ChatGPT

Context Tracker – Progressive Development Roadmap

Phase 1: Prototype Foundation (Complete)

- Website usage tracking (URL, title, scroll, duration)
- AFK detection via events
- Redis queue
- FastAPI backend
- PostgreSQL with SQLAlchemy
- Worker for async event ingestion

Phase 2: Session Stitching Engine

Step 1: `session_engine.py`

- Input: AFK + Event data per day
- Output: Time blocks (`active`, `idle`, `break`)
- Algorithms: interval merging, time bucketing

Step 2: `GET /summaries/daily`

- Returns: total active time, idle %, top categories, streaks
- Optional caching for faster access

📊 Phase 3: Productivity Scoring

Step 3: Strategy Pattern for Scoring

- Per-category scoring strategies
- E.g. `ResearchScorer`, `EntertainmentScorer`

Step 4: `ProductivityScore` Engine

- Assign scores (0–100) per session
- Aggregate into day/week summary

Phase 4: Multi-User & Auth

Step 5: Add `user_id` to Models

- Update DB and API to tag all data by user

Step 6: JWT Authentication

- `/login` returns JWT
- API extracts `user_id` from JWT claims

Phase 5: Scalable Infrastructure

Step 7: Kafka Ingestion

- Replace Redis with Kafka
- Add multiple parallel workers

Step 8: Partitioned & Indexed DB

- Partition tables by `user_id` or `date`
- Add efficient indexes for query performance

🛠 Phase 6: OOP & Code Architecture

Step 9: Backend Refactor

- `Repository` pattern for DB logic
- `Factory` for Event object creation
- `Service` layer for business logic

Phase 7: Frontend Expansion

Step 10: Popup UI Enhancements

- Show today’s time usage, streaks, categories

Step 11: Dashboard (Optional React App)

- Charts, session timelines, weekly summaries

Summary View

Phase	Step	Outcome
1	Tracking + Ingestion	Working prototype
2	Session Engine	Stitched usage blocks
3	Productivity Scoring	Meaningful session analysis
4	Multi-User + Auth	Cloud-ready, user-aware system
5	Kafka + Partitioned DB	Scalable ingestion + storage
6	OOP Refactor	Clean, modular codebase
7	UI Dashboard	Insightful user-facing features