***Phase 4: Development***

**Code:-**

1. Demographic Analysis

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

# Load the demographic data

demographic\_data = pd.read\_csv(‘demographic\_data.csv’)

# Load the electricity price data

electricity\_price\_data = pd.read\_csv(‘https://www.kaggle.com/datasets/chakradharmattapalli/electricity-price-prediction’)

# Merge the two datasets

merged\_data = pd.merge(demographic\_data, electricity\_price\_data, on=’customer\_id’)

# Get the descriptive statistics of the merged dataset

merged\_data.describe()

# Create a correlation matrix of the merged dataset

correlation\_matrix = merged\_data.corr()

# Plot the correlation matrix

sns.heatmap(correlation\_matrix, annot=True)

plt.show()

# Select the demographic features that are most correlated with electricity price

selected\_features = [‘age’, ‘income’, ‘household\_size’]

# Create a new dataframe with the selected features

selected\_data = merged\_data[selected\_features]

# Split the data into training and test sets

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(selected\_data, merged\_data[‘electricity\_price’], test\_size=0.25)

# Train a linear regression model

from sklearn.linear\_model import LinearRegression

model = LinearRegression()

model.fit(X\_train, y\_train)

# Make predictions on the test set

Y\_pred = model.predict(X\_test)

# Evaluate the model

From sklearn.metrics import mean\_squared\_error

mse = mean\_squared\_error(y\_test, y\_pred)

print(‘Mean squared error:’, mse)

1. Visualisation

Import pandas as pd

Import numpy as np

Import seaborn as sns

Import matplotlib.pyplot as plt

# Load the electricity price prediction data

electricity\_price\_prediction\_data = pd.read\_csv(‘https://www.kaggle.com/datasets/chakradharmattapalli/electricity-price-prediction’)

# Create a line plot of the predicted electricity price

sns.lineplot(x=electricity\_price\_prediction\_data[‘date’], y=electricity\_price\_prediction\_data[‘predicted\_electricity\_price’])

plt.title(‘Electricity Price Prediction’)

plt.xlabel(‘Date’)

plt.ylabel(‘Electricity Price’)

plt.show()

# Create a histogram of the predicted electricity price

sns.histplot(electricity\_price\_prediction\_data[‘predicted\_electricity\_price’])

plt.title(‘Distribution of Predicted Electricity Price’)

plt.xlabel(‘Predicted Electricity Price’)

plt.ylabel(‘Frequency’)

plt.show()

# Create a boxplot of the predicted electricity price by day of the week

sns.boxplot(X=electricity\_price\_prediction\_data[‘day\_of\_week’],Y=electricity\_price\_prediction\_data,[‘predicted\_electricity\_price’],Showmeans=True)

plt.title(‘Predicted Electricity price by Day of the Week’)

plt.xlabel(‘Day of the Week’)

plt.ylabel(‘Predicted Electricity Price’)

plt.show()

# Create a heatmap of the predicted electricity price by hour of the day and day of the week

sns.heatmap(Electricity\_price\_prediction\_data.pivot\_table,values=’predicted\_electricity\_price’,index=’hour\_of\_day’,columns=’day\_of\_week’,aggfunc=np.mean))

plt.title(‘Predicted Electricity Price by Hour of the Day and Day of the Week’)

plt.xlabel(‘Hour of the Day’)

plt.ylabel(‘Day of the Week’)

plt.show()