**To continue building your serverless IoT data processing solution using IBM Cloud Functions, real-time data processing, automation, and data storage, follow these steps:**

**Step 1: Implement Real-time Data Processing with IBM Cloud Functions**

1.Create Additional IBM Cloud Functions:

Depending on your specific use case, create additional IBM Cloud Functions for real-time data processing. For example, you can have one function to process temperature data, another for motion sensor data, and so on.

2. Real-time Data Processing Logic:

In each function, implement the logic to process the incoming data in real-time. Use the event triggers provided by IBM Cloud Functions to process data as soon as it arrives. You can use Python or Node.js as the programming language for your functions.

Here's an example of a real-time data processing function in Python:

**def main(params):**

**# Extract data from the event object**

**data = params.get("data")**

**# Implement your custom real-time processing logic here**

**# For example, you can check the temperature and take action based on predefined thresholds**

**# Trigger actions or store results as needed**

**# Implement actions or further storage logic here**

**return {"message": "Real-time data processing completed."}**

**Step 2: Develop Automation Routines**

1. Define Automation Rules: Define the automation routines you want to implement in your smart home. For example, you can create automation rules to adjust thermostat settings, turn on lights, or trigger security alerts based on sensor data.

2. Automation Functions: Create IBM Cloud Functions for each automation routine. These functions should be triggered by specific events or conditions in your real-time data processing functions.

3. Automation Logic:

In each automation function, implement the logic for the desired routine. Use the data processed in the real-time processing functions to make decisions and take actions. For example, you can adjust the thermostat temperature based on temperature data or turn on security cameras when motion is detected.

**Step 3: Store Processed Data in IBM Cloud Object Storage**

1. **Data Storage Logic:**

In your real-time data processing functions, add code to store the processed data in IBM Cloud Object Storage. You can use the IBM Cloud Object Storage client to do this.

**Here's an example of how to store data in IBM Cloud Object Storage:**

**import ibm\_boto3**

**from ibm\_botocore.client import Config**

**cos\_credentials = {**

**"apikey": " vPc\_jSjpKkr4ENC-y8iFlg0jwtQcVfdKFvCQmsu8SXSF",**

**"iam\_service\_endpoint": " crn:v1:bluemix:public:iam-identity::a/9e95d6860c864c09998b14bd52f93b68::serviceid:ServiceId-fe940556-dddb-499a-b494-af49781d1da7",**

**"resource\_instance\_id": " crn:v1:bluemix:public:cloud-object-storage:global:a/9e95d6860c864c09998b14bd52f93b68:b00228fc-6641-4a3c-92ea-e4e78cb4364e::",**

**"endpoint": "https://control.cloud-object-storage.cloud.ibm.com/v2/endpoints",**

**}**

**cos\_client = ibm\_boto3.client(**

**"s3",**

**ibm\_api\_key\_id=cos\_credentials["apikey"],**

**ibm\_service\_instance\_id=cos\_credentials["resource\_instance\_id"],**

**ibm\_auth\_endpoint=cos\_credentials["iam\_service\_endpoint"],**

**config=Config(signature\_version="oauth"),**

**endpoint\_url=cos\_credentials["endpoint"],**

**)**

**def store\_data\_in\_object\_storage(data):**

**bucket\_name = "crn:v1:bluemix:public:cloud-object-storage:global:a/9e95d6860c864c09998b14bd52f93b68:b00228fc-6641-4a3c-92ea-e4e78cb4364e:bucket:ibmnaanmudhalvan"**

**object\_name = "processed\_data.json"**

**cos\_client.put\_object(Bucket=bucket\_name, Key=object\_name, Body=data)**

**store\_data\_in\_object\_storage(processed\_data)**

2. Continuous Storage:

Ensure that data is consistently and securely stored in your designated IBM Cloud Object Storage bucket as it is processed and actioned upon.

**By following these steps, you can process data in real-time using IBM Cloud Functions, trigger automated routines, and store processed data in IBM Cloud Object Storage for future analysis.**