UNIT- III

DEVELOPING INTERNET OF THINGS

"An IoT system is successful only when devices, data, and applications work together seamlessly."

— Arshdeep Bahga & Vijay Madisetti

SREENIVASA INSTITUTE OF TECHNOLOGY AND MANAGEMENT STUDIES.
(AUTONOMOUS)
MCA DEPARTMENT

LECTURE NOTES

INTERNET OF THINGS

Unit-III

IoT platforms Design Methodology: Introduction-IoT design Methodology-Step 1:Purpose and Requirements specification, Step 2:Process specification, step 3:Domain model Specification, step 4:Information Model Specification ,step 5:Service Specifications, step 6:IoT Level Specification, Step 7:Functional view specification, step 8:Operational view specification, step 9:Device and Component Integration, Step 10:Application Development-Case study on IoT system for weather Monitoring-Motivation for using Python-IoT systems-Logical Design using Python-Introduction- Installing Python-Python data Types & Data Structures- Numbers, Strings, Lists, Tuples, Dictionaries, Type Conversions- Python Packages of interest for IoT- JSON, XML, HTTP Lib & URL Lib ,SMTP Lib.

Unit Overview

This unit introduces the IoT Design Methodology, providing a systematic approach to designing and developing IoT applications. It covers the complete design process from requirement specification to application development, including domain, information, functional, and operational views. The unit also presents a real-time weather monitoring case study and introduces Python programming for IoT system development. Students will learn Python fundamentals, data types, data structures, and essential Python libraries such as JSON, XML, HTTP, URL, and SMTP for implementing IoT applications.

Objectives of the Unit

1. Understand the IoT Design Methodology and its design steps.
2. Learn how to specify requirements, processes, domain models, and information models for IoT systems.
3. Understand service, functional, operational, and IoT level specifications.
4. Study device and component integration in IoT applications.
5. Develop IoT applications using a structured design approach.
6. Analyze a real-world IoT weather monitoring case study.
7. Learn the fundamentals of Python programming for IoT systems.
8. Use Python data types, data structures, and libraries to build IoT applications.

Learning Outcomes

After completing this unit, students will be able to:

1. Explain the steps involved in the IoT Design Methodology.
2. Develop requirement, process, domain, and information models for IoT applications.

3. Design IoT systems using functional and operational specifications.
4. Integrate IoT devices and components into complete IoT solutions.
5. Analyze real-time IoT applications through case studies.
6. Apply Python programming concepts to IoT system development.
7. Use Python libraries such as JSON, XML, HTTP, URL, and SMTP in IoT applications.
8. Design and implement simple IoT applications using Python.

Importance of Studying this Unit

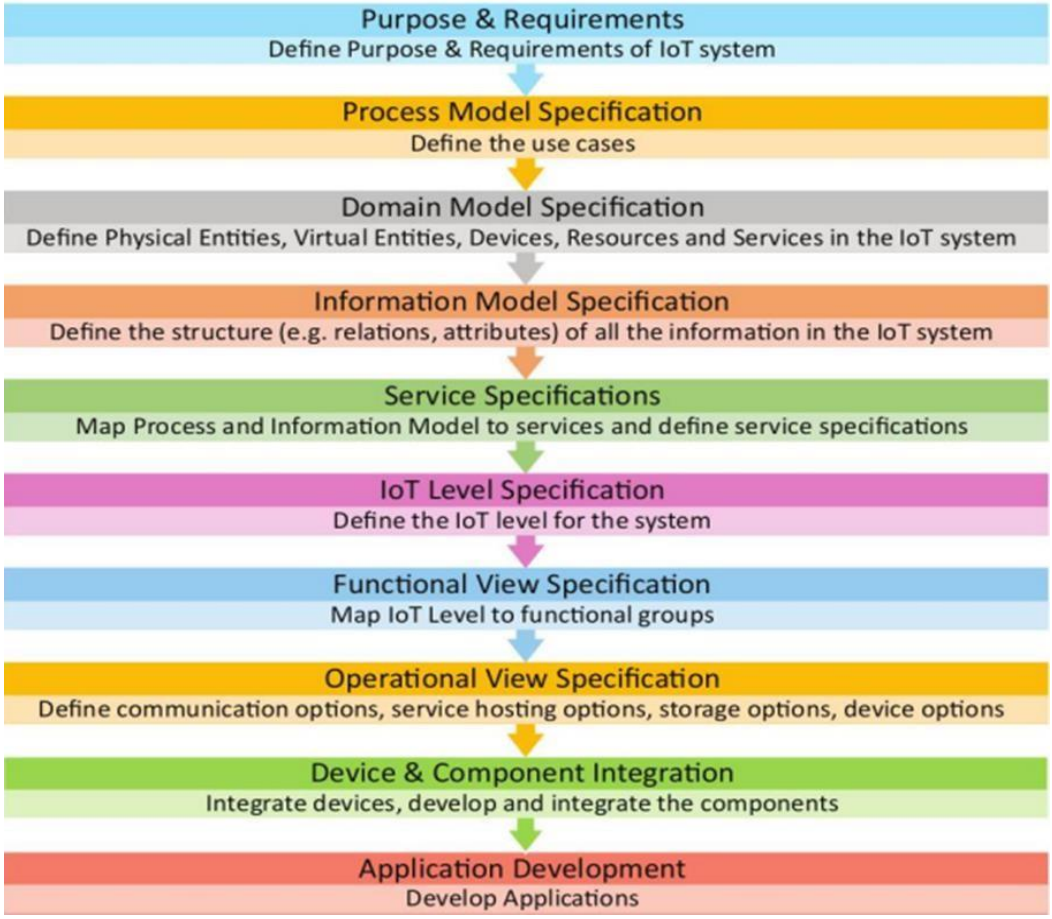
- Provides a structured methodology for designing IoT systems.
- Develops skills in planning and implementing complete IoT solutions.
- Introduces real-world IoT application development through case studies.
- Enhances problem-solving using systematic design techniques.
- Builds programming skills using Python for IoT development.
- Familiarizes students with commonly used IoT communication libraries.
- Prepares learners for industrial IoT application development and deployment.
- Bridges the gap between IoT system design and practical implementation.

IOT Platform Design Methodology

Designing IoT systems can be a complex and challenging task as these systems involve interactions between various components. A wide range of choices are available for each component. IoT designers often tend to design the system keeping specific products in mind.

We will look at a generic design methodology which is independent of specific product, service or programming language. IoT systems designed with this methodology will have reduced design time, testing time, maintenance time, complexity and better interoperability.

The steps involved in the designing of an IoT system or application can be summarized as shown in the below figure:



Step 1: Purpose & Requirements Specification

First step is to define the purpose and requirements of the system. In this step, the system purpose, behavior and requirements are captured. Requirements can be:

- Data collection requirements
- Data analysis requirements
- System management requirements
- Security requirements
- User interface requirements

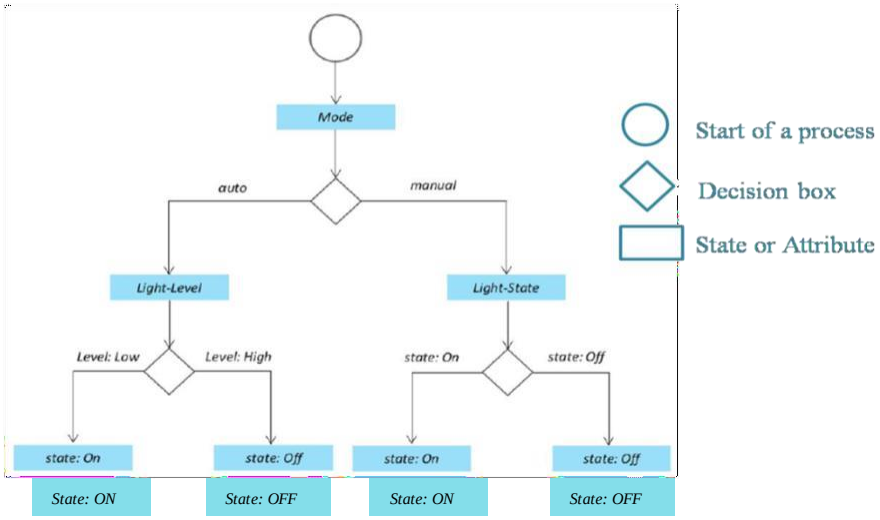
For home automation system the purpose and requirements specification is as follows:

Purpose	A home automation system that allows controlling the lights remotely using a web application
Behavior	Home automation system should support two modes: auto and manual **Auto:** System measures the light level in the room and switches on the light

	when it is dark **Manual:** Allows remotely switching lights on and off
System Management	System should provide remote monitoring and control functions
Data Analysis	System should perform local analysis of the data
Application Deployment	Application should be deployed locally, but should be accessible remotely
Security	Should provide basic security like user authentication

Step 2: Process Specification

This step, formally described the **Process specifications** of the IoT system. In this step, the use cases of the IOT system are formally described based on the derived from the purpose and requirement specification. In the following process diagram, the circle denotes the start of a process, the diamond denotes a decision box and the rectangle denotes a state or attribute (design flow chart).

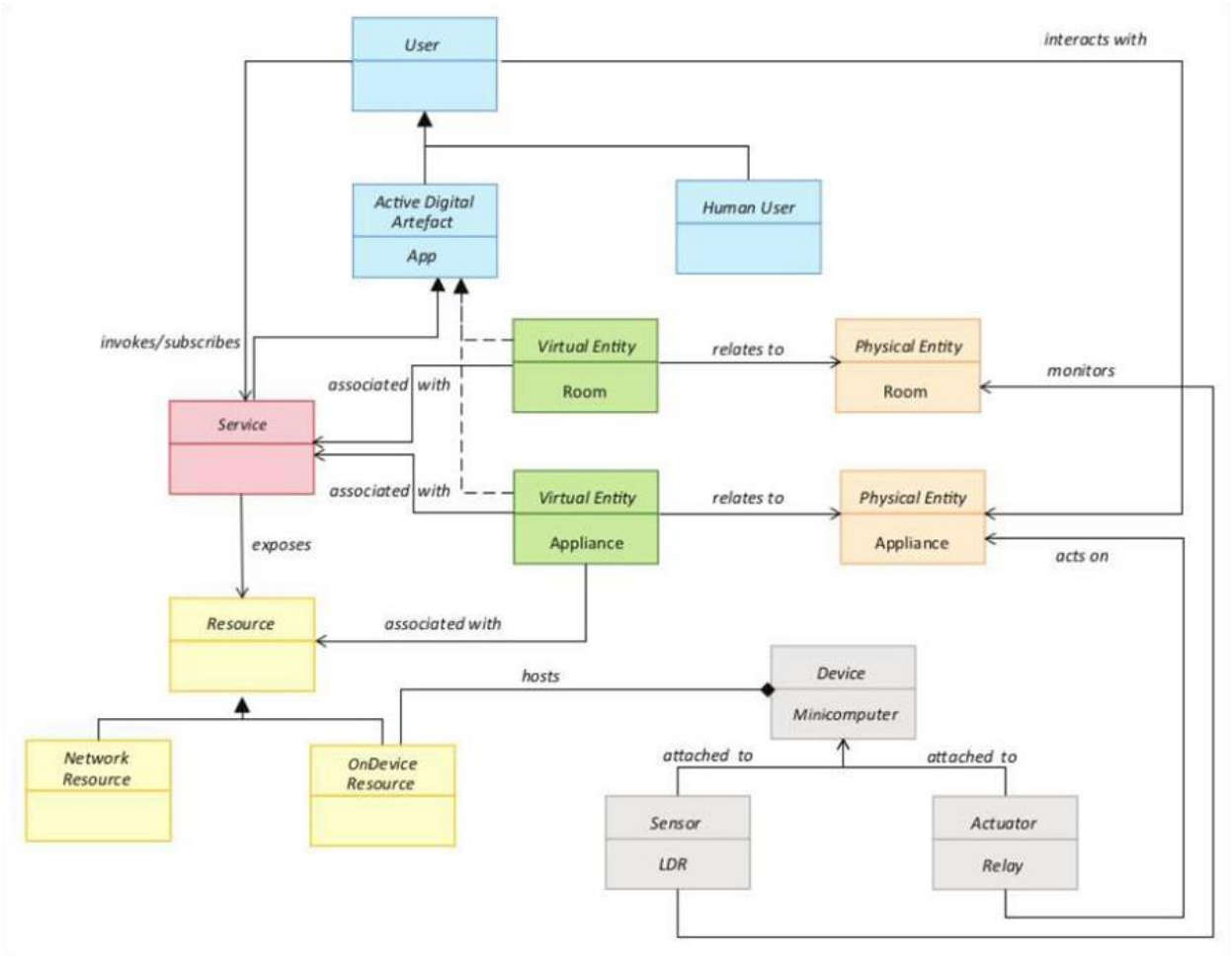


Step 3: Domain Model Specification

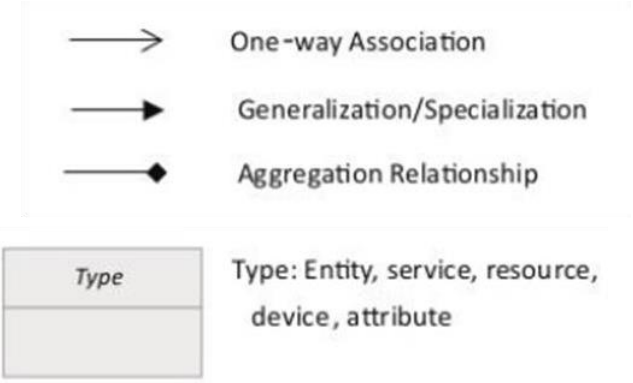
The domain model describes the main concepts, entities and objects in the domain of the IoT system to be designed. Domain model defines the attributes of the objects and relationships between objects. The domain model provides an abstract representation of the concepts, objects and entities in the IOT

domain, independent of any specific technology or platform.

Using domain model, system designers can get an understanding of the IoT domain for which the system is to be designed. The entities, objects and concepts defined in the domain model of home automation system include the following:



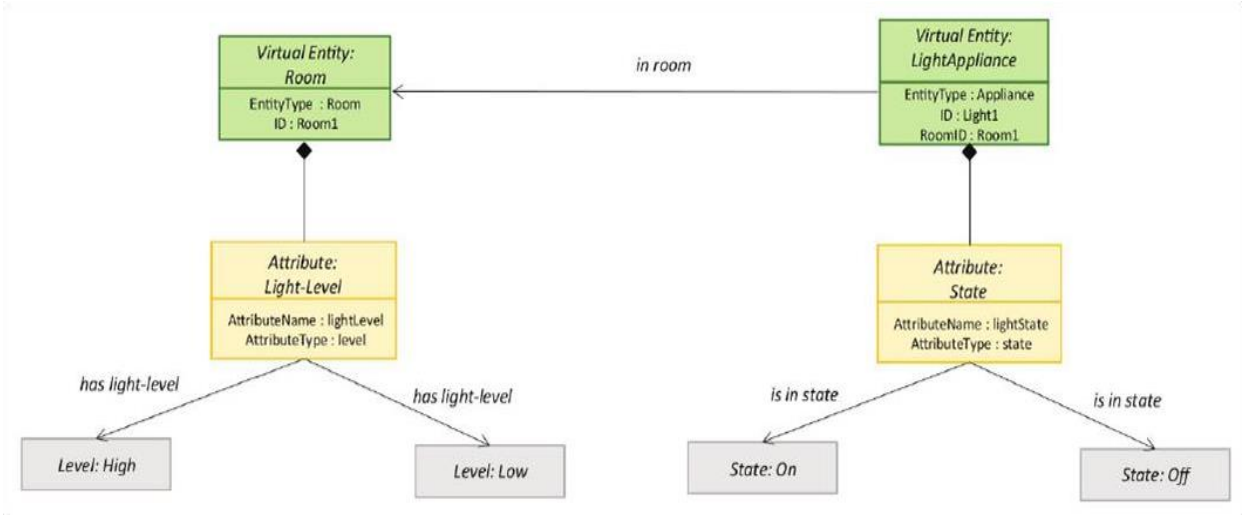
Webbased Application
Smartphone user –application
Light sensor
On/off-Actuator



Physical Entity	• The physical identifiable objects in the environment(eg. A room, a light, an environment etc.,) • IoT system provides information about the physical entity (using sensors) or performs actuation upon the physical entity (eg. Switching on a light).
Virtual Entity	• Virtual entity is a representation of the **physical entity** in the digital world • For every physical entity there is a virtual entity.
Device	• Devices provide a medium for interaction between physical and virtual entities. • Devices are used to gather information from or perform actuation on physical entities.
Resource	• Resources are software components which can be either on-device or network-resources • On-device resources are hosted on the device and provide sensing or actuation (eg: operating system) • Network-resources include software components that are available on the network (eg: database)
Service	• Services provide an **interface** for interacting with the physical entity • Services access resources to perform operations on physical entities

Step 4: Information Model Specification

This step defines Information Model. The Information Model defines the **structure** of all the information in the IoT system, example, attributes of Virtual Entities, relations, etc. The information model does not describe "how the information is represented or stored". The information model defines the list of the Virtual Entities, their attributes, and the relations of the domain model.



Step 5: Service Specifications

In this step, Service specifications define the services in the IoT system such as service types, service inputs/output, service endpoints, service schedules, service preconditions, and service effects. These services either change the state or attribute values or retrieve the current values. The Mode service is a RESTful web service that sets the mode to auto or manual (PUT request), or retrieves the current mode (GET request). The mode is updated to/retrieved from the database. The State service is a RESTful web service that sets the light appliance state to on/off (PUT request) or retrieves the current light state (GET request). The state is updated to/retrieved from the status database. The Controller service runs as a native service on the device.

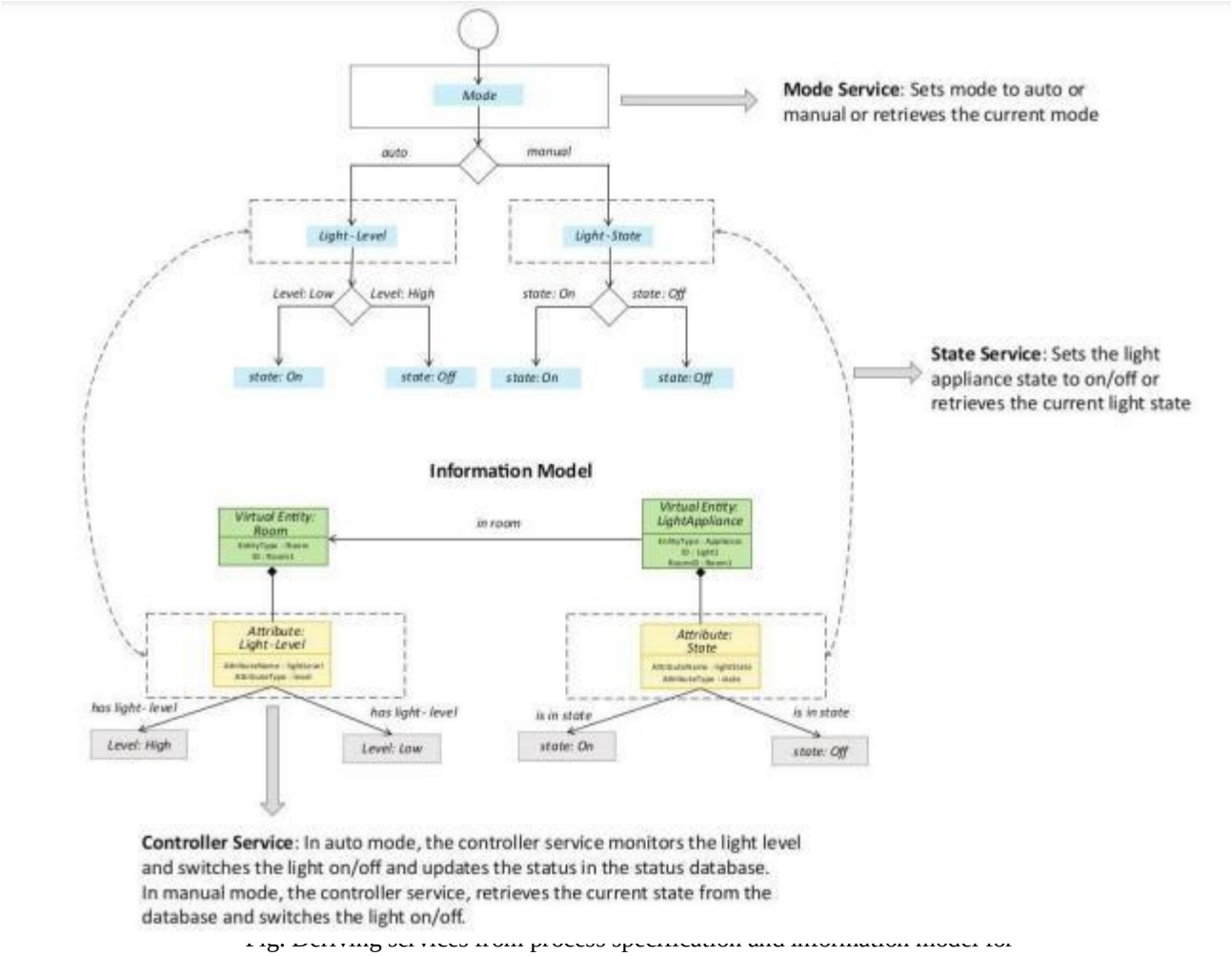
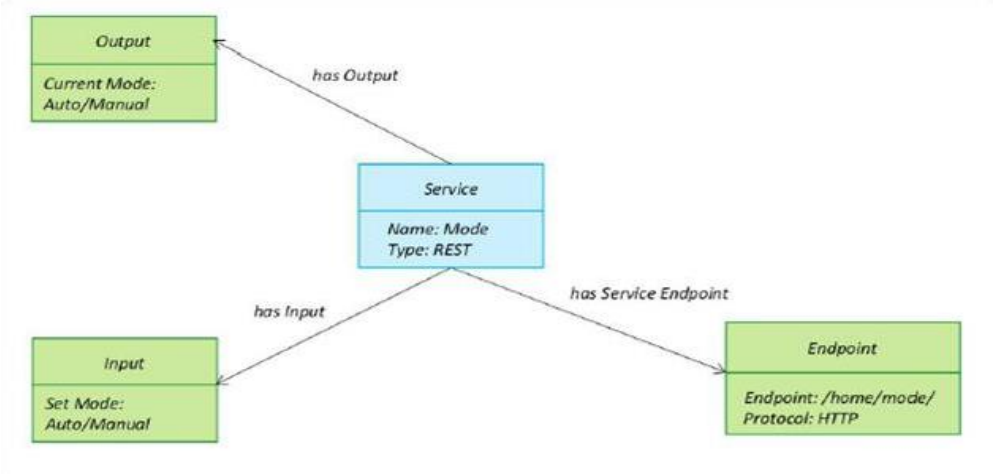


Fig. Deriving services from process specification and information model for Home automation system



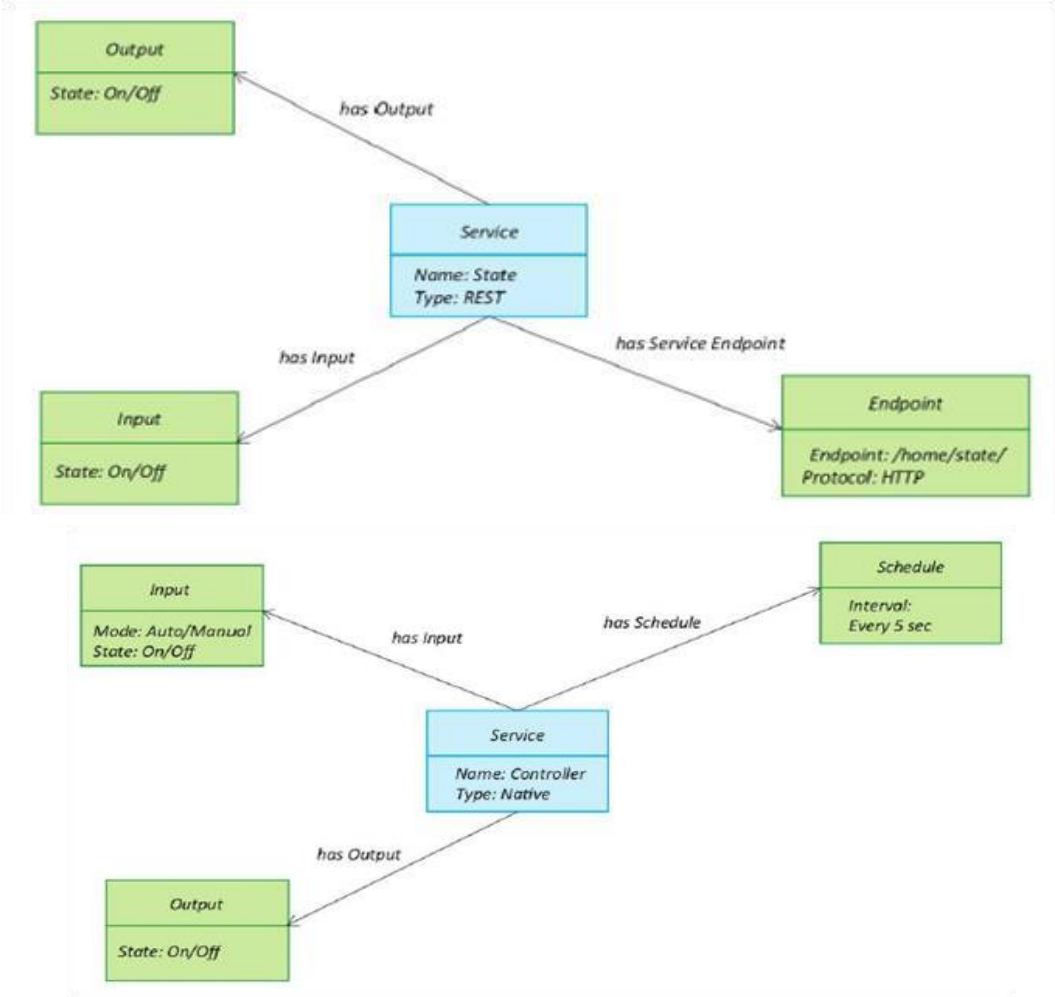
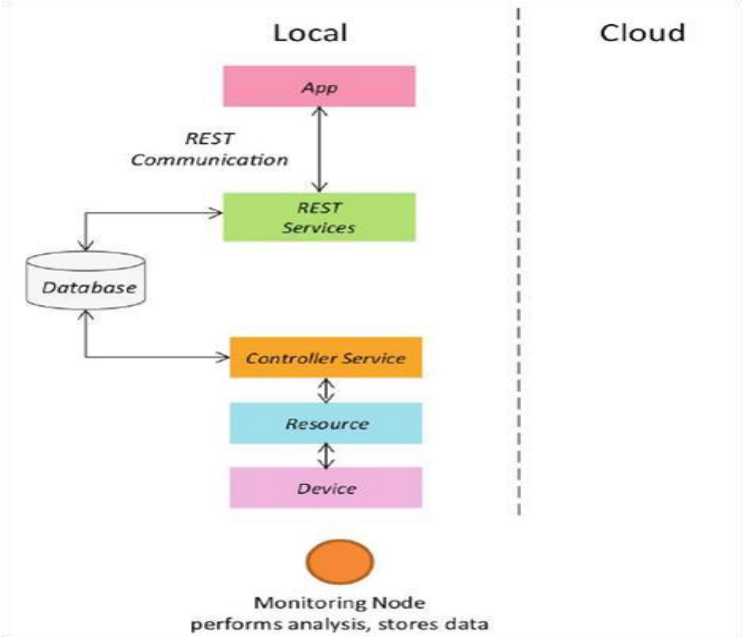


Fig: Mode Service , state service and controller service.

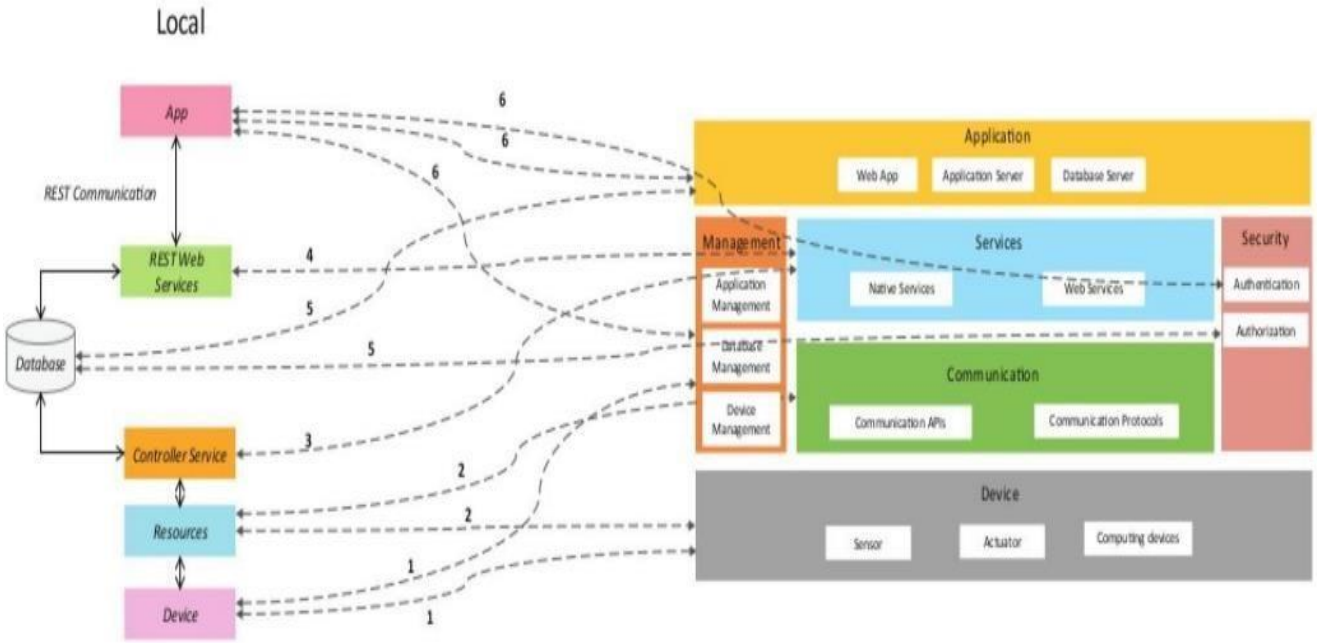
Step 6: IoT Level Specification

In this step define the IoT level for the system. Five types of IoT deployment levels are used according to different conditions.



Step 7: Functional View Specification

In the seventh step define the Functional View. The Functional View (FV) defines the functions of the loT systems grouped into various Functional Groups (FGs). Each Functional Group either provides functionalities for interacting with instances of the Domain Model. A Functional View includes:



1. IoT device maps to the Device FG (sensors, actuators devices, computing devices) and the Management FG (device management)

2. Resources map to the Device FG (on-device resource) and Communication FG (communication APIs and protocols)

3. Controller service maps to the Services FG (native service). Web Services map to Services FG (web services)

4. Web Services map to Services FG (web services)

5. Database maps to the Management FG (database management) and Security FG (database security)

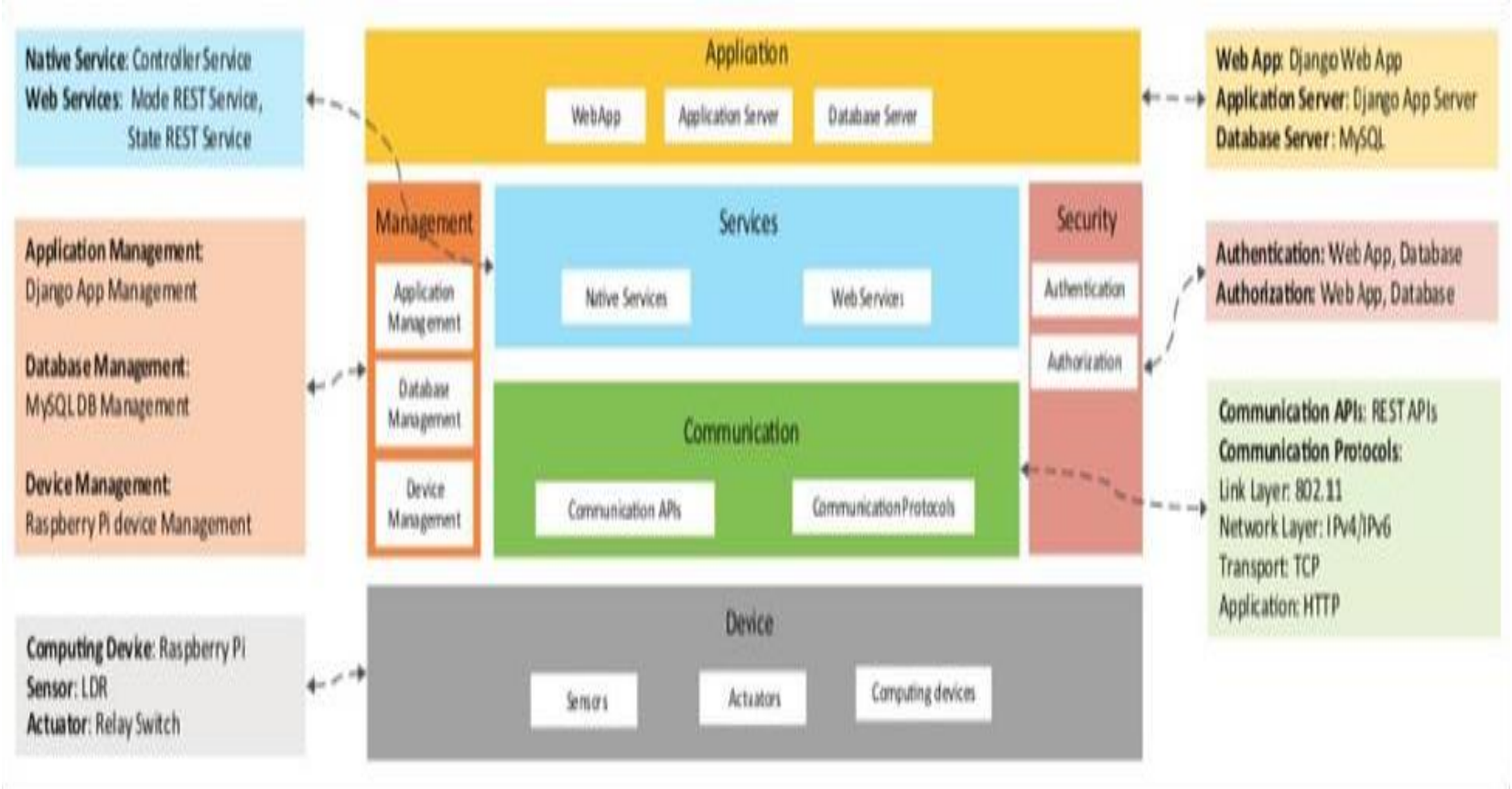
6. Application maps to the Application FG (web application, application and database servers), Management FG (app management) and Security FG (app security)

.

- **Device:**
  The device FG contains devices for monitoring and control.
- **Communication:**
  The communication FG handles the communication protocols and APIs (such as REST and WebSocket) that are used by the services and applications to exchange data over the network. for the IoT system. These are the backbone of IoT systems and enable network connectivity.
- **Services:**
  The service FG includes various services involved in the IoT system such as services for device monitoring, device control services, data publishing services, and services for device discovery.
- **Management:**
  The management FG includes all functionalities that are needed to configure and manage the loT system.
- **Security:**
  The security FG includes security mechanisms for the loT system such as authentication, authorization, data security, etc.
- **Application:**
  The application FG includes applications that provide an interface for the users to control and monitor various aspects of the loT system. Applications also allow users to view the system status and the processed data.

Step 8: Operational View Specification

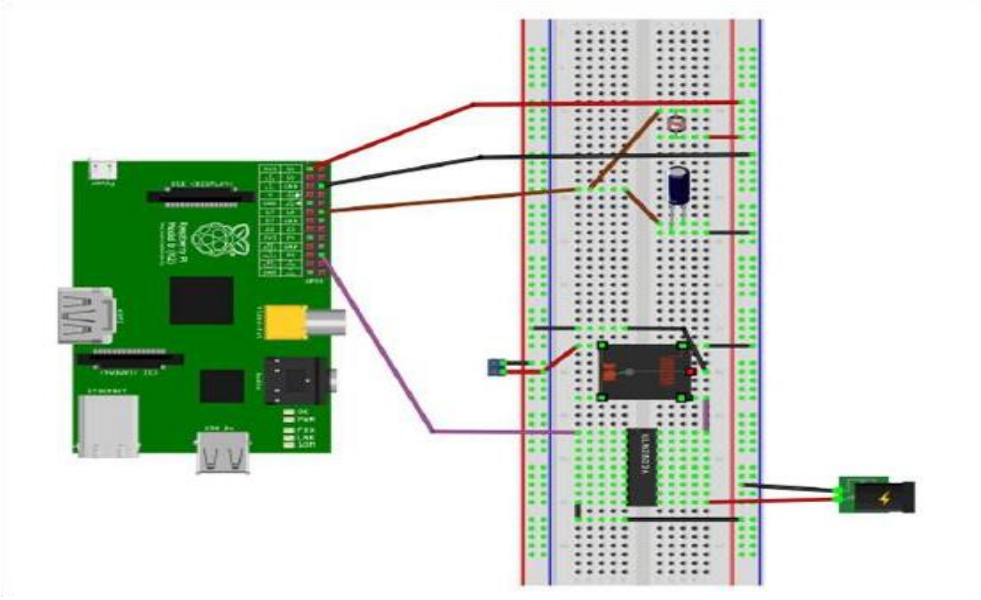
In this step define the Operational View Specifications. IoT system deployment and operation are defined, such as service hosting options, storage options, device options, application hosting options, etc.


Native Service: Controller Service
Web Services: Node REST Service, State REST Service
Application
WebApp
Application Server
Database Server
Web App: Django Web App
Application Server: Django App Server
Database Server: MySQL
Application Management: Django App Management
Database Management: MySQL DB Management
Device Management: Raspberry Pi device Management
Management
Application Management
Database Management
Device Management
Services
Native Services
Web Services
Communication
Communication APIs
Communication Protocols
Security
Authentication
Authorization
Authentication: Web App, Database
Authorization: Web App, Database
Communication APIs: REST APIs
Communication Protocols:
Link Layer: 802.11
Network Layer: IPv4/IPv6
Transport: TCP
Application: HTTP
Computing Device: Raspberry Pi
Sensor: LDR
Actuator: Relay Switch
Device
Sensors
Actuators
Computing devices

- **Devices:**
  The computing device (Raspberry Pi), light-dependent resistor (sensor), relay switch (actuator). . Communication APIS: REST APIs
- **Communication Protocols:**
  Link Layer - 802.11. Network Layer-IPv4/IPv6, Transport -TCP, Application - HTTP.
- **Services:**
  1. Controller Service - Hosted on the device, implemented in Python, and run as a native service.
  2. Mode service - REST-ful web service, hosted on a device, implemented with Django-REST Framework.
  3. State service - REST-ful web service, hosted on a device, implemented with Django-REST Framework.
- **Application:**
  Web Application - Django Web Application, Application Server - Django App Server, Database Server - MySQL.
- **Security:**
  Authentication: Web App, Database
- **Management:**
  Application Management - Django App Management. Database Management - MySQL DB Management, Device Management - Raspberry Pi device Management.

Step 9: Device & Component Integration

In this step integration of the devices and components design such as minicomputer, LDR sensor, and relay switch actuator.



Step 10: Application Development

The final step in the IoT design methodology. It is to develop the IoT application. The application has controls for the mode (auto-on or auto-off) and the light (on or off).

Text Books

1. Arshdeep Bahga & Vijay Madisetti, *Internet of Things – A Hands-on Approach*, 1st Edition, Universities Press, 2015.
2. David Hanes, Gonzalo Salgueiro, Patrick Grossetete, Rob Barton & Jerome Henry, *IoT Fundamentals: Networking Technologies, Protocols, and Use Cases for the Internet of Things*, 1st Edition, Cisco Press, 2017.
3. Fei Hu, *Internet of Things (IoT): Principles, Applications and Paradigms*, 1st Edition, CRC Press, 2016.
4. Gastón C. Hillar, *Internet of Things with Python*, 1st Edition, Packt Publishing, 2016.
5. Allen B. Downey, *Think Python: How to Think Like a Computer Scientist*, 2nd Edition, O'Reilly Media, 2015.

Reference Books

1. Olivier Hersent, David Boswarthick & Omar Elloumi, *The Internet of Things: Key Applications and Protocols*, 2nd Edition, Wiley, 2012.
2. Jan Holler, Vlasios Tsiatsis, Catherine Mulligan, Stefan Avesand, Stamatis Karnouskos & David Boyle, *From Machine-to-Machine to the Internet of Things: Introduction to a New Age of Intelligence*, 1st Edition, Academic Press (Elsevier), 2014.
3. Adrian McEwen & Hakim Cassimally, *Designing the Internet of Things*, 1st Edition, Wiley, 2014.
4. Charles Bell, *Beginning Sensor Networks with Arduino and Raspberry Pi*, 1st Edition, Apress, 2013.
5. Luciano Ramalho, *Fluent Python*, 2nd Edition, O'Reilly Media, 2022.
6. Wes McKinney, *Python for Data Analysis*, 3rd Edition, O'Reilly Media, 2022.

Case Study

Case Study: Smart Healthcare Monitoring System using IoT Design Methodology

Title

Design and Development of an IoT-Based Smart Healthcare Monitoring System using Python.

Background

Hospitals and healthcare providers need continuous monitoring of patients, especially elderly people and those with chronic diseases. Traditional monitoring requires patients to remain in hospitals, increasing healthcare costs and limiting access to timely medical attention. An IoT-based healthcare monitoring system enables doctors to monitor patients remotely by collecting real-time health data and sending alerts during emergencies.

Problem Statement

A multispecialty hospital wants to develop a smart healthcare system capable of monitoring patients' vital signs remotely. The system should continuously collect health data, store it securely in the cloud, and notify doctors immediately if abnormal conditions are detected.

Objectives

- Monitor patients' vital signs in real time.
- Reduce the need for continuous hospital visits.
- Provide instant alerts during medical emergencies.
- Store patient data securely in the cloud.
- Enable doctors to access patient information remotely.

IoT Design Methodology

Step 1: Purpose and Requirements Specification

The system should:

a. Measure body temperature, heart rate, blood oxygen level (SpO_2), and blood pressure.
b. Continuously monitor patient health.
c. Store medical records in the cloud.
d. Notify doctors during emergencies.
e. Provide remote access through mobile and web applications.

Step 2: Process Specification

The workflow includes:

1. Medical sensors collect patient health data.
2. Raspberry Pi receives sensor readings.
3. Python processes the collected information.
4. Data is converted into JSON format.
5. HTTP Library sends data to the cloud server.
6. Cloud stores patient records.
7. SMTP Library sends emergency email alerts.
8. Doctors monitor patients through a healthcare dashboard.

Step 3: Domain Model Specification

The system consists of:

- Patient
- Medical Sensors
- Raspberry Pi
- Wi-Fi Module
- Cloud Server
- Database
- Doctor
- Hospital Administrator
- Mobile Application

Relationship:
Patient → Sensors → Raspberry Pi → Cloud → Doctor

Step 4: Information Model Specification

Information stored includes:

- Patient ID
- Body Temperature
- Heart Rate
- Blood Pressure
- SpO_2 Level
- Date and Time
- Patient Location
- Doctor ID

Step 5: Service Specification

Services provided:

- Patient Health Monitoring
- Data Storage Service
- Emergency Alert Service
- Medical Report Generation
- Remote Consultation Service

Step 6: IoT Level Specification

IoT Level-3

Components:

- Medical Sensors
- Raspberry Pi Gateway
- Cloud Platform
- Doctor Dashboard
- Mobile Application

Step 7: Functional View Specification

System functions include:

- Health Data Collection
- Data Processing
- Secure Data Transmission
- Cloud Storage
- Health Analysis
- Emergency Alert Generation
- Doctor Dashboard

Step 8: Operational View Specification

Operations performed:

- Continuous patient monitoring
- Secure communication
- Automatic cloud synchronization
- Remote diagnosis
- System maintenance
- Patient data backup

Step 9: Device and Component Integration

Hardware

- Raspberry Pi
- Pulse Sensor
- Temperature Sensor
- Blood Pressure Sensor
- Pulse Oximeter (SpO_2 Sensor)
- Wi-Fi Module

Software

- Python
- Linux Operating System
- Cloud Database
- JSON Library
- HTTP Library
- SMTP Library

Step 10: Application Development

Python application performs:

- Reading sensor values
- Processing patient health data
- Creating JSON data
- Uploading data to cloud servers
- Sending emergency email notifications
- Displaying health reports on the dashboard

SREENIVASA INSTITUTE OF TECHNOLOGY AND MANAGEMENT STUDIES.
(AUTONOMOUS)
MCA DEPARTMENT

LECTURE NOTES

INTERNET OF THINGS

PART A (2 Marks)		
UNIT -III: DEVELOPING INTERNET OF THINGS		
Q.No.	**Questions**	**Bloom's Taxonomy Level**
1	Define IoT Design Methodology.	L1
2	What is the purpose of Requirement Specification in IoT Design Methodology?	L1
3	Define Process Specification in IoT system design.	L1
4	What is a Domain Model in IoT?	L1
5	Define Information Model Specification.	L1
6	What is meant by Service Specification in an IoT application?	L2
7	What is IoT Level Specification?	L1
8	Differentiate between Functional View and Operational View in IoT Design Methodology.	L2
9	What is Device and Component Integration?	L2
10	Why is Python preferred for IoT application development?	L2
11	List the basic Python data types used in IoT applications.	L1
12	What is JSON and why is it used in IoT?	L2
13	State the purpose of the XML library in IoT applications.	L1
14	What is the role of the HTTP and URL libraries in Python-based IoT systems?	L2
PART B(10 Marks)		
1	Explain the IoT Design Methodology with a neat diagram and discuss all ten design steps.	L2
2	Explain the Purpose and Requirements Specification and Process Specification stages of the IoT Design Methodology with suitable examples.	L2
3	Describe the Domain Model Specification and Information Model Specification used in IoT system design.	L2
4	Explain the Service Specification and IoT Level Specification in the IoT Design Methodology with appropriate illustrations.	L2
5	Discuss the Functional View Specification and Operational View Specification of an IoT system.	L3
6	Explain the process of Device and Component Integration in IoT application development.	L3
7	Explain the complete application development process in IoT using the IoT Design Methodology.	L3
8	Explain the design and working of an IoT-based Weather Monitoring System as a case study.	L3
9	Discuss the motivation for using Python in IoT systems and explain its advantages.	L2
10	Explain the logical design of IoT systems using Python.	L3
11	Describe Python data types and data structures with suitable examples for IoT applications.	L2
12	Explain the role of JSON and XML packages in Python-based IoT application development.	L3
13	Explain the HTTP Library and URL Library with suitable examples in IoT communication.	L3
14	Discuss the working of the SMTP Library and explain how it is used for notification services in IoT applications.	L3