Group 3 (Jesus, Hamid, Tena, Ethan, Anthony)

3/18/2023

CS 499 Capstone

**Pseudo Code Design:**

***Import required libraries and modules:***

a. random: a module that generates random numbers for simulating sensor data

b. time: a module that provides time-related functions

c. psycopg2: a PostgreSQL database adapter for the Python programming language

***Define a function to establish a connection to the PostgreSQL database:***

a. Use the psycopg2 library to establish a connection to the database

b. Return the database connection object

***Define a function to generate sensor data:***

a. Use the random module to simulate sensor data for various devices and sensors

b. Store the simulated data in a dictionary

c. Return the dictionary containing the simulated data

***Define a function to store sensor data in the database:***

a. Use the psycopg2 library to execute an SQL query that inserts the sensor data into the database

b. Commit the changes to the database

***Define a function to retrieve current sensor data from the database:***

a. Use the psycopg2 library to execute an SQL query that retrieves the most recent sensor data from the database

b. Return the retrieved sensor data as a dictionary

***Define a function to calculate utility usage and cost:***

a. Calculate the utility usage and cost based on the retrieved sensor data

b. Print the utility usage and cost to the console

***Define a main function:***

a. Call the function to establish a connection to the database

b. Use a while loop to generate and store sensor data, retrieve current sensor data, and calculate utility usage and cost

c. Wait for a few seconds before generating new sensor data

***Call the main function to start the program***

------------------------------------------------------ Psuedo Code (Python) ------------------------------------------------------

# Import required libraries and modules

import random

import psycopg2

***{Import required libraries and modules: This section imports the necessary modules to interact with PostgreSQL and generate random numbers.}***

# Define a function to generate sensor data

def generate\_sensor\_data(conn):

# Create a cursor object to execute queries on the database

cur = conn.cursor()

# Prepare the INSERT statement to add new sensor data to the database

sql = "INSERT INTO sensor\_data (temperature, humidity, garage\_door\_status, window\_status, water\_usage, electricity\_usage) VALUES (%s, %s, %s, %s, %s, %s)"

# Simulate sensor data for various devices and sensors

temperature = random.randint(60, 80)

humidity = random.randint(30, 50)

garage\_door\_status = "open" if random.random() < 0.5 else "closed"

window\_status = "open" if random.random() < 0.5 else "closed"

water\_usage = random.randint(50, 100)

electricity\_usage = random.randint(200, 400)

# Store sensor data in the database

cur.execute(sql, (temperature, humidity, garage\_door\_status, window\_status, water\_usage, electricity\_usage))

# Commit the transaction to the database

conn.commit()

# Retrieve the last row of sensor data from the database

cur.execute("SELECT \* FROM sensor\_data ORDER BY id DESC LIMIT 1")

row = cur.fetchone()

# Display the current sensor data to the user

print("Temperature: {} degrees F".format(row[1]))

print("Humidity: {}%".format(row[2]))

print("Garage Door: {}".format(row[3]))

print("Windows: {}".format(row[4]))

print("Water Usage: {} gallons".format(row[5]))

print("Electricity Usage: {} kW-hrs".format(row[6]))

# Calculate utility usage and cost

calculate\_utility\_usage\_and\_cost(cur)

***{Define a function to generate sensor data: This function generates simulated sensor data for various devices and sensors, stores the data in the database, retrieves the last row}***

# Define a function to calculate utility usage and cost

def calculate\_utility\_usage\_and\_cost(cur):

# Retrieve the latest sensor data from the database

cur.execute("SELECT \* FROM sensor\_data ORDER BY id DESC LIMIT 1")

row = cur.fetchone()

# Calculate the total utility usage and cost

water\_cost = row[5] \* 0.005 # $0.005 per gallon

electricity\_cost = row[6] \* 0.10 # $0.10 per kW-hr

total\_cost = water\_cost + electricity\_cost

# Display the total utility usage and cost to the user

print("Total Water Cost: ${:.2f}".format(water\_cost))

print("Total Electricity Cost: ${:.2f}".format(electricity\_cost))

print("Total Cost: ${:.2f}".format(total\_cost))

# Establish a connection to the PostgreSQL database

conn = psycopg2.connect(database="home\_iot", user="username", password="password", host="localhost", port="5432")

try:

# Loop indefinitely to generate sensor data

while True:

generate\_sensor\_data(conn)

# Wait for a few seconds before generating new sensor data

time.sleep(5)

except KeyboardInterrupt:

# Close the connection to the database

conn.close()