# Stripe Connect Reporting API – Test Exercise

## Objective

Build a reporting tool that interacts with the **Stripe Connect Custom Accounts API**, authenticating with the provided API keys, and generates reports on **daily, weekly, and monthly transaction data**. This exercise will test:

- API integration
- Data aggregation logic
- Report scheduling or parameterization
- Response structuring and presentation

## Task Breakdown

- Stripe Connect account with API keys
- Can pull one connected account with activity
- Can pull multiple accounts to one five-connected account with activity
- React / Node.js (preferred language)
- Schedule reports using cron jobs or job schedulers, UX/UI

### 1. API Authentication

- Use the provided Stripe **Secret Key** for authentication.
- All requests must be made over HTTPS using Bearer authentication.

### 2. UI/UX

Create a web interface with basic design. The page should contain next sections:

- API keys input fields:
    - Connected account ID;
    - Public key;
    - Secret key;
- Filtering section. Should contain:
    - Start date;
    - End date;
    - Timezone selection;
- Generate Report button

### 3. Date Range Logic

Implement logic for fetching:

- 🗓️ **Daily Reports**: Last 24 hours
- 📆 **Weekly Reports**: Last 7 days
- 📅 **Monthly Reports**: Last 30 days

---

### 4. Aggregate and Format results

The output report should be a spreadsheet showing on the same page. Should have next columns:

- Date;
- Charges_count - Amount of successful transactions
- Charges_amount - Volume of successful transactions in $$
- Refunds_count - Amount of refunds
- Refunds_amount - Volume of refunds in $$
- Chargebacks_count - Amount of chargebacks (might be 0 for your case);
- Chargebacks_amount - Volume of chargebacks in $$ (might be 0 for your case);
- Declines_count - Amount of failed transactions;
- Aprvl_pct - Equals 100% - (Declines_count/(Charges_count + Declines_count))*100%
- Totals_count - Equals (Charges_count + Refunds_count + Declines_count + Chargebacks_count )
- Totals_amount - Equals (Charges_amount - Refunds_amount - Chargebacks_amount )

**Example below**

If user chooses to show transactions between July 28 and July 29 in EST timezone, he will get next spreadsheet (data is for example purposes only and you will see a different one)

| Date | Charges_count | Charges_amount | Refunds_count | Refunds_amount | CB_count | CB_amount | Declines_count | Aprvl_pct | Total_count | Totals_amount |
|---|---|---|---|---|---|---|---|---|---|---|
| 7/28/2025 | 100 | $5000 | 1 | $100 | 1 | $150 | 7 | 93.4% | 109 | $4 750 |
| 7/29/2025 | 200 | $10000 | 3 | $500 | 0 | $0 | 10 | 95.2% | 213 | $9 500 |

- Format output as that can be selected:
  - CSV, XLS, Google Sheets or Email
  - Allow user to choose date ranges via CLI or web paramaters

---

## 📤 Submission Expectations

- Code repo with README instructions
- Sample reports for each frequency (daily/weekly/monthly)
- Any notes or assumptions made
- The task is due in 48 hours after the start

---