Abstract Code

Main Menu

Abstraction Code

- Display Link to Enter Household Information
- Display Link to Enter Reports View
- If user clicks Enter my household info, then:
    - Jump to Household Info Form
- If user clicks View reports/query data, then:
    - Jump to Reports listmp to Appliance Info Form

Household Information Form

Abstraction Code

- User enters email address, five digit postal code
- User picks HomeType from the provided List
- User enters square footage for the household
- User can enter temperature for cooling and heating OR check no Heat/Cool checkbox if they choose not to.
- User checks list of public utilities from the checkbox
- When user clicks next button following check will occur:
    - If email address is inside the database, the user should use a different email address
    - Postal code will be validated against the list of postal code provided
- When the check is successful
    - Write user's input into Household table and Address Table
    - Jump to Appliance Form

Abstraction Code with SQL Statements

- User enters email address, five digit postal code
- User picks HomeType from the provided List
- User enters square footage for the household
- User can enter temperature for cooling and heating OR check no Heat/Cool checkbox if they choose not to.
- User checks list of public utilities from the checkbox
- When user clicks next button following check will occur:

```
SELECT email FROM Household WHERE email='email';
```

    - If email exists from the query above, throw an error message that the email exists.

```
SELECT postal_code FROM Address WHERE postal_code='postal_code';
```

    - If postal_code does not exist from the query above, throw an error message that the postal code is invalid.
- When the check is successful

```
INSERT INTO HouseHold (email, square_footage, household_types, postal_code)
VALUES ('email', 1000, 'Townhome' , 22022);
```

    - If user checked 'No heat', then:

```
INSERT INTO Heating (email, temperature) VALUES ('email', NULL)
```

    - If user checked 'No cooling', then:

```
INSERT INTO Cooling (email, temperature) VALUES ('email', NULL)
```

    - If user did not check 'No heat', then:

INSERT INTO Heating (email, temperature) **VALUES** ('email', 100);

- If user did not check 'No cooling', then:
  INSERT INTO Cooling (email, temperature) **VALUES** ('email', 75);
- If user checked 'Electric' from public utilities checkbox, then:
  INSERT INTO PublicUtility (email, public_utility) **VALUES** ('email', 'Electric');
- If user checked 'Gas' from public utilities checkbox, then:
  INSERT INTO PublicUtility (email, public_utility) **VALUES** ('email', 'Gas');
- If user checked 'Steam' from public utilities checkbox, then:
  INSERT INTO PublicUtility (email, public_utility) **VALUES** ('email', 'Steam');
- If user checked 'Fuel oil' from public utilities checkbox, then:
  INSERT INTO PublicUtility (email, public_utility) **VALUES** ('email', 'Fuel oil');
- IF user didn't check any of the public utilities checkbox, then:
  - Do not perform any query
- Jump to Appliance Form

Appliance Form

Abstraction Code

- User picks Appliance type and Manufacturer
- User enter Model name and BTU rating
- User picks Appliance Types:
  - If user picks Air handler for Appliance type, then:
    - Users must pick one or more air handler types that are: Air conditioner, Heater, or Heat pump.
    - If user picks Air conditioner for air handler type, then:
      - User enters Energy efficiency ratio (EER)
    - If user picks Heater for air handler type, then:
      - User picks the energy sources that are Electric, Gas, or Fuel oil
    - If user picks Heat pump for air handler type, then:
      - User enters Seasonal Energy Efficiency Rating (SEER) and Heating Seasonal Performance Factor (HSPF)
  - Else if user picks Water heater for Appliance type, then:
    - User picks an energy source that are Electric gas, Gas, thermosolar, or heat pump
    - User enters Capacity and current temperature
- Upon clicking Add button
  - Write user's input into Appliance
  - Jump to Appliance list

Abstraction Code with SQL Statements

- User picks Appliance type
- Based on user Appliance type:
    - If user picks Air handler for Appliance type, then:

```
SELECT manufacturer_name FROM Manufacturer
```

        - Populate list of manufacturer_name from the query result above to the dropdown list
        - User types model name
        - Users must pick one or more air handler types that are: Air conditioner, Heater, or Heat pump.
        - If user picks Air conditioner for air handler type, then:
            - User enters Energy efficiency ratio (EER)
        - If user picks Heater for air handler type, then:
            - User picks the energy sources that are Electric, Gas, or Fuel oil
        - If user picks Heat pump for air handler type, then:
            - User enters Seasonal Energy Efficiency Rating (SEER) and Heating Seasonal Performance Factor (HSPF)
    - Else if user picks Water heater for Appliance type, then:

```
SELECT manufacturer_name FROM Manufacturer
```

        - Populate list of manufacturer_name from the query result above to the dropdown list
        - User types model name
        - User picks an energy source that are Electric gas, Gas, thermosolar, or heat pump
        - User enters Capacity and current temperature
- Upon clicking Add button

```
INSERT INTO Appliance (email, btu_rating, model_name, appliance_type,
manufacturer_name) VALUES ('email', 1000, 'Varmvatten' , 'Water heater',
'Yurginloogen');
```

    - If user picked Air handler for appliance type, then:
        - If user picked Air conditioner:

```
INSERT INTO AirConditioner
    SET email='email',
    appliance_number=(
        SELECT MAX(appliance_number)
        FROM Appliance
        WHERE email='email'
    ),
    eer=94.3424;
```

        - If user picked Heater:

```
INSERT INTO Heater
    SET email='email',
```

```
appliance_number=(
        SELECT MAX(appliance_number)
        FROM Appliance
        WHERE email='email'
),
energy_source='electric'
```

- If user picked Heat Pump:

```
INSERT INTO HeatPump
        SET email='email',
        appliance_number=(
                SELECT MAX(appliance_number)
                FROM Appliance
                WHERE email='email'
        ),
        seer=94.3424,
        hsbf=94.3424;
```

- If user picked Water heater for appliance type, then:

```
INSERT INTO WaterHeater
        SET email='email',
        appliance_number=(
                SELECT MAX(appliance_number)
                FROM Appliance
                WHERE email='email'
        ),
        energy_source='heat pump',
        temperature=58,
        capacity=123.56;
```

- Jump to Appliance list

Listing an Appliances

Abstraction Code

- Get the user's Appliance Number, Type, Manufacturer, and model from the Appliance and display the corresponding information.
- Upon clicking delete button:
  - Delete corresponding information from the Appliance.
  - Get updated user's Appliance Number, Type, Manufacturer, and model from the Appliance and display the corresponding information.
- Upon clicking "+Add another appliance" button:
  - Jump to appliance form the page (as described above) and have the user input the appliance information.
- Upon clicking Next button:
  - Check if there is at least one appliance list:

        - throw conditionals error saying there must be at least one appliance list
    - else:
        - Jump to the Power generation form page.

Abstraction Code with SQL Statements

- Get the user's Appliance Number, Type, Manufacturer, and model from the Appliance and display the corresponding information.

```
SELECT appliance_number,appliance_type, manufacturer_name, model_name
FROM Appliance WHERE A.email = 'email';
```

- Upon clicking delete button:
    - Delete corresponding information from the Appliance.

```
DELETE FROM Appliance WHERE email='email' AND
appliance_number='2';
```

    - Get updated user's Appliance Number, Type, Manufacturer, and model from the Appliance and display the corresponding information.

```
SELECT appliance_number, appliance_type, manufacturer_name,
model_name FROM Appliance WHERE A.email = 'email';
```

- Upon clicking "+Add another appliance" button:
    - Jump to appliance form the page (as described above) and have the user input the appliance information.
- Upon clicking Next button:
    - Check if there is at least one appliance list:

```
SELECT COUNT(*) FROM Appliance WHERE email='email';
```

        - throw conditionals error if the count is 0 saying there must be at least one appliance list
    - else:
        - Jump to the Power generation form page.

Add Power Generation

Abstraction Code

- Check if the Current User has a public utilities as "off-the-grid" via Household
- If the user has a public utility as "off-the-grid", user has to enter the information
    - User picks Energy type
    - User enters Monthly Power Generated and Battery Storage Capacity
        - Battery Storage Capacity can be left out blank, if not applicable
    - Upon clicking Add button
        - Write user's input into Appliance
        - Jump to Power Generation list
- If the user has a public utility as "on-the-grid":
    - Skip button will be available, and user doesn't have to fill out the form

    - If the “skip” button is clicked, then:
        - Jump to Power Generation list
    - User can still enter the information if needed similar to above steps
        - User picks Energy type
        - User enters Monthly Power Generated and Battery Storage Capacity
            - Battery Storage Capacity can be left out blank, if not applicable
        - Upon clicking Add button
            - Write user’s input into Appliance table
            - Jump to Power Generation list

Abstraction Code with SQL Statements

- Check if the Current User has a public utilities as “off-the-grid” via Household

  SELECT COUNT(*) **FROM** PublicUtility **WHERE** email=’email’;
- If the count from query result above is 0, meaning user has a public utility as “off-the-grid”, user has to enter the information
    - User picks Energy type
    - User enters Monthly Power Generated and Battery Storage Capacity
        - Battery Storage Capacity can be left out blank, if not applicable
    - Upon clicking Add button

      INSERT INTO PowerGeneration (email, generation_type, monthly_power_generated, battery_storage_capacity) **VALUES** (‘Solar-electric’, 300, 200);
        - Jump to Power Generation list
- If the count from query result above is greater than 0, meaning user has a public utility as “on-the-grid”:
    - Skip button will be available, and user doesn’t have to fill out the form
    - If the “skip” button is clicked, then:
        - Jump to Power Generation list
    - User can still enter the information if needed similar to above steps
        - User picks Energy type
        - User enters Monthly Power Generated and Battery Storage Capacity
            - Battery Storage Capacity can be left out blank, if not applicable
        - Upon clicking Add button

          INSERT INTO PowerGeneration (email, generation_type, monthly_power_generated, battery_storage_capacity) **VALUES** (‘Solar-electric’, 300, 200);
            - Jump to Power Generation list

Power Generation Listing

Abstraction Code

- Get the user's Power Generation Number, Generation Type, Average Power, and Storage Capacity from the Power Generation and display the corresponding information.
- Upon clicking delete button:
    - Delete corresponding information from the Power Generation.
    - Get updated user's Power Generation Number, Generation Type, Average Power, and Battery Storage Capacity from the Power Generation and display the corresponding information.
- Upon clicking "+Add another power" button:
    - Jump to the power generation form page (as described above) and have the user input the power generation information.
- Upon clicking Finish button:
    - Check if there is at least one power generation list for household with "off-the-grid" public utility:
        - throw conditionals error saying there must be at least one power generation for household with "off-the-grid" public utilities:
    - Else:
        - Jump to the Power generation form page.

Abstraction Code with SQL Statements

- Get the user's Power Generation Number, Generation Type, Average Power, and Storage Capacity from the Power Generation and display the corresponding information.

    SELECT power_generation_number, generation_type, monthly_power_generated, battery_storage_capacity **FROM** PowerGeneration **WHERE** email='email'
- Upon clicking delete button:
    - Delete corresponding information from the Power Generation.

        DELETE FROM PowerGeneration **WHERE** email='email' **AND** power_generation_number='2';
    - Get updated user's Power Generation Number, Generation Type, Average Power, and Battery Storage Capacity from the Power Generation and display the corresponding information.

        SELECT power_generation_number, generation_type, monthly_power_generated, battery_storage_capacity **FROM** PowerGeneration **WHERE** email='email'
- Upon clicking "+Add another power" button:
    - Jump to the power generation form page (as described above) and have the user input the power generation information.
- Upon clicking Finish button:

- Check if there is at least one power generation list for household with "off-the-grid" public utility:

  ```
  SELECT COUNT(*) FROM PowerGeneration WHERE email='email';
  ```

  - If the count from the result query above is 0, throw conditional error saying there must be at least one power generation for household with "off-the-grid" public utilities:
  - Else:
    - Jump to the Submission Complete page.

Reportings

Report 00 - Main Report Menu

Abstraction Code

- Display Link to Top 25 Manufacturers Report
- Display Link to Manufacturer/Model Search Report
- Display Link to Heating/Cooling Methods Detail Report
- Display Link to Water Heater Statistics by State Report
- Display Link to Off the Grid Household by grid status Report
- Display Link to Household average by Radius Report
- Upon clicking a Report link,
  - Direct to the corresponding Reporting Page
- If user clicks back button,
  - Direct to the Main menu page

Report 01 - Top 25 Popular Manufacturers Report

Abstraction Code

- User clicks on **Top 25 Manufactures** button subsection from **View Report** Menu
- Run **Top 25 Manufactures** task.
    - Get list of Manufacturer Name
    - For Each Manufacturer Name display affiliated Appliance Count
    - Order this list by Appliance Count Descending
    - Display only the first 25 of the result.
    - For Each Manufacturer Name display also button – **Drilldown Report**

- Run **Drilldown** sub-task:
    - If **Drilldown Report** button is clicked, then:
        - Get Manufacturer Name displayed at the top of a new report on a separate dialog box.
        - If Appliance Count is NULL, then:
            - display that N/A.
        - Else:
            - Get Appliance Type and Appliance Count for this Manufacturer Name

Abstraction Code with SQL Statements

- User clicks on Top 25 Manufactures button subsection from View Report Menu
- Run Top 25 Manufactures task.

  SELECT manufacturer_name, **COUNT**(*) **AS** appliances_number **FROM** Appliance **GROUP BY** manufacturer_name **ORDER BY** appliances_number **DESC LIMIT** 25;
    - For Each Manufacturer Name display also button – **Drilldown Report**
- Run Drilldown sub-task:
    - If Drilldown Report button is clicked, then:
        - Get Manufacturer Name displayed at the top of a new report on a separate dialog box. Then a table with the following details: appliance type, raw count of appliances of that type for that manufacturer **SELECT** appliance_type, **COUNT**(*) **AS** Appliance_Count **FROM** Appliance **WHERE** manufacture_name = ‘manufacture_name’ **GROUP BY** appliance_type;
        - If Appliance Count is NULL, then:
            - display that N/A.

Report 02 – Manufacturer/Model Search

Abstraction Code

- User clicked on **Manufactures/Model Search** button subsection from **View Report** Menu
- Run **Search** subtask.
    - Show *Keyword* input fields and wait for user input.
    - Show **Cancel**, and **Search** buttons.
    - Upon:
        - Click **Search** button
            - **If** input field is empty, then:
                - Do nothing
            - **Else** validate and verify input before querying the database.
                - Search keywords, case insensitive from both Manufacturer Name and/or Appliance Model_Name
                - Run **Display Result** subtask by returning results and populate the corresponding list.
        - Click **Cancel** button
            - Exits out of Manufacture/Model Search and goes back to the reports menu

- Run **Display Result** subtask
    - Show **Back** button at the end of the result table.
    - Return subset of dataset by returning results that meet all criteria populated
        - Get results from Manufacturer Name and/or Appliance Model Name displayed in two columns displayed in Ascending order for both.
            - Search *Keyword* match is case insensitive
            - Search *Keyword* string displayed is highlighted with a light green background.
    - If user clicks **Back** button, then:
        - Exits out of **Display Result** subtask and goes back to **Search** subtask

Abstraction Code with SQL Statements

- User clicked on Manufactures/Model Search button subsection from View Report Menu
- Run Search subtask.
    - Show *Keyword* input fields and wait for user input.
    - Show Cancel, and Search buttons.
    - Upon:
        - Click Search button
            - If input field is empty, then:

        - Do nothing
    - Else validate and verify input before querying the database.
        - Search keywords, case insensitive from both Manufacturer Name and/or Appliance Model_Name
        - Run Display Result subtask by returning results and populate the corresponding list.
- Click Cancel button
    - Exits out of Manufacture/Model Search and goes back to the reports menu

- Run Display Result subtask
    - Show Back button at the end of the result table.
    - Return subset of dataset by returning results that meet all criteria populated

      SELECT manufacturer_name, model_name
      FROM Appliance WHERE manufacturer_name **LIKE** '%query%' **OR** model_name **LIKE** '%query%'
      ORDER BY manufactuer_name **ASC**, model_name **ASC**;
        - Search *Keyword* match is case insensitive
        - Search *Keyword* string displayed is highlighted with a light green background.
    - If user clicks Back button, then:
        - Exits out of Display Result subtask and goes back to Search subtask

Report 03 – Heating/Cooling Method Details

Abstraction Code

- User clicked on **Heating/Cooling Method Details** button subsection from **View Report** Menu
- Show **Cancel** button
- Run subtask - **Grouped by Household Types**
    - Display list of Household Types
    - For each Household Types
        - Get list of available Air Handler and its **Heating/Cooling Method**
    - According to the **Heating/Cooling Method**, display information
        - If **Heating/Cooling Method** is Air Conditioner, then:
            - Display from Air Conditioner -> Count, Average BTU, Average EER
        - Else if **Heating/Cooling Method** is Heaters Count:
            - Display from Heaters -> Count, Average BTU, Average SEER
        - Else **Heating/Cooling Method** is Heat Pumps:
            - Display from Heat Pumps -> Average BTU, Average HSPF
- If user clicks **Cancel** button, then:
    - Exits out of current page and goes back to the main **Report Menu**

Abstraction Code with SQL Statements

- User clicked on Heating/Cooling Method Details button subsection from View Report Menu
- Show Cancel button
- Run subtask - Grouped by Household Types
    - Display list of Household Types
    - For each Household Types
        - Get list of available Air Handler and its Heating/Cooling Method
    - According to the Heating/Cooling Method, display information below:
        - Air Conditioner:

```
SELECT
        Household.household_types,
        COUNT(*) AS num_households,
        ROUND(AVERAGE(Appliance.btu_rating)) AS
average_btu,
        ROUND(AVERAGE(AirConditioner.eer)) AS average_eer
FROM Household
LEFT JOIN AirConditioner ON Household.email =
AirConditioner.email
LEFT JOIN Appliance ON Appliance.appliance_number =
AirConditioner.appliance_number
GROUP BY Household.household_types;
```

- Heaters:

```
SELECT
    Household.household_types,
    COUNT(*) AS num_households,
    ROUND(AVERAGE(Appliance.btu_rating)) AS average_btu,
    Most_common_energy_source = (
        SELECT energy_source
        FROM Heater
        RIGHT JOIN Household ON Heater.email =
        Household.email
        GROUP BY energy_source
        ORDER BY COUNT(*) DESC
        LIMIT 1
    )
FROM Household
LEFT JOIN Heater ON Household.email = Heater.email
LEFT JOIN Appliance ON Appliance.appliance_number =
Heater.appliance_number
GROUP BY Household.household_types;
```

- Heat Pumps:

```
SELECT
    Household.household_types,
    COUNT(*) AS num_households,
    ROUND(AVERAGE(Appliance.btu_rating)) AS average_btu,
    ROUND(AVERAGE(HeatPump.seer)) AS average_seer,
    ROUND(AVERAGE(HeatPump.hspf)) AS average_seer
FROM Household
LEFT JOIN HeatPump ON Household.email = HeatPump.email
LEFT JOIN Appliance ON Appliance.appliance_number =
HeatPump.appliance_number
GROUP BY Household.household_types
```

- If user clicks Cancel button, then:
    - Exits out of current page and goes back to the main Report Menu

Report 04 – Water heater statistics by state

Abstraction Code

- User clicked on **Water heater statics by state** button subsection from **View Report** Menu
- Show **Cancel** buttons
- Run subtask - **Calculate average capacity, temperature, BTU, and count Water heater with NULL:**
    - For each state that has Water heater:
        - Calculate average the Water heater capacity and round them up by whole number.
        - Calculate average the Water heater BTU rating and round them up by whole number.
        - Calculate average the Water heater temperature and round them up by tenths.
        - Count all Water heater where Water heater temperature is Null
        - Count all Water heater where Water heater temperature is not Null
    - For the state that doesn't have any Water heater associated:
        - All values will be 0.
    - Consolidate information for each state and sort it by state Abbreviation in an ascending order.
    - If user clicks **Cancel** button, then:
        - Exits out of current page and goes back to the main **Report Menu**
    - When the user clicks on the link for each state provided in a row, a drill-down report will be displayed for the selected State.

- Run subtask - **Calculate average capacity, temperature, by each energy source in a state:**
    - Show **Back** button at the end of the drilldown report.
    - For each drill down report:
        - Get a list of all Water heater Appliances for the selected State
        - Group the Water heater appliances by each Water heater energy_source
        - For each energy source:
            - Get MIN Water heater capacity and round them up by whole number.
            - Calculate average the Water heater capacity and round them up by whole number.
            - Get MAX Water heater capacity and round them up by whole number.
            - Get MIN Water heater Temperature and round them up by whole tenths.
            - Calculate average the Water heater Temperature and round them up by whole tenths.
            - Get MAX Water heater Temperature and round them up by whole tenths.

        - Consolidate information for each state and sort it by energy source in an ascending order.
    - If user clicks **Back** button, then:
        - Exits out of the current page and goes back to **Water heater statistics by state** page.

Abstraction Code with SQL Statements

- User clicked on **Water heater statics by state** button subsection from **View Report** Menu
- Show **Cancel** buttons
- Run subtask - **Calculate average capacity, temperature, BTU, and count Water heater with NULL:**
    - For each state that has Water heater:
        - Calculate average the Water heater capacity and round them up by whole number.
        - Calculate average the Water heater BTU rating and round them up by whole number.
        - Calculate average the Water heater temperature and round them up by tenths.
        - Count all Water heater where Water heater temperature is Null
        - Count all Water heater where Water heater temperature is not Null
    - For the state that doesn't have any Water heater associated:
        - All values will be 0.
    - Consolidate information for each state and sort it by state Abbreviation in an ascending order.
    - If user clicks **Cancel** button, then:
        - Exits out of current page and goes back to the main **Report Menu**
    - When the user clicks on the link for each state provided in a row, a drill-down report will be displayed for the selected State.

```
SELECT
        Address.state,
        ROUND(AVG(WaterHeater.capacity)) AS avg_capacity,
        ROUND(AVG(Appliance.btu_rating)) AS avg_btu_rating,
        ROUND(AVG(WaterHeater.temperature)) AS avg_temperature,
        COUNT(WaterHeater.temperature) AS temperature_count
FROM
        Address
LEFT JOIN
        Household ON Address.postal_code = Household.postal_code
LEFT JOIN
        WaterHeater ON Household.email = WaterHeater.email
LEFT JOIN
```

```
        Appliance ON WaterHeater.email = Appliance.email AND
        WaterHeater.appliance_number = Appliance.appliance_number
GROUP BY
        Address.state
ORDER BY
        Address.state ASC;
```

- Run subtask - **Calculate average capacity, temperature, by each energy source in a state:**
    - Show **Back** button at the end of the drilldown report.
    - For each drill down report:
        - Get a list of all Water heater Appliances for the selected State
        - Group the Water heater appliances by each Water heater energy_source
        - For each energy source:
            - Get MIN Water heater capacity and round them up by whole number.
            - Calculate average the Water heater capacity and round them up by whole number.
            - Get MAX Water heater capacity and round them up by whole number.
            - Get MIN Water heater Temperature and round them up by whole tenths.
            - Calculate average the Water heater Temperature and round them up by whole tenths.
            - Get MAX Water heater Temperature and round them up by whole tenths.
        - Consolidate information for each state and sort it by energy source in an ascending order.
    - If user clicks **Back** button, then:
        - Exits out of the current page and goes back to **Water heater statistics by state** page.

For a given State - 'NY'

```
SELECT
  WaterHeater.energy_source,
  ROUND(MIN(WaterHeater.capacity)) AS min_capacity,
  ROUND(AVG(WaterHeater.capacity)) AS avg_capacity,
  ROUND(MAX(WaterHeater.capacity)) AS max_capacity,
  ROUND(MIN(WaterHeater.temperature), 1) AS min_temperature,
  ROUND(AVG(WaterHeater.temperature), 1) AS avg_temperature,
  ROUND(MAX(WaterHeater.temperature), 1) AS max_temperature
FROM
  Address
  JOIN Household ON Address.postal_code = Household.postal_code
```

```
    JOIN WaterHeater ON Household.email = WaterHeater.email
WHERE
    Address.state = 'NY'
GROUP BY
    WaterHeater.energy_source
ORDER BY
    WaterHeater.energy_source ASC;
```

Report 05 – Off-the-grid household dashboard

Abstraction Code

User clicked on **Off-the-grid Household Dashboard** button subsection from **View Report** Menu

- Display **Cancel** button
- Group by household types (off/on-the grid)
- Run the following records:
    - Run subtask - **Off-the-grid household by states**
        - Get zero public utilities count from the Household and group them by state.
        - Get the maximum household counts.
        - Display the values with the columns of state and its off-the-grid counts in the report.
    - Run subtask - **Average battery storage subtask**
        - Get the average battery storage value with zero public utilities count from the Household and Power Generation
        - Round the average battery storage value in the whole number
        - Display the average battery storage values with the columns of the off-the-grid household in the report.
    - Run subtask - **Percentage power generation subtask**
        - Get solar-electric, wind, and mixed counts from the power generation
        - Get the percentage values for solar-electric, wind and mixed in decimals rounded by tenths.
        - Display the percentage of power generation values with the columns of solar electric, wind and mixed in the report.
    - Run subtask - **Average water heater gallon capacity subtask**
        - Get the average water heater gallon capacity with zero public utilities count from the Household and Appliance. (for off-the-grid)
        - Get the average water heater gallon capacity with the ones with at least one public utility count from the Household and Appliance. (for “on-the-grid”)
        - Get the average water heater gallon capacity for both off-the-grid and “on-the-grid” and round them up by tenths.

        - Display the average water heater gallon capacity with the columns of on/off-the-grid in the report.

    - Run subtask - **Min/Avg/Max BTU subtask**
        - Get the BTU values with zero public utilities count from the Household and Appliance and group them by appliance type.
        - Get the minimum, average, and maximum BTU values and round them up to the whole number.
        - Display the minimum, average, and maximum BTU values for each appliance type in the report.

    - If user clicks **Cancel** button, then:
        - Exits out of current page and goes back to the main **Report Menu**

Abstraction Code with SQL Statements

- User clicked on **Off-the-grid Household Dashboard** button subsection from **View Report** Menu
- Display **Cancel** button
- Group by household types (off/on-the grid)
- Run the following records:
    - Run subtask - **Off-the-grid household by states**
        - Get zero public utilities count from the Household and group them by state.
        - Get the maximum household counts.
        - Display the values with the columns of state and its off-the-grid counts in the report.

```
SELECT Address.state, COUNT(PublicUtility.public_utility) AS
off_the_grid_count
FROM Household
LEFT JOIN Address ON Household.postal_code = Address.postal_code
LEFT JOIN PublicUtility ON Household.email = PublicUtility.email
WHERE off_the_grid_count = 0
GROUP BY Address.state;
```

    - Run subtask - **Average battery storage subtask**
        - Get the average battery storage value with zero public utilities count from the Household and Power Generation
        - Round the average battery storage value in the whole number
        - Display the average battery storage values with the columns of the off-the-grid household in the report.

```
SELECT Address.state AS state,
ROUND(AVERAGE(Power_Generation.battery_storage_capacity)) AS
average_battery_capacity
FROM Household
LEFT JOIN Address ON Household.postal_code = Address.postal_code
LEFT JOIN Power_Generation ON Household.email = Power_Generation.email
LEFT JOIN PublicUtility ON Household.email = PublicUtility.email
WHERE PublicUtility.public_utility IS NULL
GROUP BY state;
```

- Run subtask - **Percentage power generation subtask**
  - Get solar-electric, wind, and mixed counts from the power generation
  - Get the percentage values for solar-electric, wind and mixed in decimals rounded by tenths.
  - Display the percentage of power generation values with the columns of solar electric, wind and mixed in the report.

```
SET @solar_electric = (SELECT
        COUNT(*) as count
FROM Household
LEFT JOIN Power_Generation ON Household.email = Power_Generation..email
LEFT JOIN PublicUtility ON Household.email = PublicUtility.email
WHERE PublicUtility.public_utility IS NULL AND WHERE
Power_Generation.generation_type = 'Solar-electric'');

SET @wind = (SELECT
        COUNT(*) as count
FROM Household
LEFT JOIN Power_Generation ON Household.email = Power_Generation..email
LEFT JOIN PublicUtility ON Household.email = PublicUtility.email
WHERE PublicUtility.public_utility IS NULL AND WHERE
Power_Generation.generation_type = 'wind'');

SET @mixed = (SELECT
        COUNT(*) as count
FROM Household
LEFT JOIN Power_Generation ON Household.email = Power_Generation..email
LEFT JOIN PublicUtility ON Household.email = PublicUtility.email
WHERE PublicUtility.public_utility IS NULL AND
WHERE  Power_Generation.generation_type IN ("solar-electric", "wind")
HAVING COUNT(DISTINCT Power_Generation.generation_type) = 2;
);

SET @total = SUM(@solar_electric, @wind.count, @mixed);
```

```
SET @solar_perc = @solar_electric/@total;
SET @wind_perc = @wind/@total;
SET @mixed_perc = @mixed/@total;

SELECT @solar_perc AS Solar, @wind_perc AS wind, @mixed_perc AS
mixed
```

- Run subtask - **Average water heater gallon capacity subtask**
  - Get the average water heater gallon capacity with zero public utilities count from the Household and Appliance. (for off-the-grid)
  - Get the average water heater gallon capacity with the ones with at least one public utility count from the Household and Appliance. (for "on-the-grid")
  - Get the average water heater gallon capacity for both off-the-grid and "on-the-grid" and round them up by tenths.
  - Display the average water heater gallon capacity with the columns of on/off-the-grid in the report.

```
SELECT Address.state AS state, ROUND(AVERAGE(Water_Heater.capacity),2)
AS average_water_heater_gallon_capacity_OFF,
(SELECT ROUND(AVERAGE(Water_Heater.capacity),2) AS
average_water_heater_gallon_capacity_ON
FROM Household
LEFT JOIN Address ON Household.postal_code = Address.postal_code
LEFT JOIN Appliance ON Household.email = Appliance.email
LEFT JOIN PublicUtility ON Household.email = PublicUtility.email
LEFT JOIN Water_Heater ON Water_Heater.email = Household.email
WHERE PublicUtility.public_utility IS NOT NULL
GROUP BY state;
)
FROM Household
LEFT JOIN Address ON Household.postal_code = Address.postal_code
LEFT JOIN Appliance ON Household.email = Appliance.email
LEFT JOIN PublicUtility ON Household.email = PublicUtility.email
LEFT JOIN Water_Heater ON Water_Heater.email = Household.email
WHERE PublicUtility.public_utility IS NULL
GROUP BY state;
```

- Run subtask - **Min/Avg/Max BTU subtask**
  - Get the BTU values with zero public utilities count from the Household and Appliance and group them by appliance type.
  - Get the minimum, average, and maximum BTU values and round them up to the whole number.

        - Display the minimum, average, and maximum BTU values for each appliance type in the report.

```
SELECT
    ROUND(MIN(Appliance.btu_rating), 0) AS min_btu,
    ROUND(AVG(Appliance.btu_rating), 0) AS avg_btu,
    ROUND(MAX(Appliance.btu_rating), 0) AS max_btu,
    COUNT(PublicUtility.public_utility) AS off_the_grid_count
FROM Household
LEFT JOIN Appliance ON Household.email = Appliance.email
LEFT JOIN PublicUtility ON Household.email = PublicUtility.email
WHERE off_the_grid_count = 0
GROUP BY Appliance.appliance_type
```

    - If user clicks **Cancel** button, then:
        - Exits out of current page and goes back to the main **Report Menu**

Report 06 – Household averages by Radius

Abstraction Code

- User clicked on **Household averages by Radius** link/button subsection from **View Report** Menu
- Display **Postal Code search bar, Radius dropdown**, **Cancel** and **Back** button
- Run Subtask - Search stats
    - When user enters postal code and chooses a radius from dropdown and clicks search button,
        - If the Postal Code exists in the records, then:
            - Show Display Results subtask
        - Else:
            - Display Error Message
    - Click **Back** button
        - Exists out of **Household Average by Radius** report and goes back to search page

- Run Subtask - Display Results
    - Display search parameters: Postal Code and *Search Radius*
    - Return subset of the dataset by returning results that meet all the criteria populated:
        - Distance = Less than or equal to distance between two Postal Code
            - Convert the Latitude and Longitude to radians value
            - Haversine formula used to calculate distance between two points defined by latitude and longitude coordinates as follows:

$$\Delta\, lat = lat2 - lat1$$
$$\Delta\, lon = lon2 - lon1$$

$$a = \sin2\,(\Delta\, lat\,/\,2) + \cos(lat1) * \cos(lat2) * \sin2\,(\,\Delta\, lon\,/\,2)$$
$$c = 2*\text{atan2}\,(\sqrt{a},\sqrt{(1-a)})$$
$$d=R*c$$

- Within this radius
    - Display all from Household Count according to Household Type including count = 0.
    - Display from Household -> Average Square Footage, Average Heating Temp, Average Cooling Temp, Public Utilities
        - Also Display - Grid Status of a Household,
    - Display from Power Generation -> Count, Average Monthly Power Generation
        - Also Display - Homes with power generation, Most common generation method,
    - if available, Display from Battery -> Battery Storage
- If the search Radius is 0, show the statistics for the households that are within that postal code.
- Click **Back** link/button goes to previous screen of subtask - Search

- If user clicks **Cancel** button, then:
    - Exits out of current page and goes back to the main **Report Menu**

```
SELECT
   COUNT(*) AS Household_count,

   SUM(CASE WHEN household_types = 'House' THEN 1 ELSE 0 END) AS type_house,
   SUM(CASE WHEN household_types = 'Apartment' THEN 1 ELSE 0 END) AS
type_apartment,
   SUM(CASE WHEN household_types = 'Townhome' THEN 1 ELSE 0 END) AS
type_townhome,
   SUM(CASE WHEN household_types = 'Condominium' THEN 1 ELSE 0 END)
type_condominium,
   SUM(CASE WHEN household_types = 'Mobile Home' THEN 1 ELSE 0 END) AS
type_mobile_home,

   ROUND(AVERAGE(Household.square_footage)) AS Avg_Sq_Ft,
   ROUND(AVERAGE(Heating.temperature), 1) AS Avg_Heating_Temp,
   ROUND(AVERAGE(Cooling.temperature), 1) AS Avg_Cooling_Temp,

   GROUP_CONCAT(DISTINCT PublicUtility.public_utility SEPARATOR ', ') AS
Used_PublicUtilities,

   COUNT(IF(PublicUtility.public_utility = 0)) AS off_the_grid_Count,

   (SELECT generation_type FROM PowerGeneration
            GROUP BY generation_type
```

```
                ORDER BY COUNT(*) DESC LIMIT 1) as Most_Common_Method

ROUND(AVERAGE(PowerGeneration.power_generation_number)) AS
Avg_Monthly_Power_Generation,
COUNT(IF(PowerGeneration.battery_storage_capacity IS NOT NULL)) AS
Count_With_Battery_Storage

FROM
   Household h
   LEFT JOIN Heating ON Household.email = Heating.email
   LEFT JOIN Cooling ON Household.email = Cooling.email
   INNER JOIN Address a ON h.postal_code = a.postal_code
   INNER JOIN PublicUtility pu ON h.email = pu.email
   LEFT JOIN PowerGeneration pg ON h.email = pg.email

WHERE
            (
            3956.75 *
            acos(cos(radians((SELECT Address.latitude FROM Address WHERE
      postal_code = [USER_POSTAL_CODE]))) *
            cos(radians(latitude)) *
            cos(radians(longitude) - radians((SELECT Address.longitude FROM Address
      WHERE postal_code = [USER_POSTAL_CODE]))) +
            sin(radians((SELECT Address.latitude FROM Address WHERE postal_code =
      [USER POSTAL CODE]))) *
            sin(radians(latitude))
            ) <= ([USER_INPUT_RADIUS] / 3956.75)
GROUP BY
   Address.postal_code;
```