```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Sun Feb 15 19:07:17 2026

@author: evebarr20
"""
# Import files from step 1 and step 3
from db_connection import create_connection
from transactions import get_transactions_by_month

# Import libraries
import matplotlib.pyplot as plt
# import pandas as pd

class BankTransactions:
    """
    Provide convenient access to monthly transactions and daily plots.

    Attributes
    ----------
    filename: str
        Path to the vault file containing username (line 1) and password (line 2)
    con: sqlalchemy.engine.Engine
        SQLAlchemy engine connected to the bank database.
    """
    def __init__(self, filename):
        """
        Initialize the BankTransactions object.

        Reads the vault filename (stored for reference) and creates
        connection using the step 1 connection function.

        Parameters
        ----------
        filename : str
            Path to the vault file.

        Returns
        -------
        None.

        """
        # Store the vault filename used to create the connection
        self.filename = filename
        # Create and store the database connection (engine)
        self.con = create_connection(filename)
    def get_transactions(self, month, year):
        """
        Retrieve all transactions for a specified month and year.
```

```
    This method uses the step 3 function to query the transactions table and
    returns a DataFrame of transactions excluding account_id. If
    invalid, the step 3 function returns a one-row DataFrame filled with -1.

    Parameters
    ----------
    month : int
        Month value (1-12).
    year : int
        Year value (>= 1800).

    Returns
    -------
    pandas.DataFrame
        DataFrame containing transaction records for the given month/year.
        If invalid input, returns a one-row DataFrame of -1 values.

    """
    return get_transactions_by_month(self.con, month, year)
def plot_transactions_by_day(self, month, year):
    """
    Generate a bar plot of transaction counts by day for a given month/year

    The method:
        1) Retrieves transactions using get_transactions().
        2) Validates results (handles empty results or -1 row)
        3) Extracts day-of-month from txn_date
        4) Groups by day and counts transactions
        5) Plots the daily counts as a bar chart

    Parameters
    ----------
    month : int
        Month value (1-12).
    year : int
        Year value (>= 1800).

    Returns
    -------
    None.

    """
    # Retrieve transactions for the month/year (may be empty or -1 row)
    transactions_df = self.get_transactions(month, year)
    # Case 1: valid input but no matching transactions
    if transactions_df.empty:
        print("Cannot plot: no transactions found for that month/year.")
        return
    # Case 2: invalid input detected by step 3 function (-1 row)
    if transactions_df.iloc[0]["txn_id"] == -1:
        print("Cannot plot: invalid month/year input.")
```

```
    return
# Extract day from txn_date
transactions_df["day"] = transactions_df["txn_date"].dt.day
# Count transactions by day
daily_counts_df = (transactions_df
                .groupby(["day"], observed = True) ["txn_id"]
                .count()
                .reset_index(name = "transactions_count")
                )
# Plot daily transaction counts
plt.bar(daily_counts_df["day"], daily_counts_df["transactions_count"])
plt.title(f"Daily Transaction Count - {month}/{year}")
plt.xlabel("Day of Month")
plt.ylabel("Number of Transactions")
plt.show()
```