Министерство образования Московской области

Государственное образовательное учреждение высшего образования Московской области

«Государственный гуманитарно-технологический университет»

**Ликино-Дулевский политехнический колледж – филиал ГГТУ**

**КУРСОВОЙ ПРОЕКТ**

Тема: «Разработка интернет-магазина по продаже велосипедов»

МДК.08.01 «Проектирование и разработка интерфейсов пользователя»

**Выполнил:**

Степанов Дмитрий

Студент 3 курса группы ИСП.21.2А

09.02.07 Информационные системы и программирование)

очной формы обучения

**Руководитель:**

Гжегожевский Сергей Владимирович

Оценка

\_\_\_\_\_\_\_\_\_/\_\_\_­­­­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/

Подпись руководителя

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ликино-Дулево

2024 год

**Введение**

Описать сферу продаж. Каждая современная, уважающая себя компания обязана иметь свой веб-магазин либо сайт, благодаря им она привлекает потенциальных покупателей к своей продукции и повышает доход из-за того, что людям даже не нужно выходить из дома для покупки чего-либо, все доставит курьер, либо вещь можно будет забрать прямо в магазине.

CycleStore это интернет магазин, он предназначен для ознакомления клиентов с компанией и ее продукцией, а так-же для продажи велосипедов. Сайт будет интересен как новичкам и любителям велоспорта, так и профи. На сайте можно создавать свой аккаунт для заказа велосипеда.

**1. Разработка системного проекта.**

**1.1. Назначение разработки**

Целью разработки является создание сайта для компании CycleStore, с возможностью просмотра продукции, добавления его в корзину. Также каждый пользователь должен создать свой аккаунт на сайте через регистрацию.

**1.2. Требования к программному продукту**

**1.2.1. Требования к функциональным характеристикам**

Для использования сайта подойдет любой браузер и понадобится доступ в Интернет

Функции:

1. Регистрация
2. Авторизация
3. Добавление/удаление/редактирование товара
4. Заказ товара
5. Поиск товара

**1.2.2. Требования к надежности и безопасности**

Для покупок нужно зарегистрировать/войти в аккаунт

Для регистрации пользователь должен ввести желаемый логин, электронную почту и пароль с его дубликатом. Для безопасности, к паролю выдвигаются следующие требования:

1. Он должен содержать минимум 8 символов
2. Он не должен быть простым или распространенным
3. Он не может состоять из одних цифр

Пароль после регистрации попадает в базу и будет хешироваться, благодаря чему не возможно украсть пароль.

**1.2.3. Требования к составу и параметрам технических средств**

Для разработки использовалась программа PyCharm, т.к. является наиболее удобной для работы с проектом.

**2. Разработка технического проекта**

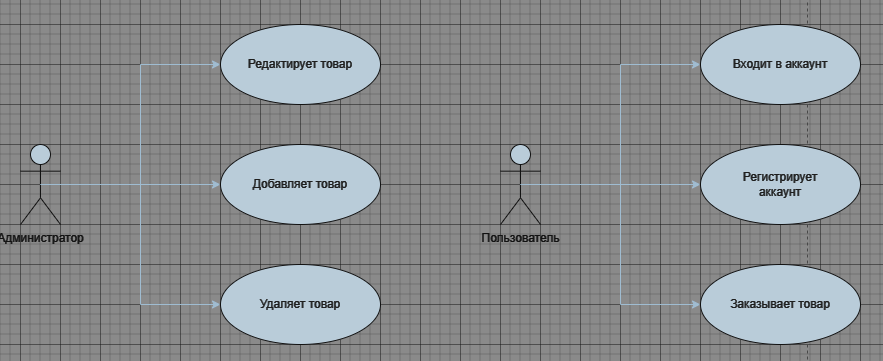
**2.1. Обоснование выбора CASE – средств**

**2.2. Проектирование модели данных**

**2.2.1. Диаграмма прецедентов**

Администратор – управляет сайтом, добавляет/редактирует/удаляет товары из каталога.

Пользователь – регистрирует и входит в аккаунт, заказывает товар.



**2.2.2. Моделирование бизнес-процессов**

1. Регистрация пользователя

2. Авторизация пользователя

3. Заказ товара

4. Добавление товара

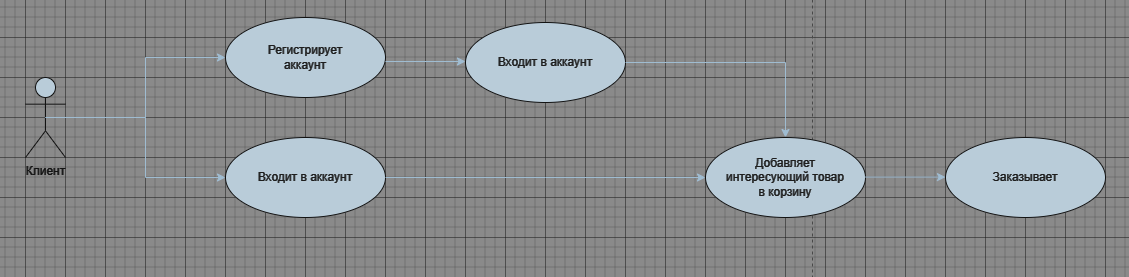
5. Удаление товара

6. Редактирование товара

**2.2.3. Словесный алгоритм бизнес-процессов**

Клиент входит/регистрируется на сайте, в каталоге выбирает нужный ему товар и заказывает чего через корзину.

**2.2.4. Построение диаграммы действий**



**3. Реализация приложения**

**3.1. Обоснование выбора средств разработки**

Вся программа написана в PyCharm, это бесплатная и хорошая программа для написания сайтов.

**3.2. Руководство программиста**

Сайт состоит из 3х основных страниц, на главной странице располагается каталог с товарами, на двух других – информация о компании и т.д. Для управления сайтом нужно авторизоваться под админом (login: admin password: 1234)

* + 1. **Построение моделей и структура БД.**

Корзина (cart)

from django.db import models  
from django.contrib.auth.models import User  
from main.models import Tovar  
  
class CartItem(models.Model):  
 tovar = models.ForeignKey(Tovar, on\_delete=models.CASCADE)  
 quantity = models.PositiveIntegerField(default=0)  
 user = models.ForeignKey(User, on\_delete=models.CASCADE)  
 date\_added = models.DateTimeField(auto\_now\_add=True)  
   
 def \_\_str\_\_(self):  
 return f'{self.quantity} x {self.tovar.cyclename}'

main

class Tovar(models.Model):  
 cyclename = models.CharField(max\_length=100, blank=False, default="")  
 tovarprice = models.FloatField()  
 category = models.ForeignKey(  
 "Categories",  
 related\_name="tovars\_categories",  
 on\_delete=models.CASCADE,  
 null=True,  
 )  
 shipping = models.TextField(max\_length=300, blank=False, default="")  
 tovardescrpt = models.TextField(max\_length=500, blank=False, default="")  
 date\_posted = models.DateTimeField(default=timezone.now, null=True)  
 tovarimage = models.ImageField(  
 upload\_to="img/tovarimages",  
 verbose\_name="Фото товара",  
 default=None,  
 blank=False,  
 )  
  
 def \_\_str\_\_(self):  
 return f"{self.cyclename} {self.category}"  
  
  
  
class Post(models.Model):  
 title = models.CharField(max\_length=100, blank=False)  
 postbody = models.TextField(max\_length=1000)  
  
 def \_\_str\_\_(self):  
 return self.title

**3.2.2. Представления**

Представление cart

def view\_cart(request):  
 if request.user.is\_authenticated:  
 cart\_items = CartItem.objects.filter(user=request.user)  
 total\_price = sum(item.tovar.tovarprice \* item.quantity for item in cart\_items)  
 return render(  
 request,  
 "cart/cart.html",  
 {"cart\_items": cart\_items, "total\_price": total\_price},  
 )  
 else:  
 return render(request, "account/signup.html")

def add\_to\_cart(request, tovar\_id):  
 tovar = Tovar.objects.get(id=tovar\_id)  
 cart\_item, created = CartItem.objects.get\_or\_create(tovar=tovar, user=request.user)  
 cart\_item.quantity += 1  
 cart\_item.save()  
 return redirect("cart:view\_cart")  
  
  
def remove\_from\_cart(request, item\_id):  
 cart\_item = CartItem.objects.get(id=item\_id)  
 cart\_item.delete()  
 return redirect("cart:view\_cart")

Представление orders

def order\_create(request):  
 cart = CartItem.objects.all()  
 if request.method == 'POST':  
 form = OrderCreateForm(request.POST)  
 if form.is\_valid():  
 order = form.save()  
 for item in cart:  
 OrderItem.objects.create(order=order,  
 tovar=item['tovar'],  
 price=item['price'],  
 quantity=item['quantity'])  
 # очистка корзины  
 cart.clear()  
 return render(request, 'orders/order-created.html',  
 {'order': order})  
 else:  
 form = OrderCreateForm  
 return render(request, 'orders/create-order.html',  
 {'cart': cart, 'form': form})

**3.2.3. Шаблоны**

Создан шаблон «base», он имеет свои стили и благодаря нему все страницы имеют одинаковый внешний вид, из-за этого страницы не будут выделяться между собой. Этот шаблон подключен ко всем страницам

**3.2.4. Формы**

Форма добавления товара:

class TovarForm(ModelForm):  
 def \_\_init\_\_(self,\*args,\*\*kwargs):  
 super().\_\_init\_\_(\*args,\*\*kwargs)  
 self.fields['category'].empty\_label = 'Категория не выбрана'  
   
 category = forms.ModelChoiceField(queryset=Categories.objects.all(), empty\_label="Категория", required=True)  
 class Meta:  
 model = Tovar  
 fields = ["cyclename", 'tovarprice', 'category', 'shipping', 'tovardescrpt','tovarimage']  
 widgets = {  
 "cyclename": TextInput(attrs={'placeholder': 'Заголовок товара', 'class': 'form-input'}),  
 "tovarprice": TextInput(attrs={'placeholder': 'Цена', 'class': 'form-input'}),  
 "category": forms.Select(attrs={'placeholder': 'Категория', 'class': 'form-input'}),  
 "shipping": Textarea(attrs={'placeholder': 'Доставка', 'class': 'form-input'}),  
 "tovardescrpt": Textarea(attrs={'placeholder': 'Опишите хаарктеристики товара, его состояние, нюансы', 'class': 'form-input'}),  
 }

**3.3. Руководство пользователя**

Для регистрации/авторизации нужно заполнить форму, чтобы перейти в нее требуется нажать на одну из двух нужных вам кнопок в правом верхнем углу.

Для добавления товара в корзину нажмите на нужный товар, это действие перенесет вас на страницу этого товара, далее нажмите «В корзину».

Для заказа товара перейдите в корзину (нажмите на пользователя) и далее «Заказать».

**4. Тестирование программного продукта**

**4.1. Тестовые сценарии**

См. картинку Тестовые-сценарии ниже. Создать не менее 3 тестовых сценариев.

**4.2. Юнит-тесты в Django**

Описать созданные тесты

**5. Размещение проекта на хостинге.**

Заключение

Список литературы

Приложения

Тестовые-сценарии

