INTERNET OF THINGS

R22CS702PC

PART-A

IV-BTech, I Sem, R22 Regular Exams / R22CS702PC

1) a) List the characteristics of IOT.

IOT has key characteristics:

1. Connectivity: devices connect to networks for data exchange
2. sensing: sensors collect real time data from environment
3. Automation & control: IOT systems automatic tasks & remotely and control devices.

b) what is mean by things in IOT?

"Things" are physical objects embedded ē sensors, processors and connectivity to communicate over the internet

ex:- smart bulbs, fitness bands, temperature sensors, smart watches, home appliances.

c) Differentiate b/w IOT and M2M.

Communication: IOT uses IP networks; M2M point-to point or cellular networks.

Scope: IOT connects many heterogenous devices, M2M is many device to device.

Cloud usage: IOT depends on cloud platforms, M2M may work without cloud.

d) what is the need for IOT system management?

1. Device Monitoring: Track device health & performance.
2. security: Apply updates & protect device from attacks.
3. Maintenance: manage configuration, firmware updates, and fault detection.

e) State features of python language.

Easy & Readable: simple syntax makes programming easier.

Interpreted: code executes line-by-line, useful for debugging.

Extensive libraries: support data science, AI, web, development, IOT, etc...

f) what are the control flow statements?

1. conditional: if, elif, else

2. Loop: for, while

3. Jump: break, continue, pass to manage flow.

g) How Raspberry Pi is different from desktop computer?

Size: Raspberry Pi is a credit-card size board; desktop is large.

Hardware Power: Pi has limited CPU/RAM, desktops are more powerful.

Purpose: Pi is used for electronic/IOT projects, desktop is for general use.

h) what is the use of GPIO pin in Raspberry Pi?

- They acts an input pins to read sensors
- they act as output pins to control LED, motors.
- used to build automation & IOT hardware projects.

i) what is Rest API?

1. stateless: Each request is independent

2. client-server: client sends request, sever process them

3. use of HTTP method. GET, POST, PUT, DELETE

j) Define Django template.

1. dynamic webpage: Displays data from views

2. separation of Logic & design: Keeps HTML seperate from Python code.

3. Reusability: Can templates can be reused with blocks.

PART-B

2) Describe various communication models used in IOT.

1. Device - to Device communication model:

This model involves direct communication b/w two IOT devices without requiring a central server or cloud.

Ex:- Bluetooth, Zigbee, NFC.

Features:- Low battery due to short distance communication

2. Device to cloud communication model:

Device connect directly to cloud services using the internet. Data is sent to the cloud for processing, storage and analytics.

Ex:- IOT device uploading data to AWS to IOT.

features: cloud manage data analytics, visualization and decisions.

3. devices to Gateway communication model:

Devices first communicate ē a gateway instead of directly connecting to the cloud. The gateway then forwards the data to cloud services.

Ex:- Smart home hubs like Amazon Echo, Google hub.

features: Gateway perform data aggregation, filtering and initial processing.

4. Gateway to cloud communication model:

The gateway connects multiple IOT devices to the cloud. Devices communicate locally ē the gateway and the gateway communicates ē cloud services

features: supports multiple sensor nodes ē low power.

5. Device to mobile communication model:

IOT devices communicate ē a mobile applications using wi-Fi bluetooth, or cellular networks.

Ex :- fitness trackers connecting to smartphones

features: mobile App acts as user interface & controller

6. machine-to machine communication method:

Device communicate autonomously without human involvement.

Ex:- Industrial machines exchanging production data.

features: uses wired or wireless networks.

7. publish subscribe communication model.

Devices send messages to a broker instead of directly to receivers

Features: Loosely coupled communication

8. Request - Response method:

The client sends a request and the server returns a response

Ex:- REST APIs using HTTP

features:- Simple & widely used on internet

9. Event driven communication method

Device send data only when an event or change occurs

features: Reduces power and bandwidth usage.

10. cloud to cloud communication method

This model involves communication b/w two or more cloud platforms to share data collected from different IOT systems.

features: Allows integration of multiple IOT ecosystems.

3. Briefly explain the technology with play key role in IOT.

The Internet of things depends on several core technologies that enable devices to sense, connect, process and exchange information.

* wireless sensor network:

WSN's consist of distributed sensors that collect environment data such as temperature, pressure, motion or humidity.

Role: Enable real-time monitoring

* RFID:

RFID uses tags and readers to identify objects wirelessly

Role: Automatic tracking of items in logistics & supply chain.

* cloud computing:

cloud platforms store, process, and analyze massive amounts of IOT data.

Role: Provides scalable storage.

* Big data Analytics:

IOT generates huge data streams, which need advanced data analytics tools.

Role: Extract meaningful pattern from huge datasets

* Communication Protocols:

IOT relies on lightweight & efficient communication standards like MQTT, COAP, Zigbee, Bluetooth, WiFi, LTE and 5G.

Role: Ensure reliable data transfer b/w devices

* Embedded systems and microcontrollers:

Devices like Arduino, Raspberry Pi, ESP32, & ARM-based controllers operate sensors and run IOT applications

Role:

* machine learning and AI:

AI techniques help IOT systems make intelligent decisions.

Role: Enables predictive analysis

* Edge and Fog computing:

Instead of sending all data to cloud, processing happens near the device

Role: Reduces latency and improves speed.

* cybersecurity technologies:

IOT devices are vulnerable to attacks, so strong security mechanism are required.

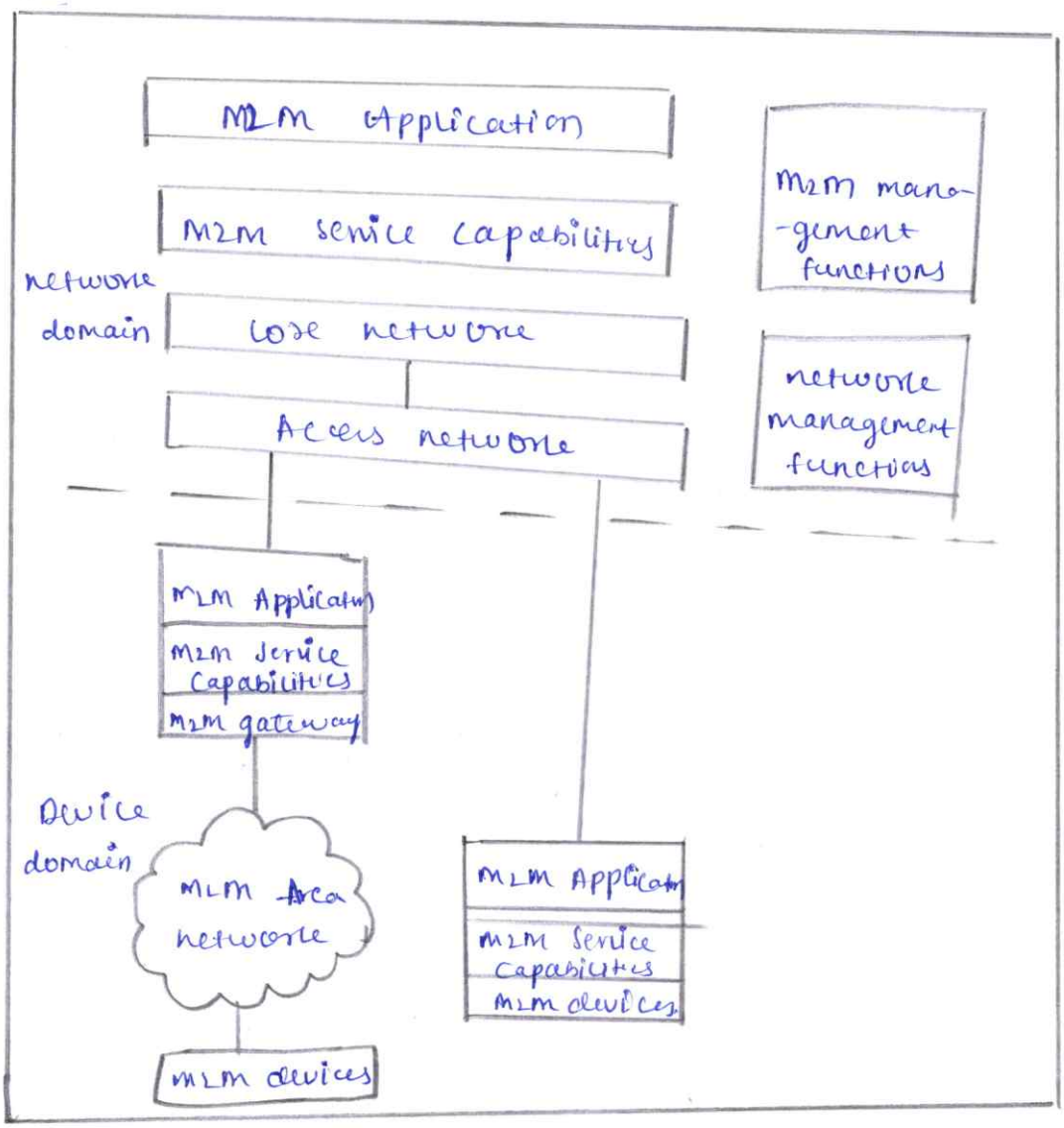
Role: Encryption protect data.

* Mobile and web Application:

App serve as the interface btw users & IOT systems

Role: Allow user to monitor & control IOT devices.

4. Illustrate end-to end architecture of M2M system & neat diagram?



An M2M system enables machine to communicate without human intervention.

1. M2M devices:

- These are physical devices equipped ē sensors & actuators
- They collect data like temp, pressure, location etc...

2- M2M Gateway:

- Device send data to a gateway for further process
- functions: Protocol translation
  data aggregation & filtering
- Examples: IOT hubs, routers, PLCs, Raspberry Pi.

3. Communication network:

- Transfer data b/w gateway & remote servers
- networks include: cellular (3g/4g/5g)
  wi-fi ethernet

4. M2M Service layer:

- Provide middleware such as:
  device management
  data storage & Analytics
- ensure integration ē cloud & external applications.

5. M2M Application layer:

- Provides dashboards, alerts, reports & decision-making tools

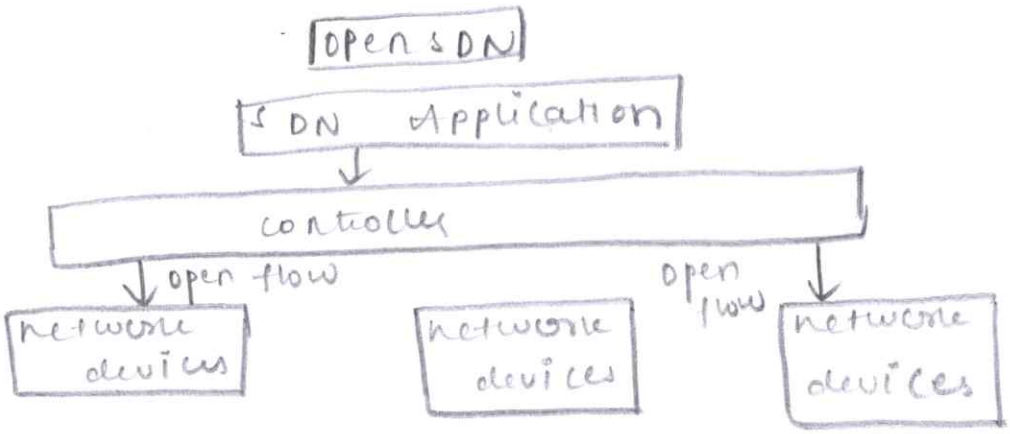
5) Discuss in detail about software define networking for IOT.

Software DN is a modern networking approach where the control of the network is seperated from the physical devices. SDN is widely used in IOT to manage large numbers of devices and ensure efficient, secure communication

1. Concept of SDN:

- control plane: make decisions about data flow
- Data plane: forward data packets.

2. SDN Architecture for IOT:



a) Application layer:
- contains IOT application like smart homes, healthcare industrial to IOT that require network services

b) control layer:
The brain of SDN, it centrally manages the entire network.

c) Data layer:
consists of networking devices such as switches, routers, gateway that forward packets.

3. Need of SDN in IOT:-

IOT has millions of devices that create complex, dynamic networks. helps in.
- centralized control
- Easy management

4. Benefits of SDN in IOT:-

a) Scalability: can handle thousands to millions of IOT devices by dynamically managing network

b) Improved security: SDN controllers monitors traffic flows detect anamolies & apply policies.

c) Reduced complexity: network changes can be made through software instead of devices.

d) Quality of service: ensures priority for important IOT data.

8. improves Readability: code becomes cleaner and easier to understand

6) b) write a python program that prints a multiplication table of a given number.

```
# python program to print the multiplication table of
  a given number
# asking user to enter a number
num = int(input("enter a number to print its
          multiplication table."))
print("\n Multiplaction Table for", num)
print("-----------")
# Printing table from 1 to 10.
for i in range(1,11):
    result = num * i
    print(num, "x", i, "=", result
print("----------")
print("end of Table")
```

Output:

```
Enter a number to print its multiplication table: 7
multiplication table for 7
-------------------
7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63
7 x 10 = 70
-------------------
end of table
```

c) Efficient Routing: SDN selects optimal paths for IOT data, reducing delays.

5. SDN with edge | Fog computing:

Combining SDN & edge computing enhances IOT performance:

- data processed near devices

6. Challenges of SDN in IOT:

- controller becomes a single point of failure
- High security requirements.

6) a) What is python function? write the characteristics of a python function.

Python function: python function is a block of reusable code that performs a specific task. it is defined using the def keyword.

Ex: def add (a, b):
    return a+b.

characteristics of python function:

1. Reusability: A function can be used multiple times in a program.
2. modularity: Breaks large programs into small, manageable parts.
3. parameter support: functions can accept parameters for processing data.
4. Return values: functions can return results using the return statement.
5. default Arguments: python allows default values for parameters.
6. Keyword & variable Arguments: Supports *args and **kwargs.
7. Built-in and user-defined: python has both built-in and custom functions.

7. what is module in python? Explain how you can use modules in your program ē example code.

module in python:

def: A module in python is a file that contains python code such as variables, function & classes. It helps in organizing large programs into smaller, manageable and reusable components.

Ex:- math, random, datetime, or your own custom

need & importance of modules:

1. Reusability: code written once can be reused in multiple programs.
2. Better organization: large program can devided into logical files.
3. Avoid repetition: common functions are stored in modules and imported wherever needed.

Types of python modules:

1. Built-in modules
   These come ē python installation
   Ex:- math, random, os, sys
2. user-defined modules:
   created by programmes
   Ex:- A file named calc.py containing custom functions.

used modules in a program:

1. import module_name.
2. from module_name import function
3. import module_name as alias

Example: using a user-defined module.

step-1: create a module file — my_module.py

```
# my_module.py
def greet(name):
    return f"Hello, {name}! welcome to python modules"
def add(a, b):
    return a+b
```

Step 2: use it in another python program - main.py.

```
# main.py
import my_module
name = input("enter your name:")
print(my_module.greet(name))
x = int(input("enter first number:"))
y = int(input("enter second number:"))
print("sum =", my_module.add(x, y))
```

Output

```
Enter your name: priya
Hello, priya! welcome to python modules
Enter first number: 10
Enter second number: 20
Sum = 30.
```

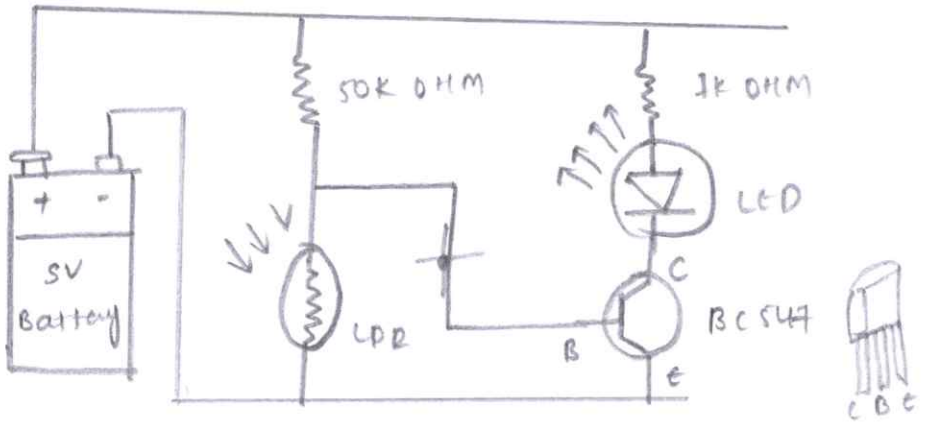
8) Explain the interfacing of a light sensor in Raspberry Pi.

- A LDR is a sensor whose resistance changes based on the intensity of light. when light increases, resistance ↓, when light decreases, resistance increases.

*components Required:

- Raspberry Pi
- LDR
- 10kΩ resistor
- MCP3008 ADC
- Jumper wires
- Breadboard.

* circuit diagram description:



voltage divider:

The LDR is used as a resistor to form a voltage divider:

3.3V --- LDR ---- A0 (MCP3008) ---- 10K Ω ---- GND

MCP3008 to Raspberry Pi connections:

- VDD → 3.3V
- VREF → 3.3V
- AGND - GND
- DGND → GND
- CLK → GPIO11 (SCLK)
- DOUT → GPIO9 (MISO)
- DIN → GPIO10 (MOSI)
- CS/SHDN → GPIO8 (CE0)
- CH0 → LDR voltage divider output.

* Python program to Read LDR values:

```
import spidev
import time

# Create SPI object

spi = spidev.SpiDev()
spi.open(0,0)  # Bus 0, device CE0
```

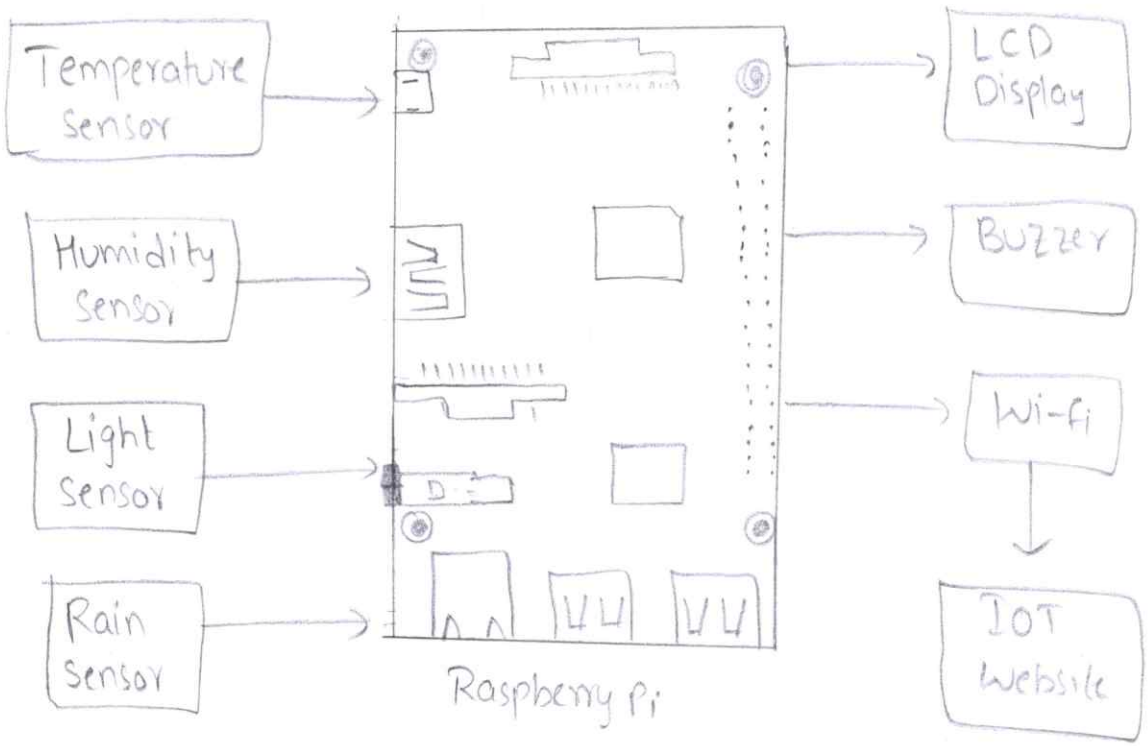
```
spi.max_speed_hz = 1350000

# function to read analog data from MCP3008.
def read_adc(channel):
    adc = spi.xfer2([1|8 + channel)<<4,0])
    data = ((adc[1)&3)<<8) + adc[2]
    return data

while true:
    light_value = read_adc(0) . # Read CH0
    print("Light Intensity Value", light_value)
    if light_value < 300:
    print("Low light - turn on LED")
    else:
    print("Bright light")
        time.sleep(1).
```

Applications:

- Automatic street lights
- smart home lightining
- Light based alarms.

a) write a neat diagram, explain wireless temperature monitoring system using Raspberry Pi.

- A Raspberry Pi wireless temperature monitoring system use a temperature sensor connected to a Raspberry Pi to read data, which is then transmitted wirelessly to a cloud platform or local server, allowing real-time monitoring, data logging and alerts via web browser or apps, creating an IOT solution for smart homes, agriculture or industry.



Components & function:

- temperature sensor: captures real-time temperature. popular choices are the digital.
- Raspberry Pi: The central brain, reads sensor data via GPIO pins, process it, and manages wireless communication
- wireless module - transmits data, wi-fi for internet
- cloud platform / database: stores data for historical analysis and visualization

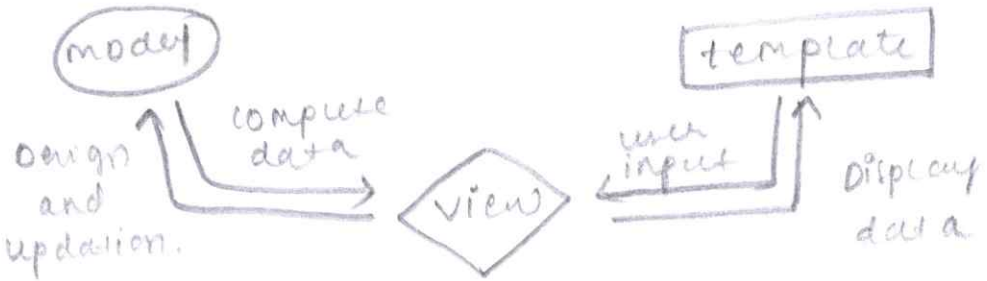
working principle:

- Data Acquisition: The Pi reads digital temperature values from the sensor
- data processing: the Pi process these readings,

Potentially converting analog signals.

- Wireless transmission: data is sent via wi-fi or zigbee to a router / gateway.
- Cloud / Server storage: The gateway pushes data to the cloud or local database.
- Monitoring & Alerts: Users view data on dashboards and can set up email/sms alerts.

10) Describe the architecture of Django application.



Django architecture follows the model view template pattern, a variation of the MVC design pattern.

* model:

- The model represents the data layer of the application.
- It defines the structure of the database, including tables, and relationships typically implemented using Django's object-Relational Mapper.
- It serves as single source of truth for the Applications data.

* view:

- The view contains the business logic of application
- views are implemented as python functions or class-based views that return HTTP responses.
- They acts an intermediary b/w the model and the template

* template:

- The template is responsible for presenting data to the user.
- It combine static content & dynamic data provided by the view.

Flow of Django Application:

1. Client sends HTTP request →
2. URL dispatcher maps it to the appoidate view →
3. view interacts & model to get / update data →
4. view sends data to template →
5. template renders HTML →
6. HTTP response sent back to client.

Features of Django Architecture:

- Seperation of concerns
- Reusabe of components
- Scalability
- security.

11) Explain web Applications frame work and web API 1.1. web Application frame work

Def: A web application framework is a software framework designed to help developers build and manage web application efficently.

features:

1. simplified development
2. MCV / MTV Architecture
3. Security
4. Scalibility
5. Reusable components-

Examples:

- python : Django , flask , pyramid
- Java Script : express, js , node.js , frameworks
- java : spring . struts.

How it works:

- Handles HTTP requests from the client
- Routes Requests to appropriate controllers / views.

2. Web API:

def: A web API is a set of rules and protocols that allows applications to communicate over the internet.

features:

1. interoperability
2. HTTP Methods
3. Data formats
4. Statelessness

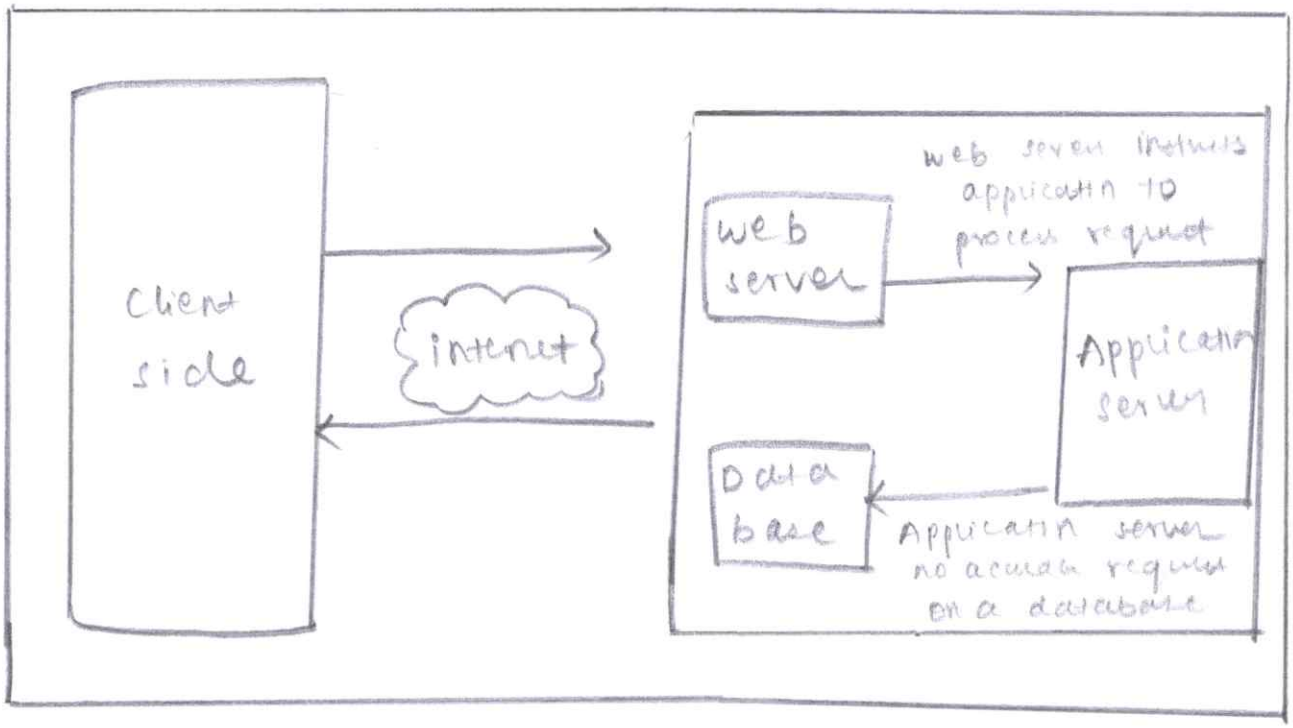
Examples:

- Rest API: Restful services using HTTP
- SOAP API: xml - based messaging protocol.
- GraphQL API: Allows querying only the required data.

How it works:

1. Client sends an HTTP request to the API endpoint
2. Server processes the request and interacts with database/ models
3. Server sends the response back to the client.

Diagram :



use in IOT / web Applications :

- IOT devices can use web APIs to send sensor data to cloud services.
- web applications use APIs to fetch data dynamically.

SCHEME OF EVALUATION

Subject: Internet of Things **Subject Code: R22CS702PC**

Q.No	Definition	Diagram	Explanation	Program	Marks
1.a)	1	-	-	-	1
1.b)	1	-	-	-	1
1.c)	1	-	-	-	1
1.d)	1	-	-	-	1
1.e)	1	-	-	-	1
1.f)	1	-	-	-	1
1.g)	1	-	-	-	1
1.h)	1	-	-	-	1
1.i)	1	-	-	-	1
1.j)	1	-	-	-	1
2.	2	-	8	-	10
3.	-	-	10	-	10
4.	2	-	8	-	10
5.	2	-	8	-	10
6.a)	2	-	3	-	05
6.b)	1	-	-	4	05
7.	2	-	4	4	10
8.	-	-	10	-	10
9.	-	3	7	-	10
10.	2	-	8	-	10
11.	2	2	6	-	10