🧠 TimeWell Backend Rubric - Detailed Task List

🦴 Core Setup

- [ ] Initialize FastAPI project with appropriate structure (`app/`, `routers/`, `models/`, `schemas/`, `services/`)
- [ ] Set up MongoDB Atlas connection with `motor` or `pymongo`
- [ ] Environment variable management (`.env` using `python-dotenv` or `pydantic.BaseSettings`)
- [ ] Install and configure CORS middleware

🤗 Authentication

- [ ] JWT-based auth or Firebase Auth integration
  - If JWT:
    - [ ] User model with hashed password
    - [ ] Signup (`POST /auth/signup`)
    - [ ] Login (`POST /auth/login`)
    - [ ] Token creation and validation utils
    - [ ] Auth middleware dependency
  - If Firebase:
    - [ ] Firebase SDK setup

  - [ ] Auth route logic via Firebase API

👤💼 User Module

- [ ] MongoDB `users` collection

- [ ] GET `/users/:id/goals` - Fetch goals

- [ ] POST `/users/:id/goals` - Update goals

- [ ] Support for `coach_voice` and user preferences

🗓️ Events Module

- [ ] MongoDB `events` collection

- [ ] POST `/events` - Add event

- [ ] GET `/events/:user_id` - Fetch user events

- [ ] POST `/events/analyze` - GPT-4 API call to check alignment with goals

  - [ ] Prompt template using LangChain

  - [ ] AI response saved to `suggestions` collection

✅ Habits Module

- [ ] MongoDB `habits` collection

- [ ] POST `/habits` - Create new habit

- [ ] PUT `/habits/:id` - Mark habit complete (update streak + timestamp)

- [ ] GET `/habits/:user_id` - Fetch all user habits

💡 Suggestions Module

- [ ] MongoDB `suggestions` collection

- [ ] GET `/suggestions/:user_id` - Fetch past suggestions

- [ ] Store `ai_prompt`, `ai_response`, `event_id`, `was_accepted`

🧘 Coach Reflection Module

- [ ] POST `/coach/reflect`

  - [ ] Input = weekly reflection or status

  - [ ] Output = GPT-based encouragement/feedback using user's coach voice

---

🧠 AI Integration (LangChain + GPT-4)

- [ ] Define prompt templates by voice (Cool Cousin, OG Big Bro, Oracle, etc.)

- [ ] Setup LangChain chains or direct OpenAI API calls

- [ ] Handle caching or storing AI results

- [ ] Include fallback messages if AI call fails

⏰ Scheduler (Optional)

- [ ] Use Celery or APScheduler

  - [ ] Daily habit reminders

  - [ ] Weekly summary reports from Coach

☁️ Storage & Media

- [ ] Firebase Storage or Cloudinary setup (for profile photos)

📦 Deployment Readiness

- [ ] Procfile / `start.sh` script for Render/Railway

- [ ] Dockerfile (optional)

- [ ] Swagger docs available via FastAPI (`/docs`)

- [ ] Logging and error handling middleware

- [ ] Unit test coverage for routes and services

🔍 Optional Extras

- [ ] Rate limiting on AI endpoints

- [ ] Data export route (`/users/:id/export`)

- [ ] Role-based access (e.g. admin dashboard)

- [ ] In-app notifications service via Firebase

Let me know when you're ready to start implementing each section or if you want code samples for any of these!