**M6.3–M6.5 Gjennomføringsmanual (CycleGraph)**

**Formål:** Fullføre Strava-integrasjonen slik at vi kan hente aktiviteter (paging + since), hente streams, skrive robuste CSV-filer, og håndtere token-refresh og rate-limits uten avbrudd.

**Steg 0 – Sanity-sjekk av koden**

**Mål:** Bekrefte at grunnstrukturen er på plass før vi begynner å endre noe.

1. Åpne cli/strava\_import.py.
2. Sjekk at følgende funksjoner finnes (de kan være tomme/skjelett, men må eksistere):
   * fetch\_activities(...) – for paging og --since.
   * fetch\_streams\_for\_activity(...) – henter time, heartrate, watts, moving, altitude.
   * write\_stream\_csv(...) – skriver CSV med riktig header.
   * Ratelimit-funksjoner: parse\_rate\_headers(...), seconds\_until\_next\_15m\_window(...), strava\_get(...).
3. Sjekk at CLI-subkommandoene activities og streams finnes i argparse-oppsettet.
4. **Ikke endre noe enda** – vi skal bare verifisere at strukturen er der.

**Kjør test:**

* Ingen kodekjøring nødvendig – dette er kun visuell inspeksjon.

**Del 1 – M6.3: Refresh-logikk**

**Mål:** Fornye access token automatisk før utløp, lagre nyeste refresh\_token hver gang.

1. Etter load\_tokens() i både activities- og streams-kommandoene:

python

KopierRediger

now = int(time.time())

if tokens["expires\_at"] < now + 3600:

tokens = refresh\_tokens(tokens)

save\_tokens(tokens)

1. Lag funksjonen refresh\_tokens(tokens):
   * POST til https://www.strava.com/api/v3/oauth/token med:

python

KopierRediger

{

"client\_id": os.getenv("STRAVA\_CLIENT\_ID"),

"client\_secret": os.getenv("STRAVA\_CLIENT\_SECRET"),

"grant\_type": "refresh\_token",

"refresh\_token": tokens["refresh\_token"]

}

* + Oppdater access\_token, refresh\_token, expires\_at fra respons.
  + Returner den oppdaterte token-dict.

1. Lag save\_tokens(tokens) som skriver JSON til data/strava\_tokens.json med UTF-8 uten BOM.
2. **Feilhåndtering:**
   * Hvis refresh feiler (401/403/5xx), logg feilmeldingen og avslutt med sys.exit(1).

**Kjør test:**

* Sett tokens["expires\_at"] = int(time.time()) + 10 manuelt i data/strava\_tokens.json.
* Kjør:

bash

KopierRediger

python cli/strava\_import.py activities --since 2025-07-01 --out data/activities.json

* Bekreft i loggen:
  + “Token refreshed”
  + Ny refresh\_token-verdi i data/strava\_tokens.json.
  + Ingen 401 Unauthorized i output.

**Del 2 – M6.4: Aktiviteter (paging + since)**

**Mål:** Hente alle aktiviteter etter en gitt dato, håndtere paging, og lagre state for inkrementell sync.

1. **Konverter --since til epoch:**

python

KopierRediger

if args.since:

after = int(datetime.strptime(args.since, "%Y-%m-%d").timestamp())

else:

state\_path = Path.home() / ".config" / "cyclegraph" / "state.json"

if state\_path.exists():

after = json.load(open(state\_path))["last\_after"]

else:

after = 0

1. **Paging-løkke:**

python

KopierRediger

page = 1

all\_activities = []

while True:

resp = strava\_get(f"/athlete/activities?after={after}&page={page}&per\_page=200", tokens)

if not resp:

break

all\_activities.extend(resp)

print(f"Hentet {len(all\_activities)} aktiviteter...")

page += 1

1. **State-fil:**
   * Etter ferdig sync, oppdater last\_after med seneste aktivitetens starttid:

python

KopierRediger

json.dump({"last\_after": newest\_epoch}, open(state\_path, "w"))

1. **Skriv til output-fil:**

python

KopierRediger

with open(args.out, "w") as f:

json.dump(all\_activities, f, indent=2)

**Kjør test:**

* Første kjøring:

bash

KopierRediger

python cli/strava\_import.py activities --since 2025-07-01 --out data/activities.json

Bekreft at:

* + Filen data/activities.json inneholder aktiviteter.
  + Loggen viser “Hentet 200 / 400 / … aktiviteter”.
* Andre kjøring (uten --since):

bash

KopierRediger

python cli/strava\_import.py activities --out data/activities.json

Bekreft at kun nye aktiviteter hentes.

**Del 3 – M6.5: Streams & CSV**

**Mål:** Hente tidsserier for aktiviteter og skrive robuste CSV-filer med konsistent kolonneantall.

1. **Hent streams:**

python

KopierRediger

keys = ["time", "heartrate", "watts", "moving", "altitude"]

resp = strava\_get(f"/activities/{activity\_id}/streams?keys={','.join(keys)}&key\_by\_type=true", tokens)

1. **Bygg datastruktur:**
   * Finn maks lengde blant streamene.
   * For hvert index:

python

KopierRediger

row = [

activity\_id,

i,

time\_stream[i] if i < len(time\_stream) else "",

hr\_stream[i] if i < len(hr\_stream) else "",

watts\_stream[i] if i < len(watts\_stream) else "",

moving\_stream[i] if i < len(moving\_stream) else "",

altitude\_stream[i] if i < len(altitude\_stream) else ""

]

1. **Skriv CSV:**

python

KopierRediger

with open(out\_path, "w", newline="") as csvfile:

writer = csv.writer(csvfile)

writer.writerow(["activity\_id", "index", "time\_s", "hr", "watts", "moving", "altitude"])

writer.writerows(rows)

1. **Manglende felt:**
   * Hvis watts mangler: skriv tom streng "" eller NaN – ikke dropp raden.

**Kjør test 1 (én aktivitet):**

bash

KopierRediger

python cli/strava\_import.py streams --activity-id <id> --out data/streams

Bekreft:

* Fil data/streams/<id>.csv eksisterer.
* Første rad er headeren.
* Alle rader har like mange kolonner.

**Kjør test 2 (flere aktiviteter):**

bash

KopierRediger

python cli/strava\_import.py streams --from-activities data/activities.json --type Ride --limit 3 --out data/streams

Bekreft:

* 3 CSV-filer opprettet.
* Konsistent format.

**Del 4 – Ratelimit & feilhåndtering**

**Mål:** Unngå at skriptet stopper ved Strava sine begrensninger.

1. **Ratelimit-sjekk:**
   * I strava\_get():

python

KopierRediger

used\_15min, used\_day = map(int, resp.headers["X-RateLimit-Usage"].split(","))

limit\_15min, limit\_day = map(int, resp.headers["X-RateLimit-Limit"].split(","))

if used\_15min > limit\_15min \* 0.8:

sleep\_time = seconds\_until\_next\_15m\_window()

print(f"Rate limit nær nådd – venter {sleep\_time} sekunder...")

time.sleep(sleep\_time)

1. **401 Unauthorized:** Trigger refresh\_tokens().
2. **403 Forbidden:** Meld om manglende scope og avslutt.
3. **429 Too Many Requests:** Vent til neste 15-minuttersvindu.
4. **5xx-feil:** Implementer retry med eksponentiell backoff.

**Presstest:**

* Sett SAFE\_HEADROOM = 1000 for å simulere venting tidlig.
* Kjør en import og bekreft at scriptet logger “venter X sekunder…”.

**Definition of Done for M6.3–M6.5**

* Token fornyes før utløp og nyeste refresh\_token lagres.
* activities.json genereres med paging + --since/state.
* Minst 3 CSV-filer med korrekt header og identisk kolonneantall.
* Manglende HR/watts gir tomme felt – ingen rader droppes.
* Ingen ubehandlede 429/401 stopper kjøringen.