**1) Kjerneberegninger (Rust)**

* **Input**: samples (time, hr, watts, moving, altitude), meta (ftp? hr\_max? start\_time duration).
* **NP/IF/VI**:
  + NP med 30s rull, ^4‑middel, ^(1/4).
  + IF = NP/FTP; **FTP‑fallback**: beste 20 min \* 0,95 (hvis tilgjengelig siste 30–60 dager).
  + VI = NP / AvgPower.
* **Pa:Hr** (drift): del i to halvdeler, relativ endring i P/HR‑ratio (krav: ≥60 min og gyldig data).
* **W/beat**: AvgPower/AvgHR; normaliser mot personlig baseline (median siste 28 dager for økter ±25 % varighet).
* **CGS v1**:
  + Intensitet (40 %): 100 \* exp(-((IF-0.90)/0.10)^2) (HR‑fallback med %HRR/%HRmax med samme klokkeform).
  + Varighet (30 %): 100 \* min(1, log1p(min/30) / log1p(180/30)).
  + Kvalitet (30 %): 0.4*DriftScore + 0.3*VIScore + 0.3\*WpBScore; re‑vekt ved manglende komponenter; default 50 om alt mangler.
  + Stykkevise skalaer som spesifisert tidligere (Pa:Hr, VI, W/beat).
* **Mini‑trend**: rullerende snitt/±% på CGS for **siste 3 økter** (og/eller 5 hvis tilgjengelig).
* **Badges** (regelmotor v1):
  + Iron Lungs → Pa:Hr ≤ 2 % på økter ≥ 90 min.
  + Metronome → VI ≤ 1.03 på økter ≥ 60 min.
  + Big Engine → W/beat ≥ +10 % over baseline.
* **Ytelse**: 2t @1 Hz analyseres ≤200 ms på normal laptop.

**2) API/binding & CLI**

* **Rust pyo3 API**:
  + analyze\_session(samples, meta, cfg) -> SessionReport
  + analyze\_batch(sessions, cfg) -> BatchReport (for mini‑trend).
  + estimate\_ftp(sessions) -> FtpEstimate
* **Konfig (serde + JSON)**:

{ "ftp": 260, "ftp\_auto\_estimate": true, "hr\_max": 190,

"cgs": {"weights":{"intensity":0.4,"duration":0.3,"quality":0.3}},

"zones": {"power":"coggan\_like","hr":"hrr\_5"} }

**CLI (Python)**:

* python analyze.py --input data/\*.csv --out output/ --cfg config.json --batch --with-trend
* Tillegg: --set-ftp, --auto-ftp, --format json|csv|both, --publish-to-strava.

**Outputformat (fil/REST)**

* **SessionReport JSON** (eksempel):

{

"session\_id":"2025-08-12\_ride\_123",

"duration\_min":120,

"avg\_power":214,"avg\_hr":135,

"np":238,"if":0.92,"vi":1.11,

"pa\_hr\_pct":2.4,"w\_per\_beat":1.59,"w\_per\_beat\_baseline":1.45,

"scores":{"intensity":93,"duration":82,"quality":88,"cgs":88},

"badges":["Iron Lungs"],

"trend":{"cgs\_last3\_avg":85,"cgs\_delta\_vs\_last3":+3.4}

}

**BatchReport JSON**: liste av SessionReport + aggregerte trend‑felter.

**Strava‑publisering (kommentar/description)**

* **Formatter** som genererer kort streng (≤ ~250–300 tegn) + en litt lengre for beskrivelse:
  + Kort (kommentar):  
    CycleGraph CGS 88 · IF 0.92 · VI 1.11 · Pa:Hr 2.4% · W/beat 1.59 (↑+10%) · Trend: +3% vs siste 3
  + Lang (beskrivelse topp‑linje + 1 linje til):  
    CGS 88 – How hard:93 | How long:82 | How steady & efficient:88. VI 1.11 · Pa:Hr 2.4% · W/beat 1.59 (↑+10%).
* **CLI‑flagget --publish-to-strava** oppdaterer automatisk siste importerte aktivitet (for innlogget bruker) med teksten.
* **Tørr‑kjøringsmodus**: --dry-run skriver kun til stdout.

**5) Robusthet & tester**

* **Unit‑tester**: NP/IF/VI/Pa:Hr/WpB (syntetiske serier, fasitverdier).
* **Golden‑tester**: 3 ekte økter (rolig langtur, terskel, intervaller) – stabile outputverdier med toleranser.
* **Edge‑tester**: uten watt, uten HR, korte økter (<10/20/60 min), pauser, outliers.
* **Feilhåndtering**: meningsfulle feilmeldinger i CLI + exit‑koder.

**6) Dokumentasjon**

* docs/CGS\_v1.md (formler, skalaer, fallbacks).
* docs/cli\_usage.md (eksempler).
* docs/strava\_publish.md (hva publiseres, hvordan skrus av/på).

**7) Comms & commits**

* Commit: feat(m7): core CGS v1 + metrics (NP/IF/VI/Pa:Hr/WpB) with pyo3
* Commit: feat(m7): batch trend + badges + JSON schema
* Commit: feat(m7): CLI flags + Strava auto-publish (+dry-run)
* Commit: test(m7): unit/golden/edge + perf guard
* Oppdater Masterplan‑tabellen til “Ferdig” når alt over er grønt.