SERVERLESS IOT DATA PROCESSING

Phase 5:Project Documentation And Submission

Team Members:

1.Navanitha.M-732721104037

2.Sowmiya.S-732721104047

3.Srija.R-732721104048

4.Sugeetha.S-732721104049

5.Varshini.N-732721104051

6.Vihashini.E-732721104058

OBJECTIVES:

Outline the primary goals and objectives of the IOT data processing project, explaining what you aimed to achieve from a technical and functional perspective.

The primary objectives were as follows:

☐ **Smart Home Transformation:** The project aimed to convert a traditional home into a modern, smart living space, leveraging IoT devices and cloud-based services.
☐ **Real-Time Data Analysis:** Enable the system to process data from various sensors and devices in real-time to make instant decisions and take automated actions.
☐ **Energy Efficiency and Home Security:** Implement automation routines that optimize energy consumption and enhance home security by detecting and responding to security threats.
☐ **Data Storage and Analysis:** Store data securely in IBM Cloud Object Storage and utilize data analysis tools for insights into energy usage and security events.
☐ **User Convenience:** Create a user-friendly smart home experience that can be customized by homeowners to meet their specific needs and preference.

DESIGN THINKING:

Applying design thinking to the development of a serverless IOT data processing system involves understanding and addressing the needs of both the end-users (such as administrators and data analysts) and the specific requirements of the IOT application. Here's how design thinking can be adapted to create an effective serverless IOT data processing solution:

1. Empathize:

Understand the key stakeholders and their needs, including IOT device operators, data analysts, and system administrators.

Conduct interviews and gather insights into their pain points, goals, and expectations for the IOT data processing system.

2. Define:

Clearly define the problem or challenge you aim to solve. For instance, this could be improving real-time data processing, enhancing scalability, or reducing operational costs.

Create a problem statement that encapsulates the needs of both end-users and the IOT application.

3. Ideate:

Brainstorm and generate creative ideas for how to design a serverless IOT data processing system that meets the defined needs and challenges.

Encourage collaboration among cross-functional teams to explore different architectural and functional possibilities.

4. Prototype:

Create low-fidelity prototypes or mock-ups of the proposed serverless architecture and data processing workflow. This might involve designing serverless functions and outlining how they interact.

Use visualization tools to help stakeholders understand the proposed solution.

5. Test:

Collect feedback from end-users and stakeholders by presenting the prototypes to them.

Analyze their reactions and identify areas for improvement or refinement in the serverless IoT data processing design.

6. Iterate:

Based on user feedback, iterate on the design of the serverless architecture, revising serverless function logic, and considering adjustments to improve its effectiveness.

DEVELOPMENT PHASES:

Developing a serverless IOT data processing system involves several phases to ensure a well-structured and effective solution. These phases are sequential but may overlap, and they can be iterative to accommodate changing requirements. Here are the key development phases for serverless IOT data processing:

1.Requirement Analysis and Planning:

Define the project's objectives and scope.

Identify the IOT data sources, data types, and expected data volume.

Understand the specific processing and analytical requirements.

Define security and compliance considerations.

Plan for source allocation and cost management.

2. Architecture Design:

Design the serverless architecture that will support the IOT data processing, considering the use of cloud services like AWS Lambda, Azure Functions, or Google Cloud Functions.

Define the event triggers and data flow within the system.

Determine the data storage and database solutions.

Address scalability, fault tolerance, and redundancy requirements.

Consider integration with IOT devices, edge computing,

and external services.

3.Data Ingestion:

Implement data ingestion mechanisms to capture data from IOT devices and sensors.

Ensure data validation, transformation, and encryption as necessary.

Configure event-driven triggers to initiate data processing when new data arrives.

4. Data Processing and Transformation:

Develop serverless functions to process and transform raw IOT data into meaningful insights or prepare it for storage.

Implement real-time analytics, filtering, aggregation, or other data processing tasks.

Optimize for low-latency processing, if required.

5.Data Storage:

Choose appropriate data storage solutions for both real-time and historical data.

Set up databases, data lakes, or cloud storage systems for efficient data retention and retrieval.

Ensure data durability and redundancy.

6. Security and Access Control:

Implement robust security measures, including data encryption, authentication, and authorization.

Set access controls to protect sensitive IOT data and system resources.

Monitor for security breaches and anomalies.

7. Scalability and Resource Optimization:

Design for automatic scalability, taking advantage of the serverless nature to handle changing workloads.

Optimize resource usage, including efficient function code and memory allocation.

8. Testing and Quality Assurance:

Develop comprehensive testing strategies, including unit tests, integration tests, and end-to-end tests.

Test the system under different data scenarios and workloads.

Validate the system's performance, security, and reliability.

9. Monitoring and Logging:

Implement monitoring and logging solutions to track the performance of the serverless IOT data processing system.

Set up alerts for system health and anomalies.

Use data analytics to gain insights into system behavior and performance.

10. Deployment and Maintenance:

Deploy the serverless IoT data processing system to a production environment.

Develop maintenance and update procedures.

Continuously monitor the system and apply patches or updates as needed.

11. Documentation and Training:

Create documentation for system architecture, data flows, and best practices.

Provide training for administrators, operators, and other stakeholders.

12.Optimization and Iteration:

Continuously optimize the system for improved performance, cost-efficiency, and security.

Iterate based on user feedback and evolving IOT requirements.

These development phases ensure that your serverless IOT data processing system is well-designed, secure, scalable, and capable of meeting the demands of your IOT application while considering the unique challenges of processing IOT data in a serverless environment.

SMART HOME SETUP:

Setting up a smart home system for serverless IOT data processing involves integrating various IOT devices and sensors, leveraging serverless computing resources, and implementing data processing workflows. Here's a high-level overview of how to create a serverless IOT data processing setup for a smart home:

.

1. Define Your IOT Data Sources: Identify the IOT devices generating data. These could be sensors, cameras, or any other connected devices.

.

. **2. Choose a Cloud Provider:** Select a cloud provider that offers a robust serverless platform. Some popular options include AWS Lambda, Google Cloud Functions, or Microsoft Azure Functions.

.

. **3. Set Up IOT Hub:** Create an IOT hub in your cloud provider's ecosystem. This hub will act as a centralized platform for receiving and processing data from your IOT devices. For example, you might use AWS IOT Core or Azure IOT Hub.

.

. **4. Configure IOT Device Communication:** Configure your IOT devices to communicate with the IOT hub securely. This usually involves setting up authentication, encryption, and secure communication protocols like MQTT or HTTPS.

.

. **5. Data Ingestion:** Configure your IOT hub to ingest the data from the IOT devices. This can often be done using the SDKs provided by the cloud provider. Ensure that the data ingestion process is reliable and can handle large volumes of data.

.

. **6. Data Processing using Serverless Functions:** Create serverless functions that process the incoming IOT data. You can use functions provided by your cloud provider, such as AWS Lambda or Azure Functions. These functions can perform various tasks like data

transformation, filtering, aggregation, or triggering other services based on the data received.

7. Data Storage: Choose a suitable data storage solution to store the processed data. Options may include cloud-based databases like Amazon DynamoDB, Google Cloud Bigtable, or Azure Cosmos DB, depending on your cloud provider.

8.Data Analysis and Visualization: Implement tools or services to analyze and visualize the processed data. You can use services like AWS QuickSight, Google Data Studio, or Microsoft Power BI to gain insights from your data.

9. Monitoring and Logging: Implement monitoring and logging solutions to track the performance and behavior of your IOT data processing pipeline. Use tools like AWS CloudWatch, Google Cloud Monitoring, or Azure Monitor to monitor the health of your serverless functions and IOT data processing system.

10. Security and Compliance: Implement robust security measures to protect your IOT data and ensure compliance with data protection regulations. This includes encryption, access control, and regular security audits.

11. Testing and Optimization: Test your IOT data processing system thoroughly to ensure its reliability and efficiency. Optimize the system based on the performance metrics and feedback gathered during the testing phase.

By following these steps, you can create a robust and scalable serverless IOT data processing setup at home.

DEVICE INTEGRATION:

Integrated IOT Devices:

Smart Climate Control Hub: This device serves as the central hub for managing temperature and humidity in your smart home.

Advanced Motion Sensors: These sensors are strategically placed throughout your home to detect motion and presence.

Intelligent Lighting Controllers: These controllers offer precise control over lighting to create the desired ambiance and enhance energy efficiency.

Smart Surveillance Cameras: These cameras provide real-time video feeds for enhanced security and monitoring.

Voice-Activated Assistant: This device allows hands-free control of your smart home through voice commands.These integrated IOT devices collectively contribute to the functionality, energy efficiency, and security of your smart living space, enhancing your overall home automation experience.

Communication Protocols:

1. MQTT (Message Queuing Telemetry Transport): MQTT is used for efficient and real-time communication between the IOT devices and the cloud functions. It plays a critical role in data transmission, particularly for data related to motion ,temperature, and lighting control.

2. **HTTP (Hypertext Transfer Protocol):** HTTP is employed for web-based communication between certain IOT devices, such as smart surveillance cameras, and the central processing unit. It allows users to access real-time video feeds and camera controls via web interfaces.

3. Custom Protocols: In some cases, custom communication protocols are used to ensure seamless integration and data exchange with specific devices. These custom protocols are designed to meet the unique requirements of each device.These communication

protocols enable data to flow from the IOT devices to the central processing unit, facilitating real-time data processing, automation, and data storage and analysis within your smart home system.

Data Collection Process for Smart Home IOT Devices:

1. Data Generation:
☐ The process begins with the various IOT devices installed in your smart home, such as thermostats, motion sensors, cameras, and intelligent lighting controllers. These devices continuously generate data as they operate.

2. Data Transmission:
☐ The data generated by these devices needs to be transmitted to a central processing unit for analysis and automation. To achieve this, communication protocols play a vital role. In your smart home project, MQTT (Message Queuing Telemetry Transport) is used for efficient and real-time data transfer.MQTT is a lightweight and efficient protocol that is well-suited for IOT applications. It enables devices to publish and subscribe to topics, facilitating seamless communication.

3. Data Ingestion:
Once the data reaches the central processing unit, it undergoes the data ingestion phase. This phase ensures that all incoming data is received and processed without loss, even during high traffic or peak usage times. Data may be temporarily stored in a buffer or queue during this phase.

4. Data Preprocessing:
☐ Data, especially when coming from multiple sources, may not always be in a uniform format. To ensure consistency and accuracy in processing, data preprocessing is performed. This step involves cleaning, formatting, and validating the incoming data Preprocessing can include tasks such as converting data

to a standardized format,checking for data integrity, and performing any necessary transformations.

5. Real-Time Data Processing:

☐ With data in a consistent and validated format, it is then sent for real- time processing. IBM Cloud Functions are at the heart of this phase, serving as serverless actions that are triggered by incoming data.These functions analyze the data in real time, making decisions and generating responses based on predefined logic and automation routines. For example, if motion is detected, the system can trigger lighting adjustments, security alerts, or other predefined actions.

6. Automation and Storage:

☐ Simultaneously, the processed data is not only used for real-time decision-making but is also stored for future analysis. In your project, IBM Cloud Object Storage is employed to securely store data.The data storage component is designed to accommodate a vast amount of data generated by your IOT devices. It provides the foundation for gaining valuable insights into energy consumption, security events, and usage patterns within your smart home.

The data collection process is essential to the functionality of your smart home. It ensures that data generated by various IOT devices is efficiently collected, processed in real time, and used to enhance energy efficiency, home security, and overall user experience. The combination of MQTT for data transmission, preprocessing for data consistency, IBM Cloud Functions for real-time processing, and IBM Cloud Object Storage for data storage and analysis creates a robust and efficient system architecture.

TECHNICAL IMPLEMENTATION:

```
from pymongo import MongoClient
from date time import datetime
import json
import time
def handle(req):
client = Mongo Client("mongodb://10.10.10.10:27017")
db = client.test
json_req = json.loads(req)
db_entry = {"Humidity": json_req["Humidity"],
"Temperature_farenheit":json_req["
Temperature_farenheit"],
"Temperature_celsius": json_req["
Temperature_celsius"],
"Latitude": json_req["Latitude"],
"Longitude": json_req["Longitude"],
"Time": datetime.now()}
db.sensordata.insert(db_entry)
return {"status Code": "200"}
```

```
import requests
import json
def handle(req):
result = {"found": False}
json_req = json.loads(req)
r = requests.get(json_req["url"])
if json_req["term"] in r.text:
result = {"found":
#Base Image
FROM ubuntu:16.04
MAINTAINER Manoj Kumar
LABEL version="1.0"
# prerquisites
```

```
RUN apt-get update && apt-get install -y apt-utils && apt-
get
install -y curl
ENV DEBIAN_FRONTEND=noninteractive
COPY application.apk /go/src/project/application.apk
WORKDIR WORKDIR /go/src/project/
#Java8 installation
ARG JAVA_URL=http://download.oracle.com/otn-
pub/java/jdk/8u131-
b11/d54c1d3a095b4ff2b6607d096fa80163/jdk-8u131-
linux-x64.tar.
gz
ARG JAVA_DIR=/usr/local/java
ARG JAVA_VER=jdk1.8.0_131
RUN mkdir -p $JAVA_DIR && cd $JAVA_DIR \
&& (curl -Lsb "oracle license=accept- securebackup-
cookie"
$JAVA_URL | tar zx) \
&& update-alternatives --install "/usr/bin/java" "java" "
$JAVA_DIR/$JAVA_VER/bin/java" 1 \
&& update-alternatives --install "/usr/bin/javac" "javac" "
$JAVA_DIR/$JAVA_VER/bin/javac" 1
EXPOSE 4723
COPY --from=build /bin/project /bin/project
ENTRYPOINT ["/bin/project"]
CMD ["--help"]
```

Real-Time Data Processing:

IBM Cloud Functions in Real-Time Data Processing:

IBM Cloud Functions is a fundamental component in the real-time data processing of your smart home project. It plays a pivotal role in enabling immediate analysis and

decision-making based on the data collected from your IOT devices.

Role of IBM Cloud Functions:

1. **Event-Driven Processing:** IBM Cloud Functions follows an event-driven architecture. This means that actions are executed in response to specific events, such as data being published by your IOT devices. Whenever an event occurs, it triggers an associated action in the form of a function.

2. **Serverless Framework:** IBM Cloud Functions operates on a serverless framework, which means you need to manage the infrastructure, servers, or scaling.You only pay for the actual compute time your function consume, making it cost-effective.

3. **Action Execution:** Actions, which are essentially code functions, are executed automatically when triggered by events. For instance, when motion is detected by your smart security cameras, it triggers a corresponding action in IBM Cloud Functions to respond to this event.

4. Scalability: IBM Cloud Functions can easily scale to handle a high volume of events simultaneously. This scalability is crucial for your smart home, where multiple IOT devices generate data continuously.

Data Processing Logic in IBM Cloud Functions:

The data processing logic implemented in IBM Cloud Functions is central to your smart home real-time data processing. It involves real-time analysis and decision making based on the data collected from your IOT devices. Heres an overview of the data processing logic:

1. Data Collection:

☐ Data is continuously collected from your IOT devices, including thermostats, motion sensors,and cameras. This data encompasses a variety of information such as temperature readings,motion detection events, and video feeds.

2. Event Detection:

IBM Cloud Functions are configured to detect specific events based on the incoming data.For instance, the system can detect a rise in temperature beyond a predefined threshold, the detection of motion by sensors, or unusual activity in the surveillance camera footage.

3. Real-Time Analysis:

☐ Once an event is detected, the data processing logic involves real-time analysis. For example,if a motion sensor detects movement in your home, the logic evaluates the event context,such as the location and time of the event.

4. Decision-Making:

☐ Decision-making is a critical aspect of the data processing logic. Based on the real-time analysis, the system decides on the appropriate action to take. In the case of motion detection, this may include activating security protocols, like sending alerts and turning on lights.

5. Action Execution:

☐ The chosen action is executed automatically. This can involve sending alerts to homeowners mobile devices, adjusting thermostat settings to save energy, or activating smart lighting to enhance security. These actions occur seamlessly and without manual intervention.

6. Response Generation:

☐ In addition to executing actions, the data processing logic generates responses. For example,the system can log the event, capture images or video clips for evidence, or store data for future analysis.

7. Customization and Configuration:
☐ The data processing logic is highly customizable. You can configure actions, event detection criteria, and decision rules to align with your specific requirements and preferences. This customization allows you to tailor your smart homes automation routines for energy efficiency and home security.

8. Continuous Monitoring:
☐ IBM Cloud Functions continually monitor the data stream from your IOT devices, repeating the event detection, analysis, and decision-making cycle as new data arrives. This ensures that your smart home operates efficiently and securely.

In summary, the data processing logic in IBM Cloud Functions is designed to provide real-time insights and make immediate decisions based on the data collected from your smart devices. It enhances the functionality of your smart home, ensuring that it responds intelligently to events, thereby increasing convenience, energy savings, and security.

Automation:

Automation Routines for Energy Efficiency and Home Security:

Automation routines are a fundamental part of your smart living space, optimizing energy usage and enhancing home security. Here, we elaborate on the automation routines, detailing the triggers and actions designed to achieve energy efficiency and bolster security within your smart home:

1. Temperature Control:
☐ Trigger: Automation routines for temperature control are typically triggered by temperature readings from your thermostats.

☐ Action: Based on the temperature data, the system takes actions such as adjusting the thermostat settings, regulating heating and cooling systems, and optimizing HVAC usage. For example, if the temperature rises too high, the system may activate the air conditioning,contributing to energy savings.

2. Lighting Control:
☐ Trigger: Triggers for lighting control routines include occupancy or ambient light levels.
☐ Action: The system automatically turns lights on or off, adjusting brightness or color based on conditions. For instance, when motion is detected in a room, the lights can illuminate, enhancing security and providing convenience. Conversely, when sufficient natural light is available, lights may dim or turn off to conserve energy.

3. Security Alerts:
☐ Trigger: Motion sensors and surveillance cameras are the primary triggers for security alerts.
☐ Action: When motion is detected by cameras or motion sensors, the system sends immediate security alerts to notify homeowners of potential intrusions. These alerts may be in the form of mobile notifications, email alerts, or audible alarms, providing a sense of security and peace of mind.

4. Notification Management:
☐ Trigger: Triggers for notifications include various important events within your smart home.
☐ Action: The system informs users about significant events, such as package deliveries or visitors at the door. Notifications can be sent via mobile apps or other communication channels, ensuring that homeowners are always aware of what happening in their smart living space.

5. Scheduled Tasks:
☐ Trigger: Scheduled tasks are triggered at specific times or under predefined conditions.

☐ Action: These routines execute pre-defined tasks at designated times, enhancing energy efficiency. For example, you can set the system to automatically dim lights in the evening to create a cozy ambiance while saving energy.

6. User Customization:

☐ Trigger: User interaction and configuration initiate customization of automation rules.

Action: Users can define their own triggers, conditions, and actions to meet their specific needs and preferences. The system allows homeowners to personalize their smart living experience, ensuring that it adapts to individual requirements.

Automation routines significantly enhance the convenience, efficiency, and security of your smart home. By employing these triggers and actions, your smart home operates seamlessly, optimizing energy consumption, and providing a heightened sense of security. Whether you want to create a cozy atmosphere or ensure the safety of your home, automation routines make it all possible.

Automation Logic:

Automation logic serves as the decision-making brain behind the automation routines in your smart home. This logic guides the system in responding to various triggers and deciding which actions to take. Here, we describe the core elements of the automation logic governing your energy efficiency and home security automation routines:

1. Conditional Statements:

☐ Conditional statements play a pivotal role in the automation logic. They are used to evaluate the data received from triggers and determine if specific conditions are met. For instance, in temperature control automation, conditional statements assess whether the current temperature is outside the desired range. If so, the system takes action to adjust the thermostat settings.

2. Event Handling:

☐ Events are a crucial part of automation logic. The system identifies events triggered by IOT devices, such as motion detected by sensors or specific time-based events, to initiate the automation routines. Depending on the type of event, different actions are taken.

3. Decision Trees:

☐ Decision trees are employed to map out the decision-making process. They provide a hierarchical structure where the system evaluates conditions and events and makes decisions accordingly. For example, in security alerts, the decision tree determines whether a detected motion event is a potential security threat or a harmless event, influencing the intensity of the response.

4. User-Defined Rules:

☐ User customization is an essential aspect of the automation logic. Users can define their own rules and conditions for automation routines. This allows homeowners to tailor the behavior of the system to their unique preferences. The logic is designed to adapt to user-defined rules and execute actions accordingly.

5. Data Analysis:

☐ Data analysis is integrated into the automation logic to provide insights for making informed decisions. For instance, in the context of lighting control, data analysis can take into account the amount of ambient light and user preferences to decide whether to turn lights on or off.

6. Event Prioritization:

☐ Automation logic often includes event prioritization. This ensures that more critical events,such as security alerts, take precedence over less urgent actions. In cases where multiple events are detected simultaneously, the logic prioritizes actions to maximize security and efficiency.

7. Scheduling:

☐ Scheduling is a fundamental component of automation logic, particularly for tasks with time-based triggers. The logic is designed to execute tasks at specific times or on certain days,allowing for functions like lighting automation and scheduled routines.

8. Feedback Loops:
☐ Feedback loops are established within the logic to continuously assess the impact of actions and make necessary adjustments. If, for instance, an action results in excessive energy consumption, the system may adjust settings to optimize efficiency.The automation logic in your smart home is both flexible and adaptable. It integrates conditional statements, event handling, decision trees, and user-defined rules to make real-time decisions based on the data it receives. This approach ensures that your smart living space responds intelligently to various triggers, providing energy efficiency and security while accommodating user preferences.

Data Storage and Analysis

IBM Cloud Object Storage:

IBM Cloud Object Storage plays a pivotal role in your smart home project, serving as a robust and scalable data repository for the vast amount of information generated by your IOT devices. Here, we provide details on how data is stored in IBM Cloud Object Storage, including the storage structure and organization:

Storage Structure:
☐ The data storage structure in IBM Cloud Object Storage is designed for efficient organization and retrieval of smart home data. It is based on the principles of object storage, which offers flexibility and scalability, making it well-suited for IOT applications.

Containers:
☐ Data is organized into containers, which serve as logical compartments for categorizing different types of data. In

the context of your smart home, containers are created to accommodate specific data categories, such as temperature readings, motion detection events, security camera footage, and other data sources.

Objects:
☐ Within each container, data is stored in the form of objects. Objects are individual data units, each containing information related to a particular event or data type. For instance, an object might store temperature data for a specific time period or a video clip from a security camera.

Metadata Tagging:
☐ Each object within IBM Cloud Object Storage is enriched with metadata. This metadata includes essential information about the data source, data type, timestamp, and any other relevant contextual details. Metadata tagging is essential for data discovery, retrieval, and efficient analysis. It allows users and applications to search for specific data objects quickly.

Data Labeling:
☐ Data objects are labeled with meaningful descriptors to simplify identification and categorization. For example, security camera footage may be labeled with location, date,and time information, making it easy for users to find specific video clips.

Hierarchical Organization:
☐ Data is organized in a hierarchical manner using directories and sub directories, mirroring the logical structure of the smart home ecosystem. This hierarchy simplifies data navigation and management. For instance, data from different IOT devices is logically separated within the storage structure, making it straightforward to locate specific data streams.

Data Segmentation:
☐ Data segmentation is used to ensure that diverse data types are kept separate from one another. For example,

temperature data is stored separately from motion detection events.This segmentation not only enhances data organization but also facilitates selective access control.

Access Control:

☐ Stringent access control and security measures are implemented to safeguard the stored data from unauthorized access. Role-based access control (RBAC) is commonly used to manage and restrict data access, ensuring that only authorized personnel and applications can interact with specific data objects.

Effective data storage structure, organization, and tagging are essential for unlocking the full potential of your smart home. This structured approach to data storage enables efficient retrieval, analysis, and visualization, ultimately providing valuable insights into energy consumption,security events, and usage patterns, while ensuring that the data remains secure and organized.

CONCLUSION:

In conclusion, this project has transformed a conventional home into a smart living space, leveraging the power of IBM Cloud Functions for IOT data processing. The journey through each phase of development has been an exciting exploration of technology and innovation.

We are immensely grateful for the opportunity to delve into this project, and we extend our appreciation to all team members, stakeholders, and supporters who have contributed to its success.

As we wrap up this documentation, it important to highlight the potential for future enhancements. The world of IOT and smart homes is dynamic and ever-evolving. As technology advances, we envision expanding our smart living space to incorporate even more IOT devices and services, further enhancing energy efficiency, security, and user convenience.