# ✅ SmartCoach – GPT Handoff Summary

This document was generated to bootstrap a Custom GPT session with complete project context.

## 🏁 Last known good state:

• Tag: v1.0.0

• Deployment: Railway (web-production-c4329.up.railway.app)

• OAuth tested: ✅

• DB Connected: ✅

## 🔐 Environment Settings:

• ADMIN\_USER=admin

• REDIRECT\_URI=https://web-production-c4329.up.railway.app/oauth/callback

• DATABASE\_URL=(set in Railway)

• SECRET\_KEY, INTERNAL\_API\_KEY, CRON\_SECRET\_KEY defined

## 🔧 Architecture Notes:

• Flask API deployed on Railway

• PostgreSQL managed via Railway plugin

• JWT-based authentication and user sessions

• Strava OAuth 2.0 integration and webhook registration planned

## 🎯 Coaching Intelligence Objectives:

• Build personalized weekly training plans

• Track Strava-recorded performance

• Monitor deviations and adapt based on user behavior

• Support natural conversation for advising and adjusting

## 🧱 System Components:

• Custom GPT logic: planning, feedback, and conversation

• PostgreSQL DB storing plan and activity data

• Strava Sync service

• Planned: Training plan generator, weekly comparator

## ⚠️ Known Issues:

• None at last deploy. Strava activity fetch untested.

## 📌 Next Tasks:

1. Test full Strava flow: code -> token -> activity fetch

2. Implement auto-sync

3. Add athlete-to-user ID mapping

4. Connect activity analysis

5. Ship Phase 2 readiness

## 🧠 GPT Boot Prompt

Create a Custom GPT with the following context:

- Name: SmartCoachDev

- Role: Flask engineer + PM for Smart Marathon Coach API

- Accesses: JWT auth, Strava OAuth, Railway deploy logs, pg DB

- Code base starts in: /src, /templates, /scripts

- Current live env: https://web-production-c4329.up.railway.app

- Start by reviewing recent work, then continue with task #1: Test full Strava activity flow.

- Reference: ‘Smart Coach - Project Reference.docx’ for architecture, history, and schema.