TASK-2

Develop a social media platform with features like user profiles, posts, comments, likes, and follows. Use html/css/javascript for frontend, a backend framework like django or express.js, and a database for data storage.

here's a basic outline of how you can develop a social media platform with features like user profiles, posts, comments, likes, and follows using Django for the backend with a PostgreSQL database and HTML/CSS/JavaScript for the frontend:

Backend (Django):

1. **models.py** (inside `profiles` app):

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

class Profile(models.Model):
    user = models.OneToOneField(User, on_delete=models.CASCADE)
    bio = models.TextField(blank=True)
    avatar = models.ImageField(upload_to='avatars/', blank=True)

class Post(models.Model):
    user = models.ForeignKey(User, on_delete=models.CASCADE)
    content = models.TextField()
    created_at = models.DateTimeField(auto_now_add=True)

class Comment(models.Model):
    user = models.ForeignKey(User, on_delete=models.CASCADE)
    post = models.ForeignKey(Post, on_delete=models.CASCADE)
    content = models.TextField()
    created_at = models.DateTimeField(auto_now_add=True)

class Like(models.Model):
    user = models.ForeignKey(User, on_delete=models.CASCADE)
    post = models.ForeignKey(Post, on_delete=models.CASCADE)
```

2. **views.py** (for handling API requests):

```
python
Copy code
from django.shortcuts import render
from django.http import JsonResponse
from .models import Post, Comment, Like

def get_posts(request):
    posts = Post.objects.all().order_by('-created_at')
    data = [{'id': post.id, 'user': post.user.username, 'content':
post.content, 'created_at': post.created_at} for post in posts]
    return JsonResponse(data, safe=False)
```

```
# Implement similar views for comments, likes, and user profiles
```

3. **urls.py** (inside `profiles` app):

```
python
Copy code
from django.urls import path
from . import views

urlpatterns = [
    path('api/posts/', views.get_posts, name='get_posts'),
    # Add URLs for comments, likes, and user profiles
]
```

Frontend (HTML/CSS/JavaScript):

1. **index.html**:

```
html
Copy code
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Social Media Platform</title>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <div id="posts"></div>
    <script src="scripts.js"></script>
</body>
</html>
```

2. **styles.css**:

```
css
Copy code
.post {
    border: 1px solid #ccc;
    margin-bottom: 10px;
    padding: 10px;
}

.comment {
    margin-left: 20px;
    margin-top: 5px;
    font-style: italic;
}
```

3. **scripts.js**:

```
javascript
Copy code
```

```
document.addEventListener('DOMContentLoaded', function() {
    fetch('/api/posts')
        .then(response => response.json())
        .then(data => {
            const postsDiv = document.getElementById('posts');
            data.forEach(post => {
                const postElement = document.createElement('div');
                postElement.className = 'post';
                postElement.innerHTML = `
                    <p><strong>${post.user}</strong> - ${post.created_at}</p>
                    <p>${post.content}</p>
                `;
                postsDiv.appendChild(postElement);
            });
        });
});
```

This is just a starting point. You would need to continue building out your Django backend with views for comments, likes, user profiles, implementing user authentication, etc. Similarly, on the frontend side, you would need to handle user interactions like posting, commenting, liking, and following other users.