Task 2

Model View Controller (MVC) and Model View Template (MVT).

➢ Comparing MVC is Highly coupled and MVT is Loosely coupled:

- MVC in Laravel: In Laravel, the Model, View, and Controller are more tightly connected. The controller has to control both the logic and which view to render. This means that controllers handle a lot of responsibilities in MVC.
- MVT in Django: In Django, the View acts more like a controller, handling both logic and directing which template to render. There is less coupling between these components because the template only deals with presenting the data without knowing where the data comes from.

Model.py:

```python
from django.db import models
from django.contrib.auth.models import User

class Assessment(models.Model):
    title = models.CharField(max_length=255)

class Review(models.Model):
    reviewer = models.ForeignKey(User, on_delete=models.CASCADE, related_name='given_reviews')
    reviewee = models.ForeignKey(User, on_delete=models.CASCADE, related_name='received_reviews')
    review_text = models.TextField()
    score = models.IntegerField()
    assessment = models.ForeignKey(Assessment, on_delete=models.CASCADE)
    created_at = models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)
```

Views.py:

```python
from django.shortcuts import render, redirect
from django.contrib.auth.decorators import login_required
from django.core.exceptions import ObjectDoesNotExist
from .models import Review, Assessment
from django.contrib.auth.models import User
from django.contrib import messages

@login_required
def submit_review(request, assessment_id):
    if request.method == 'POST':
        try:
            reviewee_id = request.POST['reviewee_id']
            review_text = request.POST['review_text']
            score = int(request.POST['score'])
            assessment = Assessment.objects.get(pk=assessment_id)
            # Create and save the review
            review = Review(
                reviewer=request.user,
                reviewee=User.objects.get(pk=reviewee_id),
                review_text=review_text,
                score=score,
                assessment=assessment
            )
            review.save()

            messages.success(request, 'Review submitted successfully.')
            return redirect('assessment_detail', assessment_id=assessment_id)
        except ObjectDoesNotExist:
            messages.error(request, 'Invalid data provided.')
    return render(request, 'submit_review.html')
```

Template (review.html):

```
<h1>Review Details</h1>
<div class="card">
    <div class="card-body">
        <h5 class="card-title">Reviewer: {{ review.reviewer.username }}</h5>
        <h5 class="card-title">Reviewee: {{ review.reviewee.username }}</h5>
        <h5 class="card-title">Assessment: {{ review.assessment.title }}</h5>
        <p class="card-text">Review Text: {{ review.review_text }}</p>
        <p class="card-text">Score: {{ review.score }}</p>
        <p class="card-text">Created At: {{ review.created_at }}</p>
        <p class="card-text">Updated At: {{ review.updated_at }}</p>
        <a href="{% url 'reviews' %}" class="btn btn-primary">Back to Reviews</a>
        <a href="{% url 'edit_review' review.id %}" class="btn btn-warning">Edit Review</a>
    </div>
</div>
```

I agree with the author: In Laravel, the controller is responsible for both the logic and view rendering, tightly connecting the layers and coordinating the data flow between the model and the view.

In contrast, MVT architecture in Django is loosely coupled. The view is more independent, and the model and template don't interact directly, allowing for a cleaner separation of concerns.