TASK 2

SOCAIL NETWORK PLATFORM

HTML CODE:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>Mini Social Network</title>

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet" />

<link rel="stylesheet" href="styles.css" />

</head>

<body>

<!-- Navbar -->

<nav class="navbar navbar-expand-lg navbar-dark bg-dark px-4">

<a class="navbar-brand" href="#">SocialNet</a>

<div class="ms-auto">

<button class="btn btn-outline-light">Login</button>

<button class="btn btn-outline-light">Register</button>

</div>

</nav>

<!-- Create Post Section -->

<div class="container mt-4">

<div class="card mb-4 shadow-sm">

<div class="card-body">

<h5 class="card-title text-primary">Create a Post</h5>

<textarea id="newPostContent" class="form-control mb-2" rows="3" placeholder="What's on your mind, Aarushi?"></textarea>

<input type="file" id="newPostImage" class="form-control mb-2" accept="image/\*" />

<button class="btn btn-success" onclick="createPost()">Post</button>

</div>

</div>

<!-- Follow a User -->

<div class="card shadow-sm mb-4">

<div class="card-body">

<h5 class="card-title text-primary">Follow a User</h5>

<input type="text" id="targetUserId" class="form-control mb-2" placeholder="Enter User ID to follow" />

<button class="btn btn-outline-primary" onclick="followUser()">Follow</button>

</div>

</div>

<!-- Feed Section -->

<div id="feed"></div>

</div>

<script src="script.js"></script>

</body>

</html>

STYLE.CSS CODE:

body {

background: linear-gradient(to right, #ffecd2, #fcb69f);

font-family: 'Poppins', sans-serif;

margin: 0;

padding: 0;

color: #333;

}

.navbar {

background: linear-gradient(to right, #667eea, #764ba2);

color: white;

box-shadow: 0 4px 10px rgba(0,0,0,0.1);

}

.navbar-brand {

font-weight: 700;

font-size: 1.6rem;

color: #fff;

}

.navbar .btn {

margin-left: 10px;

border-radius: 20px;

background-color: #fff;

color: #764ba2;

font-weight: 500;

}

.card-profile {

background: linear-gradient(to bottom right, #fdfbfb, #ebedee);

border-radius: 15px;

box-shadow: 0 6px 12px rgba(0,0,0,0.08);

overflow: hidden;

transition: transform 0.3s ease;

}

.card-profile:hover {

transform: scale(1.02);

}

.card-profile img {

width: 100%;

height: 180px;

object-fit: cover;

border-bottom: 4px solid #667eea;

}

.card-profile .card-body {

text-align: center;

padding: 15px;

}

.card-profile h5 {

color: #667eea;

font-weight: 600;

}

.card-profile p {

font-size: 0.95rem;

color: #555;

}

.card-profile .btn {

background-color: #667eea;

color: #fff;

border-radius: 20px;

font-weight: 500;

}

.card-post {

background: #fff;

border-radius: 12px;

margin-bottom: 20px;

box-shadow: 0 4px 10px rgba(0,0,0,0.05);

transition: box-shadow 0.3s ease;

}

.card-post:hover {

box-shadow: 0 6px 16px rgba(0,0,0,0.1);

}

.card-post .card-title {

color: #ff6f61;

font-weight: 600;

}

.card-post .card-text {

color: #444;

}

.btn-like {

background-color: #ffe0e0;

color: #ff6f61;

border-radius: 20px;

font-weight: 500;

}

.btn-comment {

background-color: #e0f7fa;

color: #007bff;

border-radius: 20px;

font-weight: 500;

}

.comment-box {

background-color: #f9f9f9;

border-left: 4px solid #ff6f61;

padding: 10px;

border-radius: 8px;

margin-top: 10px;

}

.comment-box p {

margin-bottom: 5px;

font-size: 0.9rem;

color: #444;

}

.comment-box input {

border-radius: 20px;

border: 1px solid #ccc;

padding: 6px 12px;

font-size: 0.85rem;

margin-top: 8px;

}

@media (max-width: 768px) {

.card-profile {

margin-bottom: 20px;

}

.navbar-brand {

font-size: 1.3rem;

}

}

SCRIPT.JS CODE:

const posts = [

{

id: 1,

author: "Aarushi",

content: "Excited to launch my new UI design project!",

likes: 12,

comments: ["Looks amazing!", "Can't wait to see it live."]

},

{

id: 2,

author: "Vidhika",

content: "Just finished a backend sprint using Django!",

likes: 8,

comments: ["Nice!", "Share your repo please."]

},

{

id: 3,

author: "Avni",

content: "Just finished a backend sprint using Django!",

likes: 8,

comments: ["Nice!", "Share your repo please."]

}

];

function createPostSection() {

const createCard = document.createElement("div");

createCard.className = "card mb-4 shadow-sm";

createCard.innerHTML = `

<div class="card-body">

<h5 class="card-title text-primary">Create a Post</h5>

<textarea id="newPostContent" class="form-control mb-2" rows="3" placeholder="What's on your mind?"></textarea>

<button class="btn btn-success" onclick="createPost()">Post</button>

</div>

`;

return createCard;

}

function renderFeed() {

const feed = document.getElementById("feed");

feed.innerHTML = "";

createPostSection();

posts.forEach(post => {

const postCard = document.createElement("div");

postCard.className = "card card-post";

postCard.innerHTML = `

<div class="card-body">

<h5 class="card-title">${post.author}</h5>

<input type="text" class="form-control form-control-sm mb-2" placeholder="Write a comment..." onkeypress="addComment(event, ${post.id})" />

<p class="card-text">${post.content}</p>

<button class="btn btn-like btn-sm me-2" onclick="likePost(${post.id})">👍 ${post.likes}</button>

<button class="btn btn-comment btn-sm" onclick="toggleComments(${post.id})">💬 ${post.comments.length}</button>

<div class="comment-box" id="comments-${post.id}" style="display:block;">

${post.comments.map(c => `<p class="mb-1">${c}</p>`).join("")}

</div>

</div>

`;

feed.appendChild(postCard);

});

}

function likePost(id) {

const post = posts.find(p => p.id === id);

post.likes++;

renderFeed();

}

function toggleComments(id) {

const box = document.getElementById(`comments-${id}`);

box.style.display = box.style.display === "none" ? "block" : "none";

}

function addComment(event, id) {

if (event.key === "Enter") {

const commentText = event.target.value.trim();

if (commentText) {

const post = posts.find(p => p.id === id);

post.comments.push(commentText);

event.target.value = "";

renderFeed();

}

}

}

function createPost() {

const content = document.getElementById("newPostContent").value.trim();

if (content) {

const newPost = {

id: posts.length + 1,

author: "Aarushi",

content: content,

likes: 0,

comments: []

};

posts.unshift(newPost);

document.getElementById("newPostContent").value = "";

renderFeed();

}

}

renderFeed();

SERVER.JS CODE:

const express = require('express');

const mongoose = require('mongoose');

const cors = require('cors');

require('dotenv').config();

const app = express();

app.use(cors());

app.use(express.json());

mongoose.connect('mongodb://localhost:27017/socialApp', {

useNewUrlParser: true,

useUnifiedTopology: true,

})

.then(() => console.log('✅ MongoDB connected'))

.catch(err => console.error('❌ MongoDB error:', err));

const userSchema = new mongoose.Schema({

username: { type: String, required: true, unique: true },

bio: String,

followers: [{ type: mongoose.Schema.Types.ObjectId, ref: 'User' }],

following: [{ type: mongoose.Schema.Types.ObjectId, ref: 'User' }]

});

const postSchema = new mongoose.Schema({

author: { type: mongoose.Schema.Types.ObjectId, ref: 'User', required: true },

content: String,

image: String,

likes: [{ type: mongoose.Schema.Types.ObjectId, ref: 'User' }],

createdAt: { type: Date, default: Date.now }

});

const commentSchema = new mongoose.Schema({

post: { type: mongoose.Schema.Types.ObjectId, ref: 'Post', required: true },

author: { type: mongoose.Schema.Types.ObjectId, ref: 'User', required: true },

text: String,

createdAt: { type: Date, default: Date.now }

});

const User = mongoose.model('User', userSchema);

const Post = mongoose.model('Post', postSchema);

const Comment = mongoose.model('Comment', commentSchema);

app.post('/api/users', async (req, res) => {

try {

const user = new User(req.body);

const saved = await user.save();

res.status(201).json(saved);

} catch (err) {

res.status(400).json({ error: err.message });

}

});

app.post('/api/users/:id/follow', async (req, res) => {

try {

const follower = await User.findById(req.body.followerId);

const target = await User.findById(req.params.id);

if (!follower || !target) return res.status(404).json({ error: 'User not found' });

if (!target.followers.includes(follower.\_id)) {

target