USN										

CMRIT
CMR INSTITUTE OF TECHNOLOGY, BENGALURU.
ACCREDITED WITH A+ GRADE BY NAAC

Dec 2025/Jan2026 VTU EXAMINATION-INTERNET OF THINGS

Subject/Code: Internet of Things/ BCS701				
Date: 30/9/25	Duration: 90 mins	Max. Marks:50	Semester: 07	Branch: CSE

.	**Answer any FIVE FULL Questions**	**Marks**	**CO**	**RBT**
1a)	Explain the characteristics and applications of IOT **Characteristics of IoT** i)**Dynamic & Self Adapting**: IoT devices and systems may have the capability to dynamically adapt with the changing contexts and take actions based on their operating conditions, user's context or sensed environment. **Eg**: The surveillance system comprising of a number of surveillance cameras. The surveillance camera can adapt modes based on whether it is day or night. The surveillance system is adapting itself based on context and changing conditions. ii)**Self Configuring:** IOT devices have self configuring capability,allowing a large number of devices to work together to provide certain functionality. These devices have the ability configure themselves setup networking, and fetch latest software upgrades with minimal manual or user interaction. iii) **Inter Operable Communication Protocols:** support a number of interoperable communication protocols and can communicate with other devices and also with infrastructure. iv) **Unique Identity:** Each IoT device has a unique identity and a unique identifier(IP address). v) **Integrated into Information Network:** that allow them to communicate and exchange data with other devices and systems. **Applications of IoT:**) Home 2) Cities 3) Environment 4) Energy 5) Retail 6) Logistics 7) Agriculture 8) Industry 9) Health &LifeStyle	10	CO1	L2

b)	List and explain various IOT enabling technologies IoT is enabled by several technologies including Wireless Sensor Networks, Cloud Computing, Big Data Analytics, Embedded Systems, Security Protocols and architectures, Communication Protocols, Web Services, Mobile internet and semantic search engines. **1.4.1 Wireless Sensor Networks** A wireless sensor network comprises of distributed devices with sensors which are used to monitor the environmental and physical conditions. A WSN consist of a number of end nodes and routers and a co- ordinator. The coordinator collects the data from all the nodes. Coordinator also acts as a gateway that connects the WSN to the internet. WSNs used in IoT systems are described as follows: • Weather Monitoring System: in which nodes collect temp, humidity and other data, which is aggregated and analyzed. • Indoor air quality monitoring systems: to collect data on the indoor air quality and concentration of various gases. • Soil Moisture Monitoring Systems: to monitor soil moisture at various locations. • Surveillance Systems: use WSNs for collecting surveillance data(motion data detection). • Smart Grids : use WSNs for monitoring grids at various points. • Structural Health Monitoring Systems: Use WSNs to monitor the health of structures(building, bridges) by collecting vibrations from sensor nodes deployed at various points in the structure. WSNs are enabled by wireless communication protocols such as IEEE 802.15.4. Zig Bee is one of the most popular wireless technologies used by WSNs .Zig Bee specifications are based on IEEE 802.15.4. Zig Bee operates 2.4 GHz frequency and offers data rates upto 250 KB/s and range from 10 to	10	CO1	L2

100meters.

1.4.2 Cloud Computing

Cloud computing is a transformative computing paradigm that involves delivering applications and services over the internet. Cloud computing involves provisioning of computing, networking and storage resources on demand and providing these resources as metered services to the users, in a "pay as you go". Cloud computing resources can be provisioned on-demand by the users, without requiring interactions with the cloud service provider. The process of provisioning resources is automated.

Cloud computing services are offered to users in different forms.

- ***Infrastructure-as-a-service(IaaS):***Provides users the ability to provision computing and storage resources. These resources are provided to the users as a virtual machine instances and virtual storage.
- ***Platform-as-a-Service(PaaS):*** Provides users the ability to develop and deploy application in cloud using the development tools, APIs, software libraries and services provided by the cloud service provider.
- ***Software-as-a-Service(SaaS):*** Provides the user a complete software application or the user interface to the application itself. The cloud service provider manages the underlying cloud infrastructure including servers, network, operating systems, storage, and application software.

1.4.3 Big data Analysis

Big data is defined as collections of data sets whose volume , velocity or variety is so large that it is difficult to store, manage, process and analyze the data using traditional databases and data processing tools.

Some examples of big data generated by IoT are

Sensor data generated by IoT systems.

- Machine sensor data collected from sensors established in industrial and energy systems.
- Health and fitness data generated IoT devices.
- Data generated by IoT systems for location and tracking vehicles.
- Data generated by retail inventory monitoring systems.

The underlying characteristics of Big Data are

Volume: There is no fixed threshold for the volume of data for big data. Big data is used for massive scale data.

Velocity: Velocity is another important characteristics of Big Data and the primary reason for exponential growth of data.

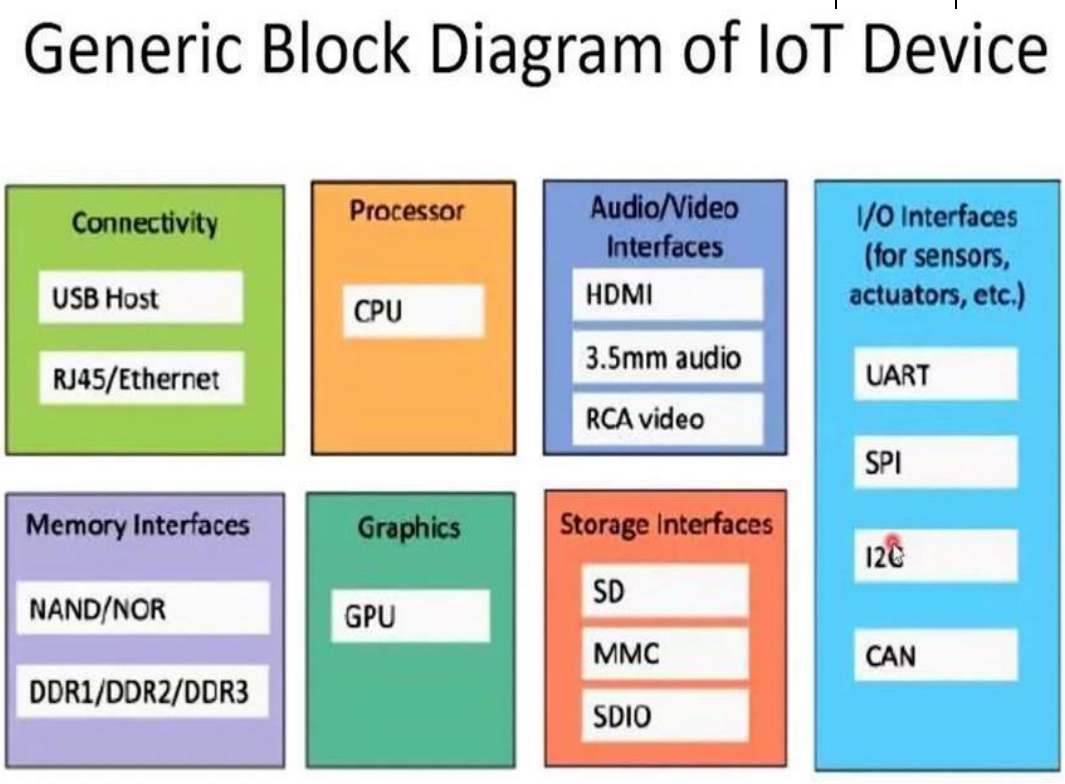
Variety: Variety refers to the form of data. Big data comes in different forms such as structured or unstructured data including test data, image , audio, video and sensor data .

1.4.4 Communication Protocols:

Communication Protocols form the back-bone of IoT systems and enable network connectivity and coupling to applications.

- Allow devices to exchange data over network.
- Define the exchange formats, data encoding addressing

	schemes for device and routing of packets from source to destination. • It includes sequence control, flow control and retransmission of lost packets. **1.4.5 Embedded Systems:** Embedded Systemsis a computer system that has computer hardware and software embedded to perform specific tasks. Key components of embedded system include microprocessor or micro controller, memory (RAM, ROM, Cache), networking units (Ethernet Wi-Fi Adaptor), input/output units (Display, Keyboard, etc..,) and storage (Flash memory). Embedded System range from low cost miniaturized devices such as digital watches to devices such as digital cameras, POS terminals, vending machines, appliances etc.,			
2	What is IOT explain the generic block diagram of an IOT device • An IoT device may consist of several interfaces for connections to other devices, both wired and wireless. • I/O interfaces for sensors • Interfaces for Internet connectivity • Memory and storage interfaces • Audio/video interfaces. • HDMI: High definition multimedia Interface. • 3.5mm: Audio Jack which headphone adapter. • RCA: Radio corporation of America. • UART: Universal Asynchronous Receiver Transmitter. • SPI: Serial Peripheral Interface. • I2C: Inter integrated circuit • CAN: Controller Area Network used for Micro-controllers and devices to communicate. • SD: Secure digital (memory card) • MMC: multimedia card • SDIO: Secure digital Input Output • GPU: Graphics processing unit. • DDR: Double data rate	10	CO1	L2

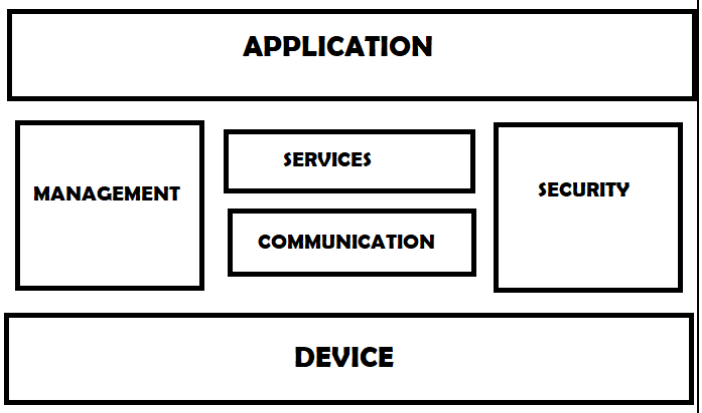


2b	Describe the components of an IOT system and explain IOT level-1 system with a diagram	10	CO1	L2

1) IoT Functional Blocks:

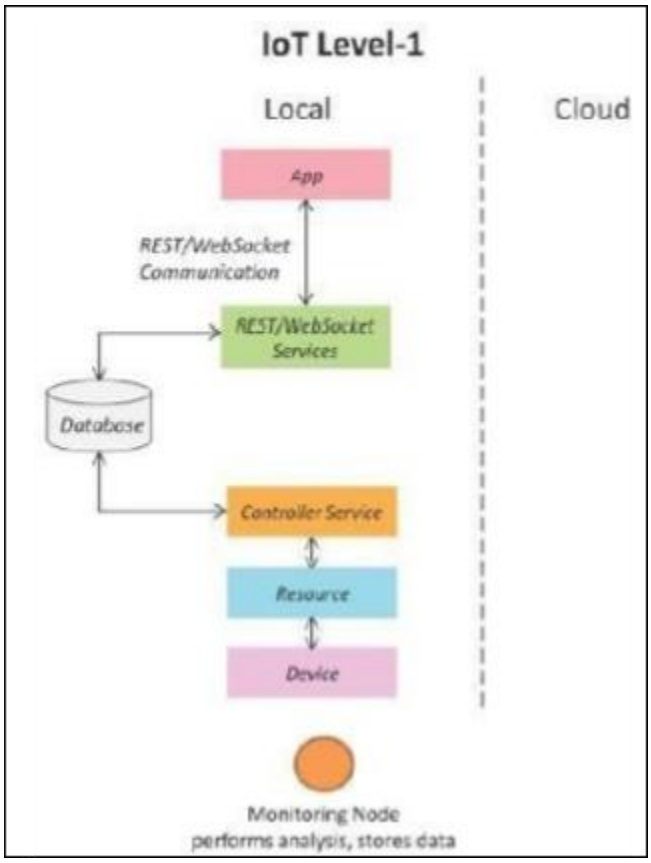
Provide the system the capabilities for identification, sensing, actuation, communication and management

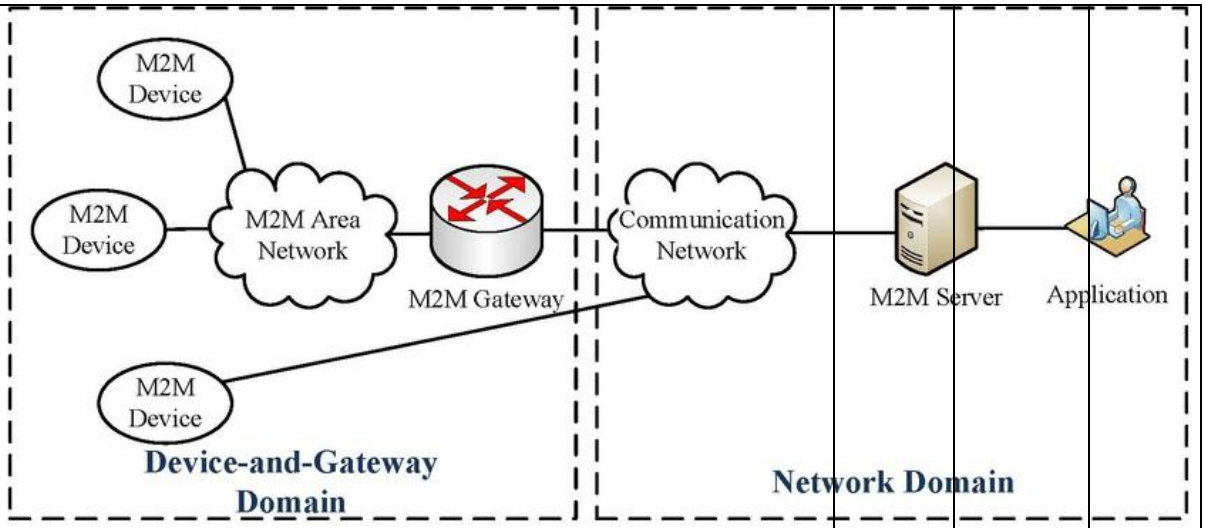
- Device: An IoT system comprises of devices that provide sensing, actuation, monitoring and control functions.
- Communication: handles the communication for IoT system.
- Services: for device monitoring, device control services, data publishing services and services for device discovery.
- Management: Provides various functions to govern the IoT system.
- Security: Secures IoT system and priority functions such as authentication, authorization, message and context integrity and data security.
- Application: IoT application provide an interface that the users can use to control and monitor various aspects of IoT system.



1.4.1 IoT Level-1

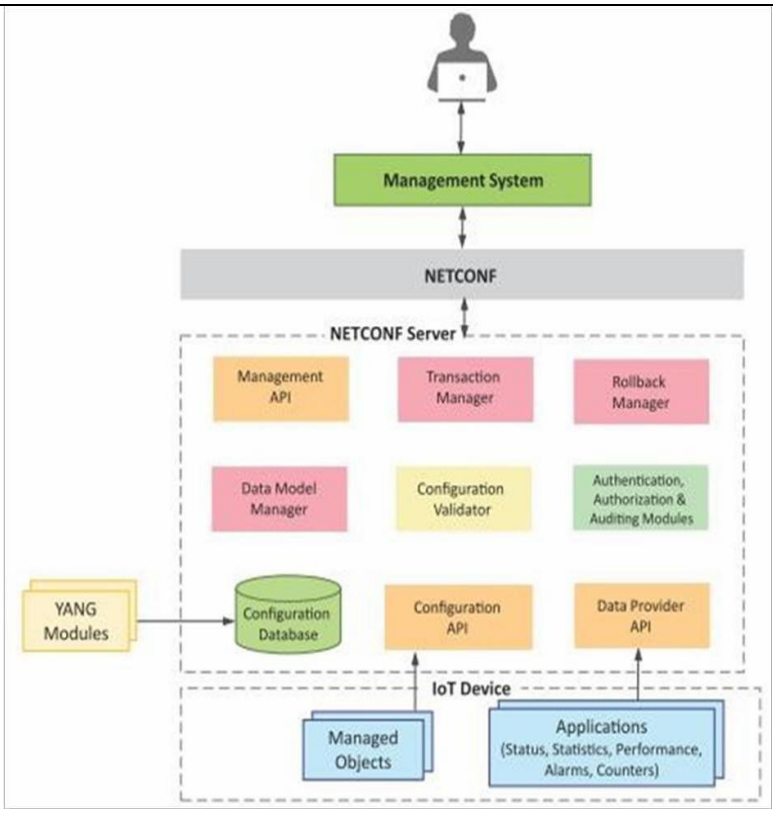
Level-1 IoT systems has a single node that performs sensing and/or actuation, stores data, performs analysis and host the application. Suitable for modeling low cost and low complexity solutions where the data involved is not big and analysis requirement are not computationally intensive. An e.g., of IoT Level1 isHomeautomation.The system consist of a single node that allows controlling the lights and appliances in a home the device used in this system interfaces with the lights and appliances using electronic rely switches. The status information of each light or appliances is maintained in a local database. REST services deployed locally allow retrieving and updating the state of each lighter appliance in the status

	database. The controller service continuously monitors the state of each light or appliance by retrieving the light from the database  			
3a	Describe M2M system architecture and M2M gateway with a block diagram ● M2M device connects to the network domain via direct connectivity or M2M gateway. In the first case, the M2M device connects to the network domain via the access network, which performs the procedures such as registration, authentication, authorization, management, and provisioning with the network domain. In the second case, the M2M device connects to the M2M gateway using the M2M area network. ● M2M area network provides connectivity between M2M devices and M2M gateways. ● M2M gateway acts as a proxy between M2M devices and the network domain. As an example, an M2M gateway can run an application that collects and treats various information (e.g., contextual parameters) from sensors and meters. ● M2M communication network provides connection between the M2M gateways/devices and the M2M servers. Usually it contains two parts: the access network and the Internet. M2M server works as a middleware layer to pass data through various application services	10	CO1	L2

	 .			
3b	Explain the need for IOT system Management **Need for IoT Systems Management** Managing multiple devices within a single system requires advanced management capabilities. 1) **AutomatingConfiguration:**IoTsyste mmanagementcapabilitiescanhelpin 2) **Monitoring Operational & Statistical Data :** Management systems can help in monitoring opeartional and statistical data of a system. This data can be used for fault diagnosis orprognosis. 3) **Improved Reliability:** A management system that allows validating the system configurationsbeforetheyareputinto effect canhelp inimprovingthe systemreliability. 4) **System Wide Configurations :** For IoT systems that consists of multiple devices	10	CO1	L2

	or nodes, ensuring system wide configuration can be critical for the correct functioning of thesystem. 5) **Multiple System Configurations :** F or some systems it m ay be desirable to have multiplevalidconfigurationswhichareapplie datdifferenttimesorin certainconditions. 6) **Retrieving&ReusingConfigurations:** Managementsystemswhichhavethecapability of retrieving configurations from devices can help in reusing the configurations for other devices of the sametype.			
4a	Discuss the difference between M2M and IOT systems. M2M, or machine-to-machine, is a direct communication between devices using wired or wireless communication channels. M2M refers to the interaction of two or more devices/machines that are connected to each other. These devices capture data and share with other connected devices, creating an intelligent network of things or systems. Devices could be sensors, actuators, embedded systems or other connected elements. M2M technology could be present in our homes, offices, shopping malls and other places. Controlling electrical appliances like bulbs and fans using RF or Bluetooth from your smartphone is a simple example of M2M applications at home. Here, the electrical appliance and your smartphone are the two machines interacting with each other. The Internet of Things (IoT) is the network of physical devices embedded with sensors, software and electronics, enabling these devices to communicate with each other and exchange data over a computer network. The things in the IoT refer to hardware devices uniquely identifiable through a network platform within the Internet infrastructure.	10	CO2	L2

	M2M versus the IoT **M2M** / **IoT** M2M is about direct communication between machines. / The IoT is about sensors automation and Internet platform. It supports point-to-point communication. / It supports cloud communication. Devices do not necessarily rely on an Internet connection. / Devices rely on an Internet connection. M2M is mostly hardware-based technology. / The IoT is both hardware- and software-based technology. Machines normally communicate with a single machine at a time. / Many users can access at one time over the Internet. A device can be connected through mobile or other network. / Data delivery depends on the Internet protocol (IP) network.			
4b	Describe IOT system management with NETCONF-YANG with a neat diagram YANGisadatamodelinglanguageusedtomodelconfigurationandstatedatamanupulated by the NETCONF protocol. ThegenericapproachofIoTdevicemanagement with NETCONF-YANG. Rolesof various componentsare: 1) ManagementSystem 2) ManagementAPI 3) TransactionManager 4) RollbackManager 5) Data ModelManager 6) ConfigurationValidator 7) ConfigurationDatabase 8) ConfigurationAPI Data ProviderAPI	10	CO2	L3



1) **Management System :** The operator uses a management system to send NETCONF messages to configure the IoT device and receives state information and notifications from the device as NETCONFmessages.

2) **ManagementAPI :**allowsmanagementapplicationtostartNETCONFsessions.

3) **Transaction Manager:** executes all the NETCONF transactions and ensures that ACID properties hold true for thetrasactions.

4) **Rollback Manager :** is responsible for generating all the transactions necessary to rollback a current configuration to its original state.

5) **DataModelManager:** KeepstrackofalltheYANG datamodelsandthecorresponding managed objects. Also keeps track oftheapplications which providedataforeach partof a datamodel.

6) **Configuration Validator :** checks if the resulting configuration after applying a transaction would be a validconfiguration.

7) **ConfigurationDatabase:**containsbothconfiguration andoperastionaldata **Configuration API :** Using the

	configuration API the application on the IoT device can be read configuration data from the configuration datastore and write opeartional data to the opearationaldatastore. 8) **DataProviderAPI:** Applicationsonthe IoTdevicecanregisterforcallbacksforvarious events using the Data Provider API. Through the Data Provider API, the applications can report statistics and opeartionaldata. **StepsforIoTdeviceManagementwithNETCONF-YANG** 1) Create a YANG model of the system that defines the configuration and state data of the system. 2) CompletetheYANGmodelwiththe_Inctool'whichcom eswithLibnetconf. 3) Fillinthe IoTdevicemangementcodeintheTransAPImodule. 4) BuildthecallbacksC filetogeneratethelibraryfile. 5) Load the YANG module and the TransAPIm 6) The operator can now connect from the management system to the Netopeer server using the NetopeerCLI. 7) Operator can issue NETCONF commands from the Netopeer CLI. Command can be issued to changew the configuration dsta, get operational dat or execute an RPC on the IoTdevice.			
5a	Explain the steps involved in IOT system design methodology with a diagram **Step 1: Purpose & Requirements Specification:** The first step in IoT system design methodology is to define the purpose and requirements of the system. In this step, the system purpose, behavior and requirements (such as data collection requirements, data analysis requirements, system management requirements, data privacy and security requirements, user interface requirements, ...) are captured. **Step 2: Process Specification:** The second step in the IoT design methodology is to define the process specification. In this step, the use cases of the IoT system are formally described based on and derived from	10	CO3	L3

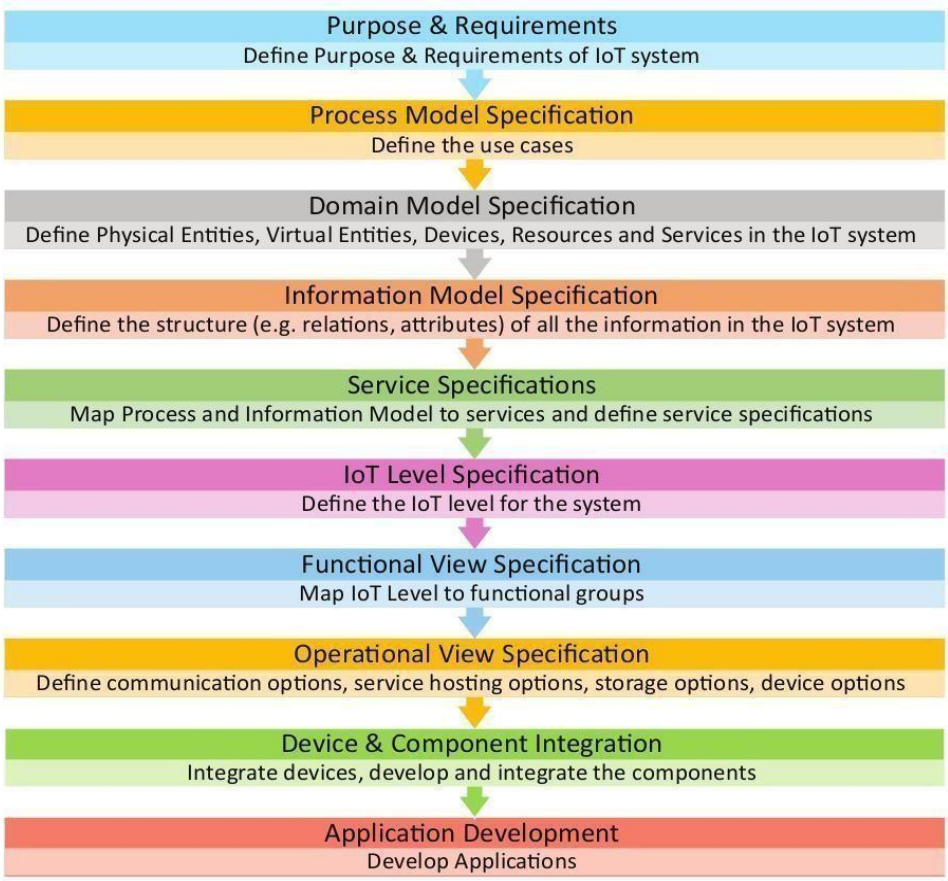
	the purpose and requirement specifications. **Step 3: Domain Model Specification:** The third step in the IoT design methodology is to define the Domain Model. The domain model describes the main concepts, entities and objects in the domain of IoT system to be designed. Domain model defines the attributes of the objects and relationships between objects. Domain model provides an abstract representation of the concepts, objects and entities in the IoT domain, independent of any specific technology or platform. With the domain model, the IoT system designers can get an understanding of the IoT domain for which the system is to be designed. **Step 4: Information Model Specification:** The fourth step in the IoT design methodology is to define the Information Model. Information Model defines the structure of all the information in the IoT system, for example, attributes of Virtual Entities, relations, etc. Information model does not describe the specifics of how the information is represented or stored. To define the information model, we first list the Virtual Entities defined in the Domain Model. Information model adds more details to the Virtual Entities by defining their attributes and relations. **Step 5: Service Specifications:** The fifth step in the IoT design methodology is to define the service specifications. Service specifications define the services in the IoT system, service types, service inputs/output, service endpoints, service schedules, service preconditions and service effects. **Step 6: IoT Level Specification:** The sixth step in the IoT design methodology is to define the IoT level for the system.			

Step 7: Functional View Specification:

The seventh step in the IoT design methodology is to define the Functional View. The Functional View (FV) defines the functions of the IoT systems grouped into various Functional Groups (FGs). Each Functional Group either provides functionalities for interacting with instances of concepts defined in the Domain Model or provides information related to these concepts.

Step 8: Operational View Specification:

The eighth step in the IoT design methodology is to define the Operational View Specifications. In this step, various options pertaining to the IoT system deployment and operation are defined, such as, service hosting options, storage options, device options, application hosting options, etc



5b	Explain the following python datatypes with an example.
	NUMBERS
	python supports different numeric types.
	Types
	• **int** – whole numbers
	• **float** – decimal numbers
	• **complex** – real + imaginary part
	Examples
	a = 10 # int
	b = 3.14 # float
	c = 2 + 3j # complex
	STRINGS
	Strings represent **text** and are enclosed in quotes.
	Examples
	name = "Python"
	message = 'Hello World'
	
	print(name)
	print(message)
	String Operations
	text = "Python Programming"
	
	print(text[0]) # P (indexing)
	print(text[0:6]) # Python (slicing)
	print(text.upper()) # PYTHON PROGRAMMING
	print(len(text)) # length
		10	CO3	L3

LISTS

Lists store **multiple items** and are **mutable**.

Example

```
numbers = [10, 20, 30, 40]
names = ["Alice", "Bob", "Charlie"]

print(numbers)
print(names)
```

List Operations

```
numbers.append(50)      # add item
numbers[1] = 25         # modify item
numbers.remove(30)      # remove item
print(numbers)
```

TUPLES

uples are like lists, but **immutable**.

Example

```
coordinates = (10, 20)
colors = ("red", "green", "blue")

print(coordinates)
print(colors)
```

Accessing Tuple Elements

```
print(coordinates[0])
print(colors[1])
```

---

5. Dictionaries in Python

Dictionaries store data in **key–value pairs**.

Example

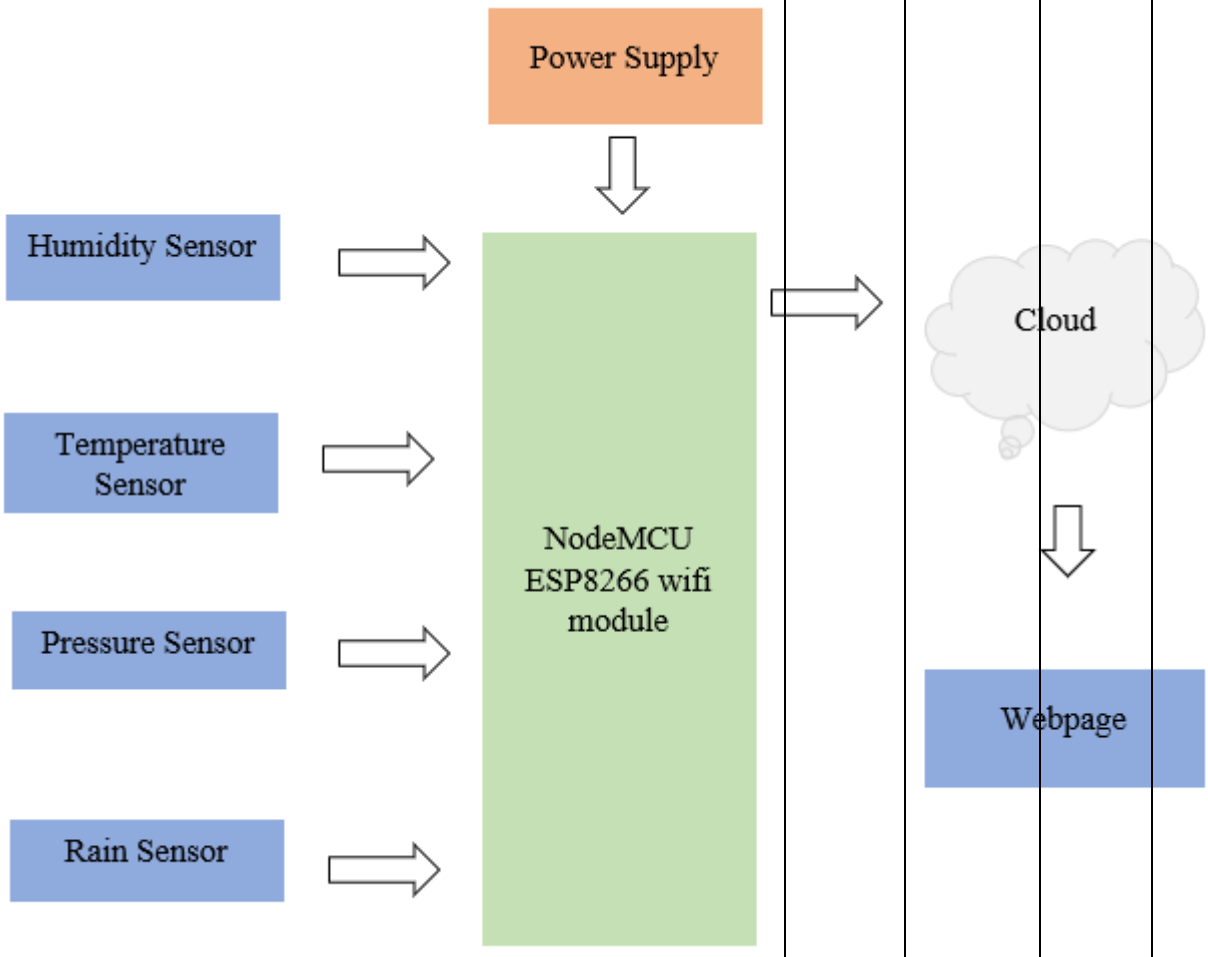
```
student = {
   "name": "Sreedevi",
   "age": 25,
   "course": "Python"
}

print(student)
```

Dictionary Operations

```
print(student["name"])     # access value
student["age"] = 26        # update
student["grade"] = "A"     # add new key

print(student)
```

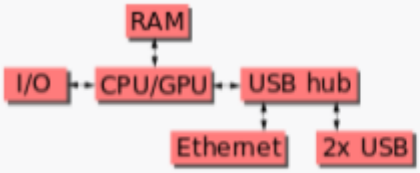
6a	Explain IOT system for weather monitoring An **IoT-based weather monitoring system** continuously measures atmospheric parameters such as **temperature, humidity, pressure, rainfall, wind speed**, etc., using sensors connected to the internet. The collected data is transmitted to a **cloud platform** for storage, analysis, and visualization in real time. sed to sense environmental parameters: **Parameter** **Sensor** Temperature & Humidity DHT11 / DHT22 Atmospheric Pressure BMP180 / BMP280 Rainfall Rain Gauge Sensor Wind Speed Anemometer Light Intensity LDR  • Sensors collect weather data • Microcontroller reads sensor outputs • Data is transmitted via Wi-Fi/GSM • Cloud platform stores and processes data • Users view real-time graphs and receive alerts	10	CO4	L2

6b	With the program describe the file handling functionality of python to read and write file object
	
	File handling in Python allows data to be **stored permanently** in files and accessed later. Python provides built-in functions to **create, read, write, and modify files** using a **file object**.
	
	--- 	
	
	◈ **Steps in File Handling**
	1. Open a file
	2. Perform read/write operations
	3. Close the file
	Opening a File
	file_object = open("filename.txt", "mode")
	File Modes
	Mode Description
	"r" Read (default)
	"w" Write (overwrites)
	"a" Append
	"r+" Read & write
	"b" Binary mod
	Writing to a File (Program)
	Example: Write Data to a File
	# Writing to a file
	file = open("data.txt", "w")
	file.write("Python File Handling\n")
	file.write("Writing data into a file.")
	file.close()
	
	print("Data written successfully")
	📌 **Explanation**:
	• "w" mode creates a new file or overwrites existing file
	• write() writes text into file
	• close() saves and closes the file
	
	--- 	
	
	◈ **Reading from a File (Program)**
	Example: Read Entire File
	# Reading from a file
	file = open("data.txt", "r")
	content = file.read()
	file.close()
	
	print(content)
	📌 **Explanation**:
	• read() reads entire content of the file
	
	--- 	
	
	◈ **Reading Line by Line**
	Example: Using readline()
	file = open("data.txt", "r")
	line = file.readline()
	print(line)
	file.close()
	Example: Using readlines()
	file = open("data.txt", "r")
		10	CO4	L3

	lines = file.readlines() print(lines) file.close() **◈ Appending to a File** **Example: Append Data** file = open("data.txt", "a") file.write("\nThis line is appended.") file.close() print("Data appended") 📌 **Explanation**: • "a" mode adds data without deleting existing content **◈ File Object Methods** **Method** **Purpose** read() Read full file readline() Read one line readlines() Read all lines write() Write data close() Close file **◈ Example: Read and Write Together** with open("input.txt", "r") as infile: content = infile.read() with open("output.txt", "w") as outfile: outfile.write(content) print("File copied successfully")			
7a.	Explain the various components and peripherals of raspberry pi board **Raspberry Pi**(/paɪ/) is a series of smallsingle-board computersdeveloped in theUnited Kingdomby theRaspberry Pi Foundationin association withBroadcom. Early on, the Raspberry Pi project leaned towards the promotion of teaching basiccomputer sciencein schools and indeveloping countries. Later, the original model became far more popular than anticipated,selling outside itstarget marketfor uses such asrobotics. It is now widely used in many areas, such as for weather monitoring,because of its low cost, modularity, and open design. After the release of the second board type, the Raspberry Pi Foundation set up a new entity, named Raspberry Pi Trading, and installedEben UptonasCEO, with the responsibility of	10	CO4	L2

developing technology.The Foundation was rededicated as an educational charity for promoting the teaching of basic computer science in schools and developing countries.The Raspberry Pi is one of the best-sellingBritish computers.

The Raspberry Pi hardware has evolved through several versions that feature variations in the type of the central processing unit, amount



ofmemorycapacity, networking support, and peripheral-device support.

This block diagram describes Model B and B+; Model A, A+, and the Pi Zero are similar, but

lack theEthernetandUSBhub components. The Ethernet adapter is internally connected to an additional USB port. In Model A, A+, and the Pi Zero, the USB port is connected directly to thesystem on a chip(SoC). On the Pi 1 Model B+ and later models the USB/Ethernet chip contains a five-port USB hub, of which four ports are available, while the Pi 1 Model B only provides two. On the Pi Zero, the USB port is also connected directly to the SoC, but it uses amicro USB(OTG) port. Unlike all other Pi models, the 40 pin GPIO connector is omitted on the Pi Zero, with solderable through-holes only in the pin locations. The Pi Zero WH remedies this.

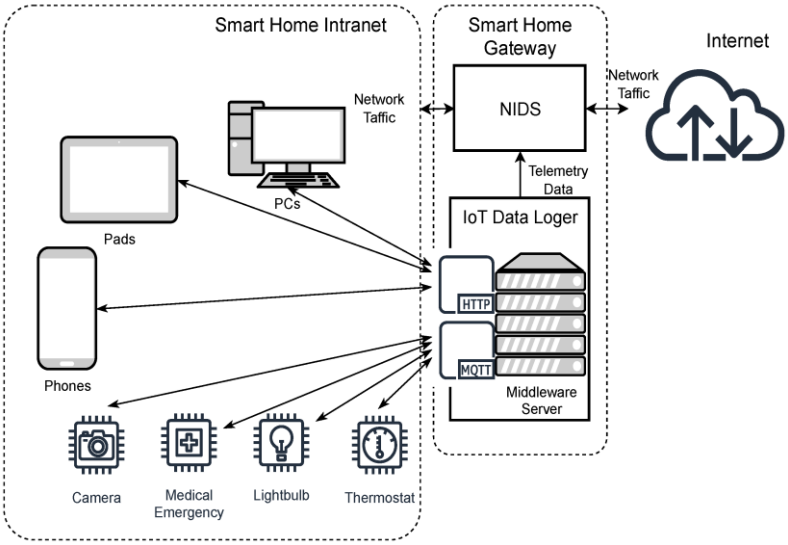
Processor speed ranges from 700 MHz to 1.4 GHz for the Pi 3 Model B+ or 1.5 GHz for the Pi 4; on-board memory ranges from 256MiBto 1GiBrandom-access memory(RAM), with up to 8 GiB available on the Pi 4.SecureDigital(SD) cards in MicroSDHC form factor (SDHC on early models) are used to store the operating system and program memory. The boards have one to fiveUSBports. For video output,HDMIandcomposite videoare supported, with a standard 3.5 mmtip-ring-sleevejack for audio output. Lower-level output is provided by a number ofGPIOpins, which support common protocols likeI²C. The B-models have an8P8CEthernetport and the Pi 3, Pi 4 and Pi Zero W have on-boardWi-Fi802.11nandBluetooth.

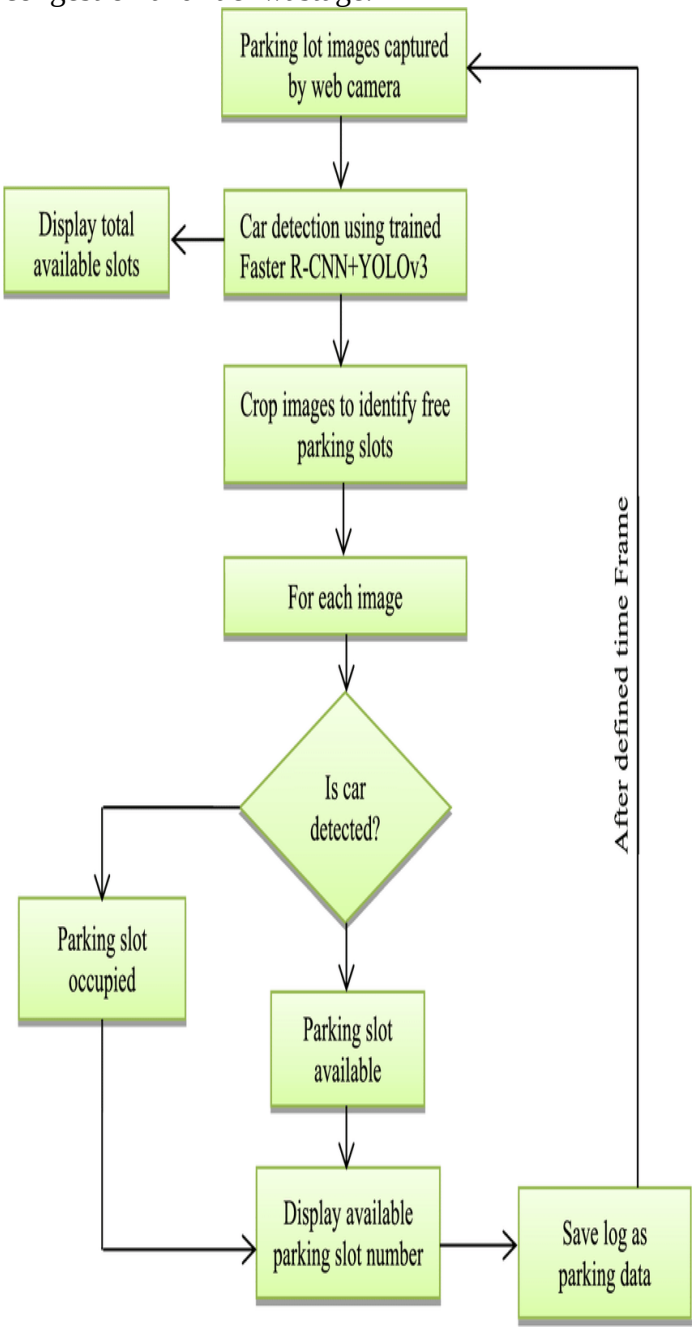
Raspberry Pi is a mono-board computing platform that's as tiny as a credit card. Initially it was developed for computer science education with later on progress

	to wider functions. Since the inception of Raspberry, the company sold out more than 8 million items. Raspberry Pi 3 is the latest version and it is the first 64-bit computing board that also comes with built-in Wi- Fi and Bluetooth functions. According to Raspberry Pi Foundation CEO Eben Upton, "*it's been a year in the making*". The Pi3 version is replaced with a quad-core 64-bit 1.1 GHz ARM Cortex A53 chip, 1GB of RAM, VideoCore IV graphics, Bluetooth 4.1 and 802.11n Wi-Fi. The developers claim the new architecture delivers an average 50% performance improvement over the Pi 2. Another peculiarity of Raspberry Pi is the *GPIO* (General Purpose Input-Output), which is a low-level interface of self-operated control by input-output ports. Raspberry has it as a 40pin connector. Raspberry Pi uses Linux as its default operating system (OS). It's also fully Android compatible. Using the system on Windows OS is enabled through any virtualization system like *XenDesktop*. If you want to develop an application for Raspberry Pi on your computer, it is necessary to download a specific toolset comprised of ARM-compiler and some libraries complied down to ARM-target platform like *glibc*.			

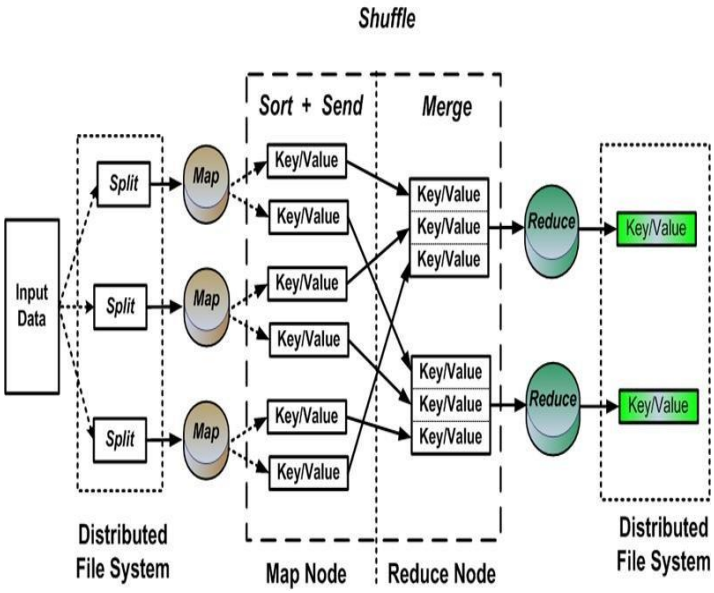
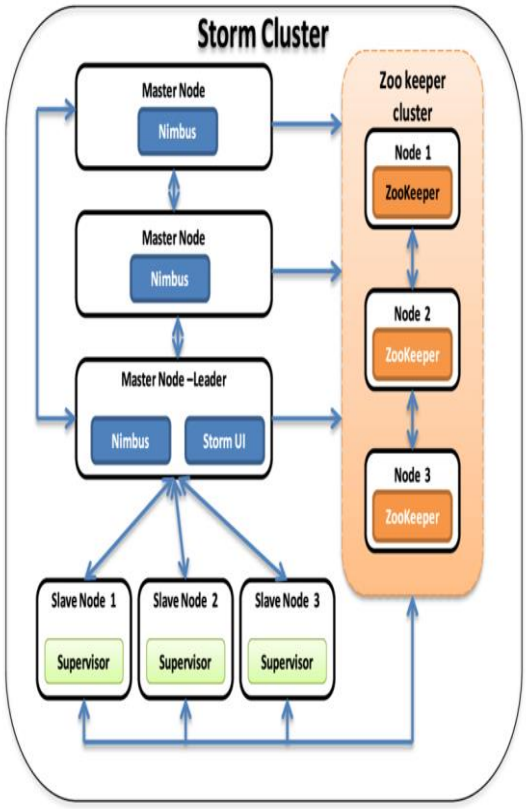
7b	Write a python program for switching LED on reading LDR reading
	
	import RPi.GPIO as GPIO
	import spidev
	import time
	
	# GPIO setup
	LED_PIN = 18
	GPIO.setmode(GPIO.BCM)
	GPIO.setup(LED_PIN, GPIO.OUT)
	
	# SPI setup for MCP3008
	spi = spidev.SpiDev()
	spi.open(0, 0) # Bus 0, Device 0
	spi.max_speed_hz = 1350000
	
	# Function to read ADC channel
	def read_adc(channel):
	adc = spi.xfer2([1, (8 + channel) << 4, 0])
	data = ((adc[1] & 3) << 8) + adc[2]
	return data
		10	CO4	L3

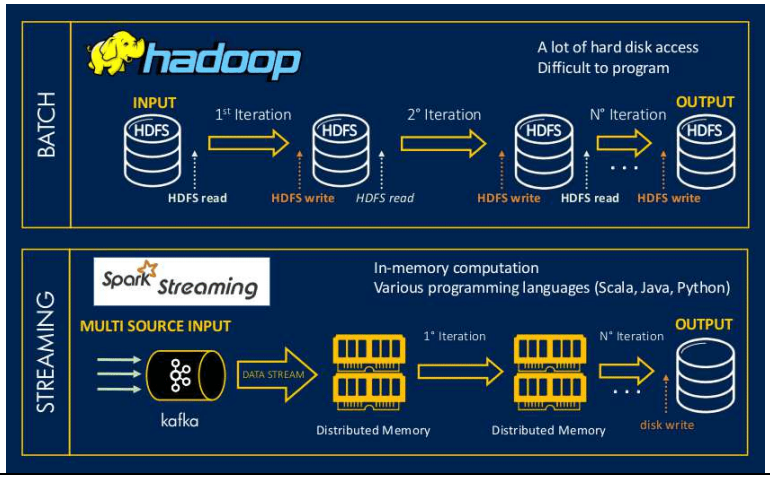
	THRESHOLD = 300 # Adjust based on environment
	
	try:
	while True:
	ldr_value = read_adc(0)
	print("LDR Value:", ldr_value)
	
	if ldr_value < THRESHOLD:
	GPIO.output(LED_PIN, GPIO.HIGH)
	print("LED ON (Dark)")
	else:
	GPIO.output(LED_PIN, GPIO.LOW)
	print("LED OFF (Bright)")
	
	time.sleep(1)
	
	except KeyboardInterrupt:
	print("Program stopped")
	
	finally:
	GPIO.cleanup()
	spi.close()			

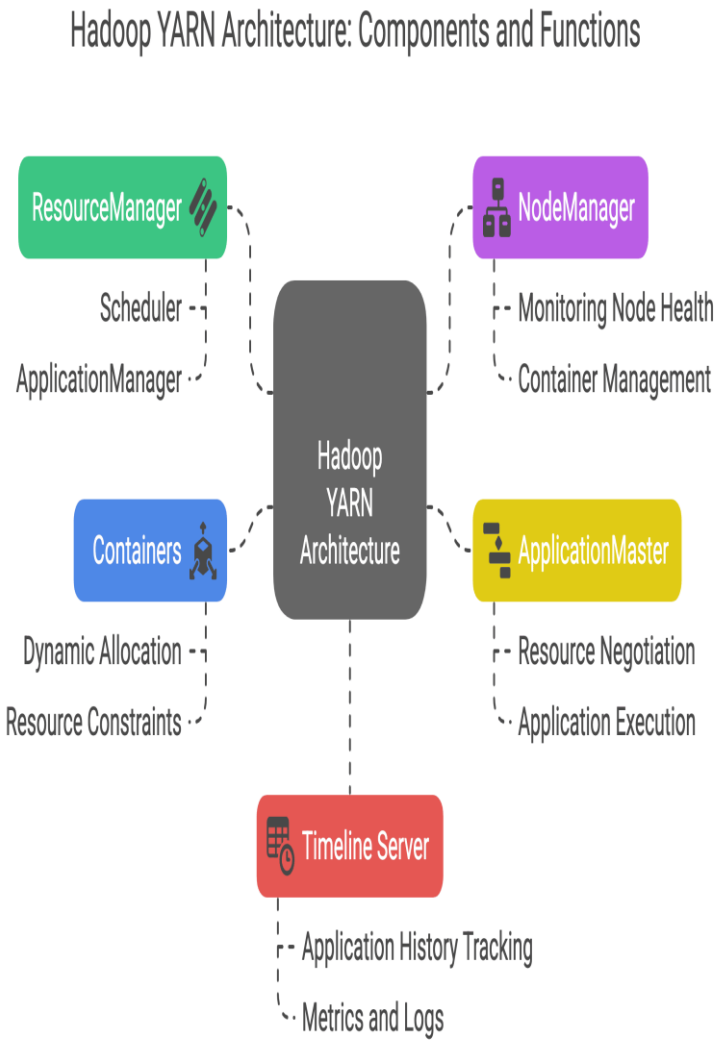
8a	Describe home intrusion detection system using IOT An IoT-based home intrusion detection system uses motion and door sensors to detect unauthorized entry. The sensor data is processed by a microcontroller and transmitted to the cloud via the internet. Alerts are sent instantly to the user's mobile device, and a local alarm is activated for security.  	10	CO5	L3

8b	Explain the smart parking IOT system A **Smart Parking System using IoT** is an intelligent solution that uses **sensors, microcontrollers, internet connectivity, and cloud platforms** to monitor parking space availability in real time and guide drivers to free slots, reducing traffic congestion and fuel wastage. • Vehicle enters parking slot • Sensor detects vehicle presence • Controller updates slot status • Data sent to cloud server • Mobile app & display boards update in real time • Driver is guided to nearest available slot smart parking system using IoT employs sensors to detect vehicle presence in parking slots. The sensor data is processed by a controller and transmitted to a cloud server through the internet. Users can view parking availability in real time using a mobile app or display board. This system reduces traffic congestion and fuel wastage.  	10	CO5	L3

9a	Explain the components of Hadoop cluster and Hadoop map reduce job execution **Apache Hadoop** is an open-source framework designed for **distributed storage and parallel processing** of large datasets across clusters of commodity hardware. **Components of a Hadoop Cluster** A Hadoop cluster mainly consists of **HDFS**, **YARN**, and **MapReduce**, running on **master and worker nodes**. **1. Hadoop Distributed File System (HDFS)** HDFS provides **fault-tolerant, distributed storage**. **a) NameNode** • Master of HDFS • Maintains **metadata** (file name, size, block locations) • Does **not** store actual data **b) DataNode** • Stores actual data blocks • Performs read/write operations • Sends heartbeat to NameNode **c) Secondary NameNode** • Periodically checkpoints metadata • Assists in NameNode recovery HDFS splits files into **blocks (128 MB)** and replicates them (default **3 copies**). **2. Yet Another Resource Negotiator (YARN)** YARN manages **cluster resources and job scheduling**. **a) ResourceManager** • Global resource allocation • Schedules applications **b) NodeManager** • Runs on each worker node • Manages containers and resources **c) ApplicationMaster** • Manages execution of a single application • Coordinates with ResourceManager **3. MapReduce** MapReduce is the **data processing model** used by Hadoop. • **Map** → processes input data into key–value pairs • **Reduce** → aggregates results **Hadoop MapReduce Job Execution (Step-by-Step)**	10	CO5	L3

				
9b	Write shortnotes on Apache storm framework Apache Storm is a distributed real-time stream processing framework used to process unbounded data streams with low latency. It uses spouts to ingest data and bolts to process it within a topology. Storm provides fault tolerance, scalability, and language flexibility, making it suitable for real-time analytics and IoT applications. 	10	CO5	L2

10a	Describe how map reduce for batch data analysis with a diagram MapReduce is a batch processing model used in Hadoop for analyzing large volumes of data. The input data is split and processed by map tasks to generate intermediate key–value pairs. These pairs are shuffled, sorted, and passed to reduce tasks for aggregation. The final output is stored back in HDFS. MapReduce is suitable for large-scale, offline batch data analysis. 	10	CO5	L2
10 b	Explain the key component of Hadoop YARN and its job execution framework Hadoop YARN is the resource management layer of Hadoop. The ResourceManager manages cluster resources, while NodeManagers handle resources on individual nodes. For each job, an ApplicationMaster is created to negotiate resources and manage task execution inside containers. This framework enables scalable, efficient, and fault-tolerant job execution.	10	CO5	L2


Hadoop YARN Architecture: Components and Functions
ResourceManager
Scheduler
ApplicationManager
NodeManager
Monitoring Node Health
Container Management
Hadoop YARN Architecture
Containers
Dynamic Allocation
Resource Constraints
ApplicationMaster
Resource Negotiation
Application Execution
Timeline Server
Application History Tracking
Metrics and Logs