Coupon code

<!-- Bootstrap Modal -->

            <div class="modal" id="successModal" tabindex="-1" aria-labelledby="successModalLabel" aria-hidden="true">

                <div class="modal-dialog">

                    <div class="modal-content">

                        <div class="modal-header">

                            <h5 class="modal-title" id="successModalLabel">Success!</h5>

                            <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>

                        </div>

                        <div class="modal-body">

                            <p>Your coupon has been applied successfully.</p>

                        </div>

                    </div>

                </div>

            </div>

            <!-- coupon -->

            <div class="col-lg-3">

                <div class="card mb-3 border shadow-0">

                    <div class="card-body">

                        <div class="table table-dark">

                            {% for coupon in active\_coupons %}

                            <th>{{coupon.coupon\_code}}</th>

                            {% endfor %}

                        </div>

                        <form method="post">

                            {% csrf\_token %}

                            <div class="form-group">

                                <label class="form-label">Have coupon?</label>

                                <div class="input-group">

                                    <input type="text" class="form-control border" name="coupon\_code"

                                        placeholder="Coupon code" />

                                    <a href="" class="btn btn-light border">Apply</a>

                                </div>

                            </div>

                        </form>

                    </div>

                </div>

class Order(models.Model):

    user = models.ForeignKey(settings.AUTH\_USER\_MODEL, on\_delete=models.CASCADE, related\_name='order\_user', null=True,blank=True)

    address = models.ForeignKey(Address, on\_delete=models.CASCADE, null=True, blank=True)

    created = models.DateTimeField(auto\_now\_add=True, null=True, blank=True)

    updated = models.DateTimeField(auto\_now=True, null=True, blank=True)

    total\_paid = models.DecimalField(max\_digits=10, decimal\_places=2, null=True, blank=True)

    # order\_key = models.CharField(max\_length=200, null=True, blank=True)

    billing\_status = models.CharField(max\_length=10, )

    product = models.ForeignKey(Product,on\_delete=models.CASCADE,null=True,blank=True)

    # New field for order status

    ORDER\_STATUS\_CHOICES = [

        ('confirmed', 'Order Confirmed'),

        ('shipped', 'Shipped'),

        ('out\_for\_delivery', 'Out for Delivery'),

        ('delivered', 'Delivered'),

        # Add more statuses as needed

    ]

    status = models.CharField(max\_length=20, choices=ORDER\_STATUS\_CHOICES, default='confirmed')

    class Meta:

        ordering = ('-created',)

    def str(self):

        return f"{self.created} - {self.status}"

class OrderItem(models.Model):

    order = models.ForeignKey(Order,related\_name='items',on\_delete=models.CASCADE)

    product = models.ForeignKey(Product,related\_name='order\_items',on\_delete=models.CASCADE)

    price = models.DecimalField(max\_digits=10, decimal\_places=2)

    quantity = models.PositiveIntegerField(default=1)

    def str(self):

        return str(self.id)

class MaxValidator:

    pass