Experiment 1

Aim: Collecting and Storing Sensor Data

Objective

- To collect temperature and pressure sensor data
- To store the collected data in a CSV file
- To perform basic data visualization

Theory:

In this experiment, we're simulating sensor data for two different parameters: temperature and pressure. We generate random data for both sensors, simulate the sensor readings, and then compute some basic statistical metrics (like average, peak, and standard deviation) on the data. The collected data is stored in a Pandas DataFrame and saved to a CSV file. We also visualize the data using `matplotlib` by plotting graphs for both temperature and pressure. Finally, we read the saved CSV file back and calculate additional statistics, such as the standard deviation.

Libraries Used:

1. **NumPy (np)**:
    - Used for numerical operations, particularly for generating random data (np.random.randn()) and performing mathematical functions like mean and max.
2. **Matplotlib (plt)**:
    - Used for plotting the sensor data. The graphs display temperature and pressure trends over time and highlight average values.
3. **Pandas (pd)**:
    - Used for creating and manipulating the DataFrame that holds the sensor data. It makes it easy to save and load data into CSV format and compute various statistics.
4. **OS (os)**:
    - Used to manage file paths and ensure the data is saved in the correct location by joining paths (os.path.join()) and interacting with the file system.

These libraries combine to facilitate data generation, analysis, visualization, and storage in this experiment.

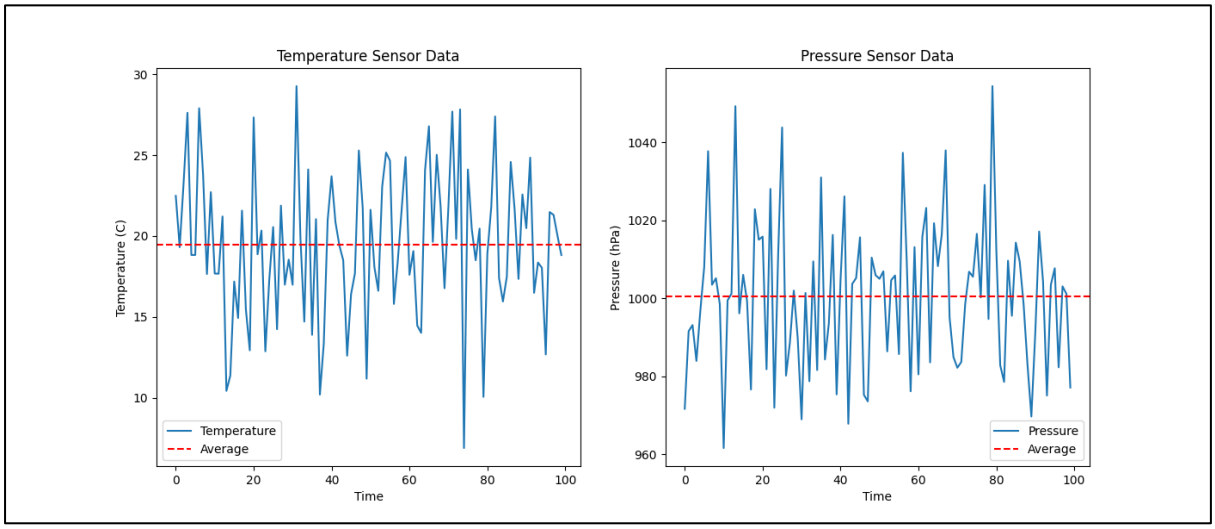
Code:

```
# Import necessary libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import os

# Set random seed for reproducibility
np.random.seed(42)

# Function to simulate sensor data
def simulate_sensor_data(sensor_type, n_pts=100):
    """
    Generates random sensor data based on the sensor type.
    :param sensor_type: Type of sensor ('temp' for temperature, 'press'
for pressure)
    :param n_pts: Number of data points to generate (default=100)
    :return: NumPy array of generated data
    """
    if sensor_type == "temp":
        data = 20 + 5 * np.random.randn(n_pts)  # Mean 20, std 5
    elif sensor_type == "press":
        data = 1000 + 20 * np.random.randn(n_pts)  # Mean 1000, std 20
    else:
        raise ValueError("Invalid sensor type")
    return data

# Collect 100 data points for temperature and pressure
t_d = simulate_sensor_data('temp')
p_d = simulate_sensor_data('press')

# Process data: Compute statistical metrics
temp_avg = np.mean(t_d)  # Average temperature
press_avg = np.mean(p_d)  # Average pressure
temp_peak = np.max(t_d)  # Peak temperature value
press_peak = np.max(p_d)  # Peak pressure value

# Display results
print(f"Average temperature: {temp_avg:.2f}, Peak: {temp_peak:.2f}")
print(f"Average pressure: {press_avg:.2f}, Peak: {press_peak:.2f}")

# Create a Pandas DataFrame to store sensor data
data = {
    'Temperature': t_d,
    'Pressure': p_d
}
```

```
df = pd.DataFrame(data)

# Specify the path where the CSV file will be stored
csv_file_path = os.path.join(os.getcwd(), 'sensor_data.csv')

# Save the DataFrame to a CSV file
df.to_csv(csv_file_path, index=False)

# Confirm that the file has been saved
print(f"CSV file saved at: {csv_file_path}")

# Plot the sensor data
plt.figure(figsize=(14, 6))

# Plot Temperature Data
plt.subplot(1, 2, 1)
plt.plot(t_d, label="Temperature")
plt.axhline(y=temp_avg, color='r', linestyle='--', label='Average')
plt.title("Temperature Sensor Data")
plt.xlabel("Time")
plt.ylabel("Temperature (C)")
plt.legend()

# Plot Pressure Data
plt.subplot(1, 2, 2)
plt.plot(p_d, label="Pressure")
plt.axhline(y=press_avg, color='r', linestyle='--', label='Average')
plt.title("Pressure Sensor Data")
plt.xlabel("Time")
plt.ylabel("Pressure (hPa)")
plt.legend()

# Save the plots as a PNG file
plt.savefig('sensor_data_plot.png')

# Display the plots
plt.show()

# Read CSV file back into a DataFrame
df_read = pd.read_csv(csv_file_path)

# Display first few rows of the DataFrame
print("Data read from CSV file: ")
print(df_read.head())

# Calculate additional statistics (optional)
```

```
temp_std = np.std(df_read['Temperature'])  # Standard deviation of
temperature
press_std = np.std(df_read['Pressure'])  # Standard deviation of pressure

# Display additional statistics
print(f"Additional Stats:")
print(f"Temperature Standard Deviation: {temp_std:.2f}")
print(f"Pressure Standard Deviation: {press_std:.2f}")
```

Output:




```
Average temperature: 19.48, Peak: 29.26
Average pressure: 1000.45, Peak: 1054.40
CSV file saved at: D:\sem6\indLab\feb-ict\sensor_data.csv
Data read from CSV file:
   Temperature    Pressure
0    22.483571  971.692585
1    19.308678  991.587094
2    23.238443  993.145710
3    27.615149  983.954455
4    18.829233  996.774286
Additional Stats:
Temperature Standard Deviation: 4.52
Pressure Standard Deviation: 18.98
```

Conclusion:

In this experiment, we successfully collected and stored temperature and pressure sensor data in a CSV file. The data was visualized using graphs, which helped identify trends and averages for both parameters over time. We calculated basic statistics such as the mean, peak, and standard deviation, which has been displayed in the output. This process demonstrated how to simulate, analyze, visualize, and store sensor data into a CSV file.