Project Code

3. vite.config.js

```
import { defineConfig } from 'vite'
import react from '@vitejs/plugin-react'
import tailwindcss from '@tailwindcss/vite'

// https://vite.dev/config/
export default defineConfig({
  plugins: [react(),
    tailwindcss()
  ],
})
```

5. backend/config/db.js

```
import mongoose from 'mongoose';

const connectDB = async () => {
  try {
    await mongoose.connect(process.env.MONGO_URI);
    console.log('MongoDB connected');
  } catch (err) {
    console.error('MongoDB connection failed:', err.message);
  }
};

export default connectDB;
```

2. index.html

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <link rel="icon" type="image/svg+xml" href="/vite.svg" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Vite + React</title>
  </head>
  <body>
    <div id="root"></div>
    <script type="module" src="/src/main.jsx"></script>
  </body>
</html>
```

4. backend/index.js

```
const express = require('express');
const mongoose = require('mongoose');
const cors = require('cors');
const authRoutes = require('./routes/auth');
require('dotenv').config();

const app = express();
app.use(cors());
app.use(express.json());

// <-- Add this here for request logging:
app.use((req, res, next) => {
  console.log(`Incoming ${req.method} request to ${req.url}`);
  console.log('Request body:', req.body);
  next();
});

mongoose.connect(process.env.MONGO_URL)
  .then(() => console.log('MongoDB connected'))
  .catch((err) => console.log('MongoDB connection error:', err));

// Mount auth routes at /api/auth
app.use('/api/auth', authRoutes);

const PORT = 5000;
app.listen(PORT, () => {
  console.log(`Server running on port ${PORT}`);
});

const planRoutes = require('./routes/plan');
app.use('/api', planRoutes);
```

1. eslint.config.js

```
import js from '@eslint/js'
import globals from 'globals'
import reactHooks from 'eslint-plugin-react-hooks'
import reactRefresh from 'eslint-plugin-react-refresh'
import { defineConfig, globalIgnores } from 'eslint/config'

export default defineConfig([
  globalIgnores(['dist']),
  {
    files: ['**/*.{js,jsx}'],
```

```
    extends: [
      js.configs.recommended,
      reactHooks.configs['recommended-latest'],
      reactRefresh.configs.vite,
    ],
    languageOptions: {
      ecmaVersion: 2020,
      globals: globals.browser,
      parserOptions: {
        ecmaVersion: 'latest',
        ecmaFeatures: { jsx: true },
        sourceType: 'module',
      },
    },
    rules: {
      'no-unused-vars': ['error', { varsIgnorePattern: '^[A-Z_]' }],
    },
  },
])
```

6. backend/models/User.js

```
const mongoose = require('mongoose');

const UserSchema = new mongoose.Schema({
  name: { type: String, required: true },
  email: { type: String, required: true, unique: true },
  password: { type: String, required: true },
});

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

module.exports = User;
```

7. backend/routes/auth.js

```
const express = require('express');
const User = require('../models/User');
const router = express.Router();

// Register route
router.post('/register', async (req, res) => {
  try {
    const { name, email, password } = req.body;
    console.log("Register data:", req.body);
```

```
    if (!name || !email || !password) {
      return res.status(400).json({ msg: "Please fill in all fields" });
    }

    const existing = await User.findOne({ email });
    if (existing) return res.status(400).json({ msg: 'User already exists' });

    const newUser = new User({ name, email, password });
    await newUser.save();

    return res.status(201).json({
      msg: 'User registered successfully',
      user: {
        _id: newUser._id,
        name: newUser.name,
        email: newUser.email,
      }
    });

  } catch (err) {
    console.error('Error in /register:', err);
    return res.status(500).json({ msg: 'Server error', error: err.message });
  }
});

// Login route
router.post('/login', async (req, res) => {
  try {
    const { email, password } = req.body;
    console.log('Login attempt:', email, password);

    if (!email || !password) {
      return res.status(400).json({ msg: 'Please fill in all fields' });
    }

    const user = await User.findOne({ email });
    if (!user) {
      return res.status(401).json({ msg: 'User not found' });
    }

    if (user.password !== password) {
      return res.status(401).json({ msg: 'Incorrect password' });
    }

    return res.status(200).json({
      msg: 'Login successful',
```

```
      user: {
        _id: user._id,
        name: user.name,
        email: user.email,
      }
    });

  } catch (err) {
    console.error('  Error in /login:', err.message);
    return res.status(500).json({ msg: 'Server error', error: err.message });
  }
});

module.exports = router;
```

10. src/App.jsx

```
// App.jsx
import React from 'react';
import { BrowserRouter, Routes, Route } from 'react-router-dom';

import AuthForm from './components/AuthForm';
import UserDetailsForm from './pages/UserDetailsForm';
import Dashboard from './pages/Dashboard'; //   IMPORT THIS

function App() {
  return (
    <BrowserRouter>
      <Routes>
        <Route path="/" element={<AuthForm />} />
        <Route path="/form" element={<UserDetailsForm />} />
        <Route path="/dashboard" element={<Dashboard />} /> {/*   NEW ROUTE */}
      </Routes>
    </BrowserRouter>
  );
}

export default App;
```

9. src/App.css

```
#root {
  max-width: 1280px;
  margin: 0 auto;
  padding: 2rem;
  text-align: center;
```

```
}

.logo {
  height: 6em;
  padding: 1.5em;
  will-change: filter;
  transition: filter 300ms;
}
.logo:hover {
  filter: drop-shadow(0 0 2em #646cffaa);
}
.logo.react:hover {
  filter: drop-shadow(0 0 2em #61dafbaa);
}

@keyframes logo-spin {
  from {
    transform: rotate(0deg);
  }
  to {
    transform: rotate(360deg);
  }
}

@media (prefers-reduced-motion: no-preference) {
  a:nth-of-type(2) .logo {
    animation: logo-spin infinite 20s linear;
  }
}

.card {
  padding: 2em;
}

.read-the-docs {
  color: #888;
}
```

8. backend/routes/plan.js

```
const express = require('express');
const router = express.Router();
const { GoogleGenerativeAI } = require('@google/generative-ai');

const genAI = new GoogleGenerativeAI(process.env.GEMINI_API_KEY);
```

```
router.post('/generate-plan', async (req, res) => {
  try {
    const user = req.body;

    const prompt = `
      Act as an expert Indian fitness coach and nutritionist.
      Create a personalized fitness and diet plan for an Indian user with the following deta
      Format the entire response using markdown.

      **User Details:**
      - **Name:** ${user.name}
      - **Age:** ${user.age}
      - **Gender:** ${user.gender}
      - **Height:** ${user.height} cm
      - **Weight:** ${user.weight} kg
      - **Activity Level:** ${user.activity}
      - **Primary Goal:** ${user.goal}

      Please generate a comprehensive plan with the following sections, using these exact he

      # Your Personalized Plan for ${user.name}

      ## 1. Workout Routine
      Provide a 3-day sample workout plan. Include exercises, sets, and reps.

      ## 2. Meal Plan
      Provide a sample one-day meal plan (Breakfast, Lunch, Dinner, Snacks). Focus on commor

      ## 3. Lifestyle Tips
      Provide 3-5 actionable lifestyle tips to help achieve the user's goal.
    `;

    const model = genAI.getGenerativeModel({ model: "gemini-2.0-flash" });

    const result = await model.generateContent(prompt);
    const text = result.text;

    res.json({ plan: text });
  } catch (err) {
    console.error('Gemini API Error:', err);
    res.status(500).json({ msg: 'Failed to generate plan', error: err.message });
  }
});

module.exports = router;
```

12. src/main.jsx

```jsx
// main.jsx
import React from 'react';
import ReactDOM from 'react-dom/client';
import App from './App.jsx';
import './index.css'; // Tailwind import

ReactDOM.createRoot(document.getElementById('root')).render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
```

11. src/index.css

```css
@import "tailwindcss";

:root {
  font-family: system-ui, Avenir, Helvetica, Arial, sans-serif;
  line-height: 1.5;
  font-weight: 400;

  color-scheme: light dark;
  color: rgba(255, 255, 255, 0.87);
  background-color: #242424;

  font-synthesis: none;
  text-rendering: optimizeLegibility;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
}

a {
  font-weight: 500;
  color: #646cff;
  text-decoration: inherit;
}
a:hover {
  color: #535bf2;
}

body {
  margin: 0;
  display: flex;
  place-items: center;
```

```
  min-width: 320px;
  min-height: 100vh;
}

h1 {
  font-size: 3.2em;
  line-height: 1.1;
}

button {
  border-radius: 8px;
  border: 1px solid transparent;
  padding: 0.6em 1.2em;
  font-size: 1em;
  font-weight: 500;
  font-family: inherit;
  background-color: #1a1a1a;
  cursor: pointer;
  transition: border-color 0.25s;
}
button:hover {
  border-color: #646cff;
}
button:focus,
button:focus-visible {
  outline: 4px auto -webkit-focus-ring-color;
}

@media (prefers-color-scheme: light) {
  :root {
    color: #213547;
    background-color: #ffffff;
  }
  a:hover {
    color: #747bff;
  }
  button {
    background-color: #f9f9f9;
  }
}
```

13. src/components/AuthForm.jsx

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

```
function AuthForm() {
  const [isSignup, setIsSignup] = useState(false);
  const [formData, setFormData] = useState({
    name: '',
    email: '',
    password: '',
  });

  const navigate = useNavigate();

  const handleChange = (e) => {
    setFormData((prev) => ({
      ...prev,
      [e.target.name]: e.target.value,
    }));
  };

const handleSubmit = async (e) => {
  e.preventDefault();

  const endpoint = isSignup ? '/api/auth/register' : '/api/auth/login';

  try {
    const res = await fetch(`${import.meta.env.VITE_API_BASE_URL}${endpoint}`, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify(formData),
    });

    const data = await res.json();

    if (!res.ok) {
      alert(data.msg || 'Something went wrong');
      return;
    }

    console.log('Success:', data);
    navigate('/form');
  } catch (err) {
    console.error('Error:', err);
    alert('Server error');
  }
};
```

```
return (
  <div className="min-h-screen w-screen flex items-center justify-center bg-gray-900">
    <div className="w-full max-w-md p-8 bg-white rounded-lg shadow-md">
      <form onSubmit={handleSubmit}>
        <h2 className="text-2xl font-bold mb-6 text-center text-gray-800">
          {isSignup ? 'Sign Up' : 'Login'}
        </h2>

        {isSignup && (
          <input
            type="text"
            name="name"
            placeholder="Name"
            value={formData.name}
            onChange={handleChange}
            required
            className="w-full mb-4 p-3 border rounded focus:outline-none focus:ring-2 focu
          />
        )}

        <input
          type="email"
          name="email"
          placeholder="Email"
          value={formData.email}
          onChange={handleChange}
          required
          className="w-full mb-4 p-3 border rounded focus:outline-none focus:ring-2 focus
        />

        <input
          type="password"
          name="password"
          placeholder="Password"
          value={formData.password}
          onChange={handleChange}
          required
          className="w-full mb-4 p-3 border rounded focus:outline-none focus:ring-2 focus
        />

        <button
          type="submit"
          className="w-full bg-blue-600 text-white py-3 rounded hover:bg-blue-700 transiti
        >
          {isSignup ? 'Create Account' : 'Login'}
```

```
          </button>

          <p
            onClick={() => setIsSignup(!isSignup)}
            className="mt-4 text-center text-sm text-gray-600 cursor-pointer hover:text-blue
          >
            {isSignup
              ? 'Already have an account? Login'
              : "Don't have an account? Sign Up"}
          </p>
        </form>
      </div>
    </div>
  );
}

export default AuthForm;
```

15. src/pages/Dashboard.jsx

```
import React, { useState, useEffect } from 'react';
import { useLocation, useNavigate } from 'react-router-dom';
import ReactMarkdown from 'react-markdown';
import { Menu, Dumbbell, Utensils, Heart, Sun, Zap, Coffee, Leaf } from 'lucide-react';
import Sidebar from '../components/Sidebar';

const navLinks = [
  { to: '/dashboard', text: 'Home', icon: Leaf },
  { to: '/workouts', text: 'Workouts', icon: Dumbbell },
  { to: '/meals', text: 'Meals', icon: Utensils },
];

// Parse markdown into sections cleanly
const parsePlan = (markdown) => {
  if (!markdown) return null;
  const sections = { title: '', workout: '', mealPlan: '', lifestyle: '' };
  let currentSection = '';

  markdown.split('\n').forEach(line => {
    if (line.startsWith('# ')) {
      sections.title = line.slice(2).trim();
    } else if (line.startsWith('## 1. Workout Routine')) {
      currentSection = 'workout';
    } else if (line.startsWith('## 2. Meal Plan')) {
      currentSection = 'mealPlan';
    } else if (line.startsWith('## 3. Lifestyle Tips')) {
```

```
      currentSection = 'lifestyle';
    } else if (currentSection) {
      sections[currentSection] += line + '\n';
    }
  });
  return sections;
};

const Dashboard = () => {
  const location = useLocation();
  const navigate = useNavigate();
  const [isSidebarOpen, setSidebarOpen] = useState(false);

  // Plan from router or session storage fallback
  const planFromState = location.state?.plan;
  const [generatedPlan, setGeneratedPlan] = useState(planFromState || null);
  const [parsedPlan, setParsedPlan] = useState(null);
  const [activeWorkoutDay, setActiveWorkoutDay] = useState('');

  useEffect(() => {
    if (!generatedPlan) {
      const savedPlan = sessionStorage.getItem('generatedPlan');
      if (savedPlan) setGeneratedPlan(savedPlan);
      else navigate('/');
    } else {
      sessionStorage.setItem('generatedPlan', generatedPlan);
    }
  }, [generatedPlan, navigate]);

  useEffect(() => {
    if (generatedPlan) {
      const planData = parsePlan(generatedPlan);
      setParsedPlan(planData);

      const dayMatches = [...(planData.workout.matchAll(/Day\s*\d+/g))];
      if (dayMatches.length > 0) setActiveWorkoutDay(dayMatches[0][0]);
    }
  }, [generatedPlan]);

  if (!parsedPlan) {
    return (
      <div className="bg-gray-800 min-h-screen flex items-center justify-center text-white">
        Loading your plan...
      </div>
    );
  }
```

```
const { title, workout, mealPlan, lifestyle } = parsedPlan;

// Split workout by days keeping delimiter
const workoutDayBlocks = workout.split(/(?=Day\s*\d+)/);
const workoutDays = workoutDayBlocks.map(block => block.split('\n')[0].trim());

const mealIcons = {
  Breakfast: <Sun className="text-yellow-400" />,
  Lunch: <Zap className="text-orange-400" />,
  Snack: <Coffee className="text-amber-600" />,
  Dinner: <Heart className="text-red-400" />,
};

const activeWorkoutContent = workoutDayBlocks.find(block => block.startsWith(activeWorkout

return (
  <div className="relative min-h-screen w-full bg-gray-800 flex">
    <Sidebar navLinks={navLinks} isOpen={isSidebarOpen} />
    {isSidebarOpen && (
      <div
        onClick={() => setSidebarOpen(false)}
        className="fixed inset-0 bg-black/60 z-40 md:hidden"
      />
    )}

    <main
      className={`flex-1 transition-all duration-300 ease-in-out p-6 md:p-10 overflow-y-au
        isSidebarOpen ? 'md:ml-64' : 'ml-0'
      }`}
      style={{ minHeight: '100vh' }}
    >
      <button
        onClick={() => setSidebarOpen(!isSidebarOpen)}
        className="mb-6 p-2 rounded-md bg-gray-700 text-white hover:bg-gray-600 transitior
        aria-label="Toggle sidebar"
      >
        <Menu size={24} />
      </button>

      <header className="mb-10">
        <h1 className="text-4xl md:text-5xl font-bold text-white tracking-tight">{title ||
        <p className="text-lg text-gray-400 mt-2">
          Here is your personalized roadmap to success. Let's get started!
        </p>
      </header>
```

```
<div className="grid grid-cols-1 lg:grid-cols-3 gap-8">
  {/* Workout Card */}
  <section className="lg:col-span-2 bg-gray-900/70 backdrop-blur-sm border border-gr
    <h2 className="text-2xl font-bold text-white mb-5 flex items-center gap-3">
      <Dumbbell className="text-emerald-400" />
      Workout Routine
    </h2>

    <div className="flex space-x-2 border-b border-gray-700 mb-5 overflow-x-auto pb-
      {workoutDays.map(day => (
        <button
          key={day}
          onClick={() => setActiveWorkoutDay(day)}
          className={`px-4 py-2 text-sm font-medium rounded-t-lg flex-shrink-0 trans
            activeWorkoutDay === day
              ? 'bg-emerald-600 text-white'
              : 'text-gray-400 hover:bg-gray-700'
          }`}
        >
          {day}
        </button>
      ))}
    </div>

    <article className="prose prose-invert max-w-none text-gray-300 whitespace-pre-l
      <ReactMarkdown
        components={{
          table: ({ node, ...props }) => (
            <div className="overflow-x-auto">
              <table className="w-full text-left min-w-[500px]" {...props} />
            </div>
          ),
          thead: ({ node, ...props }) => <thead className="border-b border-gray-600"
          th: ({ node, ...props }) => <th className="p-2 font-semibold" {...props} /
          tbody: ({ node, ...props }) => <tbody className="divide-y divide-gray-700"
          tr: ({ node, ...props }) => <tr className="hover:bg-gray-800" {...props} /
          td: ({ node, ...props }) => <td className="p-2" {...props} />,
        }}
      >
        {activeWorkoutContent.replace(activeWorkoutDay, '').trim()}
      </ReactMarkdown>
    </article>
  </section>

  {/* Meal Plan Card */}
```

```
          <section className="lg:col-span-1 bg-gray-900/70 backdrop-blur-sm border border-gr
            <h2 className="text-2xl font-bold text-white mb-5 flex items-center gap-3">
              <Utensils className="text-emerald-400" />
              Daily Meal Plan
            </h2>

            <div className="space-y-5">
              <ReactMarkdown
                components={{
                  h3: ({ node, ...props }) => {
                    const mealName = String(props.children[0]);
                    return (
                      <h3 className="font-bold text-emerald-300 flex items-center gap-2" {..
                        {mealIcons[mealName.split(' ')[0]] || <Leaf />} {mealName}
                      </h3>
                    );
                  },
                  p: ({ node, ...props }) => (
                    <p className="text-gray-400 pl-6 border-l-2 border-gray-700" {...props}
                  ),
                }}
              >
                {mealPlan}
              </ReactMarkdown>
            </div>
          </section>

          {/* Lifestyle Tips Card */}
          <section className="lg:col-span-3 bg-gray-900/70 backdrop-blur-sm border border-gr
            <h2 className="text-2xl font-bold text-white mb-5 flex items-center gap-3">
              <Heart className="text-emerald-400" />
              Lifestyle Tips
            </h2>
            <ReactMarkdown
              components={{
                ol: ({ node, ...props }) => <ol className="list-decimal list-inside space-y-
                li: ({ node, ...props }) => <li className="text-gray-300" {...props} />,
              }}
            >
              {lifestyle}
            </ReactMarkdown>
          </section>
        </div>
      </main>
    </div>
  );
```

```
};

export default Dashboard;
```

14. src/components/Sidebar.jsx

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

// ================================================================================
// Mocking dependencies for this self-contained example.
// In your project, you'll use the actual 'react-router-dom', 'lucide-react', and 'marked' p
// ================================================================================
const useLocation = () => ({
  pathname: '/dashboard',
  state: { plan: '## Weekly Goal: Strength Gain\n\n* **Monday:** Upper Body Focus\n* **Wedne
});
const Link = ({ to, children, className, onClick }) => <a href={to} className={className} on
const Home = () => <svg xmlns="http://www.w3.org/2000/svg" width="24" height="24" viewBox="0
const BarChart3 = () => <svg xmlns="http://www.w3.org/2000/svg" width="24" height="24" viewB
const Utensils = () => <svg xmlns="http://www.w3.org/2000/svg" width="24" height="24" viewBo
const Dumbbell = () => <svg xmlns="http://www.w3.org/2000/svg" width="24" height="24" viewBo
const Menu = () => <svg xmlns="http://www.w3.org/2000/svg" width="24" height="24" viewBox="0
const X = () => <svg xmlns="http://www.w3.org/2000/svg" width="24" height="24" viewBox="0 0
const LogOut = () => <svg xmlns="http://www.w3.org/2000/svg" width="24" height="24" viewBox=

const marked = {
  parse: (text) => {
    if (typeof text !== 'string') return '';
    return text
      .replace(/## (.*)/g, '<h2 class="text-2xl font-bold mb-4 text-gray-800">$1</h2>')
      .replace(/\* (.*)/g, '<li class="ml-5 list-disc text-gray-700">$1</li>');
  }
};

// ================================================================================
// Sidebar.jsx - Updated to accept state from its parent
// ================================================================================
export const Sidebar = ({ isOpen, setIsOpen }) => {
  const location = useLocation();

  // This function now closes the sidebar by updating the state in the parent component
  const handleClose = () => {
    setIsOpen(false);
  };

  const navLinks = [
```

```
  { to: "/dashboard", icon: Home, text: "Dashboard" },
  { to: "/meal-plan", icon: Utensils, text: "Meal Plan" },
  { to: "/workout-plan", icon: Dumbbell, text: "Workout Plan" },
  { to: "/progress", icon: BarChart3, text: "Progress" },
];

return (
  <>
    {/* --- Overlay (Mobile) --- */}
    {/* This now also closes the sidebar when clicked, which is good for mobile usability
    <div
      onClick={handleClose}
      className={`fixed inset-0 bg-black transition-opacity duration-300
                  ${isOpen ? 'opacity-50 z-30 md:hidden' : 'opacity-0 -z-10'}`}
      aria-hidden="true"
    ></div>

    {/* --- Sidebar Panel --- */}
    <aside
      className={`fixed inset-y-0 left-0 w-64 bg-gray-900 text-white p-6
                transform ${isOpen ? 'translate-x-0' : '-translate-x-full'}
                transition-transform duration-300 ease-in-out z-40 flex flex-col`}
    >
      <div className="flex items-center justify-between mb-10">
        {/* FIX: Added 'break-words' to prevent the title from overflowing its container

        {/* This close button is inside the sidebar, useful for mobile */}
        <button onClick={handleClose} className="md:hidden p-1 rounded-full hover:bg-gray-
          <X />
        </button>
      </div>

      <nav className="flex-grow">
        <ul className="space-y-3">
          {navLinks.map((link) => (
            <li key={link.to}>
              <Link
                to={link.to}
                onClick={handleClose} // Close sidebar on link click for better mobile UX
                className={`flex items-center gap-4 p-3 rounded-lg transition-colors
                                ${location.pathname === link.to
                                  ? 'bg-emerald-600 text-white'
                                  : 'text-gray-400 hover:bg-gray-700 hover:text-white'
                                }`}
              >
                <link.icon />
```

```
                <span className="font-medium">{link.text}</span>
              </Link>
            </li>
          ))}
        </ul>
      </nav>

      <div className="mt-auto">
          <Link
            to="/logout"
            onClick={handleClose}
            className="flex items-center gap-4 p-3 rounded-lg text-gray-400 hover:bg-red-6
          >
            <LogOut />
            <span className="font-medium">Logout</span>
          </Link>
      </div>
    </aside>
  </>
 );
};
```

16. src/pages/UserDetailsForm.jsx

```
import { useState } from 'react';
import axios from 'axios';
import { useNavigate } from 'react-router-dom';
import ReactMarkdown from 'react-markdown';

function UserDetailsForm() {
  const [formData, setFormData] = useState({
    name: '',
    age: '',
    gender: '',
    height: '',
    weight: '',
    activity: 'sedentary',
    goal: 'weight_loss',
    dietaryPreference: 'vegetarian',
  });

  const [generatedPlan, setGeneratedPlan] = useState('');
  const [loading, setLoading] = useState(false);
  const navigate = useNavigate();
```

```
const handleChange = (e) => {
  setFormData({ ...formData, [e.target.name]: e.target.value });
};

const handleSubmit = async (e) => {
  e.preventDefault();
  setLoading(true);
  setGeneratedPlan('');

  try {
    const response = await axios.post('http://localhost:5000/api/generate-plan', formData)
    setGeneratedPlan(response.data.plan);
  } catch (error) {
    console.error('Error generating plan:', error);
    setGeneratedPlan('# Error\n\nSorry, something went wrong while generating the plan. Pl
  } finally {
    setLoading(false);
  }
};

const handleFinalize = () => {
  navigate('/dashboard', { state: { plan: generatedPlan } });
};

const handleEdit = () => {
  setGeneratedPlan('');
};

// <- Moved inside component, this was missing
const markdownComponents = {
  h1: ({ node, ...props }) => <h1 className="text-3xl font-bold text-white my-4" {...props
  h2: ({ node, ...props }) => <h2 className="text-2xl font-semibold text-white my-3" {...p
  h3: ({ node, ...props }) => <h3 className="text-xl font-semibold text-teal-300 my-2" {..
  p: ({ node, ...props }) => <p className="text-gray-300 mb-2 leading-relaxed" {...props}
  ul: ({ node, ...props }) => <ul className="list-disc list-inside pl-4 mb-4" {...props} /
  li: ({ node, ...props }) => <li className="text-gray-300 mb-1" {...props} />,
  strong: ({ node, ...props }) => <strong className="font-bold text-white" {...props} />,
};

return (
  <div className="min-h-screen w-screen flex bg-gray-900 text-white">
    {/* Left - Form Section */}
    <div className="w-full md:w-1/2 flex items-center justify-center p-6">
      <div className="w-full max-w-xl bg-gray-800 p-8 rounded-lg shadow-2xl">
        {!generatedPlan ? (
```

```
<>
  <h2 className="text-3xl font-bold text-center mb-6">Create Your Personalized P
  <form onSubmit={handleSubmit} className="space-y-4">
    <input
      type="text"
      name="name"
      placeholder="Name"
      value={formData.name}
      onChange={handleChange}
      required
      className="w-full p-3 bg-gray-700 border border-gray-600 rounded-md focus:
    />
    <div className="grid grid-cols-1 md:grid-cols-2 gap-4">
      <input
        type="number"
        name="age"
        placeholder="Age"
        value={formData.age}
        onChange={handleChange}
        required
        className="w-full p-3 bg-gray-700 border border-gray-600 rounded-md focu
      />
      <select
        name="gender"
        value={formData.gender}
        onChange={handleChange}
        required
        className="w-full p-3 bg-gray-700 border border-gray-600 rounded-md focu
      >
        <option value="">Select Gender</option>
        <option value="male">Male</option>
        <option value="female">Female</option>
        <option value="other">Other</option>
      </select>
    </div>
    <div className="grid grid-cols-1 md:grid-cols-2 gap-4">
      <input
        type="number"
        name="height"
        placeholder="Height (cm)"
        value={formData.height}
        onChange={handleChange}
        required
        className="w-full p-3 bg-gray-700 border border-gray-600 rounded-md focu
      />
      <input
```

```
    type="number"
    name="weight"
    placeholder="Weight (kg)"
    value={formData.weight}
    onChange={handleChange}
    required
    className="w-full p-3 bg-gray-700 border border-gray-600 rounded-md focu
  />
</div>
<select
  name="activity"
  value={formData.activity}
  onChange={handleChange}
  className="w-full p-3 bg-gray-700 border border-gray-600 rounded-md focus
>
  <option value="">Activity Level</option>
  <option value="sedentary">Sedentary (little or no exercise)</option>
  <option value="light">Lightly active</option>
  <option value="moderate">Moderately active</option>
  <option value="very">Very active</option>
</select>
<select
  name="goal"
  value={formData.goal}
  onChange={handleChange}
  className="w-full p-3 bg-gray-700 border border-gray-600 rounded-md focus
>
  <option value="">Select Goal</option>
  <option value="weight_loss">Weight Loss</option>
  <option value="muscle_gain">Muscle Gain</option>
  <option value="maintenance">Fitness Maintenance</option>
</select>
<select
  name="dietaryPreference"
  value={formData.dietaryPreference}
  onChange={handleChange}
  className="w-full p-3 bg-gray-700 border border-gray-600 rounded-md focus
>
  <option value="">Dietary Preference</option>
  <option value="vegetarian">Vegetarian</option>
  <option value="non_vegetarian">Non-Vegetarian</option>
  <option value="vegan">Vegan</option>
</select>
<button
  type="submit"
  className="w-full bg-blue-600 hover:bg-blue-700 text-white font-bold py-3
```

```
                disabled={loading}
              >
                {loading ? 'Generating Your Plan...' : 'Generate Plan'}
              </button>
            </form>
          </>
        ) : (
          <div>
            <h3 className="text-2xl font-bold mb-4 text-center">Your Personalized Plan</h3
            <div className="prose prose-invert max-w-none max-h-[60vh] overflow-y-auto p-4
              <ReactMarkdown components={markdownComponents}>
                {generatedPlan}
              </ReactMarkdown>
            </div>
            <div className="flex gap-4 justify-center mt-6">
              <button
                className="bg-green-600 hover:bg-green-700 text-white font-bold px-6 py-2
                onClick={handleFinalize}
              >
                Finalize & Start
              </button>
              <button
                className="bg-yellow-500 hover:bg-yellow-600 text-white font-bold px-6 py-
                onClick={handleEdit}
              >
                Edit Details
              </button>
            </div>
          </div>
        )}
      </div>
    </div>

    {/* Right Section - Preview (Only shows when plan is generated) */}
    <div className="hidden md:flex w-1/2 bg-black p-6 items-center justify-center">
      {generatedPlan ? (
        <div className="max-w-2xl max-h-[80vh] overflow-y-auto p-6 rounded-lg border borde
          <ReactMarkdown components={markdownComponents}>
            {generatedPlan}
          </ReactMarkdown>
        </div>
      ) : (
        <h2 className="text-2xl font-bold text-white">Plan Preview</h2>
      )}
    </div>
  </div>
```

```
  );
}

export default UserDetailsForm;
```