**AFRICHAMA PROJECT STRUCTURE**

africhama-app/

├── frontend/

│ ├── public/

│ │ ├── index.html

│ │ ├── favicon.ico

│ │ ├── manifest.json

│ │ ├── robots.txt

│ │ ├── images/

│ │ │ ├── slide1-desktop.jpg

│ │ │ ├── slide1-mobile.jpg

│ │ │ ├── slide2-desktop.jpg

│ │ │ ├── slide2-mobile.jpg

│ │ │ ├── slide3-desktop.jpg

│ │ │ └── slide3-mobile.jpg

│ │ └── icons/

│ │ ├── education.svg

│ │ ├── network.svg

│ │ └── gift.svg

│ ├── src/

│ │ ├── components/

│ │ │ ├── Home.js

│ │ │ ├── Home.module.css

│ │ │ ├── Register.js

│ │ │ ├── Register.module.css

│ │ │ ├── Login.js

│ │ │ ├── Login.module.css

│ │ │ ├── Profile.js

│ │ │ ├── Profile.module.css

│ │ │ ├── NetworkAndUpgrade.js

│ │ │ ├── NetworkAndUpgrade.module.css

│ │ │ ├── GiftSystem.js

│ │ │ ├── GiftSystem.module.css

│ │ │ ├── SubscriptionMaintenance.js

│ │ │ ├── SubscriptionMaintenance.module.css

│ │ │ ├── Support.js

│ │ │ ├── Support.module.css

│ │ │ ├── Layout.js

│ │ │ └── Layout.module.css

│ │ ├── contexts/

│ │ │ └── AuthContext.js

│ │ ├── utils/

│ │ │ └── api.js

│ │ ├── App.js

│ │ ├── index.js

│ │ └── index.css

│ ├── package.json

│ ├── package-lock.json

│ └── .env

├── backend/

│ ├── api/

│ │ ├── register.php

│ │ ├── login.php

│ │ ├── profile.php

│ │ ├── network.php

│ │ ├── gift.php

│ │ ├── subscription.php

│ │ └── support.php

│ ├── config.php

│ ├── .htaccess

│ └── php\_errors.log

└── README.md

**AFRICHAMA CODES**

frontend

/public/index.html

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="utf-8" />

    <link rel="icon" href="%PUBLIC\_URL%/favicon.ico" />

    <meta name="viewport" content="width=device-width, initial-scale=1" />

    <meta name="theme-color" content="#2ecc71" />

    <meta

      name="description"

      content="Africhama - Empowering Success, Empowering Humanity"

    />

    <link rel="apple-touch-icon" href="%PUBLIC\_URL%/logo192.png" />

    <link rel="manifest" href="%PUBLIC\_URL%/manifest.json" />

    <title>Africhama</title>

  </head>

  <body>

    <noscript>You need to enable JavaScript to run this app.</noscript>

    <div id="root"></div>

  </body>

</html>

/public/manifest.json

{

    "short\_name": "Africhama",

    "name": "Africhama - Empowering Success",

    "icons": [

      {

        "src": "favicon.ico",

        "sizes": "64x64 32x32 24x24 16x16",

        "type": "image/x-icon"

      },

      {

        "src": "logo192.png",

        "type": "image/png",

        "sizes": "192x192"

      },

      {

        "src": "logo512.png",

        "type": "image/png",

        "sizes": "512x512"

      }

    ],

    "start\_url": ".",

    "display": "standalone",

    "theme\_color": "#000000",

    "background\_color": "#ffffff"

  }

/src/components/contexts/AuthContext.js

import React, { createContext, useContext, useEffect } from 'react';

import { useDispatch, useSelector } from 'react-redux';

import { loginSuccess, logout } from '../store/authSlice';

const AuthContext = createContext();

export const AuthProvider = ({ children }) => {

  const dispatch = useDispatch();

  const { user, token } = useSelector(state => state.auth);

  useEffect(() => {

    const storedUser = localStorage.getItem('user');

    const storedToken = localStorage.getItem('token');

    if (storedUser && storedToken) {

      dispatch(loginSuccess({ user: JSON.parse(storedUser), token: storedToken }));

    }

  }, [dispatch]);

  const login = (userData, token) => {

    localStorage.setItem('user', JSON.stringify(userData));

    localStorage.setItem('token', token);

    dispatch(loginSuccess({ user: userData, token }));

  };

  const logoutUser = () => {

    localStorage.removeItem('user');

    localStorage.removeItem('token');

    dispatch(logout());

  };

  return (

    <AuthContext.Provider value={{ user, token, login, logout: logoutUser }}>

      {children}

    </AuthContext.Provider>

  );

};

export const useAuth = () => {

  const context = useContext(AuthContext);

  if (!context) {

    throw new Error('useAuth must be used within an AuthProvider');

  }

  return context;

};

/src/components/Alert.js

import React from 'react';

import styles from './Alert.module.css';

const Alert = ({ message, type }) => (

  <div className={`${styles.alert} ${styles[type]}`}>

    {message}

  </div>

);

export default Alert;

/src/components/Alert.module.css

.alert {

    padding: 10px 15px;

    margin-bottom: 15px;

    border-radius: 4px;

    font-weight: bold;

  }

  .error {

    background-color: #f8d7da;

    color: #721c24;

    border: 1px solid #f5c6cb;

  }

  .success {

    background-color: #d4edda;

    color: #155724;

    border: 1px solid #c3e6cb;

  }

/src/components/GiftSystem.js

import React, { useState, useEffect } from 'react';

import axios from 'axios';

import styles from './GiftSystem.module.css';

const GiftSystem = () => {

  const [giftData, setGiftData] = useState({

    receiverCode: '',

    amount: '',

  });

  const [gifts, setGifts] = useState({ sent: [], received: [] });

  const [error, setError] = useState('');

  const [success, setSuccess] = useState('');

  useEffect(() => {

    fetchGifts();

  }, []);

  const fetchGifts = async () => {

    try {

      const response = await axios.get(`${process.env.REACT\_APP\_API\_URL}/gifts`, {

        headers: { Authorization: `Bearer ${localStorage.getItem('token')}` }

      });

      setGifts(response.data);

    } catch (error) {

      setError('Failed to fetch gifts');

    }

  };

  const handleChange = (e) => {

    setGiftData({ ...giftData, [e.target.name]: e.target.value });

  };

  const handleSubmit = async (e) => {

    e.preventDefault();

    setError('');

    setSuccess('');

    try {

      await axios.post(`${process.env.REACT\_APP\_API\_URL}/gifts/send`, giftData, {

        headers: { Authorization: `Bearer ${localStorage.getItem('token')}` }

      });

      setSuccess('Gift sent successfully!');

      setGiftData({ receiverCode: '', amount: '' });

      fetchGifts();

    } catch (error) {

      setError(error.response?.data?.message || 'Failed to send gift');

    }

  };

  return (

    <div className={styles.giftSystem}>

      <h1 className={styles.title}>Africhama Gift System</h1>

      <div className={styles.sendGiftSection}>

        <h2>Send a Gift</h2>

        {error && <p className={styles.error}>{error}</p>}

        {success && <p className={styles.success}>{success}</p>}

        <form onSubmit={handleSubmit} className={styles.giftForm}>

          <div className={styles.formGroup}>

            <label htmlFor="receiverCode">Receiver's Member Code</label>

            <input

              type="text"

              id="receiverCode"

              name="receiverCode"

              value={giftData.receiverCode}

              onChange={handleChange}

              required

            />

          </div>

          <div className={styles.formGroup}>

            <label htmlFor="amount">Amount (USD)</label>

            <input

              type="number"

              id="amount"

              name="amount"

              value={giftData.amount}

              onChange={handleChange}

              required

              min="1"

            />

          </div>

          <button type="submit" className={styles.sendButton}>Send Gift</button>

        </form>

      </div>

      <div className={styles.giftHistory}>

        <h2>Gift History</h2>

        <div className={styles.giftTables}>

          <div className={styles.sentGifts}>

            <h3>Sent Gifts</h3>

            <table>

              <thead>

                <tr>

                  <th>Receiver</th>

                  <th>Amount</th>

                  <th>Date</th>

                </tr>

              </thead>

              <tbody>

                {gifts.sent.map((gift, index) => (

                  <tr key={index}>

                    <td>{gift.receiverCode}</td>

                    <td>${gift.amount}</td>

                    <td>{new Date(gift.date).toLocaleDateString()}</td>

                  </tr>

                ))}

              </tbody>

            </table>

          </div>

          <div className={styles.receivedGifts}>

            <h3>Received Gifts</h3>

            <table>

              <thead>

                <tr>

                  <th>Sender</th>

                  <th>Amount</th>

                  <th>Date</th>

                </tr>

              </thead>

              <tbody>

                {gifts.received.map((gift, index) => (

                  <tr key={index}>

                    <td>{gift.senderCode}</td>

                    <td>${gift.amount}</td>

                    <td>{new Date(gift.date).toLocaleDateString()}</td>

                  </tr>

                ))}

              </tbody>

            </table>

          </div>

        </div>

      </div>

    </div>

  );

};

export default GiftSystem;

/src/components/GiftSystem.module.css

.giftSystem {

    max-width: 1200px;

    margin: 0 auto;

    padding: 2rem;

    font-family: Arial, sans-serif;

  }

  .title {

    color: #2ecc71;

    text-align: center;

    margin-bottom: 2rem;

  }

  .sendGiftSection {

    background-color: #fff;

    border-radius: 8px;

    padding: 2rem;

    box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

    margin-bottom: 2rem;

  }

  .sendGiftSection h2 {

    color: #2ecc71;

    margin-bottom: 1rem;

  }

  .giftForm {

    display: flex;

    flex-direction: column;

  }

  .formGroup {

    margin-bottom: 1rem;

  }

  .formGroup label {

    display: block;

    margin-bottom: 0.5rem;

    color: #333;

    font-weight: bold;

  }

  .formGroup input {

    width: 100%;

    padding: 0.75rem;

    border: 1px solid #ddd;

    border-radius: 4px;

    font-size: 1rem;

  }

  .sendButton {

    background-color: #2ecc71;

    color: white;

    border: none;

    padding: 0.75rem 1rem;

    border-radius: 4px;

    font-size: 1rem;

    cursor: pointer;

    transition: background-color 0.3s ease;

  }

  .sendButton:hover {

    background-color: #27ae60;

  }

  .error {

    color: #e74c3c;

    margin-bottom: 1rem;

  }

  .success {

    color: #2ecc71;

    margin-bottom: 1rem;

  }

  .giftHistory {

    background-color: #fff;

    border-radius: 8px;

    padding: 2rem;

    box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

  }

  .giftHistory h2 {

    color: #2ecc71;

    margin-bottom: 1rem;

  }

  .giftTables {

    display: flex;

    justify-content: space-between;

    flex-wrap: wrap;

  }

  .sentGifts, .receivedGifts {

    width: 48%;

  }

  .sentGifts h3, .receivedGifts h3 {

    color: #333;

    margin-bottom: 1rem;

  }

  table {

    width: 100%;

    border-collapse: collapse;

  }

  th, td {

    text-align: left;

    padding: 0.5rem;

    border-bottom: 1px solid #ddd;

  }

  th {

    background-color: #f2f2f2;

    font-weight: bold;

  }

  tr:hover {

    background-color: #f5f5f5;

  }

  @media (max-width: 768px) {

    .giftTables {

      flex-direction: column;

    }

    .sentGifts, .receivedGifts {

      width: 100%;

      margin-bottom: 2rem;

    }

  }

/src/components/Home.js

import React from 'react';

import { Link } from 'react-router-dom';

import Slider from 'react-slick';

import "slick-carousel/slick/slick.css";

import "slick-carousel/slick/slick-theme.css";

import styles from './Home.module.css';

const Home = () => {

  const sliderSettings = {

    dots: true,

    infinite: true,

    speed: 500,

    slidesToShow: 1,

    slidesToScroll: 1,

    autoplay: true,

    autoplaySpeed: 5000,

    responsive: [

      {

        breakpoint: 768,

        settings: {

          arrows: false

        }

      }

    ]

  };

  return (

    <div className={styles.home}>

      <section className={styles.hero}>

        <div className={styles.heroContent}>

          <h1>Welcome to Africhama</h1>

          <p>Empowering Success, Empowering Humanity</p>

          <Link to="/register" className={styles.ctaButton}>Join Now</Link>

        </div>

      </section>

      <section className={styles.slider}>

        <Slider {...sliderSettings}>

          <div className={styles.slide}>

            <picture>

              <source media="(max-width: 768px)" srcSet="/images/slide1-mobile.jpg" />

              <img src="/images/slide1-desktop.jpg" alt="Empowering Communities" />

            </picture>

            <div className={styles.slideCaption}>Empowering Communities</div>

          </div>

          <div className={styles.slide}>

            <picture>

              <source media="(max-width: 768px)" srcSet="/images/slide2-mobile.jpg" />

              <img src="/images/slide2-desktop.jpg" alt="Building Networks" />

            </picture>

            <div className={styles.slideCaption}>Building Networks</div>

          </div>

          <div className={styles.slide}>

            <picture>

              <source media="(max-width: 768px)" srcSet="/images/slide3-mobile.jpg" />

              <img src="/images/slide3-desktop.jpg" alt="Fostering Growth" />

            </picture>

            <div className={styles.slideCaption}>Fostering Growth</div>

          </div>

        </Slider>

      </section>

      <section className={styles.features}>

        <h2>Why Choose Africhama?</h2>

        <div className={styles.featureGrid}>

          <div className={styles.featureItem}>

            <img src="/icons/education.svg" alt="Education" />

            <h3>Free Educational Materials</h3>

            <p>Access a wide range of free downloadable content on entrepreneurship, health, nature, and life skills.</p>

          </div>

          <div className={styles.featureItem}>

            <img src="/icons/network.svg" alt="Network" />

            <h3>Networking Opportunities</h3>

            <p>Connect with like-minded individuals and grow your professional network across Africa and beyond.</p>

          </div>

          <div className={styles.featureItem}>

            <img src="/icons/gift.svg" alt="Gift" />

            <h3>Unique Gift System</h3>

            <p>Participate in our innovative gift donation system and support fellow members in their journey to success.</p>

          </div>

        </div>

      </section>

      <section className={styles.about}>

        <h2>About Africhama</h2>

        <p>Africhama is an online platform dedicated to empowering individuals through knowledge sharing and community support. Our name combines "Afri-" from Africa and "Chama," a Swahili word embodying the principle of collective success.</p>

        <p>We believe in the power of collaboration and mutual support to drive personal and professional growth across the African continent and beyond.</p>

      </section>

      <section className={styles.cta}>

        <h2>Ready to Start Your Journey?</h2>

        <p>Join Africhama today and take the first step towards empowering your success and contributing to a brighter future for all.</p>

        <Link to="/register" className={styles.ctaButton}>Get Started</Link>

      </section>

    </div>

  );

};

export default Home;

/src/components/[Home.module.css](http://Home.module.css)

.home {

  color: #333;

}

.hero {

  background-color: #2ecc71;

  color: white;

  padding: 4rem 1rem;

  text-align: center;

}

.heroContent {

  max-width: 800px;

  margin: 0 auto;

}

.hero h1 {

  font-size: 3rem;

  margin-bottom: 1rem;

}

.hero p {

  font-size: 1.2rem;

  margin-bottom: 2rem;

}

.ctaButton {

  display: inline-block;

  background-color: white;

  color: #2ecc71;

  padding: 0.75rem 1.5rem;

  border-radius: 4px;

  text-decoration: none;

  font-weight: bold;

  transition: background-color 0.3s ease;

}

.ctaButton:hover {

  background-color: #f0f0f0;

}

.slider {

  margin: 2rem 0;

}

.slide {

  position: relative;

}

.slide img {

  width: 100%;

  height: auto;

  object-fit: cover;

}

.slideCaption {

  position: absolute;

  bottom: 20px;

  left: 20px;

  background-color: rgba(46, 204, 113, 0.8);

  color: white;

  padding: 10px 20px;

  border-radius: 4px;

  font-size: 1.2rem;

}

.features, .about {

  padding: 4rem 1rem;

}

.features {

  background-color: #f9f9f9;

}

.features h2, .about h2 {

  text-align: center;

  color: #2ecc71;

  margin-bottom: 2rem;

  font-size: 2.5rem;

}

.featureGrid {

  display: grid;

  grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));

  gap: 2rem;

  max-width: 1200px;

  margin: 0 auto;

}

.featureItem {

  text-align: center;

  padding: 1.5rem;

  background-color: white;

  border-radius: 8px;

  box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);

  transition: transform 0.3s ease;

}

.featureItem:hover {

  transform: translateY(-5px);

}

.featureItem img {

  width: 64px;

  height: 64px;

  margin-bottom: 1rem;

}

.featureItem h3 {

  color: #2ecc71;

  margin-bottom: 0.5rem;

  font-size: 1.5rem;

}

.about {

  max-width: 800px;

  margin: 0 auto;

  text-align: center;

}

.about p {

  margin-bottom: 1rem;

  font-size: 1.1rem;

  line-height: 1.6;

}

.cta {

  background-color: #2ecc71;

  color: white;

  text-align: center;

  padding: 4rem 1rem;

}

.cta h2 {

  font-size: 2.5rem;

  margin-bottom: 1rem;

}

.cta p {

  margin-bottom: 2rem;

  font-size: 1.1rem;

}

@media (max-width: 768px) {

  .hero h1 {

    font-size: 2.5rem;

  }

  .hero p {

    font-size: 1rem;

  }

  .slideCaption {

    font-size: 1rem;

    padding: 8px 16px;

  }

  .features h2, .about h2, .cta h2 {

    font-size: 2rem;

  }

  .featureItem h3 {

    font-size: 1.3rem;

  }

  .about p, .cta p {

    font-size: 1rem;

  }

}

@media (max-width: 480px) {

  .hero h1 {

    font-size: 2rem;

  }

  .ctaButton {

    padding: 0.6rem 1.2rem;

    font-size: 0.9rem;

  }

  .slideCaption {

    font-size: 0.9rem;

    padding: 6px 12px;

  }

  .features h2, .about h2, .cta h2 {

    font-size: 1.8rem;

  }

  .featureItem {

    padding: 1rem;

  }

  .featureItem img {

    width: 48px;

    height: 48px;

  }

  .featureItem h3 {

    font-size: 1.2rem;

  }

}

/src/components/Layout.js

import React from 'react';

import { Link } from 'react-router-dom';

import styles from './Layout.module.css';

const Layout = ({ children }) => {

  return (

    <div className={styles.layout}>

      <header className={styles.header}>

        <nav className={styles.nav}>

          <Link to="/" className={styles.logo}>

            <img src="/logo.png" alt="Africhama Logo" />

          </Link>

          <ul className={styles.navLinks}>

            <li><Link to="/">Home</Link></li>

            <li><Link to="/register">Register</Link></li>

            <li><Link to="/login">Login</Link></li>

            <li><Link to="/profile">Profile</Link></li>

            <li><Link to="/network">Network</Link></li>

            <li><Link to="/gift-system">Gift System</Link></li>

            <li><Link to="/subscription">Subscription</Link></li>

            <li><Link to="/support">Support</Link></li>

          </ul>

        </nav>

      </header>

      <main className={styles.main}>{children}</main>

      <footer className={styles.footer}>

        <p>&copy; 2023 Africhama. All rights reserved.</p>

      </footer>

    </div>

  );

};

export default Layout;

/src/components/Layout.module.css

.layout {

  display: flex;

  flex-direction: column;

  min-height: 100vh;

}

.header {

  background-color: #2ecc71;

  padding: 1rem 0;

}

.nav {

  display: flex;

  justify-content: space-between;

  align-items: center;

  max-width: 1200px;

  margin: 0 auto;

  padding: 0 1rem;

}

.logo {

  color: white;

  font-size: 1.5rem;

  font-weight: bold;

  text-decoration: none;

}

.navLinks {

  display: flex;

  list-style-type: none;

}

.navLinks li {

  margin-left: 1rem;

}

.navLinks a {

  color: white;

  text-decoration: none;

}

.main {

  flex-grow: 1;

  max-width: 1200px;

  margin: 0 auto;

  padding: 2rem 1rem;

}

.footer {

  background-color: #2ecc71;

  color: white;

  text-align: center;

  padding: 1rem 0;

}

/src/components/LoadingSpinner.js

import React from 'react';

import styles from './LoadingSpinner.module.css';

const LoadingSpinner = () => {

  return (

    <div className={styles.spinnerOverlay}>

      <div className={styles.spinner}></div>

    </div>

  );

};

export default LoadingSpinner;

/src/components/LoadingSpinner.module.css

.spinnerOverlay {

    position: fixed;

    top: 0;

    left: 0;

    width: 100%;

    height: 100%;

    background-color: rgba(0, 0, 0, 0.5);

    display: flex;

    justify-content: center;

    align-items: center;

    z-index: 1000;

  }

  .spinner {

    width: 50px;

    height: 50px;

    border: 5px solid #f3f3f3;

    border-top: 5px solid #2ecc71;

    border-radius: 50%;

    animation: spin 1s linear infinite;

  }

  @keyframes spin {

    0% { transform: rotate(0deg); }

    100% { transform: rotate(360deg); }

  }

/src/components/Login.js

import React, { useState } from 'react';

import { useNavigate, Link } from 'react-router-dom';

import axios from 'axios';

import styles from './Login.module.css';

const Login = () => {

  const [formData, setFormData] = useState({

    username: '',

    password: '',

  });

  const [error, setError] = useState('');

  const navigate = useNavigate();

  const handleChange = (e) => {

    setFormData({ ...formData, [e.target.name]: e.target.value });

  };

  const handleSubmit = async (e) => {

    e.preventDefault();

    setError('');

    try {

      const response = await axios.post(`${process.env.REACT\_APP\_API\_URL}/login`, formData);

      if (response.data.success) {

        // Here you would typically store the token in localStorage or a state management solution

        console.log('Login successful:', response.data);

        navigate('/profile');

      } else {

        setError(response.data.message || 'Login failed');

      }

    } catch (error) {

      console.error('Login error:', error);

      setError(error.response?.data?.message || 'An error occurred during login');

    }

  };

  return (

    <div className={styles.loginContainer}>

      <div className={styles.loginCard}>

        <h2 className={styles.loginTitle}>Welcome Back to Africhama</h2>

        {error && <p className={styles.error}>{error}</p>}

        <form onSubmit={handleSubmit} className={styles.loginForm}>

          <div className={styles.formGroup}>

            <label htmlFor="username">Username</label>

            <input

              type="text"

              id="username"

              name="username"

              value={formData.username}

              onChange={handleChange}

              required

            />

          </div>

          <div className={styles.formGroup}>

            <label htmlFor="password">Password</label>

            <input

              type="password"

              id="password"

              name="password"

              value={formData.password}

              onChange={handleChange}

              required

            />

          </div>

          <button type="submit" className={styles.submitButton}>Login</button>

        </form>

        <div className={styles.links}>

          <Link to="/forgot-password" className={styles.forgotPassword}>Forgot Password?</Link>

          <p className={styles.registerLink}>

            Don't have an account? <Link to="/register">Register here</Link>

          </p>

        </div>

      </div>

    </div>

  );

};

export default Login;

/src/components/Login.module.css

.loginContainer {

    display: flex;

    justify-content: center;

    align-items: center;

    min-height: 100vh;

    background-color: #f0f0f0;

    padding: 2rem;

  }

  .loginCard {

    background-color: white;

    border-radius: 8px;

    box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

    padding: 2rem;

    width: 100%;

    max-width: 400px;

  }

  .loginTitle {

    color: #2ecc71;

    text-align: center;

    margin-bottom: 1.5rem;

    font-size: 2rem;

  }

  .loginForm {

    display: flex;

    flex-direction: column;

  }

  .formGroup {

    margin-bottom: 1rem;

  }

  .formGroup label {

    display: block;

    margin-bottom: 0.5rem;

    color: #333;

    font-weight: bold;

  }

  .formGroup input {

    width: 100%;

    padding: 0.75rem;

    border: 1px solid #ddd;

    border-radius: 4px;

    font-size: 1rem;

    transition: border-color 0.3s ease;

  }

  .formGroup input:focus {

    outline: none;

    border-color: #2ecc71;

  }

  .submitButton {

    background-color: #2ecc71;

    color: white;

    border: none;

    padding: 1rem;

    border-radius: 4px;

    font-size: 1rem;

    cursor: pointer;

    transition: background-color 0.3s ease;

    margin-top: 1rem;

  }

  .submitButton:hover {

    background-color: #27ae60;

  }

  .error {

    color: #e74c3c;

    text-align: center;

    margin-bottom: 1rem;

  }

  .links {

    margin-top: 1rem;

    text-align: center;

  }

  .forgotPassword, .registerLink a {

    color: #2ecc71;

    text-decoration: none;

    font-weight: bold;

  }

  .forgotPassword:hover, .registerLink a:hover {

    text-decoration: underline;

  }

  .registerLink {

    margin-top: 0.5rem;

  }

  @media (max-width: 480px) {

    .loginCard {

      padding: 1.5rem;

    }

  }

/src/components/NetworkAndUpgrade.js

import React, { useState, useEffect } from 'react';

import axios from 'axios';

import styles from './NetworkAndUpgrade.module.css';

const NetworkAndUpgrade = () => {

  const [user, setUser] = useState(null);

  const [network, setNetwork] = useState({ E: [], P: [], B: [] });

  const [loading, setLoading] = useState(true);

  const [error, setError] = useState('');

  useEffect(() => {

    fetchUserAndNetwork();

  }, []);

  const fetchUserAndNetwork = async () => {

    try {

      const token = localStorage.getItem('token'); // Assume token is stored in localStorage

      const userResponse = await axios.get(`${process.env.REACT\_APP\_API\_URL}/user`, {

        headers: { Authorization: `Bearer ${token}` }

      });

      setUser(userResponse.data);

      const networkResponse = await axios.get(`${process.env.REACT\_APP\_API\_URL}/network`, {

        headers: { Authorization: `Bearer ${token}` }

      });

      setNetwork(networkResponse.data);

      setLoading(false);

    } catch (err) {

      setError('Failed to fetch user data and network');

      setLoading(false);

    }

  };

  const handleUpgrade = async (newClass) => {

    try {

      const token = localStorage.getItem('token');

      await axios.post(`${process.env.REACT\_APP\_API\_URL}/upgrade`, { newClass }, {

        headers: { Authorization: `Bearer ${token}` }

      });

      // Refetch user data after successful upgrade

      fetchUserAndNetwork();

    } catch (err) {

      setError('Failed to upgrade membership class');

    }

  };

  if (loading) return <div className={styles.loading}>Loading...</div>;

  if (error) return <div className={styles.error}>{error}</div>;

  return (

    <div className={styles.container}>

      <h1 className={styles.title}>Your Network & Membership</h1>

      <section className={styles.userInfo}>

        <h2>Your Information</h2>

        <p><strong>Name:</strong> {user.first\_name} {user.last\_name}</p>

        <p><strong>Member Code:</strong> {user.member\_code}</p>

        <p><strong>Current Class:</strong> {user.class}</p>

      </section>

      <section className={styles.upgrade}>

        <h2>Upgrade Your Membership</h2>

        {user.class === 'E' && (

          <>

            <button onClick={() => handleUpgrade('P')} className={styles.upgradeButton}>Upgrade to P-Class</button>

            <button onClick={() => handleUpgrade('B')} className={styles.upgradeButton}>Upgrade to B-Class</button>

          </>

        )}

        {user.class === 'P' && (

          <button onClick={() => handleUpgrade('B')} className={styles.upgradeButton}>Upgrade to B-Class</button>

        )}

        {user.class === 'B' && (

          <p>You have reached the highest membership class!</p>

        )}

      </section>

      <section className={styles.network}>

        <h2>Your Network</h2>

        <div className={styles.networkGrid}>

          <div className={styles.networkClass}>

            <h3>E-Class Members</h3>

            <ul>

              {network.E.map(member => (

                <li key={member.id}>{member.first\_name} {member.last\_name} - {member.member\_code}</li>

              ))}

            </ul>

          </div>

          <div className={styles.networkClass}>

            <h3>P-Class Members</h3>

            <ul>

              {network.P.map(member => (

                <li key={member.id}>{member.first\_name} {member.last\_name} - {member.member\_code}</li>

              ))}

            </ul>

          </div>

          <div className={styles.networkClass}>

            <h3>B-Class Members</h3>

            <ul>

              {network.B.map(member => (

                <li key={member.id}>{member.first\_name} {member.last\_name} - {member.member\_code}</li>

              ))}

            </ul>

          </div>

        </div>

      </section>

    </div>

  );

};

export default NetworkAndUpgrade;

/src/components/NetworkAndUpgrade.module.css

.container {

    max-width: 1200px;

    margin: 0 auto;

    padding: 2rem;

    font-family: Arial, sans-serif;

  }

  .title {

    color: #2ecc71;

    text-align: center;

    margin-bottom: 2rem;

  }

  .userInfo, .upgrade, .network {

    background-color: #fff;

    border-radius: 8px;

    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

    padding: 1.5rem;

    margin-bottom: 2rem;

  }

  .userInfo h2, .upgrade h2, .network h2 {

    color: #2ecc71;

    margin-bottom: 1rem;

  }

  .upgradeButton {

    background-color: #2ecc71;

    color: white;

    border: none;

    padding: 0.75rem 1.5rem;

    border-radius: 4px;

    font-size: 1rem;

    cursor: pointer;

    transition: background-color 0.3s ease;

    margin-right: 1rem;

    margin-bottom: 1rem;

  }

  .upgradeButton:hover {

    background-color: #27ae60;

  }

  .networkGrid {

    display: grid;

    grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));

    gap: 1.5rem;

  }

  .networkClass {

    background-color: #f9f9f9;

    border-radius: 8px;

    padding: 1rem;

  }

  .networkClass h3 {

    color: #2ecc71;

    margin-bottom: 0.5rem;

  }

  .networkClass ul {

    list-style-type: none;

    padding: 0;

  }

  .networkClass li {

    margin-bottom: 0.5rem;

    padding: 0.5rem;

    background-color: #fff;

    border-radius: 4px;

    box-shadow: 0 1px 2px rgba(0, 0, 0, 0.1);

  }

  .loading, .error {

    text-align: center;

    padding: 2rem;

    font-size: 1.2rem;

  }

  .error {

    color: #e74c3c;

  }

  @media (max-width: 768px) {

    .networkGrid {

      grid-template-columns: 1fr;

    }

    .upgradeButton {

      width: 100%;

      margin-right: 0;

    }

  }

/src/components/PrivateRoute.js

import React from 'react';

import { Navigate } from 'react-router-dom';

import { useSelector } from 'react-redux';

const PrivateRoute = ({ children }) => {

  const isAuthenticated = useSelector(state => state.auth.isAuthenticated);

  return isAuthenticated ? children : <Navigate to="/login" />;

};

export default PrivateRoute;

/src/components/Profile.js

import React, { useState, useEffect } from 'react';

import axios from 'axios';

import styles from './Profile.module.css';

const Profile = () => {

  const [user, setUser] = useState(null);

  const [loading, setLoading] = useState(true);

  const [error, setError] = useState('');

  useEffect(() => {

    fetchUserData();

  }, []);

  const fetchUserData = async () => {

    try {

      const token = localStorage.getItem('token'); // Assume token is stored in localStorage

      const response = await axios.get(`${process.env.REACT\_APP\_API\_URL}/user`, {

        headers: { Authorization: `Bearer ${token}` }

      });

      setUser(response.data);

      setLoading(false);

    } catch (error) {

      console.error('Error fetching user data:', error);

      setError('Failed to load user data');

      setLoading(false);

    }

  };

  if (loading) return <div className={styles.loading}>Loading...</div>;

  if (error) return <div className={styles.error}>{error}</div>;

  if (!user) return <div className={styles.error}>No user data found</div>;

  return (

    <div className={styles.profileContainer}>

      <header className={styles.profileHeader}>

        <div className={styles.profilePicture}>

          {user.profile\_picture ? (

            <img src={user.profile\_picture} alt={user.username} />

          ) : (

            <div className={styles.initials}>{user.username[0].toUpperCase()}</div>

          )}

        </div>

        <h1>{user.first\_name} {user.last\_name}</h1>

        <p className={styles.username}>@{user.username}</p>

        <p className={styles.memberCode}>Member Code: {user.member\_code}</p>

      </header>

      <section className={styles.profileInfo}>

        <h2>Personal Information</h2>

        <div className={styles.infoGrid}>

          <div className={styles.infoItem}>

            <strong>Email:</strong> {user.email}

          </div>

          <div className={styles.infoItem}>

            <strong>Country:</strong> {user.country}

          </div>

          <div className={styles.infoItem}>

            <strong>City:</strong> {user.city}

          </div>

          <div className={styles.infoItem}>

            <strong>Age:</strong> {user.age}

          </div>

          <div className={styles.infoItem}>

            <strong>Class:</strong> {user.class}

          </div>

          <div className={styles.infoItem}>

            <strong>Sponsor Code:</strong> {user.sponsor\_code}

          </div>

        </div>

      </section>

      <section className={styles.profileStats}>

        <h2>Account Statistics</h2>

        <div className={styles.statsGrid}>

          <div className={styles.statItem}>

            <span className={styles.statValue}>{user.network\_size}</span>

            <span className={styles.statLabel}>Network Size</span>

          </div>

          <div className={styles.statItem}>

            <span className={styles.statValue}>{user.gifts\_sent}</span>

            <span className={styles.statLabel}>Gifts Sent</span>

          </div>

          <div className={styles.statItem}>

            <span className={styles.statValue}>{user.gifts\_received}</span>

            <span className={styles.statLabel}>Gifts Received</span>

          </div>

        </div>

      </section>

      <section className={styles.profileActions}>

        <h2>Account Actions</h2>

        <div className={styles.actionButtons}>

          <button className={styles.actionButton}>Edit Profile</button>

          <button className={styles.actionButton}>Change Password</button>

          <button className={styles.actionButton}>Upgrade Membership</button>

        </div>

      </section>

    </div>

  );

};

export default Profile;

/src/components/Profile.module.css

.profileContainer {

  max-width: 800px;

  margin: 2rem auto;

  padding: 2rem;

  background-color: white;

  border-radius: 8px;

  box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

}

.profileHeader {

  text-align: center;

  margin-bottom: 2rem;

}

.profilePicture {

  width: 150px;

  height: 150px;

  border-radius: 50%;

  margin: 0 auto 1rem;

  overflow: hidden;

  background-color: #2ecc71;

  display: flex;

  align-items: center;

  justify-content: center;

}

.profilePicture img {

  width: 100%;

  height: 100%;

  object-fit: cover;

}

.initials {

  font-size: 4rem;

  color: white;

}

.profileHeader h1 {

  color: #2ecc71;

  margin-bottom: 0.5rem;

}

.username {

  color: #777;

  font-size: 1.1rem;

  margin-bottom: 0.5rem;

}

.memberCode {

  font-weight: bold;

  color: #2ecc71;

}

.profileInfo, .profileStats, .profileActions {

  margin-bottom: 2rem;

}

.profileInfo h2, .profileStats h2, .profileActions h2 {

  color: #2ecc71;

  border-bottom: 2px solid #2ecc71;

  padding-bottom: 0.5rem;

  margin-bottom: 1rem;

}

.infoGrid, .statsGrid {

  display: grid;

  grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));

  gap: 1rem;

}

.infoItem, .statItem {

  padding: 1rem;

  background-color: #f9f9f9;

  border-radius: 4px;

}

.statItem {

  text-align: center;

}

.statValue {

  display: block;

  font-size: 2rem;

  font-weight: bold;

  color: #2ecc71;

}

.statLabel {

  color: #777;

}

.actionButtons {

  display: flex;

  gap: 1rem;

  flex-wrap: wrap;

}

.actionButton {

  flex: 1;

  min-width: 200px;

  padding: 0.75rem 1rem;

  background-color: #2ecc71;

  color: white;

  border: none;

  border-radius: 4px;

  font-size: 1rem;

  cursor: pointer;

  transition: background-color 0.3s ease;

}

.actionButton:hover {

  background-color: #27ae60;

}

.loading, .error {

  text-align: center;

  padding: 2rem;

  font-size: 1.2rem;

}

.error {

  color: #e74c3c;

}

@media (max-width: 600px) {

  .profileContainer {

    padding: 1rem;

  }

  .infoGrid, .statsGrid {

    grid-template-columns: 1fr;

  }

  .actionButtons {

    flex-direction: column;

  }

  .actionButton {

    width: 100%;

  }

}

/src/components/Register.js

import React, { useState } from 'react';

import { useNavigate, Link } from 'react-router-dom';

import axios from 'axios';

import styles from './Register.module.css';

const Register = () => {

  const [formData, setFormData] = useState({

    username: '',

    email: '',

    password: '',

    confirmPassword: '',

    firstName: '',

    lastName: '',

    country: '',

    city: '',

    age: '',

    gender: '',

    occupation: '',

    interests: [],

    sponsorCode: '',

    agreeTOS: false

  });

  const [error, setError] = useState('');

  const [success, setSuccess] = useState('');

  const navigate = useNavigate();

  const interestOptions = [

    'Entrepreneurship', 'Health', 'Education', 'Technology',

    'Agriculture', 'Finance', 'Arts', 'Environmental Sustainability'

  ];

  const handleChange = (e) => {

    const { name, value, type, checked } = e.target;

    setFormData(prevState => ({

      ...prevState,

      [name]: type === 'checkbox' ? checked : value

    }));

  };

  const handleInterestChange = (interest) => {

    setFormData(prevState => ({

      ...prevState,

      interests: prevState.interests.includes(interest)

        ? prevState.interests.filter(i => i !== interest)

        : [...prevState.interests, interest]

    }));

  };

  const handleSubmit = async (e) => {

    e.preventDefault();

    setError('');

    setSuccess('');

    if (formData.password !== formData.confirmPassword) {

      setError('Passwords do not match');

      return;

    }

    if (!formData.agreeTOS) {

      setError('You must agree to the Terms of Service');

      return;

    }

    try {

      const response = await axios.post(`${process.env.REACT\_APP\_API\_URL}/register`, formData);

      if (response.data.success) {

        setSuccess('Registration successful! Redirecting to login...');

        setTimeout(() => navigate('/login'), 3000);

      } else {

        setError(response.data.message || 'Registration failed');

      }

    } catch (error) {

      console.error('Registration error:', error);

      setError(error.response?.data?.message || 'An error occurred during registration');

    }

  };

  return (

    <div className={styles.registerContainer}>

      <div className={styles.registerCard}>

        <h2 className={styles.registerTitle}>Join Africhama</h2>

        <p className={styles.registerSubtitle}>Empower your success, connect with a global community</p>

        {error && <p className={styles.error}>{error}</p>}

        {success && <p className={styles.success}>{success}</p>}

        <form onSubmit={handleSubmit} className={styles.registerForm}>

          <div className={styles.formSection}>

            <h3>Account Information</h3>

            <div className={styles.formGroup}>

              <label htmlFor="username">Username</label>

              <input type="text" id="username" name="username" value={formData.username} onChange={handleChange} required />

            </div>

            <div className={styles.formGroup}>

              <label htmlFor="email">Email</label>

              <input type="email" id="email" name="email" value={formData.email} onChange={handleChange} required />

            </div>

            <div className={styles.formGroup}>

              <label htmlFor="password">Password</label>

              <input type="password" id="password" name="password" value={formData.password} onChange={handleChange} required />

            </div>

            <div className={styles.formGroup}>

              <label htmlFor="confirmPassword">Confirm Password</label>

              <input type="password" id="confirmPassword" name="confirmPassword" value={formData.confirmPassword} onChange={handleChange} required />

            </div>

          </div>

          <div className={styles.formSection}>

            <h3>Personal Information</h3>

            <div className={styles.formRow}>

              <div className={styles.formGroup}>

                <label htmlFor="firstName">First Name</label>

                <input type="text" id="firstName" name="firstName" value={formData.firstName} onChange={handleChange} required />

              </div>

              <div className={styles.formGroup}>

                <label htmlFor="lastName">Last Name</label>

                <input type="text" id="lastName" name="lastName" value={formData.lastName} onChange={handleChange} required />

              </div>

            </div>

            <div className={styles.formRow}>

              <div className={styles.formGroup}>

                <label htmlFor="country">Country</label>

                <input type="text" id="country" name="country" value={formData.country} onChange={handleChange} required />

              </div>

              <div className={styles.formGroup}>

                <label htmlFor="city">City</label>

                <input type="text" id="city" name="city" value={formData.city} onChange={handleChange} required />

              </div>

            </div>

            <div className={styles.formRow}>

              <div className={styles.formGroup}>

                <label htmlFor="age">Age</label>

                <input type="number" id="age" name="age" value={formData.age} onChange={handleChange} required min="19" />

              </div>

              <div className={styles.formGroup}>

                <label htmlFor="gender">Gender</label>

                <select id="gender" name="gender" value={formData.gender} onChange={handleChange} required>

                  <option value="">Select gender</option>

                  <option value="male">Male</option>

                  <option value="female">Female</option>

                  <option value="other">Other</option>

                  <option value="prefer-not-to-say">Prefer not to say</option>

                </select>

              </div>

            </div>

            <div className={styles.formGroup}>

              <label htmlFor="occupation">Occupation</label>

              <input type="text" id="occupation" name="occupation" value={formData.occupation} onChange={handleChange} required />

            </div>

          </div>

          <div className={styles.formSection}>

            <h3>Interests</h3>

            <div className={styles.interestsGrid}>

              {interestOptions.map(interest => (

                <label key={interest} className={styles.interestLabel}>

                  <input

                    type="checkbox"

                    checked={formData.interests.includes(interest)}

                    onChange={() => handleInterestChange(interest)}

                  />

                  {interest}

                </label>

              ))}

            </div>

          </div>

          <div className={styles.formSection}>

            <h3>Africhama Network</h3>

            <div className={styles.formGroup}>

              <label htmlFor="sponsorCode">Sponsor Code</label>

              <input type="text" id="sponsorCode" name="sponsorCode" value={formData.sponsorCode} onChange={handleChange} required />

            </div>

          </div>

          <div className={styles.formGroup}>

            <label className={styles.checkboxLabel}>

              <input

                type="checkbox"

                name="agreeTOS"

                checked={formData.agreeTOS}

                onChange={handleChange}

                required

              />

              I agree to the <Link to="/terms-of-service">Terms of Service</Link> and <Link to="/privacy-policy">Privacy Policy</Link>

            </label>

          </div>

          <button type="submit" className={styles.submitButton}>Register</button>

        </form>

        <p className={styles.loginLink}>

          Already have an account? <Link to="/login">Login here</Link>

        </p>

      </div>

    </div>

  );

};

export default Register;

/src/components/Register.module.css

.registerContainer {

  display: flex;

  justify-content: center;

  align-items: center;

  min-height: 100vh;

  background-color: #f0f8f0;

  padding: 2rem;

}

.registerCard {

  background-color: white;

  border-radius: 12px;

  box-shadow: 0 8px 16px rgba(46, 204, 113, 0.1);

  padding: 2rem;

  width: 100%;

  max-width: 600px;

}

.registerTitle {

  color: #2ecc71;

  text-align: center;

  margin-bottom: 0.5rem;

  font-size: 2.5rem;

}

.registerSubtitle {

  text-align: center;

  color: #666;

  margin-bottom: 2rem;

}

.registerForm {

  display: flex;

  flex-direction: column;

}

.formSection {

  margin-bottom: 2rem;

  border-bottom: 1px solid #eee;

  padding-bottom: 1rem;

}

.formSection h3 {

  color: #2ecc71;

  margin-bottom: 1rem;

}

.formGroup {

  margin-bottom: 1rem;

}

.formRow {

  display: flex;

  gap: 1rem;

}

.formRow .formGroup {

  flex: 1;

}

.formGroup label {

  display: block;

  margin-bottom: 0.5rem;

  color: #333;

  font-weight: bold;

}

.formGroup input,

.formGroup select {

  width: 100%;

  padding: 0.75rem;

  border: 1px solid #ddd;

  border-radius: 4px;

  font-size: 1rem;

  transition: border-color 0.3s ease, box-shadow 0.3s ease;

}

.formGroup input:focus,

.formGroup select:focus {

  outline: none;

  border-color: #2ecc71;

  box-shadow: 0 0 0 2px rgba(46, 204, 113, 0.2);

}

.interestsGrid {

  display: grid;

  grid-template-columns: repeat(auto-fill, minmax(150px, 1fr));

  gap: 0.5rem;

}

.interestLabel {

  display: flex;

  align-items: center;

  font-size: 0.9rem;

}

.interestLabel input {

  margin-right: 0.5rem;

}

.checkboxLabel {

  display: flex;

  align-items: center;

  font-size: 0.9rem;

}

.checkboxLabel input {

  margin-right: 0.5rem;

}

.submitButton {

  background-color: #2ecc71;

  color: white;

  border: none;

  padding: 1rem;

  border-radius: 4px;

  font-size: 1rem;

  cursor: pointer;

  transition: background-color 0.3s ease, transform 0.1s ease;

  margin-top: 1rem;

}

.submitButton:hover {

  background-color: #27ae60;

}

.submitButton:active {

  transform: translateY(1px);

}

.error {

  color: #e74c3c;

  text-align: center;

  margin-bottom: 1rem;

  padding: 0.5rem;

  background-color: #fdeaea;

  border-radius: 4px;

}

.success {

  color: #2ecc71;

  text-align: center;

  margin-bottom: 1rem;

  padding: 0.5rem;

  background-color: #e8f8f0;

  border-radius: 4px;

}

.loginLink {

  text-align: center;

  margin-top: 1rem;

}

.loginLink a,

.checkboxLabel a {

  color: #2ecc71;

  text-decoration: none;

  font-weight: bold;

}

.loginLink a:hover,

.checkboxLabel a:hover {

  text-decoration: underline;

}

@media (max-width: 600px) {

  .formRow {

    flex-direction: column;

    gap: 0;

  }

  .registerCard {

    padding: 1.5rem;

  }

  .interestsGrid {

    grid-template-columns: repeat(auto-fill, minmax(120px, 1fr));

  }

}

/src/components/SubscriptionMaintenance.js

import React, { useState, useEffect } from 'react';

import axios from 'axios';

import styles from './SubscriptionMaintenance.module.css';

const SubscriptionMaintenance = () => {

  const [subscriptionData, setSubscriptionData] = useState(null);

  const [maintenanceData, setMaintenanceData] = useState(null);

  const [loading, setLoading] = useState(true);

  const [error, setError] = useState('');

  useEffect(() => {

    fetchData();

  }, []);

  const fetchData = async () => {

    try {

      const token = localStorage.getItem('token'); // Assume token is stored in localStorage

      const response = await axios.get(`${process.env.REACT\_APP\_API\_URL}/subscription-maintenance`, {

        headers: { Authorization: `Bearer ${token}` }

      });

      setSubscriptionData(response.data.subscription);

      setMaintenanceData(response.data.maintenance);

      setLoading(false);

    } catch (err) {

      setError('Failed to fetch subscription and maintenance data');

      setLoading(false);

    }

  };

  const handleSubscribe = async () => {

    try {

      const token = localStorage.getItem('token');

      await axios.post(`${process.env.REACT\_APP\_API\_URL}/subscribe`, {}, {

        headers: { Authorization: `Bearer ${token}` }

      });

      fetchData(); // Refetch data after subscribing

    } catch (err) {

      setError('Failed to subscribe. Please try again.');

    }

  };

  const handlePayMaintenance = async () => {

    try {

      const token = localStorage.getItem('token');

      await axios.post(`${process.env.REACT\_APP\_API\_URL}/pay-maintenance`, {}, {

        headers: { Authorization: `Bearer ${token}` }

      });

      fetchData(); // Refetch data after paying maintenance

    } catch (err) {

      setError('Failed to pay maintenance fee. Please try again.');

    }

  };

  if (loading) return <div className={styles.loading}>Loading...</div>;

  return (

    <div className={styles.container}>

      <h1 className={styles.title}>Subscription & Maintenance</h1>

      {error && <p className={styles.error}>{error}</p>}

      <div className={styles.card}>

        <h2>Subscription Status</h2>

        {subscriptionData ? (

          <>

            <p><strong>Status:</strong> {subscriptionData.status}</p>

            <p><strong>Expiry Date:</strong> {new Date(subscriptionData.expiryDate).toLocaleDateString()}</p>

            {subscriptionData.status === 'active' ? (

              <p className={styles.activeStatus}>Your subscription is active</p>

            ) : (

              <button onClick={handleSubscribe} className={styles.button}>Renew Subscription</button>

            )}

          </>

        ) : (

          <>

            <p>You are not currently subscribed.</p>

            <button onClick={handleSubscribe} className={styles.button}>Subscribe Now</button>

          </>

        )}

      </div>

      <div className={styles.card}>

        <h2>Yearly Maintenance</h2>

        {maintenanceData ? (

          <>

            <p><strong>Last Payment:</strong> {new Date(maintenanceData.lastPaymentDate).toLocaleDateString()}</p>

            <p><strong>Next Due Date:</strong> {new Date(maintenanceData.nextDueDate).toLocaleDateString()}</p>

            {new Date(maintenanceData.nextDueDate) <= new Date() ? (

              <button onClick={handlePayMaintenance} className={styles.button}>Pay Maintenance Fee</button>

            ) : (

              <p className={styles.activeStatus}>Your maintenance fee is up to date</p>

            )}

          </>

        ) : (

          <>

            <p>No maintenance payment record found.</p>

            <button onClick={handlePayMaintenance} className={styles.button}>Pay Initial Maintenance Fee</button>

          </>

        )}

      </div>

      <div className={styles.infoBox}>

        <h3>Why Maintenance Fee?</h3>

        <p>The yearly maintenance fee helps us keep Africhama running smoothly, ensuring you always have access to the best features and support. It's a small investment in your success and the success of our community.</p>

      </div>

    </div>

  );

};

export default SubscriptionMaintenance;

/src/components/SubscriptionMaintenance.module.css

.container {

    max-width: 800px;

    margin: 0 auto;

    padding: 2rem;

  }

  .title {

    color: #2ecc71;

    text-align: center;

    margin-bottom: 2rem;

  }

  .card {

    background-color: white;

    border-radius: 8px;

    box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

    padding: 1.5rem;

    margin-bottom: 2rem;

  }

  .card h2 {

    color: #2ecc71;

    margin-bottom: 1rem;

  }

  .button {

    background-color: #2ecc71;

    color: white;

    border: none;

    padding: 0.75rem 1.5rem;

    border-radius: 4px;

    font-size: 1rem;

    cursor: pointer;

    transition: background-color 0.3s ease;

    margin-top: 1rem;

  }

  .button:hover {

    background-color: #27ae60;

  }

  .activeStatus {

    color: #2ecc71;

    font-weight: bold;

    margin-top: 1rem;

  }

  .error {

    color: #e74c3c;

    text-align: center;

    margin-bottom: 1rem;

  }

  .loading {

    text-align: center;

    color: #2ecc71;

    font-size: 1.2rem;

    margin-top: 2rem;

  }

  .infoBox {

    background-color: #f0f0f0;

    border-left: 4px solid #2ecc71;

    padding: 1rem;

    margin-top: 2rem;

  }

  .infoBox h3 {

    color: #2ecc71;

    margin-bottom: 0.5rem;

  }

  @media (max-width: 600px) {

    .container {

      padding: 1rem;

    }

    .card {

      padding: 1rem;

    }

  }

/src/components/Support.js

import React, { useState } from 'react';

import axios from 'axios';

import styles from './Support.module.css';

const Support = () => {

  const [activeTab, setActiveTab] = useState('faq');

  const [contactForm, setContactForm] = useState({

    name: '',

    email: '',

    subject: '',

    message: ''

  });

  const [submitStatus, setSubmitStatus] = useState(null);

  const handleContactChange = (e) => {

    setContactForm({ ...contactForm, [e.target.name]: e.target.value });

  };

  const handleContactSubmit = async (e) => {

    e.preventDefault();

    try {

      // Replace with your actual API endpoint

      await axios.post(`${process.env.REACT\_APP\_API\_URL}/support`, contactForm);

      setSubmitStatus('success');

      setContactForm({ name: '', email: '', subject: '', message: '' });

    } catch (error) {

      setSubmitStatus('error');

    }

  };

  const faqData = [

    {

      question: "What is Africhama?",

      answer: "Africhama is an online platform that empowers individuals through knowledge sharing and community support. We offer free educational materials, networking opportunities, and a unique gift system to foster growth and success."

    },

    {

      question: "How do I join Africhama?",

      answer: "To join Africhama, click on the 'Register' button in the top menu. You'll need to provide some basic information and a sponsor code from an existing member."

    },

    {

      question: "What is the gift system?",

      answer: "The Africhama gift system is a unique feature that allows members to support each other through direct donations. It's a way to foster community and mutual growth within our platform."

    },

    {

      question: "How can I upgrade my membership class?",

      answer: "You can upgrade your membership class through your profile dashboard. Different classes offer various benefits and opportunities within the Africhama community."

    },

    {

      question: "What kind of educational materials are available?",

      answer: "We offer a wide range of free downloadable content covering topics such as entrepreneurship, health, nature, and life skills. These resources are designed to support your personal and professional growth."

    }

  ];

  return (

    <div className={styles.supportContainer}>

      <h1 className={styles.supportTitle}>Africhama Support Center</h1>

      <div className={styles.tabContainer}>

        <button

          className={`${styles.tabButton} ${activeTab === 'faq' ? styles.active : ''}`}

          onClick={() => setActiveTab('faq')}

        >

          FAQ

        </button>

        <button

          className={`${styles.tabButton} ${activeTab === 'contact' ? styles.active : ''}`}

          onClick={() => setActiveTab('contact')}

        >

          Contact Us

        </button>

      </div>

      {activeTab === 'faq' && (

        <div className={styles.faqSection}>

          <h2>Frequently Asked Questions</h2>

          {faqData.map((faq, index) => (

            <div key={index} className={styles.faqItem}>

              <h3>{faq.question}</h3>

              <p>{faq.answer}</p>

            </div>

          ))}

        </div>

      )}

      {activeTab === 'contact' && (

        <div className={styles.contactSection}>

          <h2>Contact Us</h2>

          <form onSubmit={handleContactSubmit} className={styles.contactForm}>

            <div className={styles.formGroup}>

              <label htmlFor="name">Name</label>

              <input

                type="text"

                id="name"

                name="name"

                value={contactForm.name}

                onChange={handleContactChange}

                required

              />

            </div>

            <div className={styles.formGroup}>

              <label htmlFor="email">Email</label>

              <input

                type="email"

                id="email"

                name="email"

                value={contactForm.email}

                onChange={handleContactChange}

                required

              />

            </div>

            <div className={styles.formGroup}>

              <label htmlFor="subject">Subject</label>

              <input

                type="text"

                id="subject"

                name="subject"

                value={contactForm.subject}

                onChange={handleContactChange}

                required

              />

            </div>

            <div className={styles.formGroup}>

              <label htmlFor="message">Message</label>

              <textarea

                id="message"

                name="message"

                value={contactForm.message}

                onChange={handleContactChange}

                required

              ></textarea>

            </div>

            <button type="submit" className={styles.submitButton}>Send Message</button>

          </form>

          {submitStatus === 'success' && (

            <p className={styles.successMessage}>Your message has been sent successfully!</p>

          )}

          {submitStatus === 'error' && (

            <p className={styles.errorMessage}>There was an error sending your message. Please try again.</p>

          )}

        </div>

      )}

      <div className={styles.additionalResources}>

        <h2>Additional Resources</h2>

        <ul>

          <li><a href="/terms">Terms of Service</a></li>

          <li><a href="/privacy">Privacy Policy</a></li>

          <li><a href="/community-guidelines">Community Guidelines</a></li>

        </ul>

      </div>

    </div>

  );

};

export default Support;

/src/components/Support.module.css

.supportContainer {

    max-width: 800px;

    margin: 0 auto;

    padding: 2rem;

    background-color: #fff;

    border-radius: 8px;

    box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

  }

  .supportTitle {

    color: #2ecc71;

    text-align: center;

    margin-bottom: 2rem;

    font-size: 2.5rem;

  }

  .tabContainer {

    display: flex;

    justify-content: center;

    margin-bottom: 2rem;

  }

  .tabButton {

    background-color: #f0f0f0;

    border: none;

    padding: 1rem 2rem;

    font-size: 1rem;

    cursor: pointer;

    transition: background-color 0.3s ease;

  }

  .tabButton:first-child {

    border-top-left-radius: 4px;

    border-bottom-left-radius: 4px;

  }

  .tabButton:last-child {

    border-top-right-radius: 4px;

    border-bottom-right-radius: 4px;

  }

  .tabButton.active {

    background-color: #2ecc71;

    color: white;

  }

  .faqSection, .contactSection {

    margin-bottom: 2rem;

  }

  .faqItem {

    margin-bottom: 1.5rem;

  }

  .faqItem h3 {

    color: #2ecc71;

    margin-bottom: 0.5rem;

  }

  .contactForm {

    display: flex;

    flex-direction: column;

  }

  .formGroup {

    margin-bottom: 1rem;

  }

  .formGroup label {

    display: block;

    margin-bottom: 0.5rem;

    color: #333;

    font-weight: bold;

  }

  .formGroup input,

  .formGroup textarea {

    width: 100%;

    padding: 0.75rem;

    border: 1px solid #ddd;

    border-radius: 4px;

    font-size: 1rem;

    transition: border-color 0.3s ease;

  }

  .formGroup input:focus,

  .formGroup textarea:focus {

    outline: none;

    border-color: #2ecc71;

  }

  .formGroup textarea {

    min-height: 150px;

    resize: vertical;

  }

  .submitButton {

    background-color: #2ecc71;

    color: white;

    border: none;

    padding: 1rem;

    border-radius: 4px;

    font-size: 1rem;

    cursor: pointer;

    transition: background-color 0.3s ease;

    align-self: flex-start;

  }

  .submitButton:hover {

    background-color: #27ae60;

  }

  .successMessage {

    color: #2ecc71;

    font-weight: bold;

    margin-top: 1rem;

  }

  .errorMessage {

    color: #e74c3c;

    font-weight: bold;

    margin-top: 1rem;

  }

  .additionalResources {

    border-top: 1px solid #ddd;

    padding-top: 2rem;

  }

  .additionalResources ul {

    list-style-type: none;

    padding: 0;

  }

  .additionalResources li {

    margin-bottom: 0.5rem;

  }

  .additionalResources a {

    color: #2ecc71;

    text-decoration: none;

    transition: color 0.3s ease;

  }

  .additionalResources a:hover {

    color: #27ae60;

  }

  @media (max-width: 600px) {

    .supportContainer {

      padding: 1rem;

    }

    .tabButton {

      padding: 0.75rem 1rem;

    }

    .submitButton {

      width: 100%;

    }

  }

/src/store/authSlice.js

import { createSlice } from '@reduxjs/toolkit';

const initialState = {

  user: null,

  token: null,

  isAuthenticated: false,

  isLoading: false,

  error: null

};

const authSlice = createSlice({

  name: 'auth',

  initialState,

  reducers: {

    loginStart: (state) => {

      state.isLoading = true;

      state.error = null;

    },

    loginSuccess: (state, action) => {

      state.isLoading = false;

      state.isAuthenticated = true;

      state.user = action.payload.user;

      state.token = action.payload.token;

      state.error = null;

    },

    loginFailure: (state, action) => {

      state.isLoading = false;

      state.isAuthenticated = false;

      state.user = null;

      state.token = null;

      state.error = action.payload;

    },

    logout: (state) => {

      state.isAuthenticated = false;

      state.user = null;

      state.token = null;

      state.error = null;

    },

    updateUser: (state, action) => {

      state.user = { ...state.user, ...action.payload };

    },

    setError: (state, action) => {

      state.error = action.payload;

    },

    clearError: (state) => {

      state.error = null;

    }

  }

});

export const {

  loginStart,

  loginSuccess,

  loginFailure,

  logout,

  updateUser,

  setError,

  clearError

} = authSlice.actions;

export default authSlice.reducer;

// Selectors

export const selectUser = (state) => state.auth.user;

export const selectToken = (state) => state.auth.token;

export const selectIsAuthenticated = (state) => state.auth.isAuthenticated;

export const selectIsLoading = (state) => state.auth.isLoading;

export const selectError = (state) => state.auth.error;

/src/store/index.js

import { configureStore } from '@reduxjs/toolkit';

import authReducer from './authSlice';

import subscriptionReducer from './subscriptionSlice';

const store = configureStore({

  reducer: {

    auth: authReducer,

    subscription: subscriptionReducer,

  },

});

export default store;

/src/store/subscriptionSlice.js

import { createSlice } from '@reduxjs/toolkit';

const initialState = {

  subscription: null,

  maintenanceStatus: null,

  isLoading: false,

  error: null

};

const subscriptionSlice = createSlice({

  name: 'subscription',

  initialState,

  reducers: {

    fetchSubscriptionStart: (state) => {

      state.isLoading = true;

      state.error = null;

    },

    fetchSubscriptionSuccess: (state, action) => {

      state.isLoading = false;

      state.subscription = action.payload.subscription;

      state.maintenanceStatus = action.payload.maintenanceStatus;

      state.error = null;

    },

    fetchSubscriptionFailure: (state, action) => {

      state.isLoading = false;

      state.error = action.payload;

    },

    subscribeStart: (state) => {

      state.isLoading = true;

      state.error = null;

    },

    subscribeSuccess: (state, action) => {

      state.isLoading = false;

      state.subscription = action.payload;

      state.error = null;

    },

    subscribeFailure: (state, action) => {

      state.isLoading = false;

      state.error = action.payload;

    },

    payMaintenanceStart: (state) => {

      state.isLoading = true;

      state.error = null;

    },

    payMaintenanceSuccess: (state, action) => {

      state.isLoading = false;

      state.maintenanceStatus = action.payload;

      state.error = null;

    },

    payMaintenanceFailure: (state, action) => {

      state.isLoading = false;

      state.error = action.payload;

    },

    clearError: (state) => {

      state.error = null;

    }

  }

});

export const {

  fetchSubscriptionStart,

  fetchSubscriptionSuccess,

  fetchSubscriptionFailure,

  subscribeStart,

  subscribeSuccess,

  subscribeFailure,

  payMaintenanceStart,

  payMaintenanceSuccess,

  payMaintenanceFailure,

  clearError

} = subscriptionSlice.actions;

export default subscriptionSlice.reducer;

// Selectors

export const selectSubscription = (state) => state.subscription.subscription;

export const selectMaintenanceStatus = (state) => state.subscription.maintenanceStatus;

export const selectIsLoading = (state) => state.subscription.isLoading;

export const selectError = (state) => state.subscription.error;

/src/styles/global.css

/\* Reset and base styles \*/

\* {

    margin: 0;

    padding: 0;

    box-sizing: border-box;

  }

  body {

    font-family: 'Arial', sans-serif;

    line-height: 1.6;

    color: #333;

    background-color: #f4f4f4;

  }

  /\* Typography \*/

  h1, h2, h3, h4, h5, h6 {

    margin-bottom: 0.5em;

    font-weight: 600;

    line-height: 1.2;

  }

  h1 { font-size: 2.5em; }

  h2 { font-size: 2em; }

  h3 { font-size: 1.75em; }

  h4 { font-size: 1.5em; }

  h5 { font-size: 1.25em; }

  h6 { font-size: 1em; }

  p {

    margin-bottom: 1em;

  }

  a {

    color: #2ecc71;

    text-decoration: none;

    transition: color 0.3s ease;

  }

  a:hover {

    color: #27ae60;

  }

  /\* Layout \*/

  .container {

    width: 100%;

    max-width: 1200px;

    margin: 0 auto;

    padding: 0 15px;

  }

  /\* Buttons \*/

  .btn {

    display: inline-block;

    padding: 10px 20px;

    font-size: 1em;

    font-weight: 600;

    text-align: center;

    text-decoration: none;

    background-color: #2ecc71;

    color: #fff;

    border: none;

    border-radius: 4px;

    cursor: pointer;

    transition: background-color 0.3s ease;

  }

  .btn:hover {

    background-color: #27ae60;

  }

  .btn-secondary {

    background-color: #3498db;

  }

  .btn-secondary:hover {

    background-color: #2980b9;

  }

  /\* Forms \*/

  .form-group {

    margin-bottom: 1em;

  }

  .form-group label {

    display: block;

    margin-bottom: 0.5em;

    font-weight: 600;

  }

  .form-group input,

  .form-group textarea,

  .form-group select {

    width: 100%;

    padding: 10px;

    border: 1px solid #ddd;

    border-radius: 4px;

    font-size: 1em;

  }

  /\* Utility classes \*/

  .text-center { text-align: center; }

  .text-right { text-align: right; }

  .text-left { text-align: left; }

  .mt-1 { margin-top: 0.5em; }

  .mt-2 { margin-top: 1em; }

  .mt-3 { margin-top: 1.5em; }

  .mb-1 { margin-bottom: 0.5em; }

  .mb-2 { margin-bottom: 1em; }

  .mb-3 { margin-bottom: 1.5em; }

  .p-1 { padding: 0.5em; }

  .p-2 { padding: 1em; }

  .p-3 { padding: 1.5em; }

  /\* Responsive design \*/

  @media (max-width: 768px) {

    h1 { font-size: 2em; }

    h2 { font-size: 1.75em; }

    h3 { font-size: 1.5em; }

    h4 { font-size: 1.25em; }

    h5 { font-size: 1.1em; }

    h6 { font-size: 1em; }

    .btn {

      display: block;

      width: 100%;

    }

  }

  /\* Animations \*/

  @keyframes fadeIn {

    from { opacity: 0; }

    to { opacity: 1; }

  }

  .fade-in {

    animation: fadeIn 0.3s ease-in;

  }

/src/utils/auth.js

import axios from 'axios';

// Set the AUTH\_TOKEN\_KEY to a unique string for your application

const AUTH\_TOKEN\_KEY = 'africhama\_auth\_token';

// Set the base URL for your API

const API\_BASE\_URL = process.env.REACT\_APP\_API\_BASE\_URL || 'http://localhost:5000/api';

// Function to set the authentication token

export const setAuthToken = (token) => {

  if (token) {

    localStorage.setItem(AUTH\_TOKEN\_KEY, token);

    axios.defaults.headers.common['Authorization'] = `Bearer ${token}`;

  } else {

    localStorage.removeItem(AUTH\_TOKEN\_KEY);

    delete axios.defaults.headers.common['Authorization'];

  }

};

// Function to get the authentication token

export const getAuthToken = () => {

  return localStorage.getItem(AUTH\_TOKEN\_KEY);

};

// Function to check if the user is authenticated

export const isAuthenticated = () => {

  const token = getAuthToken();

  return !!token;

};

// Function to login

export const login = async (username, password) => {

  try {

    const response = await axios.post(`${API\_BASE\_URL}/auth/login`, { username, password });

    const { token, user } = response.data;

    setAuthToken(token);

    return user;

  } catch (error) {

    throw error.response.data;

  }

};

// Function to register

export const register = async (userData) => {

  try {

    const response = await axios.post(`${API\_BASE\_URL}/auth/register`, userData);

    const { token, user } = response.data;

    setAuthToken(token);

    return user;

  } catch (error) {

    throw error.response.data;

  }

};

// Function to logout

export const logout = () => {

  setAuthToken(null);

};

// Function to get the current user's data

export const getCurrentUser = async () => {

  try {

    const response = await axios.get(`${API\_BASE\_URL}/auth/me`);

    return response.data;

  } catch (error) {

    throw error.response.data;

  }

};

// Function to update the user's profile

export const updateProfile = async (userData) => {

  try {

    const response = await axios.put(`${API\_BASE\_URL}/auth/profile`, userData);

    return response.data;

  } catch (error) {

    throw error.response.data;

  }

};

// Function to check if the token is expired

export const isTokenExpired = (token) => {

  if (!token) return true;

  const decodedToken = JSON.parse(atob(token.split('.')[1]));

  return decodedToken.exp \* 1000 < Date.now();

};

// Function to refresh the token

export const refreshToken = async () => {

  try {

    const response = await axios.post(`${API\_BASE\_URL}/auth/refresh-token`);

    const { token } = response.data;

    setAuthToken(token);

    return token;

  } catch (error) {

    throw error.response.data;

  }

};

// Axios interceptor to handle token expiration

axios.interceptors.response.use(

  (response) => response,

  async (error) => {

    const originalRequest = error.config;

    if (error.response.status === 401 && !originalRequest.\_retry) {

      originalRequest.\_retry = true;

      try {

        await refreshToken();

        return axios(originalRequest);

      } catch (refreshError) {

        logout();

        throw refreshError;

      }

    }

    return Promise.reject(error);

  }

);

/src/utils/errorHandler.js

import { toast } from 'react-toastify';

// Generic error handler

export const handleError = (error, defaultMessage = 'An unexpected error occurred') => {

  console.error('Error:', error);

  if (error.response) {

    // The request was made and the server responded with a status code

    // that falls out of the range of 2xx

    const status = error.response.status;

    const message = error.response.data.message || defaultMessage;

    switch (status) {

      case 400:

        toast.error(`Bad Request: ${message}`);

        break;

      case 401:

        toast.error('Unauthorized: Please login again');

        // You might want to trigger a logout action here

        break;

      case 403:

        toast.error('Forbidden: You do not have permission to perform this action');

        break;

      case 404:

        toast.error('Not Found: The requested resource could not be found');

        break;

      case 500:

        toast.error('Server Error: Please try again later');

        break;

      default:

        toast.error(message);

    }

  } else if (error.request) {

    // The request was made but no response was received

    toast.error('Network Error: Please check your internet connection');

  } else {

    // Something happened in setting up the request that triggered an Error

    toast.error(defaultMessage);

  }

  // You can choose to rethrow the error if you want calling code to handle it as well

  // throw error;

};

// Specific error handlers for different parts of your application

export const handleLoginError = (error) => {

  handleError(error, 'Login failed. Please check your credentials and try again.');

};

export const handleRegisterError = (error) => {

  handleError(error, 'Registration failed. Please check your information and try again.');

};

export const handleProfileUpdateError = (error) => {

  handleError(error, 'Failed to update profile. Please try again.');

};

export const handleSubscriptionError = (error) => {

  handleError(error, 'Subscription action failed. Please try again.');

};

export const handleNetworkError = (error) => {

  handleError(error, 'Network action failed. Please try again.');

};

export const handleGiftError = (error) => {

  handleError(error, 'Gift action failed. Please try again.');

};

export const handleSupportTicketError = (error) => {

  handleError(error, 'Failed to submit support ticket. Please try again.');

};

// You can add more specific error handlers as needed for different parts of your application

/src/App.js

import React from 'react';

import { BrowserRouter as Router, Route, Routes } from 'react-router-dom';

import { ToastContainer } from 'react-toastify';

import 'react-toastify/dist/ReactToastify.css';

import Layout from './components/Layout';

import Home from './components/Home';

import Register from './components/Register';

import Login from './components/Login';

import Profile from './components/Profile';

import NetworkAndUpgrade from './components/NetworkAndUpgrade';

import GiftSystem from './components/GiftSystem';

import SubscriptionMaintenance from './components/SubscriptionMaintenance';

import Support from './components/Support';

function App() {

  return (

    <Router>

      <Layout>

        <Routes>

          <Route path="/" element={<Home />} />

          <Route path="/register" element={<Register />} />

          <Route path="/login" element={<Login />} />

          <Route path="/profile" element={<Profile />} />

          <Route path="/network" element={<NetworkAndUpgrade />} />

          <Route path="/gift-system" element={<GiftSystem />} />

          <Route path="/subscription" element={<SubscriptionMaintenance />} />

          <Route path="/support" element={<Support />} />

        </Routes>

        <ToastContainer />

      </Layout>

    </Router>

  );

}

export default App;

/src/config.js

// Environment-specific configuration

const env = process.env.NODE\_ENV || 'development';

const config = {

  development: {

    API\_BASE\_URL: 'http://localhost:5000/api',

    APP\_NAME: 'Africhama (Dev)',

    DEBUG: true,

  },

  test: {

    API\_BASE\_URL: 'http://localhost:5000/api',

    APP\_NAME: 'Africhama (Test)',

    DEBUG: true,

  },

  production: {

    API\_BASE\_URL: 'https://api.africhama.com',

    APP\_NAME: 'Africhama',

    DEBUG: false,

  },

};

// Default configuration

const defaultConfig = {

  // Token key for local storage

  AUTH\_TOKEN\_KEY: 'africhama\_auth\_token',

  // Default currency

  CURRENCY: 'USD',

  // Pagination defaults

  DEFAULT\_PAGE\_SIZE: 10,

  // Subscription levels

  SUBSCRIPTION\_LEVELS: ['E', 'P', 'B'],

  // Support ticket categories

  SUPPORT\_CATEGORIES: [

    'Account Issues',

    'Subscription Problems',

    'Gift System',

    'Network Issues',

    'Other',

  ],

  // Maintenance fee amount

  YEARLY\_MAINTENANCE\_FEE: 20,

  // Minimum age for registration

  MIN\_AGE: 19,

  // Maximum file upload size (in bytes)

  MAX\_UPLOAD\_SIZE: 5 \* 1024 \* 1024, // 5MB

};

// Merge the environment-specific config with the default config

const mergedConfig = { ...defaultConfig, ...config[env] };

export default mergedConfig;

/src/index.css

body {

    font-family: Arial, sans-serif;

    margin: 0;

    padding: 0;

    background-color: #f0f0f0;

  }

  .container {

    max-width: 1200px;

    margin: 0 auto;

    padding: 20px;

  }

  button {

    cursor: pointer;

    padding: 10px 15px;

    background-color: #2ecc71;

    color: white;

    border: none;

    border-radius: 4px;

    font-size: 16px;

  }

  button:hover {

    background-color: #27ae60;

  }

  input[type="text"],

  input[type="email"],

  input[type="password"],

  input[type="number"] {

    width: 100%;

    padding: 10px;

    margin-bottom: 10px;

    border: 1px solid #ddd;

    border-radius: 4px;

  }

/src/index.js

import React from 'react';

import { createRoot } from 'react-dom/client';

import { Provider } from 'react-redux';

import store from './store';

import App from './App';

import './index.css';

const container = document.getElementById('root');

const root = createRoot(container);

root.render(

  <React.StrictMode>

    <Provider store={store}>

      <App />

    </Provider>

  </React.StrictMode>

);

/.env

REACT\_APP\_API\_URL=http://localhost/africhama-app/backend/api

/.env.production

# API Configuration

REACT\_APP\_API\_BASE\_URL=https://api.africhama.com

# App Configuration

REACT\_APP\_NAME=Africhama

# Feature Flags

REACT\_APP\_ENABLE\_GIFT\_SYSTEM=true

REACT\_APP\_ENABLE\_SUBSCRIPTION\_SYSTEM=true

# Third-party Services

REACT\_APP\_GOOGLE\_ANALYTICS\_ID=UA-XXXXXXXXX-X

# Misc

REACT\_APP\_MAX\_UPLOAD\_SIZE=5242880

# Performance

REACT\_APP\_SENTRY\_DSN=https://examplePublicKey@o0.ingest.sentry.io/0

# DO NOT STORE SENSITIVE INFORMATION LIKE API KEYS HERE

# Use a secure method for managing secrets in production

# Build Configuration

GENERATE\_SOURCEMAP=false

/index.html

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="utf-8" />

    <link rel="icon" href="%PUBLIC\_URL%/favicon.ico" />

    <meta name="viewport" content="width=device-width, initial-scale=1" />

    <meta name="theme-color" content="#2ecc71" />

    <meta

      name="description"

      content="Africhama - Empowering Success, Empowering Humanity"

    />

    <link rel="apple-touch-icon" href="%PUBLIC\_URL%/logo192.png" />

    <!--

      manifest.json provides metadata used when your web app is installed on a

      user's mobile device or desktop. See https://developers.google.com/web/fundamentals/web-app-manifest/

    -->

    <link rel="manifest" href="%PUBLIC\_URL%/manifest.json" />

    <!--

      Notice the use of %PUBLIC\_URL% in the tags above.

      It will be replaced with the URL of the `public` folder during the build.

      Only files inside the `public` folder can be referenced from the HTML.

      Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC\_URL%/favicon.ico" will

      work correctly both with client-side routing and a non-root public URL.

      Learn how to configure a non-root public URL by running `npm run build`.

    -->

    <title>Africhama - Empowering Success</title>

  </head>

  <body>

    <noscript>You need to enable JavaScript to run this app.</noscript>

    <div id="root"></div>

    <!--

      This div with id="root" is where your React app will be rendered.

      You can add webfonts, meta tags, or analytics to this file.

      The build step will place the bundled scripts into the <body> tag.

      To begin the development, run `npm start` or `yarn start`.

      To create a production bundle, use `npm run build` or `yarn build`.

    -->

  </body>

</html>

/robots.txt

# https://www.robotstxt.org/robotstxt.html

User-agent: \*

Allow: /

Sitemap: <https://www.africhama.com/sitemap.xml>

/sitemap.xml

<?xml version="1.0" encoding="UTF-8"?>

<urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">

  <url>

    <loc>https://www.africhama.com/</loc>

    <lastmod>2023-06-29</lastmod>

    <changefreq>daily</changefreq>

    <priority>1.0</priority>

  </url>

  <url>

    <loc>https://www.africhama.com/login</loc>

    <lastmod>2023-06-29</lastmod>

    <changefreq>monthly</changefreq>

    <priority>0.8</priority>

  </url>

  <url>

    <loc>https://www.africhama.com/register</loc>

    <lastmod>2023-06-29</lastmod>

    <changefreq>monthly</changefreq>

    <priority>0.8</priority>

  </url>

  <url>

    <loc>https://www.africhama.com/about</loc>

    <lastmod>2023-06-29</lastmod>

    <changefreq>monthly</changefreq>

    <priority>0.7</priority>

  </url>

  <url>

    <loc>https://www.africhama.com/products</loc>

    <lastmod>2023-06-29</lastmod>

    <changefreq>weekly</changefreq>

    <priority>0.9</priority>

  </url>

</urlset>

backend/

api/register.php

import React, { useState } from 'react';

import { useNavigate } from 'react-router-dom';

import axios from 'axios';

const Register = () => {

  const [formData, setFormData] = useState({

    username: '',

    email: '',

    password: '',

    first\_name: '',

    last\_name: '',

    country: '',

    city: '',

    age: '',

    sponsor\_code: ''

  });

  const [error, setError] = useState('');

  const navigate = useNavigate();

  const handleChange = (e) => {

    setFormData({ ...formData, [e.target.name]: e.target.value });

  };

  const handleSubmit = async (e) => {

    e.preventDefault();

    setError('');

    try {

      const response = await axios.post(`${process.env.REACT\_APP\_API\_URL}/register`, formData);

      if (response.data.success) {

        console.log('Registration successful:', response.data);

        navigate('/profile');

      } else {

        setError(response.data.message || 'Registration failed');

      }

    } catch (error) {

      console.error('Registration error:', error);

      setError(error.response?.data?.message || 'An error occurred during registration');

    }

  };

  return (

    <form onSubmit={handleSubmit}>

      <input

        type="text"

        name="username"

        value={formData.username}

        onChange={handleChange}

        placeholder="Username"

        required

      />

      <input

        type="email"

        name="email"

        value={formData.email}

        onChange={handleChange}

        placeholder="Email"

        required

      />

      <input

        type="password"

        name="password"

        value={formData.password}

        onChange={handleChange}

        placeholder="Password"

        required

      />

      <input

        type="text"

        name="first\_name"

        value={formData.first\_name}

        onChange={handleChange}

        placeholder="First Name"

        required

      />

      <input

        type="text"

        name="last\_name"

        value={formData.last\_name}

        onChange={handleChange}

        placeholder="Last Name"

        required

      />

      <input

        type="text"

        name="country"

        value={formData.country}

        onChange={handleChange}

        placeholder="Country"

        required

      />

      <input

        type="text"

        name="city"

        value={formData.city}

        onChange={handleChange}

        placeholder="City"

        required

      />

      <input

        type="number"

        name="age"

        value={formData.age}

        onChange={handleChange}

        placeholder="Age"

        required

      />

      <input

        type="text"

        name="sponsor\_code"

        value={formData.sponsor\_code}

        onChange={handleChange}

        placeholder="Sponsor Code"

        required

      />

      {error && <p className="error">{error}</p>}

      <button type="submit">Register</button>

    </form>

  );

};

export default Register;

.htaccess

RewriteEngine On

RewriteRule ^api/(.\*)$ api/$1.php [L]

Header set Access-Control-Allow-Origin "\*"

Header set Access-Control-Allow-Methods "GET, POST, PUT, DELETE, OPTIONS"

Header set Access-Control-Allow-Headers "Content-Type, Access-Control-Allow-Headers, Authorization, X-Requested-With"

config.php

<?php

// Database configuration

$host = 'localhost:3305';  // Note the port specification

$db   = 'africhama';

$user = 'root';

$pass = '';

$charset = 'utf8mb4';

$dsn = "mysql:host=$host;dbname=$db;charset=$charset";

$options = [

    PDO::ATTR\_ERRMODE            => PDO::ERRMODE\_EXCEPTION,

    PDO::ATTR\_DEFAULT\_FETCH\_MODE => PDO::FETCH\_ASSOC,

    PDO::ATTR\_EMULATE\_PREPARES   => false,

];

try {

     $pdo = new PDO($dsn, $user, $pass, $options);

} catch (\PDOException $e) {

     throw new \PDOException($e->getMessage(), (int)$e->getCode());

}

// Error logging setup

ini\_set('display\_errors', 1);

ini\_set('display\_startup\_errors', 1);

error\_reporting(E\_ALL);

$logFile = \_\_DIR\_\_ . '/error\_log.txt';

ini\_set('error\_log', $logFile);

function logError($message) {

    error\_log(date('[Y-m-d H:i:s] ') . $message . "\n", 3, $GLOBALS['logFile']);

}

// Other configuration settings

define('JWT\_SECRET', 'your\_jwt\_secret\_here');  // Replace with a secure random string

define('JWT\_EXPIRATION', 3600);  // 1 hour in seconds

?>

gift\_system.php

<?php

require\_once 'config.php';

require\_once 'auth\_middleware.php';

header("Content-Type: application/json; charset=UTF-8");

header("Access-Control-Allow-Origin: \*");

header("Access-Control-Allow-Methods: GET, POST, PUT");

header("Access-Control-Max-Age: 3600");

header("Access-Control-Allow-Headers: Content-Type, Access-Control-Allow-Headers, Authorization, X-Requested-With");

$token = getBearerToken();

$user\_id = validateToken($token);

if (!$user\_id) {

    http\_response\_code(401);

    echo json\_encode(array("message" => "Invalid or expired token."));

    exit();

}

if ($\_SERVER['REQUEST\_METHOD'] === 'GET') {

    // Fetch gift history

    try {

        $stmt = $pdo->prepare("

            SELECT g.id, g.amount, g.status, g.created\_at,

                   s.username as sender\_username, r.username as receiver\_username

            FROM gifts g

            JOIN users s ON g.sender\_id = s.id

            JOIN users r ON g.receiver\_id = r.id

            WHERE g.sender\_id = :user\_id OR g.receiver\_id = :user\_id

            ORDER BY g.created\_at DESC

        ");

        $stmt->bindParam(':user\_id', $user\_id);

        $stmt->execute();

        $gifts = $stmt->fetchAll(PDO::FETCH\_ASSOC);

        http\_response\_code(200);

        echo json\_encode(array("gifts" => $gifts));

    } catch(PDOException $e) {

        http\_response\_code(500);

        echo json\_encode(array("message" => "Error fetching gift history: " . $e->getMessage()));

    }

} elseif ($\_SERVER['REQUEST\_METHOD'] === 'POST') {

    // Send a gift

    $data = json\_decode(file\_get\_contents("php://input"));

    if (!isset($data->receiver\_member\_code) || !isset($data->amount)) {

        http\_response\_code(400);

        echo json\_encode(array("message" => "Receiver member code and amount are required."));

        exit();

    }

    try {

        $pdo->beginTransaction();

        // Get receiver's user ID

        $stmt = $pdo->prepare("SELECT id, class FROM users WHERE member\_code = :member\_code");

        $stmt->bindParam(':member\_code', $data->receiver\_member\_code);

        $stmt->execute();

        $receiver = $stmt->fetch(PDO::FETCH\_ASSOC);

        if (!$receiver) {

            $pdo->rollBack();

            http\_response\_code(400);

            echo json\_encode(array("message" => "Invalid receiver member code."));

            exit();

        }

        // Check if amount is valid for sender's class

        $stmt = $pdo->prepare("SELECT class FROM users WHERE id = :user\_id");

        $stmt->bindParam(':user\_id', $user\_id);

        $stmt->execute();

        $sender\_class = $stmt->fetchColumn();

        $valid\_amounts = array(

            'E' => array(10),

            'P' => array(30),

            'B' => array(100)

        );

        if (!in\_array($data->amount, $valid\_amounts[$sender\_class])) {

            $pdo->rollBack();

            http\_response\_code(400);

            echo json\_encode(array("message" => "Invalid gift amount for your class."));

            exit();

        }

        // Create gift

        $stmt = $pdo->prepare("

            INSERT INTO gifts (sender\_id, receiver\_id, amount, status)

            VALUES (:sender\_id, :receiver\_id, :amount, 'pending')

        ");

        $stmt->bindParam(':sender\_id', $user\_id);

        $stmt->bindParam(':receiver\_id', $receiver['id']);

        $stmt->bindParam(':amount', $data->amount);

        $stmt->execute();

        $pdo->commit();

        http\_response\_code(201);

        echo json\_encode(array("message" => "Gift sent successfully."));

    } catch(PDOException $e) {

        $pdo->rollBack();

        http\_response\_code(500);

        echo json\_encode(array("message" => "Error sending gift: " . $e->getMessage()));

    }

} elseif ($\_SERVER['REQUEST\_METHOD'] === 'PUT') {

    // Approve a received gift

    $data = json\_decode(file\_get\_contents("php://input"));

    if (!isset($data->gift\_id)) {

        http\_response\_code(400);

        echo json\_encode(array("message" => "Gift ID is required."));

        exit();

    }

    try {

        $pdo->beginTransaction();

        // Check if the gift exists and belongs to the user

        $stmt = $pdo->prepare("

            SELECT \* FROM gifts

            WHERE id = :gift\_id AND receiver\_id = :user\_id AND status = 'pending'

        ");

        $stmt->bindParam(':gift\_id', $data->gift\_id);

        $stmt->bindParam(':user\_id', $user\_id);

        $stmt->execute();

        $gift = $stmt->fetch(PDO::FETCH\_ASSOC);

        if (!$gift) {

            $pdo->rollBack();

            http\_response\_code(400);

            echo json\_encode(array("message" => "Invalid gift ID or gift already approved."));

            exit();

        }

        // Approve the gift

        $stmt = $pdo->prepare("UPDATE gifts SET status = 'approved' WHERE id = :gift\_id");

        $stmt->bindParam(':gift\_id', $data->gift\_id);

        $stmt->execute();

        $pdo->commit();

        http\_response\_code(200);

        echo json\_encode(array("message" => "Gift approved successfully."));

    } catch(PDOException $e) {

        $pdo->rollBack();

        http\_response\_code(500);

        echo json\_encode(array("message" => "Error approving gift: " . $e->getMessage()));

    }

} else {

    http\_response\_code(405);

    echo json\_encode(array("message" => "Method not allowed."));

}

function getBearerToken() {

    $headers = getallheaders();

    if (isset($headers['Authorization'])) {

        if (preg\_match('/Bearer\s(\S+)/', $headers['Authorization'], $matches)) {

            return $matches[1];

        }

    }

    return null;

}

Login.php

<?php

require\_once 'config.php';

require\_once 'vendor/autoload.php';

use Firebase\JWT\JWT;

header("Content-Type: application/json; charset=UTF-8");

header("Access-Control-Allow-Origin: \*");

header("Access-Control-Allow-Methods: POST");

header("Access-Control-Max-Age: 3600");

header("Access-Control-Allow-Headers: Content-Type, Access-Control-Allow-Headers, Authorization, X-Requested-With");

// Get posted data

$data = json\_decode(file\_get\_contents("php://input"));

// Check if username and password are set

if (!isset($data->username) || !isset($data->password)) {

    http\_response\_code(400);

    echo json\_encode(array("message" => "Username and password are required."));

    exit();

}

$username = $data->username;

$password = $data->password;

try {

    // Prepare a select statement

    $sql = "SELECT id, username, password, email, first\_name, last\_name, class FROM users WHERE username = :username";

    $stmt = $pdo->prepare($sql);

    // Bind parameters

    $stmt->bindParam(':username', $username, PDO::PARAM\_STR);

    // Execute the prepared statement

    $stmt->execute();

    if ($stmt->rowCount() == 1) {

        $row = $stmt->fetch();

        if (password\_verify($password, $row['password'])) {

            $token = array(

                "iss" => API\_BASE\_URL,

                "aud" => API\_BASE\_URL,

                "iat" => time(),

                "nbf" => time(),

                "exp" => time() + JWT\_EXPIRATION,

                "data" => array(

                    "id" => $row['id'],

                    "username" => $row['username'],

                    "email" => $row['email']

                )

            );

            // Generate JWT

            $jwt = JWT::encode($token, JWT\_SECRET, 'HS256');

            http\_response\_code(200);

            echo json\_encode(

                array(

                    "message" => "Successful login.",

                    "jwt" => $jwt,

                    "user" => array(

                        "id" => $row['id'],

                        "username" => $row['username'],

                        "email" => $row['email'],

                        "firstName" => $row['first\_name'],

                        "lastName" => $row['last\_name'],

                        "class" => $row['class']

                    )

                )

            );

        } else {

            http\_response\_code(401);

            echo json\_encode(array("message" => "Login failed. Invalid username or password."));

        }

    } else {

        http\_response\_code(401);

        echo json\_encode(array("message" => "Login failed. Invalid username or password."));

    }

} catch(PDOException $e) {

    http\_response\_code(500);

    echo json\_encode(array("message" => "Login failed: " . $e->getMessage()));

}

network\_system.php

<?php

require\_once 'config.php';

require\_once 'auth\_middleware.php';

header("Content-Type: application/json; charset=UTF-8");

header("Access-Control-Allow-Origin: \*");

header("Access-Control-Allow-Methods: GET, PUT");

header("Access-Control-Max-Age: 3600");

header("Access-Control-Allow-Headers: Content-Type, Access-Control-Allow-Headers, Authorization, X-Requested-With");

$token = getBearerToken();

$user\_id = validateToken($token);

if (!$user\_id) {

    http\_response\_code(401);

    echo json\_encode(array("message" => "Invalid or expired token."));

    exit();

}

if ($\_SERVER['REQUEST\_METHOD'] === 'GET') {

    // Fetch network members

    try {

        $stmt = $pdo->prepare("

            SELECT id, username, email, first\_name, last\_name, member\_code, class

            FROM users

            WHERE sponsor\_code = (SELECT member\_code FROM users WHERE id = :user\_id)

        ");

        $stmt->bindParam(':user\_id', $user\_id);

        $stmt->execute();

        $network\_members = $stmt->fetchAll(PDO::FETCH\_ASSOC);

        // Group members by class

        $network = array(

            'E' => array(),

            'P' => array(),

            'B' => array()

        );

        foreach ($network\_members as $member) {

            $network[$member['class']][] = $member;

        }

        // Fetch user's current class

        $stmt = $pdo->prepare("SELECT class FROM users WHERE id = :user\_id");

        $stmt->bindParam(':user\_id', $user\_id);

        $stmt->execute();

        $user\_class = $stmt->fetchColumn();

        http\_response\_code(200);

        echo json\_encode(array(

            "network" => $network,

            "user\_class" => $user\_class

        ));

    } catch(PDOException $e) {

        http\_response\_code(500);

        echo json\_encode(array("message" => "Error fetching network data: " . $e->getMessage()));

    }

} elseif ($\_SERVER['REQUEST\_METHOD'] === 'PUT') {

    // Upgrade class

    $data = json\_decode(file\_get\_contents("php://input"));

    if (!isset($data->new\_class)) {

        http\_response\_code(400);

        echo json\_encode(array("message" => "New class is required."));

        exit();

    }

    $valid\_upgrades = array(

        'E' => array('P', 'B'),

        'P' => array('B')

    );

    try {

        $pdo->beginTransaction();

        // Get current class

        $stmt = $pdo->prepare("SELECT class FROM users WHERE id = :user\_id");

        $stmt->bindParam(':user\_id', $user\_id);

        $stmt->execute();

        $current\_class = $stmt->fetchColumn();

        // Validate upgrade

        if (!isset($valid\_upgrades[$current\_class]) || !in\_array($data->new\_class, $valid\_upgrades[$current\_class])) {

            $pdo->rollBack();

            http\_response\_code(400);

            echo json\_encode(array("message" => "Invalid class upgrade."));

            exit();

        }

        // Perform upgrade

        $stmt = $pdo->prepare("UPDATE users SET class = :new\_class WHERE id = :user\_id");

        $stmt->bindParam(':new\_class', $data->new\_class);

        $stmt->bindParam(':user\_id', $user\_id);

        $stmt->execute();

        // Record upgrade transaction

        $stmt = $pdo->prepare("

            INSERT INTO class\_upgrades (user\_id, from\_class, to\_class, upgrade\_date)

            VALUES (:user\_id, :from\_class, :to\_class, NOW())

        ");

        $stmt->bindParam(':user\_id', $user\_id);

        $stmt->bindParam(':from\_class', $current\_class);

        $stmt->bindParam(':to\_class', $data->new\_class);

        $stmt->execute();

        $pdo->commit();

        http\_response\_code(200);

        echo json\_encode(array("message" => "Class upgraded successfully.", "new\_class" => $data->new\_class));

    } catch(PDOException $e) {

        $pdo->rollBack();

        http\_response\_code(500);

        echo json\_encode(array("message" => "Error upgrading class: " . $e->getMessage()));

    }

} else {

    http\_response\_code(405);

    echo json\_encode(array("message" => "Method not allowed."));

}

function getBearerToken() {

    $headers = getallheaders();

    if (isset($headers['Authorization'])) {

        if (preg\_match('/Bearer\s(\S+)/', $headers['Authorization'], $matches)) {

            return $matches[1];

        }

    }

    return null;

}

subscription\_system.php

<?php

require\_once 'config.php';

require\_once 'auth\_middleware.php';

header("Content-Type: application/json; charset=UTF-8");

header("Access-Control-Allow-Origin: \*");

header("Access-Control-Allow-Methods: GET, POST");

header("Access-Control-Max-Age: 3600");

header("Access-Control-Allow-Headers: Content-Type, Access-Control-Allow-Headers, Authorization, X-Requested-With");

$token = getBearerToken();

$user\_id = validateToken($token);

if (!$user\_id) {

    http\_response\_code(401);

    echo json\_encode(array("message" => "Invalid or expired token."));

    exit();

}

if ($\_SERVER['REQUEST\_METHOD'] === 'GET') {

    // Fetch subscription and maintenance status

    try {

        $stmt = $pdo->prepare("SELECT s.status as subscription\_status, s.end\_date as subscription\_end\_date,

                               m.status as maintenance\_status, m.last\_payment\_date

                               FROM users u

                               LEFT JOIN subscriptions s ON u.id = s.user\_id

                               LEFT JOIN maintenance\_fees m ON u.id = m.user\_id

                               WHERE u.id = :user\_id

                               ORDER BY s.end\_date DESC, m.last\_payment\_date DESC

                               LIMIT 1");

        $stmt->bindParam(':user\_id', $user\_id);

        $stmt->execute();

        $result = $stmt->fetch(PDO::FETCH\_ASSOC);

        http\_response\_code(200);

        echo json\_encode(array(

            "subscription" => array(

                "status" => $result['subscription\_status'] ?? 'inactive',

                "endDate" => $result['subscription\_end\_date'] ?? null

            ),

            "maintenance" => array(

                "status" => $result['maintenance\_status'] ?? 'unpaid',

                "lastPaymentDate" => $result['last\_payment\_date'] ?? null

            )

        ));

    } catch(PDOException $e) {

        http\_response\_code(500);

        echo json\_encode(array("message" => "Error fetching subscription data: " . $e->getMessage()));

    }

} elseif ($\_SERVER['REQUEST\_METHOD'] === 'POST') {

    $data = json\_decode(file\_get\_contents("php://input"));

    if (!isset($data->action)) {

        http\_response\_code(400);

        echo json\_encode(array("message" => "Action is required."));

        exit();

    }

    if ($data->action === 'subscribe') {

        try {

            $pdo->beginTransaction();

            // Check if user already has an active subscription

            $stmt = $pdo->prepare("SELECT id FROM subscriptions WHERE user\_id = :user\_id AND end\_date > NOW()");

            $stmt->bindParam(':user\_id', $user\_id);

            $stmt->execute();

            if ($stmt->rowCount() > 0) {

                $pdo->rollBack();

                http\_response\_code(400);

                echo json\_encode(array("message" => "User already has an active subscription."));

                exit();

            }

            // Create new subscription

            $stmt = $pdo->prepare("INSERT INTO subscriptions (user\_id, status, start\_date, end\_date)

                                   VALUES (:user\_id, 'active', NOW(), DATE\_ADD(NOW(), INTERVAL 1 YEAR))");

            $stmt->bindParam(':user\_id', $user\_id);

            $stmt->execute();

            $pdo->commit();

            http\_response\_code(200);

            echo json\_encode(array("message" => "Subscription created successfully."));

        } catch(PDOException $e) {

            $pdo->rollBack();

            http\_response\_code(500);

            echo json\_encode(array("message" => "Error creating subscription: " . $e->getMessage()));

        }

    } elseif ($data->action === 'pay\_maintenance') {

        try {

            $pdo->beginTransaction();

            // Check if maintenance fee is already paid for this year

            $stmt = $pdo->prepare("SELECT id FROM maintenance\_fees

                                   WHERE user\_id = :user\_id AND YEAR(last\_payment\_date) = YEAR(CURDATE())");

            $stmt->bindParam(':user\_id', $user\_id);

            $stmt->execute();

            if ($stmt->rowCount() > 0) {

                $pdo->rollBack();

                http\_response\_code(400);

                echo json\_encode(array("message" => "Maintenance fee already paid for this year."));

                exit();

            }

            // Record maintenance fee payment

            $stmt = $pdo->prepare("INSERT INTO maintenance\_fees (user\_id, status, last\_payment\_date, amount)

                                   VALUES (:user\_id, 'paid', NOW(), :amount)");

            $stmt->bindParam(':user\_id', $user\_id);

            $stmt->bindParam(':amount', config('YEARLY\_MAINTENANCE\_FEE'));

            $stmt->execute();

            $pdo->commit();

            http\_response\_code(200);

            echo json\_encode(array("message" => "Maintenance fee paid successfully."));

        } catch(PDOException $e) {

            $pdo->rollBack();

            http\_response\_code(500);

            echo json\_encode(array("message" => "Error paying maintenance fee: " . $e->getMessage()));

        }

    } else {

        http\_response\_code(400);

        echo json\_encode(array("message" => "Invalid action."));

    }

} else {

    http\_response\_code(405);

    echo json\_encode(array("message" => "Method not allowed."));

}

function getBearerToken() {

    $headers = getallheaders();

    if (isset($headers['Authorization'])) {

        if (preg\_match('/Bearer\s(\S+)/', $headers['Authorization'], $matches)) {

            return $matches[1];

        }

    }

    return null;

}

support\_system.php

<?php

require\_once 'config.php';

require\_once 'auth\_middleware.php';

header("Content-Type: application/json; charset=UTF-8");

header("Access-Control-Allow-Origin: \*");

header("Access-Control-Allow-Methods: GET, POST");

header("Access-Control-Max-Age: 3600");

header("Access-Control-Allow-Headers: Content-Type, Access-Control-Allow-Headers, Authorization, X-Requested-With");

$token = getBearerToken();

$user\_id = validateToken($token);

if (!$user\_id) {

    http\_response\_code(401);

    echo json\_encode(array("message" => "Invalid or expired token."));

    exit();

}

if ($\_SERVER['REQUEST\_METHOD'] === 'GET') {

    // Fetch support tickets

    try {

        $stmt = $pdo->prepare("

            SELECT id, subject, message, status, created\_at, updated\_at

            FROM support\_tickets

            WHERE user\_id = :user\_id

            ORDER BY created\_at DESC

        ");

        $stmt->bindParam(':user\_id', $user\_id);

        $stmt->execute();

        $tickets = $stmt->fetchAll(PDO::FETCH\_ASSOC);

        http\_response\_code(200);

        echo json\_encode(array("tickets" => $tickets));

    } catch(PDOException $e) {

        http\_response\_code(500);

        echo json\_encode(array("message" => "Error fetching support tickets: " . $e->getMessage()));

    }

} elseif ($\_SERVER['REQUEST\_METHOD'] === 'POST') {

    // Create a new support ticket

    $data = json\_decode(file\_get\_contents("php://input"));

    if (!isset($data->subject) || !isset($data->message)) {

        http\_response\_code(400);

        echo json\_encode(array("message" => "Subject and message are required."));

        exit();

    }

    try {

        $stmt = $pdo->prepare("

            INSERT INTO support\_tickets (user\_id, subject, message, status, created\_at, updated\_at)

            VALUES (:user\_id, :subject, :message, 'open', NOW(), NOW())

        ");

        $stmt->bindParam(':user\_id', $user\_id);

        $stmt->bindParam(':subject', $data->subject);

        $stmt->bindParam(':message', $data->message);

        $stmt->execute();

        $ticket\_id = $pdo->lastInsertId();

        http\_response\_code(201);

        echo json\_encode(array(

            "message" => "Support ticket created successfully.",

            "ticket\_id" => $ticket\_id

        ));

    } catch(PDOException $e) {

        http\_response\_code(500);

        echo json\_encode(array("message" => "Error creating support ticket: " . $e->getMessage()));

    }

} else {

    http\_response\_code(405);

    echo json\_encode(array("message" => "Method not allowed."));

}

function getBearerToken() {

    $headers = getallheaders();

    if (isset($headers['Authorization'])) {

        if (preg\_match('/Bearer\s(\S+)/', $headers['Authorization'], $matches)) {

            return $matches[1];

        }

    }

    return null;

}

user\_profile.php

<?php

require\_once 'config.php';

require\_once 'auth\_middleware.php';

header("Content-Type: application/json; charset=UTF-8");

header("Access-Control-Allow-Origin: \*");

header("Access-Control-Allow-Methods: GET, PUT");

header("Access-Control-Max-Age: 3600");

header("Access-Control-Allow-Headers: Content-Type, Access-Control-Allow-Headers, Authorization, X-Requested-With");

// Get the JWT token from the Authorization header

$token = getBearerToken();

// Validate the token

$user\_id = validateToken($token);

if (!$user\_id) {

    http\_response\_code(401);

    echo json\_encode(array("message" => "Invalid or expired token."));

    exit();

}

// Handle GET request to fetch user profile

if ($\_SERVER['REQUEST\_METHOD'] === 'GET') {

    try {

        $stmt = $pdo->prepare("SELECT username, email, first\_name, last\_name, country, city, age, member\_code, class FROM users WHERE id = :user\_id");

        $stmt->bindParam(':user\_id', $user\_id);

        $stmt->execute();

        if ($stmt->rowCount() > 0) {

            $user = $stmt->fetch(PDO::FETCH\_ASSOC);

            http\_response\_code(200);

            echo json\_encode(array("user" => $user));

        } else {

            http\_response\_code(404);

            echo json\_encode(array("message" => "User not found."));

        }

    } catch(PDOException $e) {

        http\_response\_code(500);

        echo json\_encode(array("message" => "Error fetching user profile: " . $e->getMessage()));

    }

}

// Handle PUT request to update user profile

elseif ($\_SERVER['REQUEST\_METHOD'] === 'PUT') {

    $data = json\_decode(file\_get\_contents("php://input"));

    $updateFields = array();

    $params = array(':user\_id' => $user\_id);

    // Check which fields are provided and add them to the update query

    $allowedFields = ['first\_name', 'last\_name', 'country', 'city', 'age'];

    foreach ($allowedFields as $field) {

        if (isset($data->$field)) {

            $updateFields[] = "$field = :$field";

            $params[":$field"] = $data->$field;

        }

    }

    if (empty($updateFields)) {

        http\_response\_code(400);

        echo json\_encode(array("message" => "No fields to update."));

        exit();

    }

    try {

        $sql = "UPDATE users SET " . implode(', ', $updateFields) . " WHERE id = :user\_id";

        $stmt = $pdo->prepare($sql);

        $stmt->execute($params);

        if ($stmt->rowCount() > 0) {

            // Fetch the updated user profile

            $stmt = $pdo->prepare("SELECT username, email, first\_name, last\_name, country, city, age, member\_code, class FROM users WHERE id = :user\_id");

            $stmt->bindParam(':user\_id', $user\_id);

            $stmt->execute();

            $updatedUser = $stmt->fetch(PDO::FETCH\_ASSOC);

            http\_response\_code(200);

            echo json\_encode(array("message" => "Profile updated successfully.", "user" => $updatedUser));

        } else {

            http\_response\_code(400);

            echo json\_encode(array("message" => "No changes were made to the profile."));

        }

    } catch(PDOException $e) {

        http\_response\_code(500);

        echo json\_encode(array("message" => "Error updating user profile: " . $e->getMessage()));

    }

}

else {

    http\_response\_code(405);

    echo json\_encode(array("message" => "Method not allowed."));

}

function getBearerToken() {

    $headers = getallheaders();

    if (isset($headers['Authorization'])) {

        if (preg\_match('/Bearer\s(\S+)/', $headers['Authorization'], $matches)) {

            return $matches[1];

        }

    }

    return null;

}