Accounting & Backend Technical Test

Background

You are given a simplified table of accounting entries for **January 2025**:

`AccountingLedgerEntry`

id	date	account	debit	credit	party	note	bankaccount	reference	reconciled
1	2025-01-02	Cash	10000	0	Investor	Capital Contribution	MainBank	DEP001	TRUE
2	2025-01-05	Office Rent	0	2000	Landlord Ltd.	January rent	MainBank	CHQ101	TRUE
3	2025-01-10	Inventory	0	3000	Supplier A	Purchase inventory	MainBank	CHQ102	FALSE
4	2025-01-15	Sales	0	8000	Customer B	Sales Invoice	NULL	NULL	NULL
5	2025-01-16	Cash	8000	0	Customer B	Payment received	MainBank	DEP002	TRUE
6	2025-01-20	Utilities Expense	0	500	Power Co	Electricity bill	MainBank	CHQ103	TRUE
7	2025-01-25	Bank Loan	0	7000	BigBank	Loan received	MainBank	DEP003	TRUE
8	2025-01-26	Cash	7000	0	BigBank	Loan deposit	MainBank	DEP003	TRUE
9	2025-01-28	Bank Charges	0	500	BigBank	Monthly service charge	MainBank	CHQ104	FALSE

(Create table schema and insert statements in PostgreSQL can be found at the end of this document)
Note: You may extend the schema if needed (e.g., adding helper columns such as `cashflow_category` or reconciliation flags). If you do so, explain your assumption and how it simplifies your query.

Task 1 — Cash Flow Statement API

Using the data above, build a **Cash Flow Statement API** for January 2025.

Requirements:

1. Write a **PostgreSQL SQL query** to retrieve and classify cash flows into:

    - **Operating activities**
    - **Investing activities**
    - **Financing activities**
2. Implement an **Express.js endpoint** in Node.js that:
    - Accepts `companyid`, `fromDate`, and `toDate` as query parameters
    - Executes your SQL query
    - Returns the Cash Flow Statement in JSON format with:
        - Cash inflows
        - Cash outflows
        - Net change in cash
        - Closing cash balance

Task 2 — Bank Reconciliation Statement API

At 31 January 2025:

- The Ledger shows a **Cash account balance = 22,500**.
- The Bank Statement shows **MainBank balance = 19,000**.

From the table above:

- Cheque `CHQ102` for 3,000 issued to Supplier A has **not cleared**.
- Bank charges `CHQ104` for 500 have **not yet been recorded in the Ledger**.

Requirements:

1. Write a **PostgreSQL SQL query** that identifies unreconciled transactions (e.g., `reconciled = FALSE`).
2. Implement an **Express.js endpoint** in Node.js that:
    - Accepts `companyid` and `bankaccount` as query parameters
    - Executes your SQL query
    - Returns a JSON response showing:
        - Ledger balance
        - Bank statement balance
        - Reconciling items
        - Adjusted balance after reconciliation

Deliverables for each task:

PostgreSQL SQL query used Express.js controller code (the endpoint handler that runs the SQL and returns JSON) Screenshot of JSON response of the controller

AccountingLedgerEntry Schema and Sample Data SQL:

```
-- Schema
CREATE TABLE AccountingLedgerEntry (
  id SERIAL PRIMARY KEY,
  date DATE NOT NULL,
  account VARCHAR(255) NOT NULL,
  debit NUMERIC DEFAULT 0,
  credit NUMERIC DEFAULT 0,
  party VARCHAR(255),
  note TEXT,
  bankaccount VARCHAR(255),
  reference VARCHAR(255),
  reconciled BOOLEAN DEFAULT FALSE,
  companyid INT NOT NULL DEFAULT 1
);

-- Sample Data
INSERT INTO AccountingLedgerEntry
(date, account, debit, credit, party, note, bankaccount, reference, reconciled, companyid)
VALUES
('2025-01-02', 'Cash',              10000, 0, 'Investor',    'Capital Contribution',   'MainBank',
'DEP001', TRUE, 1),
('2025-01-05', 'Office Rent',       0, 2000, 'Landlord Ltd.', 'January rent',           'MainBank',
'CHQ101', TRUE, 1),
('2025-01-10', 'Inventory',         0, 3000, 'Supplier A',   'Purchase inventory',     'MainBank',
'CHQ102', FALSE, 1),
('2025-01-15', 'Sales',             0, 8000, 'Customer B',   'Sales Invoice',          NULL, NULL,
NULL, 1),
('2025-01-16', 'Cash',              8000, 0, 'Customer B',   'Payment received',       'MainBank',
'DEP002', TRUE, 1),
('2025-01-20', 'Utilities Expense', 0, 500, 'Power Co',      'Electricity bill',       'MainBank',
'CHQ103', TRUE, 1),
('2025-01-25', 'Bank Loan',         0, 7000, 'BigBank',      'Loan received',          'MainBank',
'DEP003', TRUE, 1),
('2025-01-26', 'Cash',              7000, 0, 'BigBank',      'Loan deposit',           'MainBank',
'DEP003', TRUE, 1),
('2025-01-28', 'Bank Charges',      0, 500, 'BigBank',       'Monthly service charge',
'MainBank', 'CHQ104', FALSE, 1);
```