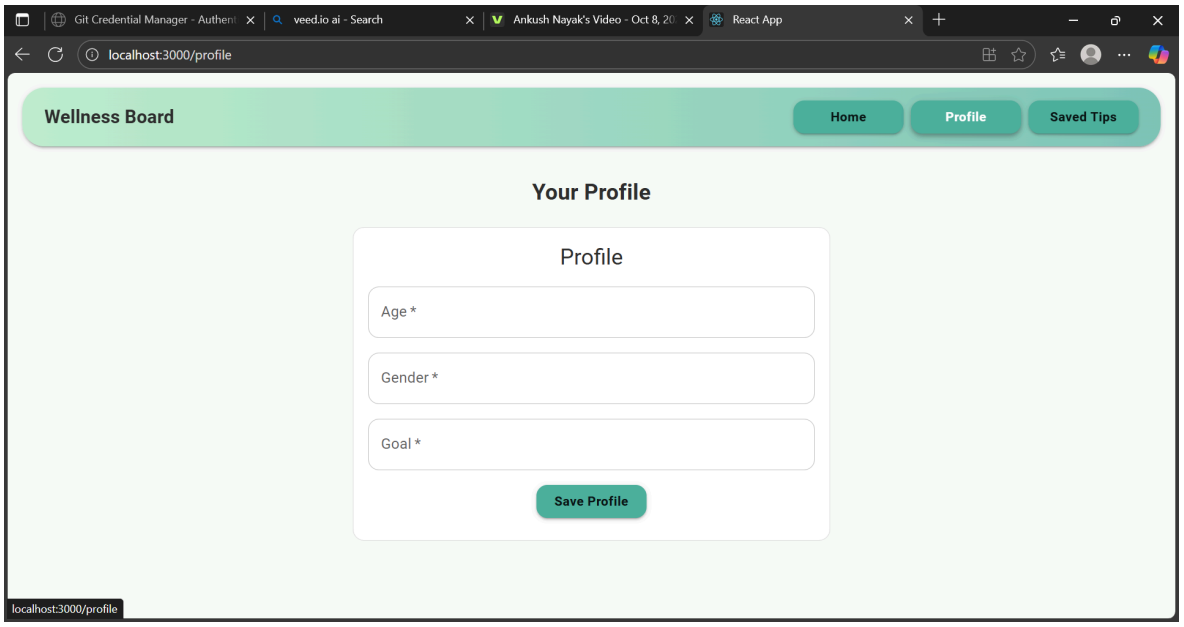
AI-Generated Wellness Recommendation Board (MERN + Gemini)

A modern, responsive wellness assistant that generates personalized health tips using Google's Gemini, with a React + Material UI frontend and an Express + MongoDB backend.

1) Project Setup & Demo

Live Link : https://drive.google.com/drive/folders/1GOCqfumUeUPlCKqr0TO8V5f2z2T2vKOp



Wellness Board Home Profile Saved Tips

Generate Tips

#1
Power Up with Protein
Protein

#2
Lift Smart, Grow Big
Weights

#3
Sleep Your Way Strong
Sleep

#4
Stay Hydrated, Build More
Water

#5
Consistency Fuels Muscle Gains
Train

Power Up with Protein

Powering up with protein is absolutely fundamental when your goal is to build muscle, as protein is the essential building block for all muscle tissue. When you engage in resistance training, you create microscopic tears in your muscle fibers. It's during the recovery process that these fibers are repaired and rebuilt, becoming stronger and larger – a process known as muscle hypertrophy. Protein supplies the necessary amino acids, which are the raw materials your body uses for this crucial repair and growth. Without an adequate and consistent supply of protein, your body simply cannot effectively repair the damage from your workouts or build new muscle, significantly hindering your progress despite your best efforts in the gym. To effectively build muscle, it's not just about consuming protein, but consuming enough of it consistently. A general guideline for muscle building is to aim for approximately 1.6 to 2.2 grams of protein per kilogram of body weight, or roughly 0.7 to 1 gram per pound of body weight, daily. This intake should be distributed throughout your day rather than consumed in one or two large sittings. Spreading your protein intake helps maintain a positive nitrogen balance, which is optimal for continuous muscle protein synthesis and recovery. Focus on incorporating high-quality protein sources into every meal. Excellent options include lean meats like chicken breast, turkey, and lean beef, as well as fish, eggs, dairy products such as Greek yogurt and cottage cheese, and plant-based sources like lentils, beans, and tofu. While whole foods should be your primary focus, protein supplements like whey or casein powder can be a convenient and effective way to meet your daily targets, especially around workouts or when you're short on time. Aim to consume a protein-rich meal or snack within an hour or two after your training sessions to kickstart the recovery process.

1. Calculate your daily protein target: Multiply your body weight in pounds by 0.8 to 1.0 to get a good starting range for your daily protein intake in grams.
2. Include a protein source in every main meal: Ensure your breakfast, lunch, and dinner each contain at least 20-30 grams of protein (e.g., 3-4 eggs, a chicken breast, a serving of fish).
3. Choose high-quality whole food proteins: Prioritize lean meats, poultry, fish, eggs, Greek yogurt, cottage cheese, and legumes as your primary protein sources.
4. Incorporate protein-rich snacks: Keep easy options like a handful of almonds, a hard-boiled egg, a small container of Greek yogurt, or a protein shake handy between meals.
5. Consume protein post-workout: Within 1-2 hours after your training session, have a meal or snack containing 20-40 grams of protein to aid muscle recovery and growth.

Save Tip

Saved Tips

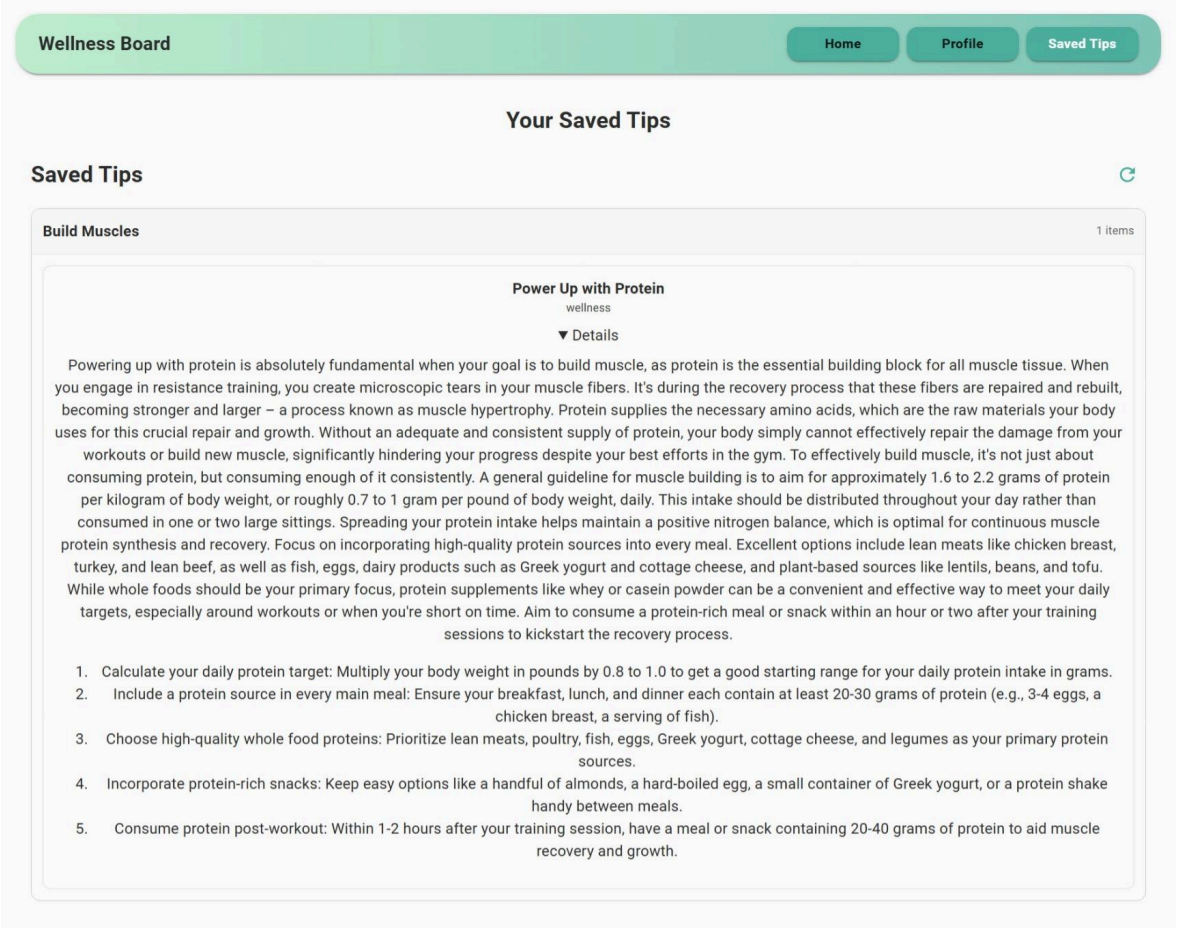
Build Muscles 1 items

Power Up with Protein
wellness

▼ Details

Powering up with protein is absolutely fundamental when your goal is to build muscle, as protein is the essential building block for all muscle tissue. When you engage in resistance training, you create microscopic tears in your muscle fibers. It's during the recovery process that these fibers are repaired and rebuilt, becoming stronger and larger – a process known as muscle hypertrophy. Protein supplies the necessary amino acids, which are the raw materials your body uses for this crucial repair and growth. Without an adequate and consistent supply of protein, your body simply cannot effectively repair the damage from your workouts or build new muscle, significantly hindering your progress despite your best efforts in the gym. To effectively build muscle, it's not just about consuming protein, but consuming enough of it consistently. A general guideline for muscle building is to aim for approximately 1.6 to 2.2 grams of protein per kilogram of body weight, or roughly 0.7 to 1 gram per pound of body weight, daily. This intake should be distributed throughout your day rather than consumed in one or two large sittings. Spreading your protein intake helps maintain a positive nitrogen balance, which is optimal for continuous muscle protein synthesis and recovery. Focus on incorporating high-quality protein sources into every meal. Excellent options include lean meats like chicken breast, turkey, and lean beef, as well as fish, eggs, dairy products such as Greek yogurt and cottage cheese, and plant-based sources like lentils, beans, and tofu. While whole foods should be your primary focus, protein supplements like whey or casein powder can be a convenient and effective way to meet your daily targets, especially around workouts or when you're short on time. Aim to consume a protein-rich meal or snack within an hour or two after your training sessions to kickstart the recovery process.

1. Calculate your daily protein target: Multiply your body weight in pounds by 0.8 to 1.0 to get a good starting range for your daily protein intake in grams.
2. Include a protein source in every main meal: Ensure your breakfast, lunch, and dinner each contain at least 20-30 grams of protein (e.g., 3-4 eggs, a chicken breast, a serving of fish).
3. Choose high-quality whole food proteins: Prioritize lean meats, poultry, fish, eggs, Greek yogurt, cottage cheese, and legumes as your primary protein sources.
4. Incorporate protein-rich snacks: Keep easy options like a handful of almonds, a hard-boiled egg, a small container of Greek yogurt, or a protein shake handy between meals.
5. Consume protein post-workout: Within 1-2 hours after your training session, have a meal or snack containing 20-40 grams of protein to aid muscle recovery and growth.



Prerequisites
- Node.js 18+
- npm 9+
- A MongoDB deployment (Atlas or local)
- A Google Gemini API key

Quick Start (Web)
1) Server
```bash
cd server
npm install
# Create .env with your secrets (see Environment Variables)
npm run dev
```
2) Client
```bash
cd client
npm install
npm start
```
The client runs on http://localhost:3000 and talks to the server at http://localhost:5000 by default.

Environment Variables (server/.env)
```
GEMINI_API_KEY=your_gemini_key
MONGODB_URI=your_mongodb_connection_string
MONGODB_DB=your_db_name   # optional
PORT=5000                 # optional
```

Optional (Client API base)
Create `client/.env` if your server is not on localhost:5000
```
REACT_APP_API_BASE=https://your-server-host
```

Demo
- Local: Open http://localhost:3000
- Deploy: Host `server` (Render/Fly/Heroku) and `client` (Netlify/Vercel). Set `REACT_APP_API_BASE` to the API URL.

Mobile (not included in this repo): If you later build a native shell, include platform-specific run steps (Xcode/Android Studio) and link to a screen recording.

2) Problem Understanding
We need a simple, structured wellness board that:
- Captures a short profile (age, gender, goal)
- Generates exactly 5 concise, personalized tips (Screen 2)
- Provides a detailed explanation and 5 steps for a selected tip (Screen 3)
- Saves tips grouped under user goals (Screen 4)

Assumptions:
- Tips are guidance, not medical advice. Users should consult healthcare professionals as needed.
- The same goal may accumulate multiple saved tasks (tips) over time.
- We prefer fast, schema-constrained responses from the model for predictable rendering.

3) AI Prompts, JSON Schemas, and Iterations

Model: `gemini-2.5-flash`
Response type: `application/json`

Prompt 1 (Screen 2: Generate 5 Concise Tips)
- Temperature: 0.7
- Prompt: generate exactly 5 distinct, actionable tips for a user (gender, age, goal), each with a short title (≤ 5 words) and a single `icon_keyword`.
- JSON Schema:

```json
{
  "type": "array",
  "items": {
    "type": "object",
    "properties": {
      "tip_id": { "type": "integer" },
      "title": { "type": "string" },
      "icon_keyword": { "type": "string" }
    },
    "required": ["tip_id", "title", "icon_keyword"]
  }
}
```

Prompt 2 (Screen 3: Detailed Advice for Selected Tip)
- Temperature: 0.3
- Prompt: given user (gender, age, goal) and selected `tip_title`, return a 2-3 paragraph explanation plus 5 actionable steps.
- JSON Schema:
```json
{
  "type": "object",
  "properties": {
    "explanation_long": { "type": "string" },
    "steps": { "type": "array", "items": { "type": "string" } }
  },
  "required": ["explanation_long", "steps"]
}
```

Iterations/Notes:
- Using schema-constrained generation and `responseMimeType: application/json` ensures predictable parsing on the backend.
- Temperatures are tuned for variety in tips and focus/consistency in detailed advice.

4) Architecture & Code Structure

Monorepo: `client/` (React) + `server/` (Express). All backend logic is intentionally in `server/server.js` (single-file backend) per requirements.

```
/wellness-board (this repo)
├── client/
│   ├── src/
│   │   ├── components/
│   │   │   ├── ProfileForm.jsx       # Screen 1
```

```
│   │   │   ├── TipsBoard.jsx          # Screen 2
│   │   │   ├── TipDetail.jsx          # Screen 3
│   │   │   ├── SavedTips.jsx          # Screen 4
│   │   │   └── SavedTipsPage.jsx      # Standalone Saved Tips page
│   │   ├── App.js                     # Routing + composition
│   │   └── index.js                   # Theme + bootstrapping
│   └── package.json
└── server/
    ├── server.js                      # All routes + model + Gemini calls
    ├── package.json
    └── README.md                      # Backend notes
```

State & Routing:
- `App.js` uses `react-router-dom` for `/` (home) and `/saved` (Saved Tips page).
- Profile is persisted in `localStorage` to allow navigating directly to Saved Tips.

AI Integration:
- Server uses `@google/generative-ai`. Two helpers encapsulate the two prompts and schemas.

Styling & UX:
- Material UI theme in `client/src/index.js` with your palette:
  - Primary: Mint Green `#4CAF9D`
  - Secondary: Light Teal `#A7E8BD`
  - Accent: Coral Orange `#FF6B6B`
  - Background: Soft White `#F9FAF9`
  - Text: Charcoal Gray `#2E2E2E`
- Translucent gradient AppBar; modern cards; consistent button sizing; tilt hover for generated tips.

5) Technologies Used (and Why)

Frontend (React + MUI):
- React (Create React App) -- fast SPA development.
- Material UI (`@mui/material`, `@mui/icons-material`, `@emotion/react`, `@emotion/styled`) -- accessible, responsive, themeable component library.
- React Router (`react-router-dom`) -- client-side routing for a clean "Saved Tips" page.

Backend (Node/Express):
- Express -- simple, familiar HTTP server.
- Mongoose -- MongoDB ODM with schema validation.
- dotenv -- environment variable management.
- cors -- cross-origin requests from the client dev server.
- nodemon -- live-reload during backend development.

AI:
- `@google/generative-ai` -- official Gemini SDK for schema-constrained JSON output.

Database:
- MongoDB Atlas/local -- stores users and nested goals with saved tasks.

6) Data Model (MongoDB via Mongoose)

```js
User: {
  user_id: String,                // mirrors _id as string for convenience
  age: Number,
  gender: String,
  goals: [
    {
      name: String,               // goal name (e.g., "build muscle")
      saved_tasks: [
        {
          title: String,
          icon_keyword: String,
          explanation_long: String,
          steps: [String]
        }
      ]
    }
  ]
}
```

Indexes:
- `user_id` unique (sparse) and `collection: 'users'` explicitly set.

7) API Endpoints

Base URL: `{server}/api`

- POST `/profile`
  - Body: `{ age:number, gender:string, goal?:string }`
  - Creates a user, initializes goals with the provided goal (if any)
  - Returns: `{ userId }`

- POST `/tips/generate`
  - Body: `{ age:number, gender:string, goal:string }`
  - Returns: `{ tips: Array<{ tip_id:number, title:string, icon_keyword:string }> }`

- POST `/tips/detail`
  - Body: `{ age:number, gender:string, goal:string, tip_title:string }`
  - Returns: `{ explanation_long:string, steps:string[] }`

- POST `/tips/save`
  - Body: `{ userId:string, goalName?:string, tip:{ title, icon_keyword, explanation_long, steps:string[] } }`
  - Saves the tip under the specified goal (creates goal if missing)
  - Returns: `{ ok:true, goal:string }`

- GET `/tips/saved/:userId`
  - Returns: `{ goals:[ { name, saved_tasks:[ ... ] } ] }`

8) Frontend Libraries to Install (Client)
Already included in `client/package.json`, but if adding manually:
```bash
npm install @mui/material @mui/icons-material @emotion/react @emotion/styled react-router-dom
```
(Optional) Icons/Enhancements:
```bash
npm install @mui/lab
```

9) Backend Libraries to Install (Server)
Already included in `server/package.json`, but if adding manually:
```bash
npm install express mongoose dotenv cors @google/generative-ai
npm install -D nodemon
```

10) Development Workflow
- Start server first (`npm run dev` in `server/`), then start client (`npm start` in `client/`).
- Configure environment variables before invoking AI endpoints.
- Use the Home screen to enter a profile, then Generate Tips, open details, and Save.
- Navigate to `Saved Tips` page to review saved tasks by goal.

11) Troubleshooting
- CORS errors: confirm server is running and `REACT_APP_API_BASE` matches the server origin.
- 500 errors on AI calls: verify `GEMINI_API_KEY` is valid and has quota.
- Mongo connection errors: verify `MONGODB_URI` and IP access list (Atlas).
- Empty saved tips: ensure you saved at least one tip; use the Refresh icon on Saved Tips.

12) Future Improvements

- Modal/right-drawer for `TipDetail` on large screens; full-screen on mobile.
- Rich icons from `icon_keyword` via a mapping or icon library.
- User authentication to support multiple devices securely.
- Progress tracking and reminders.
- Dark mode toggle via MUI theme.

13) Notes on Parity with Sample Outline
- Web focus (React). Mobile instructions are placeholders if a native shell is added later.
- Navigation handled by `react-router-dom` instead of native NavigationHost.
- AI integration lives on the server (`server/server.js`) using the official Node SDK.