CodeAlpha_ Social Media Platform

Creating a social media platform is an extensive project, but here's a breakdown of how you can design it using the technologies you mentioned:

1. Frontend (HTML/CSS/JavaScript)

a. HTML

Structure your pages:

- Login/Register Page
- Home Feed
- User Profile
- Post Details Page

Example Code: Login Page

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-
width, initial-scale=1.0">
    <title>Login - SocialMedia</title>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
```

```
    <div class="login-container">
        <h1>Login</h1>
        <form id="loginForm">
            <label for="email">Email:</label>
            <input type="email" id="email" required>
            <label for="password">Password:</label>
            <input type="password" id="password"
required>
            <button type="submit">Login</button>
        </form>
        <p>Don't have an account? <a
href="/register.html">Register</a></p>
    </div>
</body>
</html>
```

b. CSS

Basic styling:

```
body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    background-color: #f9f9f9;
}

.login-container {
    max-width: 400px;
    margin: 50px auto;
    padding: 20px;
    background: #fff;
    border-radius: 8px;
    box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
}

form {
    display: flex;
    flex-direction: column;
}

input {
    margin-bottom: 15px;
    padding: 10px;
    border: 1px solid #ccc;
    border-radius: 4px;
}
```

c. JavaScript

Handling form submissions:

```
document.getElementById('loginForm').addEventListener
('submit', async (e) => {
    e.preventDefault();
    const email =
document.getElementById('email').value;
    const password =
document.getElementById('password').value;
```

```
    const response = await fetch('/api/login', {
        method: 'POST',
        headers: { 'Content-Type':
'application/json' },
        body: JSON.stringify({ email, password }),
    });

    if (response.ok) {
        window.location.href = '/home.html';
    } else {
        alert('Login failed!');
    }
});
```

2. Backend (Django)

a. Setup Django

1. Install Django: `pip install django`

2. Start a new project: `django-admin startproject socialmedia`
   ```
   cd socialmedia
   python manage.py startapp users
   ```

b. Models

Define user, post, and interaction models.

```
# users/models.py
from django.contrib.auth.models import AbstractUser
from django.db import models

class CustomUser(AbstractUser):
    bio = models.TextField(blank=True)

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

class Like(models.Model):
    user = models.ForeignKey(CustomUser,
on_delete=models.CASCADE)
    post = models.ForeignKey(Post,
on_delete=models.CASCADE, related_name="likes")
    created_at =
models.DateTimeField(auto_now_add=True)
```

c. Views

Create APIs for frontend interaction:

```
from django.shortcuts import get_object_or_404
from django.http import JsonResponse
from django.views.decorators.csrf import csrf_exempt
from .models import Post, CustomUser
```

```
@csrf_exempt
def create_post(request):
    if request.method == 'POST':
        user = get_object_or_404(CustomUser,
id=request.user.id)
        content = request.POST.get('content')
        post = Post.objects.create(user=user,
content=content)
        return JsonResponse({'id': post.id,
'content': post.content, 'user': user.username})
```

d. URLs

```
from django.urls import path
from . import views

urlpatterns = [
    path('api/create-post', views.create_post,
name='create-post'),
]
```

3. Database (SQLite for Development)

Database Configuration

Django uses SQLite by default, so no additional setup is needed for development. Run migrations:

```
python manage.py makemigrations
python manage.py migrate
```

4. Connecting Frontend to Backend

Ensure that the frontend JavaScript fetches data from Django APIs using `fetch` or Axios.

Example Fetch:

```
async function createPost(content) {
    const response = await fetch('/api/create-post',
{
        method: 'POST',
        headers: { 'Content-Type':
'application/json' },
        body: JSON.stringify({ content }),
    });

    if (response.ok) {
        const data = await response.json();
        console.log('Post created:', data);
    } else {
        console.error('Error creating post');
    }
}
```

5. Source Code Repository

You can store the full project on GitHub or a similar platform. Use commands like:

```
git init
git add .
git commit -m "Initial commit"
git branch -M main
git remote add origin https://github.com/yourusername/socialmedia-platform.git
git push -u origin main
```