</**button**>   </**form**>  </**div**>  </**div**>  </**div**>  {% **endblock** %} |

1. Ба файли urls.py рафта муҳтавои онро ба сурати зерин иҷро мекунем:

|  |
| --- |
| **from** django.contrib **import** admin **from** django.urls **import** path **from** store.views.home **import** Index, store **from** store.views.signup **import** Signup  **from** django.conf.urls.static **import** static  urlpatterns = [  path(**''**, Index.as\_view(), name=**'homepage'**),  path(**'store'**, store, name=**'store'**),  path(**'signup'**, Signup.as\_view(), name=**'signup'**),]+static(settings.MEDIA\_URL, document\_root=settings.MEDIA\_ROOT) |

Запустим программу.



Страница логин

1. Теперь давайте создадим логин. Для этого сначала создайте представление с именем login.py.

|  |
| --- |
| **from** django.shortcuts **import** render , redirect , HttpResponseRedirect **from** django.contrib.auth.hashers **import** check\_password **from** store.models.customer **import** Customer **from** django.views **import** View  **class** Login(View):  return\_url = **None   def** get(self, request):  Login.return\_url = request.GET.get (**'return\_url'**)  **return** render (request, **'login.html'**)   **def** post(self, request):  email = request.POST.get (**'email'**)  password = request.POST.get (**'password'**)  customer = Customer.get\_customer\_by\_email (email)  error\_message = **None  if** customer:  flag = check\_password (password, customer.password)  **if** flag:  request.session[**'customer'**] = customer.id   **if** Login.return\_url:  **return** HttpResponseRedirect (Login.return\_url)  **else**:  Login.return\_url = **None  return** redirect (**'homepage'**)  **else**:  error\_message = **'Invalid !!'  else**:  error\_message = **'Invalid !!'** print (email, password)  **return** render (request, **'login.html'**, {**'error'**: error\_message}) |

1. 2) В соответствии с ним создадим и его шаблон, т.е. login.html

|  |
| --- |
| {% **extends 'base.html'** %} {% **block content** %} <**div class="container"**>  <**div class="p-3 m-3"**>  <**div class="col-lg-5 rounded mx-auto border pt-4"**>  <**div class="text-center col"**>  <**img  src="https://th.bing.com/th/id/Rd51d7c0fde6d96a31538a86cbd57f3c0?rik=%2fTJxRkU2V371QQ&riu=http%3a%2f%2ftechnologyend.com%2fwp-content%2fuploads%2f2015%2f11%2fwhy-shop-online.jpg&ehk=v1KiYT0i5U1ABCRueDSdLqnazh2BdlVD89Z4qXE0Kgs%3d&risl=&pid=ImgRaw" alt=""  class="" style="height**: 150**px**;**"**>  <**hr**>  </**div**>  <**h3 class="alert alert-light rounded-pill" style="text-align**:**center"** >Login</**h3**>  <**form action="/login" method="POST"**>    {% **csrf\_token** %}   {% **if error**%}  <**div class="alert alert-danger" role="alert"**>  {{**error**}}  </**div**>  {% **endif** %}  *<!-- email -->* <**div class="form-group"**>  <**label for=""**>Email</**label**>  <**input required type="email" name="email" id=""  value="**{{**values**.**email**}}**"   class=" form-control-sm form-control" placeholder="abc@gmail.com"**>  </**div**>    *<!-- password -->* <**div class="form-group"**>  <**label for=""**>Password</**label**>  <**input type="password"   name="password"   id=""  class="form-control form-control-sm"** >  </**div**>   *<!--Checkme button-->* <**div class="mb-3 form-check"**>  <**input type="checkbox" class="form-check-input" id="exampleCheck1"**>  <**label class="form-check-label d-grid gap-2" for="exampleCheck1"**>Check me out</**label**>  </**div**>  <**hr**>  <**button type="submit" class="btn btn-sm btn-success col-lg-12"**>Login</**button**>  </**form**>  </**div**>  </**div**> </**div**> {% **endblock** %} |

1. Для выхода из программы добавьте в конец файла views/login.py следующий метод:

|  |
| --- |
| **def** logout(request):  request.session.clear()  **return** redirect(**'login'**) |

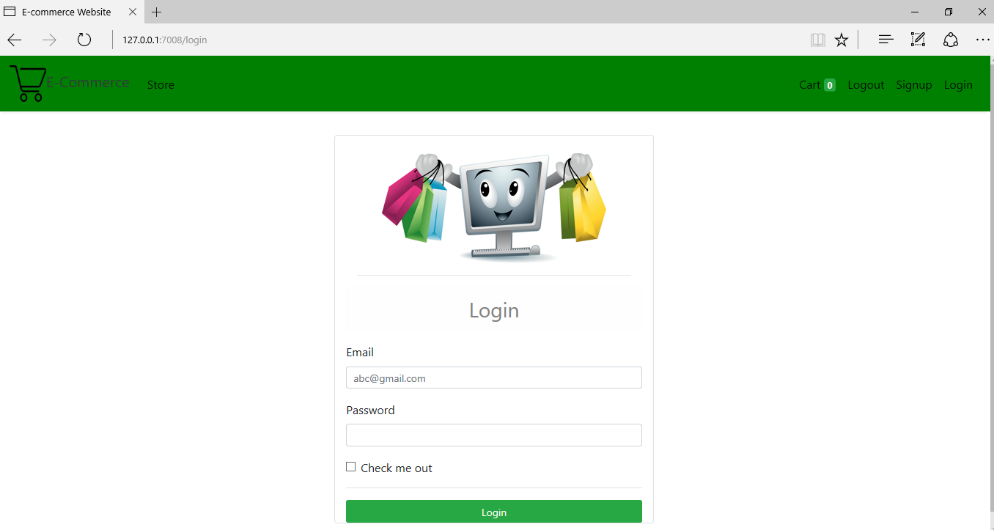
В результате файл принимает следующий вид:

|  |
| --- |
| **from** django.shortcuts **import** render , redirect , HttpResponseRedirect **from** django.contrib.auth.hashers **import** check\_password **from** store.models.customer **import** Customer **from** django.views **import** View  **class** Login(View):  return\_url = **None   def** get(self, request):  Login.return\_url = request.GET.get (**'return\_url'**)  **return** render (request, **'login.html'**)   **def** post(self, request):  email = request.POST.get (**'email'**)  password = request.POST.get (**'password'**)  customer = Customer.get\_customer\_by\_email (email)  error\_message = **None  if** customer:  flag = check\_password (password, customer.password)  **if** flag:  request.session[**'customer'**] = customer.id   **if** Login.return\_url:  **return** HttpResponseRedirect (Login.return\_url)  **else**:  Login.return\_url = **None  return** redirect (**'homepage'**)  **else**:  error\_message = **'Invalid !!'  else**:  error\_message = **'Invalid !!'** print (email, password)  **return** render (request, **'login.html'**, {**'error'**: error\_message})  **def** logout(request):  request.session.clear()  **return** redirect(**'login'**) |

1. Перейдите к файлу urls.py и измените его содержимое следующим образом:

|  |
| --- |
| **from** django.contrib **import** admin **from** django.urls **import** path **from** store.views.home **import** Index, store **from** store.views.signup **import** Signup **from** store.views.login **import** Login, logout **from** django.conf **import** settings **from** django.conf.urls.static **import** static  urlpatterns = [  path(**''**, Index.as\_view(), name=**'homepage'**),  path(**'store'**, store, name=**'store'**),  path(**'signup'**, Signup.as\_view(), name=**'signup'**),  path(**'login'**, Login.as\_view(), name=**'login'**),  path(**'logout'**, logout, name=**'logout'**),]+static(settings.MEDIA\_URL, document\_root=settings.MEDIA\_ROOT) |

Login



**Структура карточки**

1. Теперь приступим к созданию карты. Для этого создайте представление в файле представления с именем views/cart.py.

|  |
| --- |
| **from** django.shortcuts **import** render , redirect **from** django.contrib.auth.hashers **import** check\_password **from** store.models.customer **import** Customer **from** django.views **import** View **from** store.models.product **import** Products  **class** Cart(View):  **def** get(self , request):  ids = list(request.session.get(**'cart'**).keys())  products = Products.get\_products\_by\_id(ids)  print(products)  **return** render(request , **'cart.html'** , {**'products'** : products} ) |

1. В конец файла templatetags/cart добавляем следующие методы:

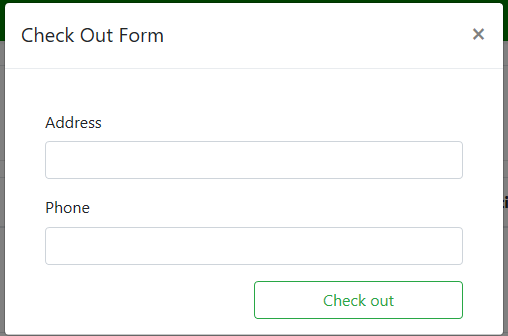
|  |
| --- |
| @register.filter (name=**'price\_total'**) **def** price\_total(product, cart):  **return** product.price \* cart\_quantity (product, cart)   @register.filter (name=**'total\_cart\_price'**) **def** total\_cart\_price(products, cart):  sum = 0;  **for** p **in** products:  sum += price\_total (p, cart)   **return** sum |

В соответствии с этим мы создадим шаблон в файле шаблонов с именем cart.html.

|  |
| --- |
| {% **extends 'base.html'** %} {% **block content** %}  {% **load cart** %}  {% **load custom\_filter** %}  <**div class="container"**>  <**div class="border rounded p-4 m-4"**>  <**p class="display-4 pl-4 ml-4 btn btn-success rounded-pill"**>Your Cart</**p**>  <**hr**>  <**table class="table"**>  <**thead**>  <**tr**>  <**th**>Sno.</**th**>  <**th**>Image</**th**>  <**th**>Product</**th**>  <**th**>Price</**th**>  <**th**>Quantity</**th**>  <**th**>Total</**th**>  </**tr**>  </**thead**>  <**tbody**>   {% **for product in products** %}  <**tr**>  <**td**>{{ **forloop**.**counter** }}</**td**>  <**td**><**img style="height**: 80**px**;**" class="rounded-circle" src="**{{ **product**.**image**.**url** }}**" alt=""**></**td**>  <**td**>{{ **product**.**name** }}</**td**>  <**td**>{{ **product**.**price**|**currency** }}</**td**>  <**td**>{{ **product**|**cart\_quantity**:**request**.**session**.**cart** }}</**td**>  <**td**>{{ **product**|**price\_total**:**request**.**session**.**cart**|**currency** }}</**td**>  </**tr**>  {% **endfor** %}  </**tbody**>  <**tfoot**>  <**tr**>  <**th colspan="4"**></**th**>  <**th class="" colspan=""**>Total</**th**>  <**th**>{{ **products**|**total\_cart\_price**:**request**.**session**.**cart**|**currency** }}</**th**>  </**tr**>  </**tfoot**>  </**table**>  <**hr**>  <**div class="m-3 p-3"**>  <**a href="#" data-toggle="modal" data-target="#exampleModal"  class="btn btn-outline-success border rounded col-lg-3 float-right"**>Check out</**a**>  </**div**>  </**div**>  </**div**>  *<!-- modal -->  <!-- Modal -->* <**div class="modal fade" id="exampleModal" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel"  aria-hidden="true"**>  <**div class="modal-dialog" role="document"**>  <**div class="modal-content"**>  <**div class="modal-header"**>  <**h5 class="modal-title" id="exampleModalLabel"**>  Check Out Form  </**h5**>  <**hr**>  <**button type="button" class="close" data-dismiss="modal" aria-label="Close"**>  <**span aria-hidden="true"**>**&times;**</**span**>  </**button**>  </**div**>  <**div class="modal-body"**>  <**div class="m-2 p-3"**>  <**form action="/check-out" method="POST"**>  {% **csrf\_token** %}  <**div class="form-group"**>  <**label for=""**>Address</**label**>  <**input type="text" name="address" id="" class="form-control" placeholder=""  aria-describedby="helpId"**>  </**div**>  <**div class="form-group"**>  <**label for=""**>Phone</**label**>  <**input type="text" name="phone" id="" class="form-control" placeholder=""  aria-describedby="helpId"**>  </**div**>  <**input type="submit" class="btn float-right btn-outline-success col-lg-6" value="Check out"**>  </**form**>  </**div**>  </**div**>  </**div**>  </**div**>  </**div**> {% **endblock** %} |

1. Здесь мы создаем представление для проверки перед выходом с именем views/checkout.py.

|  |
| --- |
| **from** django.shortcuts **import** render, redirect **from** django.contrib.auth.hashers **import** check\_password **from** store.models.customer **import** Customer **from** django.views **import** View **from** store.models.product **import** Products **from** store.models.orders **import** Order **class** CheckOut(View):  **def** post(self, request):  address = request.POST.get(**'address'**)  phone = request.POST.get(**'phone'**)  customer = request.session.get(**'customer'**)  cart = request.session.get(**'cart'**)  products = Products.get\_products\_by\_id(list(cart.keys()))  print(address, phone, customer, cart, products)   **for** product **in** products:  print(cart.get(str(product.id)))  order = Order(customer=Customer(id=customer),  product=product,  price=product.price,  address=address,  phone=phone,  quantity=cart.get(str(product.id)))  order.save()  request.session[**'cart'**] = {}  **return** redirect(**'cart'**) |



1. Кроме того, в парке магазина мы создадим пакет промежуточных программ и создадим файл с именем auth.py.

|  |
| --- |
| **from** django.shortcuts **import** redirect  **def** auth\_middleware(get\_response):  *# One-time configuration and initialization.* **def** middleware(request):  print(request.session.get(**'customer'**))  returnUrl = request.META[**'PATH\_INFO'**]  print(request.META[**'PATH\_INFO'**])  **if not** request.session.get(**'customer'**):  **return** redirect(**f'login?return\_url={**returnUrl**}'**)   response = get\_response(request)  **return** response   **return** middleware |

1. Перейдите к файлу urls.py и измените его содержимое следующим образом:

|  |
| --- |
| **from** django.contrib **import** admin **from** django.urls **import** path **from** store.views.home **import** Index, store **from** store.views.signup **import** Signup **from** store.views.login **import** Login, logout **from** store.views.cart **import** Cart **from** store.views.checkout **import** CheckOut **from** store.middlewares.auth **import** auth\_middleware **from** django.conf **import** settings **from** django.conf.urls.static **import** static  urlpatterns = [  path(**''**, Index.as\_view(), name=**'homepage'**),  path(**'store'**, store, name=**'store'**),  path(**'signup'**, Signup.as\_view(), name=**'signup'**),  path(**'login'**, Login.as\_view(), name=**'login'**),  path(**'logout'**, logout, name=**'logout'**),  path(**'cart'**, Cart.as\_view(), name=**'cart'**),  path(**'check-out'**, CheckOut.as\_view(), name=**'checkout'**),  ] + static(settings.MEDIA\_URL, document\_root=settings.MEDIA\_ROOT) |

**Старница покупки**

1. В представлении мы создаем файл с именем orders.py

|  |
| --- |
| **from** django.shortcuts **import** render **from** django.views **import** View **from** store.models.orders **import** Order  **class** OrderView(View):  **def** get(self, request):  customer = request.session.get(**'customer'**)  orders = Order.get\_orders\_by\_customer(customer)  print(orders)  **return** render(request, **'orders.html'**, {**'orders'**: orders}) |

1. В соответствии с этим файлом в файле шаблонов мы создаем файл с именем orders.html

|  |
| --- |
| {% **extends 'base.html'** %} {% **block content** %}  {% **load cart** %}  {% **load custom\_filter** %}  <**div class="container"**>  <**div class="border rounded p-4 m-4"**>  <**p class="display-4 pl-4 ml-4"**>Your Orders</**p**>  <**hr**>  <**table class="table"**>  <**thead**>  <**tr**>  <**th**>Sno.</**th**>  <**th**>Image</**th**>  <**th**>Product</**th**>  <**th**>Date</**th**>  <**th**>Price</**th**>  <**th**>Quantity</**th**>  <**th**>Total</**th**>  <**th**>Status</**th**>  </**tr**>  </**thead**>  <**tbody**>   {% **for order in orders** %}  <**tr**>  <**td**>{{ **forloop**.**counter** }}</**td**>  <**td**><**img style="height**: 80**px**;**" class="rounded-circle" src="**{{ **order**.**product**.**image**.**url** }}**"  alt=""**></**td**>  <**td**>{{ **order**.**product**.**name** }}</**td**>  <**td**>{{ **order**.**date** }}</**td**>  <**td**>{{ **order**.**price**|**currency** }}</**td**>  <**td**>{{ **order**.**quantity** }}</**td**>  <**td**>{{ **order**.**quantity**|**multiply**:**order**.**price**|**currency** }}</**td**>  {% **if order**.**status** %}  <**td**><**small class="badge badge-success"**>Completed</**small**></**td**>  {% **else** %}  <**td**><**small class="badge badge-warning"**>Pending</**small**></**td**>  {% **endif** %}  </**tr**>   {% **endfor** %}    </**tbody**>  </**table**>  </**div**>  </**div**> {% **endblock** %} |

3) В результате файл urls.py принимает следующий вид

|  |
| --- |
| **from** django.contrib **import** admin **from** django.urls **import** path **from** store.views.home **import** Index, store **from** store.views.signup **import** Signup **from** store.views.login **import** Login, logout **from** store.views.cart **import** Cart **from** store.views.checkout **import** CheckOut **from** store.views.orders **import** OrderView **from** store.middlewares.auth **import** auth\_middleware **from** django.conf **import** settings **from** django.conf.urls.static **import** static  urlpatterns = [  path(**''**, Index.as\_view(), name=**'homepage'**),  path(**'store'**, store, name=**'store'**),  path(**'signup'**, Signup.as\_view(), name=**'signup'**),  path(**'login'**, Login.as\_view(), name=**'login'**),  path(**'logout'**, logout, name=**'logout'**),  path(**'cart'**, Cart.as\_view(), name=**'cart'**),  path(**'check-out'**, CheckOut.as\_view(), name=**'checkout'**),  path(**'orders'**, auth\_middleware(OrderView.as\_view()), name=**'orders'**),  ] + static(settings.MEDIA\_URL, document\_root=settings.MEDIA\_ROOT) |

1. Наконец, измените страницу Base.html на следующую форму

|  |
| --- |
| <!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://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"  integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous"**>   <**title**>E-commerce Website</**title**>   <**style**>  **nav**{  **background-color**: **green**;  }  .**one-edge-shadow** {  **box-shadow**: 0 8**px** 2**px** -5**px rgb**(246, 245, 245); } .**display-8**{  **font-weight**: 200;  **font-size**: 30**px**;  **font-colour**: $**teal-200**; }  </**style**> </**head**>  <**body**>  *<!-- navbar -->* <**nav class="one-edge-shadow  navbar navbar-light  sticky-top navbar  navbar-expand-lg  text-white"**>  <**a class="navbar-brand ms-3 text-dark " href="/"**>  <**link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.0/css/all.css" integrity="sha384-lZN37f5QGtY3VHgisS14W3ExzMWZxybE1SJSEsQp9S+oqd12jhcu+A56Ebc1zFSJ" crossorigin="anonymous"**>   <**img src="https://www.freepnglogos.com/uploads/shopping-cart-png/shopping-cart-svg-png-icon-download-28.png" style="width**:50**px**;**height**:50**px**,**color**:**text-light**;**"**>E-Commerce</**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 class="collapse navbar-collapse" id="navbarSupportedContent"**>  <**ul class="navbar-nav mr-auto"**>  <**li class="nav-item active"**>  <**a class="nav-link" href="/store"**>Store <**span class="sr-only"**>(current)</**span**></**a**>  </**li**>   </**ul**>      <**ul class=" navbar-nav my-2 my-lg-0"**>   <**li class="nav-item active"**>  <**a class="nav-link" href="/cart"**>Cart  <**span class="badge badge-success"**>{{**request**.**session**.**cart**.**keys**|**length**}}</**span**>  <**span class="sr-only"**>(current)</**span**></**a**>  </**li**>   {% **if request**.**session**.**customer** %}   <**li class="nav-item active "**>  <**a class="nav-link" href="/orders"**>Orders<**span class="sr-only"**>(current)</**span**></**a**>  </**li**>  <**li class="nav-item active "**>  <**a class="nav-link" href="/logout"**>Logout<**span class="sr-only"**>(current)</**span**></**a**>  </**li**>  {% **else** %}   <**li class="nav-item active"**>  <**a class="nav-link" href="/signup"**>Signup<**span class="sr-only"**>(current)</**span**></**a**>  </**li**>  <**li class="nav-item active"**>  <**a class="nav-link" href="/login"**>Login<**span class="sr-only"**>(current)</**span**></**a**>  </**li**>  {% **endif** %}    </**ul**>    </**div**>  </**nav**>  *<!---->* {% **block content** %}{% **endblock** %}   *<!-- Optional JavaScript -->  <!-- jQuery first, then Popper.js, then Bootstrap JS -->* <**script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"  integrity="sha384-KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"  crossorigin="anonymous"**></**script**>  <**script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"  integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"  crossorigin="anonymous"**></**script**>  <**script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"  integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"  crossorigin="anonymous"**></**script**> </**body**>  </**html**> |