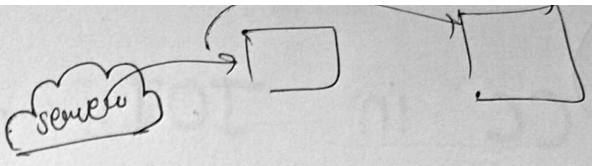
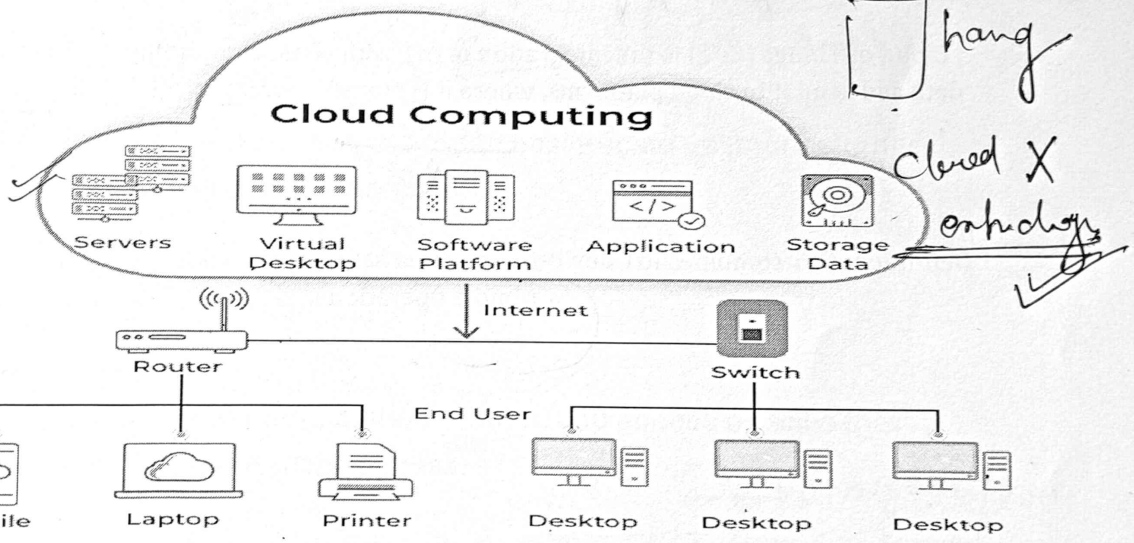


1. Cloud :The cloud refers to a collection of remote servers connected through the internet that store data, run applications, and provide various services.

   Instead of keeping data on your local computer or mobile, you store and access it on the internet.

   Example: Google Drive

2. Cloud Computing :is the technology that delivers computing services—like storage, servers, networking, databases



-Anytime, Anywhere Data can be accessed

-Scalability Storage space can be increased or decreased easily as per requirement.

-Cost-Effective No need to buy physical storage devices

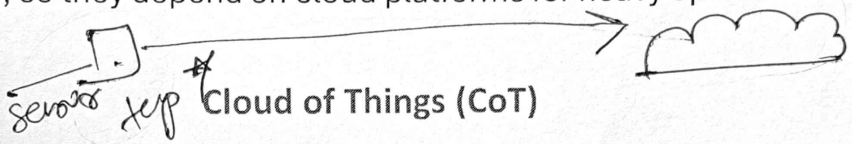
Cloud Computing as an IoT Enabling Technology

CC in IOT.....

Introduction to IoT and Cloud Computing

IoT	Cloud computing
refers to a network of physical devices—sensors, machines, vehicles, appliances that collect and exchange data.	refers to a network of physical devices—sensors, machines, vehicles, appliances—that collect and exchange data.

IoT devices generally have low memory, low processing power, and limited battery, so they depend on cloud platforms for heavy operations.

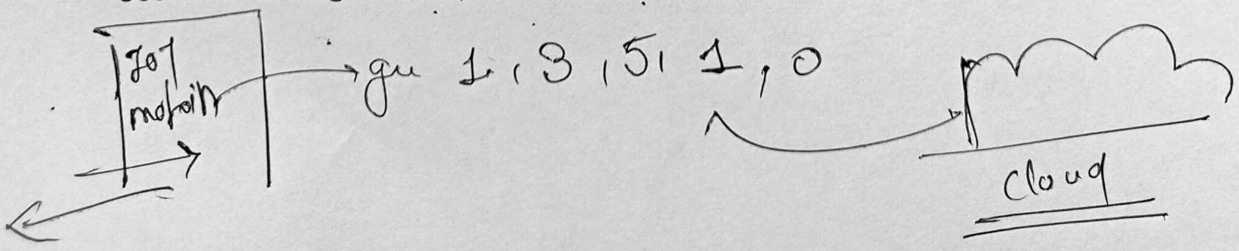


Cloud of Things (CoT) is the integration of IoT with cloud computing. IoT devices collect data and send it to cloud platforms, where it is stored, processed, analyzed, and controlled remotely.

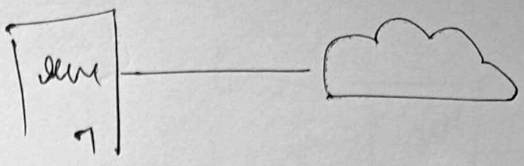
Definition: CoT combines IoT devices with cloud services to enable intelligent, scalable, and remote operations.

a) Data Storage

IoT devices continuously generate data. Cloud storage offers unlimited, scalable storage to keep sensor data, logs, images, and analytics.



b) High Processing Power



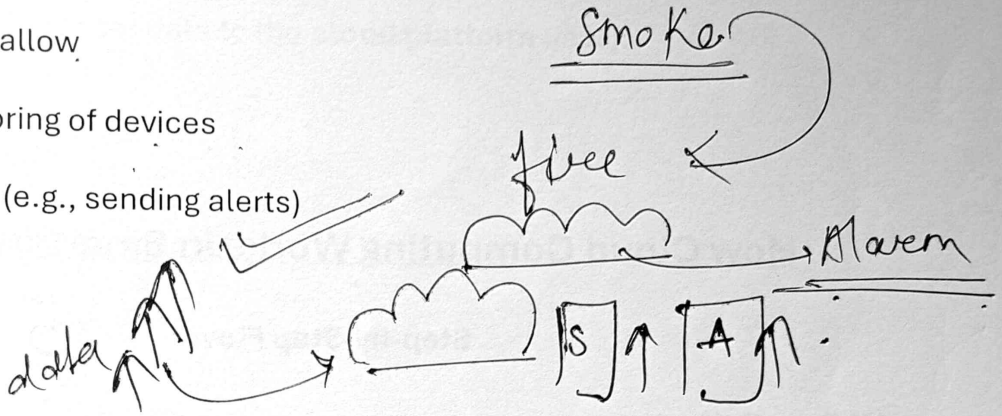
Cloud servers can analyze huge volumes of IoT data (Big Data analytics) that small sensors cannot process themselves.

c) Real-Time Data Access

Cloud platforms allow

real-time monitoring of devices

decision-making (e.g., sending alerts)

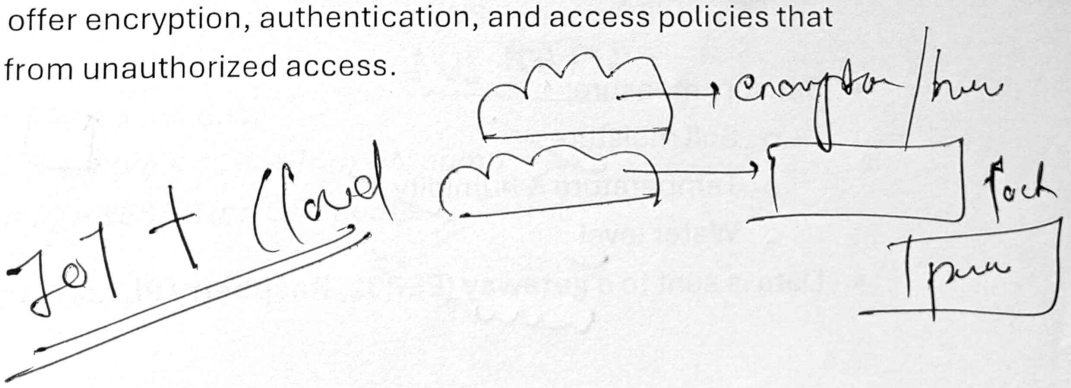


d) Scalability

IoT networks grow rapidly. Cloud services can easily scale up (more storage, more compute power) as devices increase.

f)Security

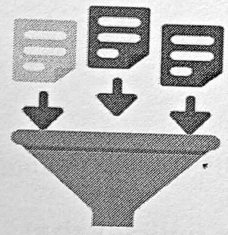
Cloud providers offer encryption, authentication, and access policies that protect IoT data from unauthorized access.

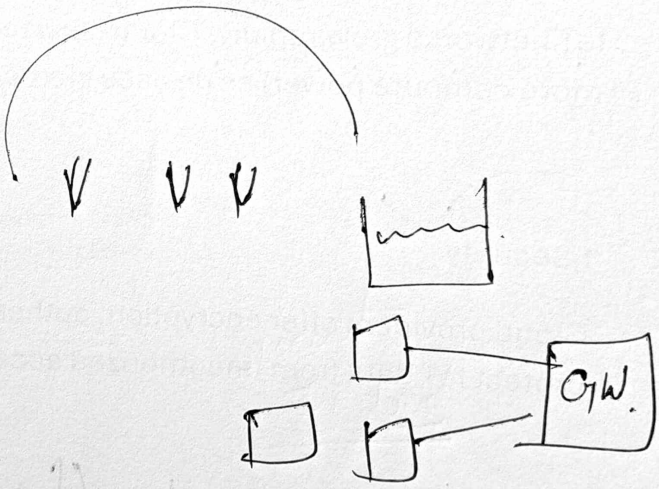


How Cloud Computing Works in Smart Irrigation

Step-by-Step Flow

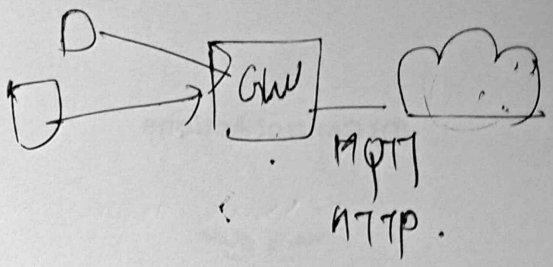
(A) Data Collection





- Sensors measure:
  - Soil moisture
  - Temperature & humidity
  - Water level
- Data is sent to a **gateway (ESP32, Raspberry Pi, LoRa node).**

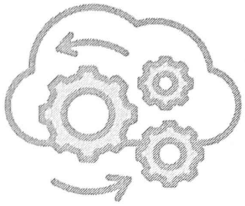
(B) Cloud Upload



The gateway sends sensor data to the **cloud platform** using:

- MQTT
- HTTP/REST API

(C) Cloud Storage & Processing

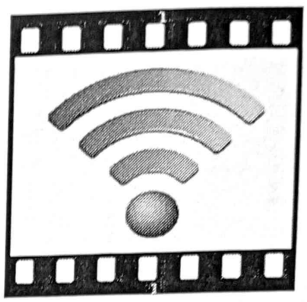


The cloud performs:

1. **Real-time storage**
   – Stores all sensor data in cloud DBs (AWS DynamoDB).
2. **Analytics & Automation**
   – Cloud checks conditions:
   a. *If soil moisture < 40% → Turn ON pump*
   b. *If raining = YES → Turn OFF pump*

(D) Control Actions

Actuators receive the command via:



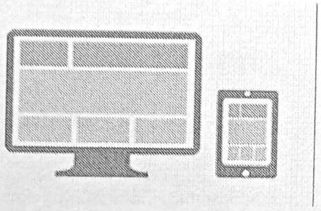
Cloud sends control commands:

Wi-Fi

- Sprinkers ON/OFF
- ON/OFF pump

240V.

(E) User Interface



Through cloud dashboard/mobile app the user can:

- See analytics, water usage history
- Get alerts (SMS/Email)

Smart T Cloud

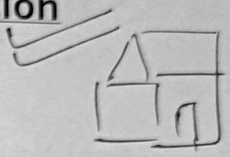
Advantages

Remote Monitoring

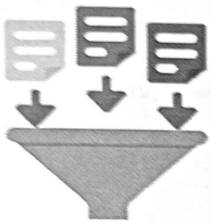
Automation

Big Data Analytics

How Cloud Computing Supports Home Automation



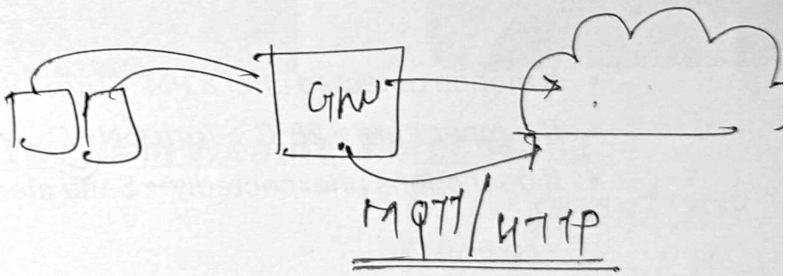
Step 1: Data Collection



-
- Room temperature
- Motion detection
- Door/window status

These devices send data to an **IoT gateway** (Home Hub, ESP32, Alexa Echo, Zigbee Hub).

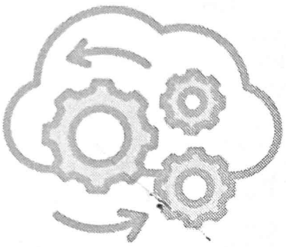
Step 2 Cloud Upload



The gateway transfers collected data to the **cloud** using WiFi or Ethernet through:

- MQTT
- HTTP REST API

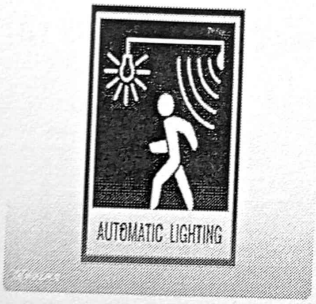
Step 3: Cloud Storage & Processing



Real-time Data Storage

Data saved in cloud databases (AWS DynamoDB).

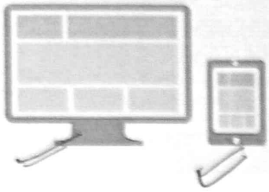
Step 4: Control Actions



Cloud rules:

- *If motion detected after 8 PM → Turn ON lights*
- *If temperature > 28°C → Turn ON AC*
- *If door opens unexpectedly → Send alert*

Step 5: User Interface



Cloud sends:

- Security alerts
- Energy reports

Advantages

Remote Monitoring

Automation

Saves Energy

How Cloud Computing Works in Weather Forecasting

Step 1: Massive Data Collection

Weather data is gathered from:

- Satellites
- IoT weather stations
- Historical weather.

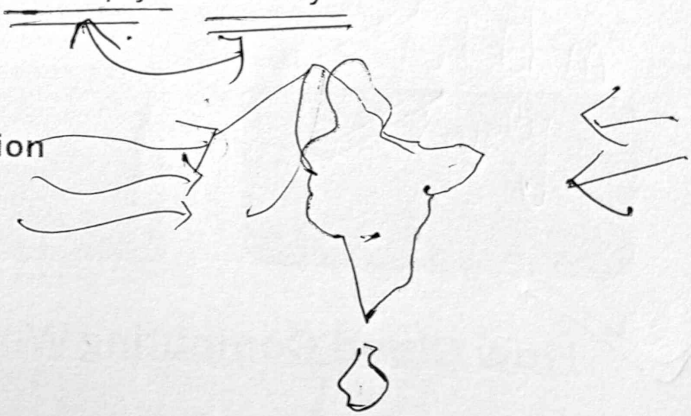
This generates **terabytes of data per day.**

Step 2 Cloud Upload

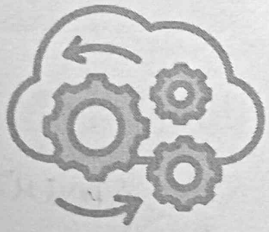
Data is uploaded to cloud storage systems such as:

- AWS S3

Cloud can store extremely large datasets cheaply and safely.



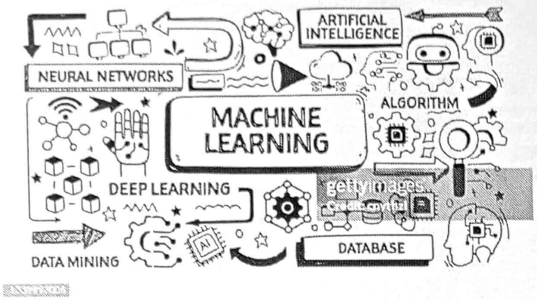
Step 3: Cloud Processing & Computation



Weather prediction requires **complex mathematical models**, such as:

- Numerical Weather Prediction (NWP)
- Cloud AI model analyzes humidity, pressure, wind speed

Step 4: Machine Learning Models



- Temperature trends
- Rainfall patterns
- Storm formation

ML predicts: Model predicts **high chance of rainfall in next 3 hour.**

Step 5: Delivery to Users

Forecast results are displayed on:

- Public portals
- Government alert systems

Cloud also sends:

- SMS alerts
- Flood alerts

Advantages

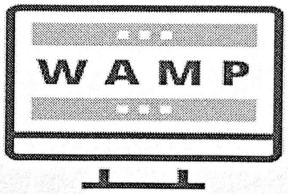
Real-Time Updates

Forecasts are updated every few minutes.

Global Accessibility

Weather data available worldwide through cloud APIs.

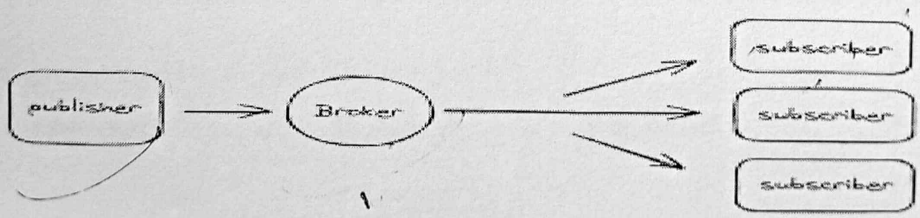
What is WAMP in IoT?



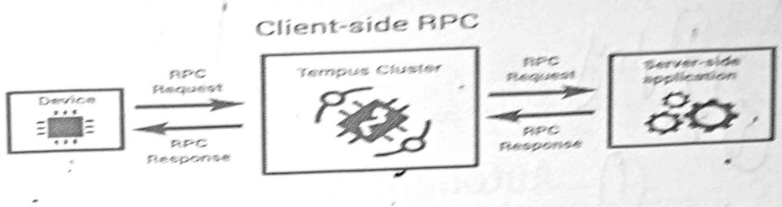
WAMP (Web Application Messaging Protocol)

real-time communication protocol used in IoT

- **Publish–Subscribe (Pub/Sub)**



- **Remote Procedure Call (RPC)**



WAMP

1. Uses Two Communication Patterns

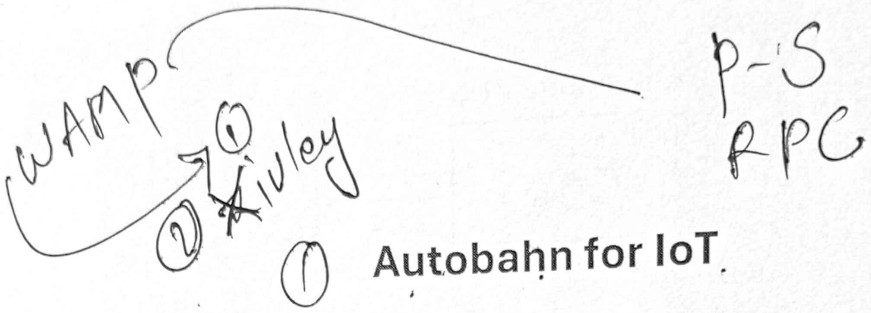
- **Publish–Subscribe**
  For broadcasting sensor data to many subscribers.
- **RPC (Remote Procedure Call)**
  For sending commands to IoT devices (like turning ON a ~~relay~~).

2. Runs Over WebSocket's

- Enables **real-time** messaging
- Both client and server can send messages anytime
- No need to refresh or poll

> WAMP is a WebSocket-based messaging protocol used in IoT to support real-time communication using Publish–Subscribe and RPC methods. It allows sensors, devices, and applications to exchange data instantly and control IoT devices efficiently.

> WAMP is a WebSocket-based messaging protocol used in IoT to support real-time communication using Publish–Subscribe and RPC methods. It allows sensors, devices, and applications to exchange data instantly and control IoT devices efficiently.



Autobahn for IoT

Autobahn is an open-source framework that implements the WAMP protocol for building real-time IoT applications using WebSockets.

1. Implements WAMP Protocol (Pub/Sub + RPC)

- Publish–Subscribe for data
- RPC for device control
- **Example:**
  Temperature sensor publishes readings → The dashboard subscribes.
  RPC "Set_Fan_Speed(3)" is sent → Fan adjusts speed.

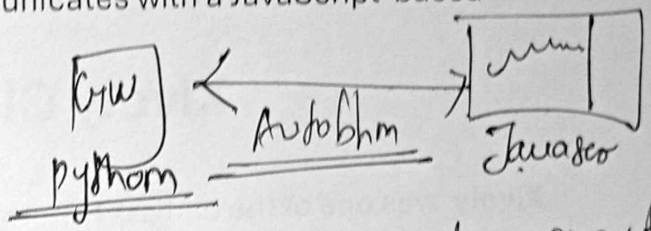
2. Real-Time Communication

- Autobahn uses WebSockets for instant messages.
- **Example:**
  A security camera detects motion → Alert instantly appears on the user's mobile app.

3. Multi-Platform Support

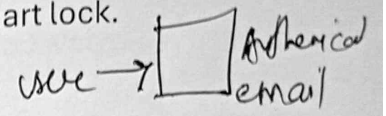
- Available for Python, JavaScript, Android, C++,

- **Example:**
  A Python-based IoT gateway communicates with a JavaScript-based web dashboard using Autobahn.



4. Secure Communication

- Supports encryption , authentication, and permissions.
- **Example:**
  Only authorized users can send RPC "Unlock Door" to a smart lock.

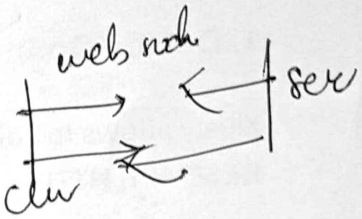


5. Scalable for Large IoT Networks

- Can handle many devices and message flows.
- **Example:**
  A city-wide smart street lighting system with 10,000 poles uses Autobahn+Crossbar for switching lights remotely.

6. Very Reliable for Industrial IoT

- Handles continuous communication without failures.
- **Example:**
  Factory machines continuously publish motor vibration data; Autobahn ensures no data loss.



Xively Cloud for IoT

Xively was one of the earliest **IoT cloud platforms** designed to help businesses connect, manage, and analyze their IoT devices.

Google Cloud

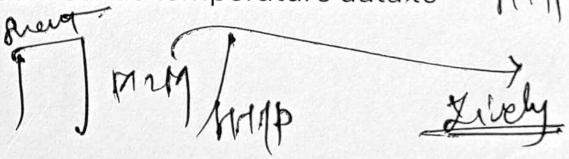
It was provided by **LogMeIn** and later acquired by **Google** for integration with Google Cloud IoT.

1. Device Connectivity

Xively allows IoT devices to connect to the cloud using protocols like **MQTT, REST API, HTTP**.

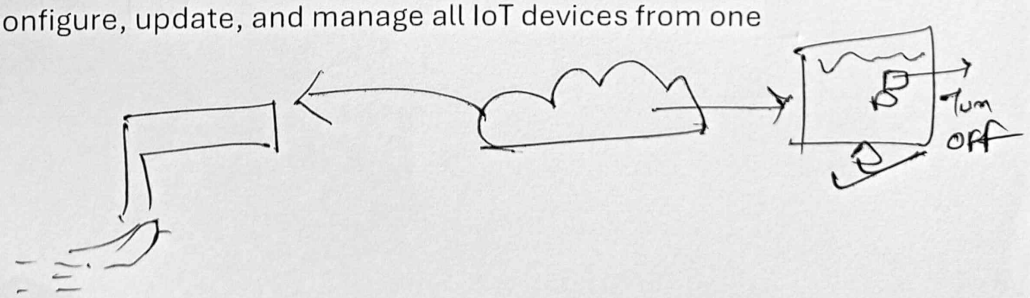
Example:

A **temperature sensor** in a cold-storage room sends temperature data to Xively using MQTT.



2. Device Management

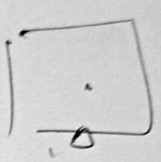
You can monitor, configure, update, and manage all IoT devices from one dashboard.



Example:

You update the **firmware of 500 smart meters** remotely through the Xively dashboard.

3. Data Storage & Real-Time Data Streaming

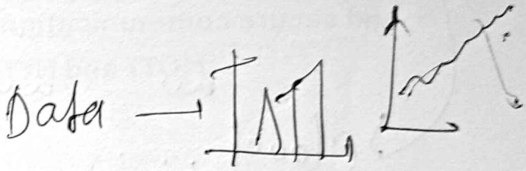


Xively stores live device data and allows real-time monitoring.

Example:

A **smart water flow sensor** continuously streams flow rate data to Xively, which updates the dashboard instantly.

4. Data Analytics & Visualization



Xively provides charts, graphs, and analytics tools to interpret IoT data.

Example:

A **smart agriculture system** shows a graph of soil moisture levels over a week to help farmers plan irrigation.

5. Security

Xively ensures device authentication, data encryption, and secure communication.

Example:

A **smart home device** registers with a secure API key so no unknown device can connect.

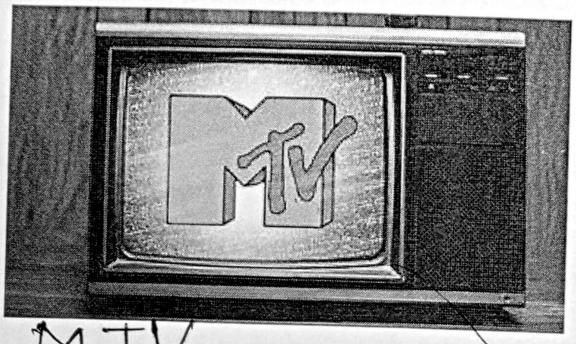
7. Scalability

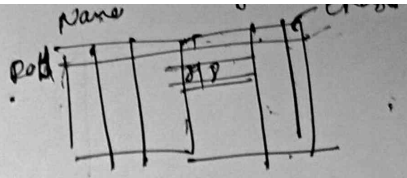
It supports thousands or millions of IoT devices.

Example:
A smart city project connects **10,000 streetlights**, and Xively handles all their data smoothly.

Xively Cloud for IoT is a cloud-based IoT platform that provides device connectivity, device management, real-time data streaming, analytics, and secure communication for IoT devices. It uses APIs and protocols like MQTT and HTTP to collect and manage IoT data.

Django's MTV (Model–Template–View) architecture





1. Model (M) – Handles the Data

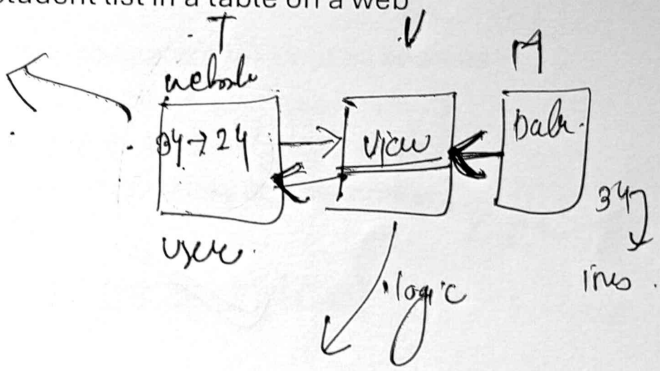
Think of **Model** as the place where all the information about your data lives.

- It decides **what data you want to store**, like name, email, price, etc.
- It controls **how the data is saved in the database**
  how it is updated.
- Example: Imagine a **Student** model. It stores details like student name, roll number, marks, etc.

2. Template (T) – Handles the Display

Template is the **front-end part** that controls how the web page looks.

- It decides **what the user will see** on the screen.
- It contains HTML code with placeholders to show data.
- Example: A template may show the student list in a table on a web page.

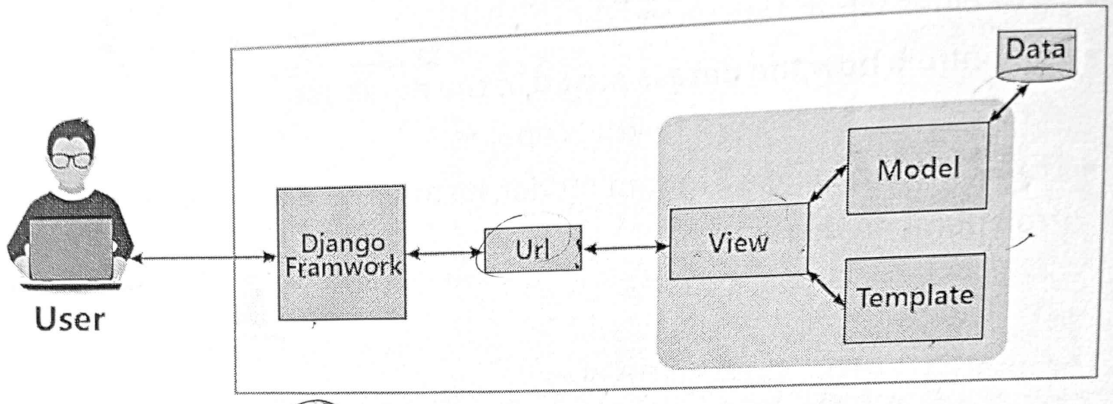


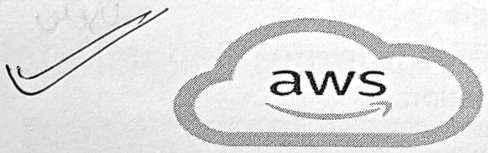
3. View (V) – Handles the Logic

View is the brain of the application.

- It takes a user request, fetches the needed data from the Model, and sends it to the Template.
- It controls **what should happen** when a user clicks a button or opens a page.
- Example: A view fetches all student records from the model and sends them to a template to display.

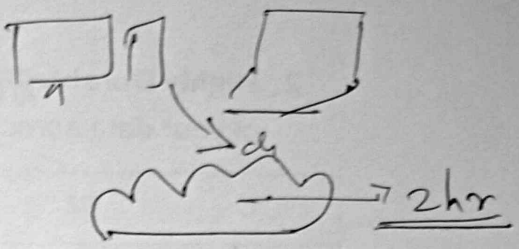
Control Flow Of MVT





1.Amazon EC2 (Elastic Compute Cloud)

EC2 provides virtual machines in the cloud where you can run applications, software, and IoT processing logic.



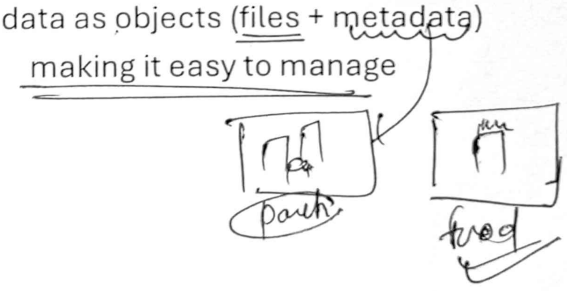
1. **Virtual Server on Cloud:** EC2 gives you a computer in the cloud, which behaves exactly like a physical computer but runs on Amazon's data centers.
2. **Customizable Hardware:** You can select how much RAM, CPU, and storage your IoT application needs.
3. **On-Demand Billing:** You pay only for the time your EC2 instance is running. This saves cost.
4. **Real-Time Data Processing:** EC2 is used to analyze IoT data as soon as it comes from sensors—like detecting temperature changes instantly.
5. **High Scalability & Control:** You can add more EC2 instances when your IoT traffic increases and remove them during low activity times.

2. Amazon S3 (Simple Storage Service)

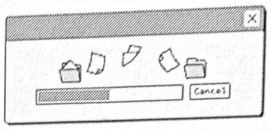
S3 is used to store and retrieve large volumes of data from anywhere. It is widely used in IoT because sensors continuously generate data.

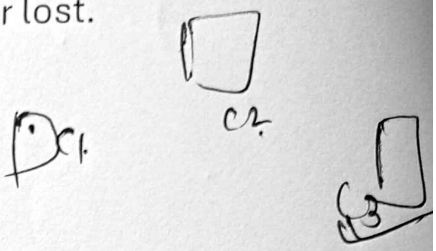
5 Detailed Points

1. **Object-Based Storage:** S3 stores data as objects (files + metadata) making it easy to manage

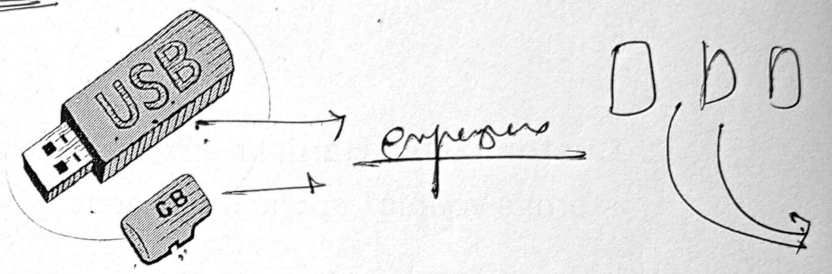


2. **Highly Durable Storage (99.999999999%):** AWS stores multiple copies of your data across many servers, ensuring it is never lost.



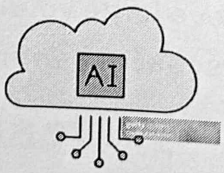


3. **Low-Cost Long-Term Storage:** IoT projects that generate years of historical data find S3 cheaper than traditional storage.



4. **Accessible Anytime:** Data stored in S3 can be accessed from IoT devices, mobile apps, or analytics platforms using simple APIs.

5. Supports Analytics & AI: IoT data saved in S3 can easily be connected to AI, machine learning tools.



Difference Between EC2 and S3

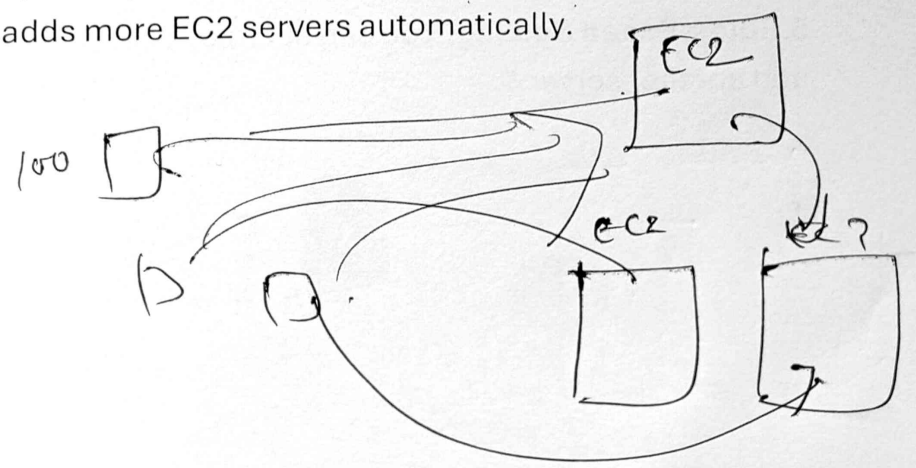
EC2	S3
EC2 is a virtual computer	S3 is cloud storage
Used to run applications	Used to store data/files
Good for processing IoT data	Good for storing IoT history
Example: Running IoT logic	Example: Storing sensor logs

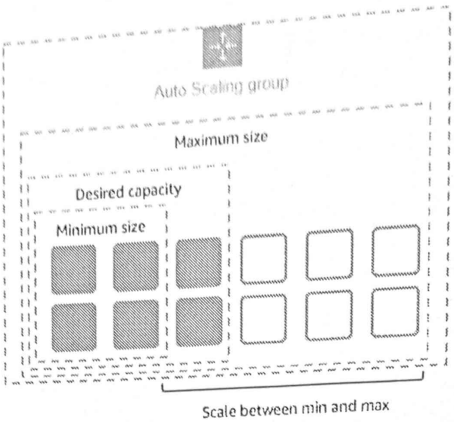
3. Amazon Auto Scaling

EC2 → virtual machine
S3 → storage

Automatically adjusts the number of EC2 instances based on the workload.

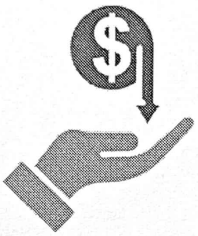
1. **Automatic Adjustment:** When many IoT devices send data at once, Auto Scaling adds more EC2 servers automatically.



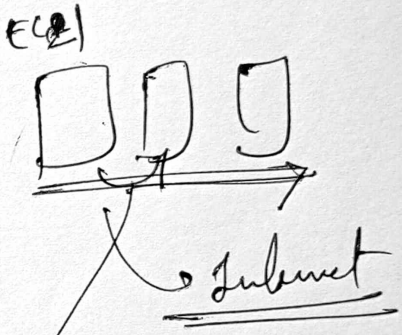


1 EC2

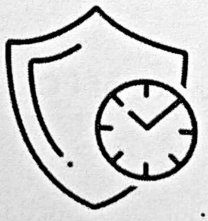
2. **Handles Sudden Traffic Spikes:** IoT systems often receive a sudden burst of data; Auto Scaling prevents server overload.
3. **Cost Reduction:** When sensor activity reduces, Auto Scaling shuts down unnecessary servers, saving cost.

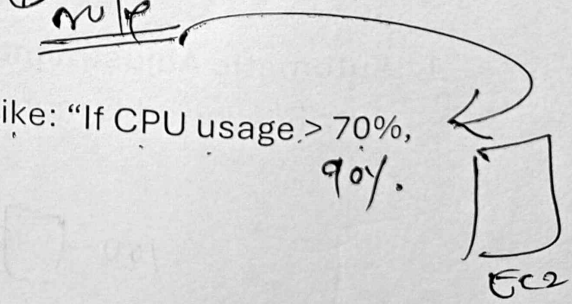


2 4/7

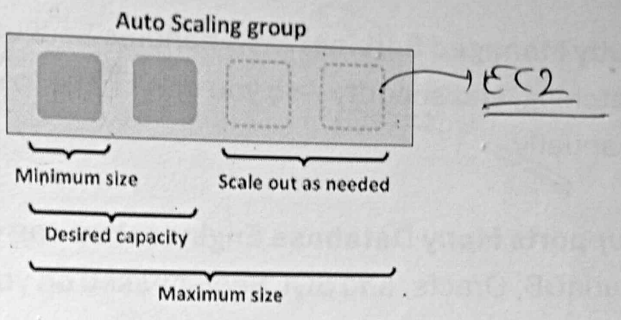


4.Improved Application Reliability: Keeps IoT apps stable by ensuring enough computing power is always available





5.Rules-Based Scaling: You can set rules like: "If CPU usage > 70%, add another server."



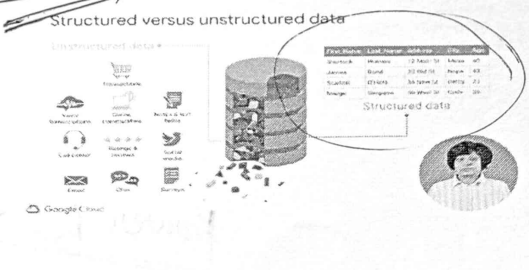
Example:

During a festival, a city's IoT traffic cameras generate high data → Auto Scaling increases servers → System stays fast.

4. Amazon RDS (Relational Database Service)

RDS is a managed relational (SQL) database for IoT systems where data is structured in tables.

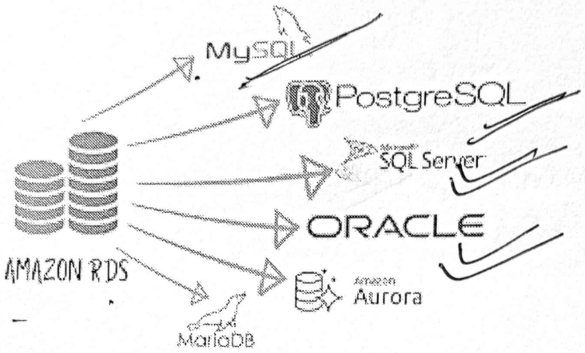
1. **Structured Storage:** Stores IoT data like sensor ID, timestamp, readings in table formats.



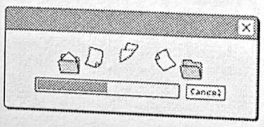
RDS

2. **Fully Managed Service:** AWS handles backups, maintenance, patching, and security—so you don't have to manage the database manually.

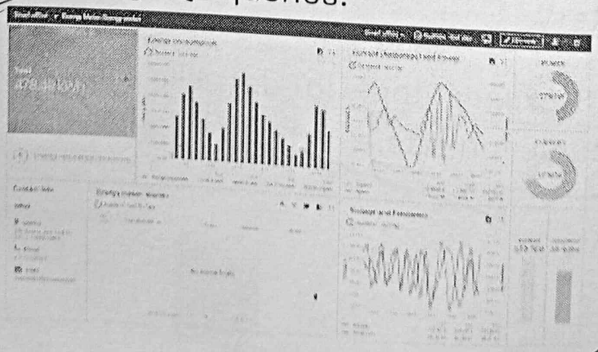
3. **Supports Many Database Engines:** You can use MySQL, PostgreSQL, MariaDB, Oracle, and SQL Server based on your IoT application need.



4. **High Availability:** RDS automatically creates standby databases to ensure the system is always running.



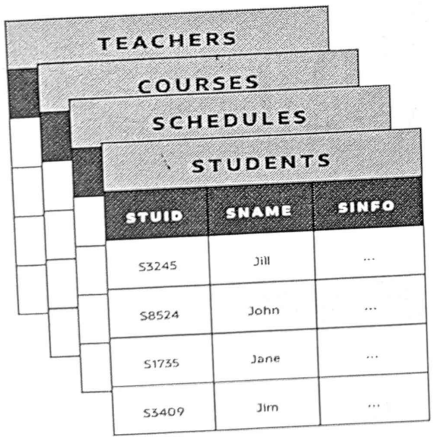
5. **Easy Integration:** IoT apps, dashboards, and analytics tools can easily fetch data from RDS using SQL queries.



5. Amazon MongoDB

It is a managed NoSQL database for flexible and fast IoT data storage.

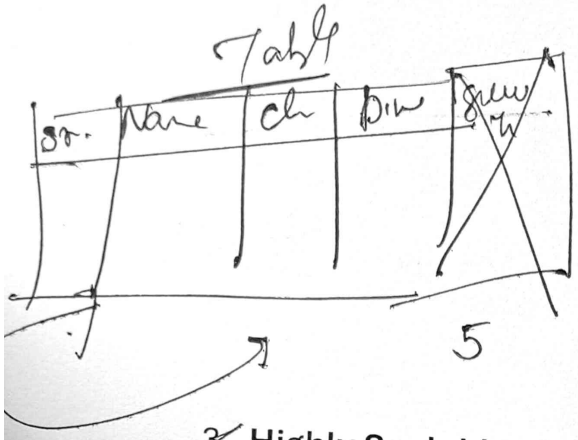
1. **Document-Oriented Storage:** Stores data in JSON-like documents, making it perfect for IoT events.

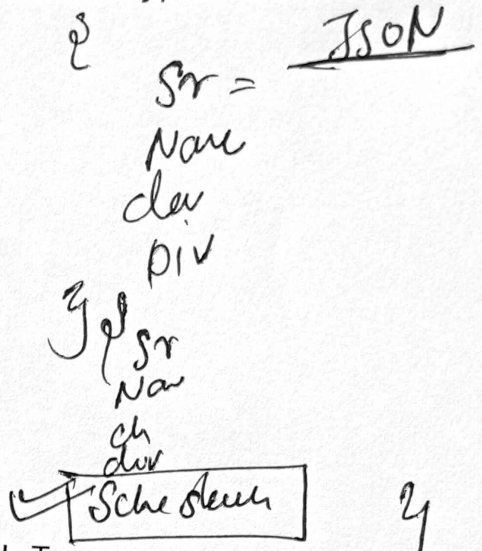


A Single JSON Document

```
{
 student     : "Jill",
 schedule    :
   [  {
            time    : "14:00",
            course  : "Math 101",
            room    : "A102",
            teacher : "Adam"
      },
      {
            time    : "16:00",
            course  : "Science 102",
            room    : "B405",
            teacher : "Anita"
      }
   ]
}
```

2. **Flexible Schema:** Devices sending different types of data do not require fixed table formats.





3. **Highly Scalable:** Can store millions of IoT events per hour without performance issues.

4. **Fast Real-Time Reads/Writes:** Ideal for IoT applications like smart homes where data changes rapidly.

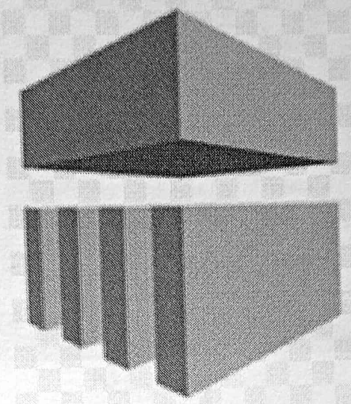
5. **MongoDB Compatibility:** Existing MongoDB applications can run without code change.

Example:

Saving smart home activity logs such as “motion detected”, “fan turned on”, “door opened”.

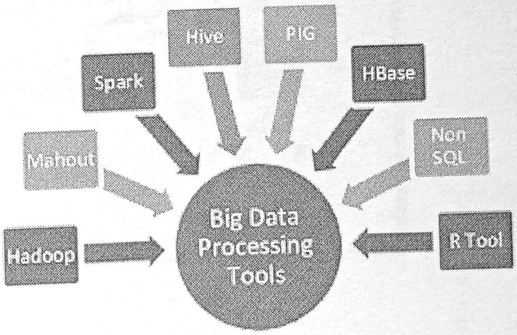
5. Amazon EMR (Elastic MapReduce)

EMR is a cloud big data processing service used for analyzing large IoT datasets.



Amazon EMR

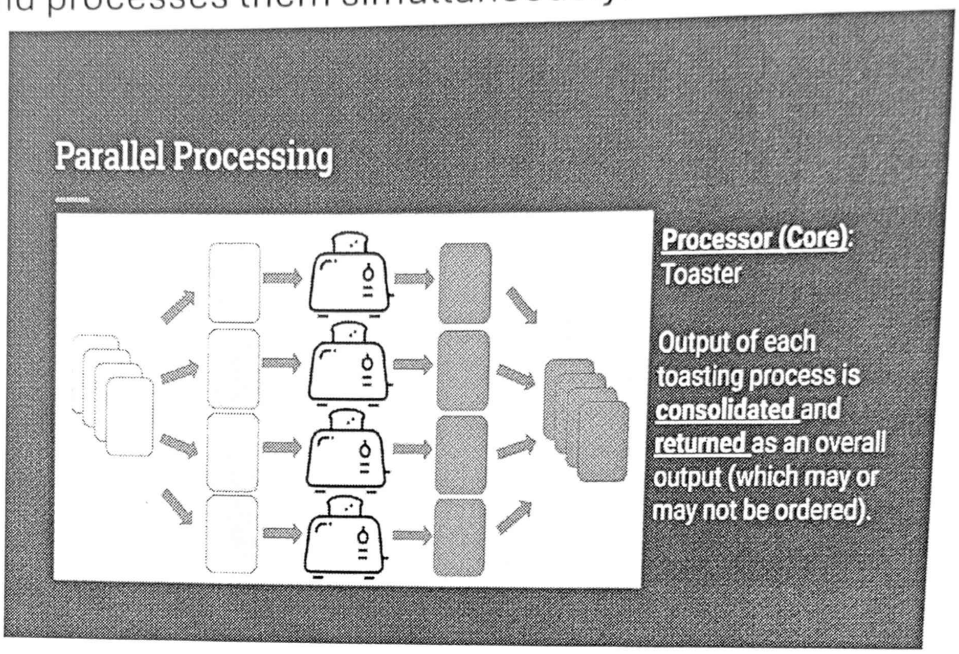
1. **Big Data Processing:** IoT devices generate huge volumes of data; EMR can handle terabytes or petabytes.



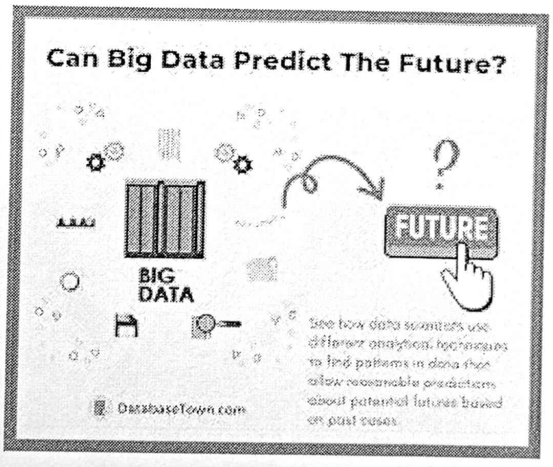
2. **Uses Tools Like Hadoop & Spark:** These tools can do batch processing, filtering, and machine learning on IoT datasets.

3. **Cost-Effective for Big Data:** Instead of setting up expensive servers, EMR processes data on demand.

4. **Fast Parallel Processing:** Breaks large IoT datasets into smaller chunks and processes them simultaneously.



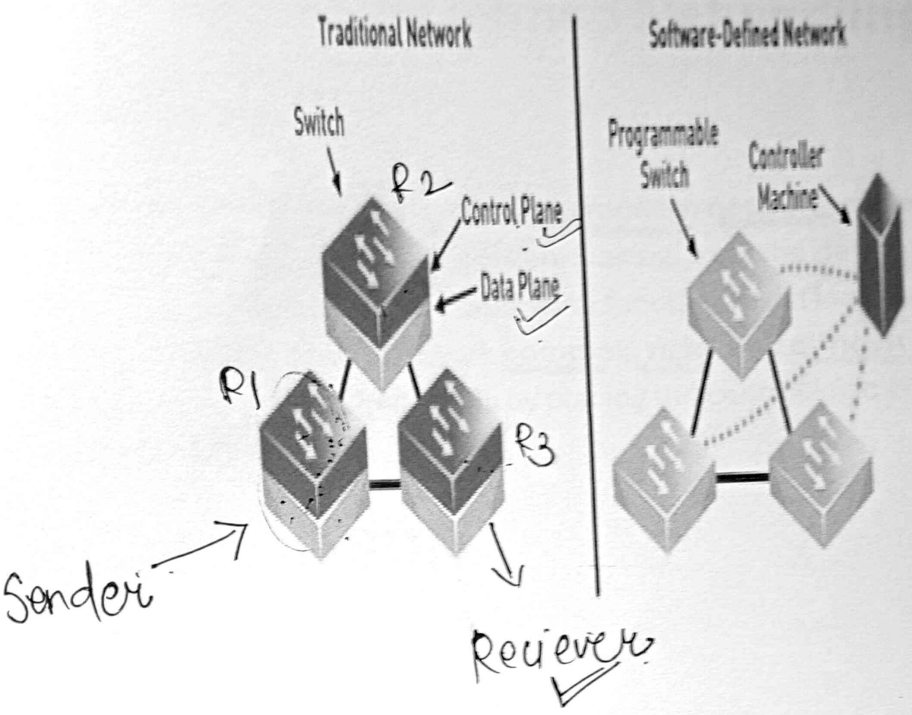
5. **Supports Machine Learning:** You can predict trends (like traffic patterns, weather changes) using ML models on EMR.

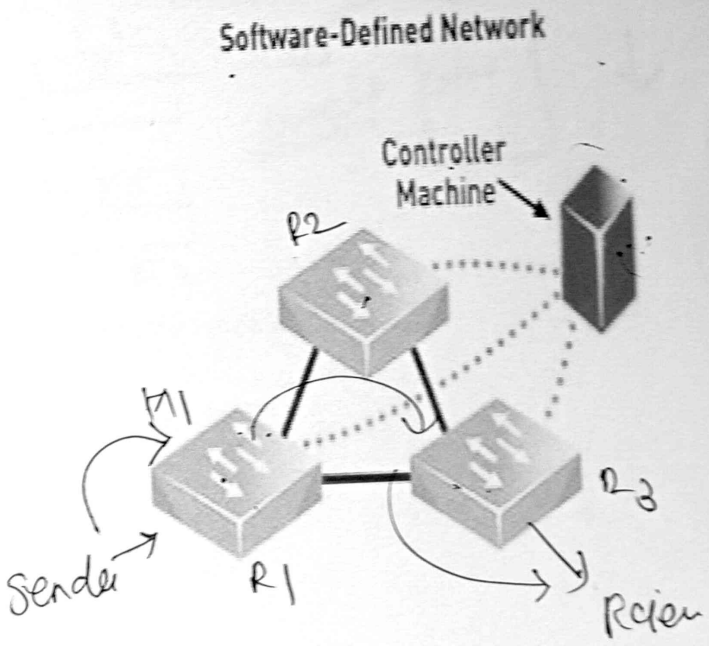


Example:

Analyzing 1 year of city air pollution sensor data to identify pollution peak hours.

Traditional Network VS SDN





SDN – Software Defined Networking

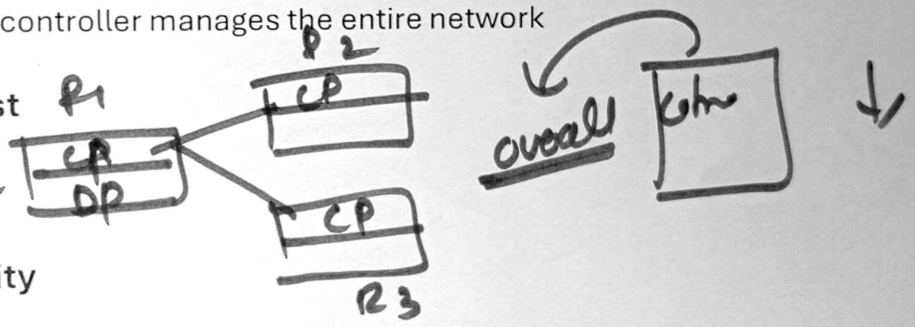
1. Introduction

Software Defined Networking (SDN) is a **modern networking paradigm** that separates the **control plane** (decision-making) from the **data plane** (packet forwarding). In traditional networks, both functions exist inside each router or switch, which makes the network **complex, rigid, and difficult to manage**. SDN overcomes these limitations by placing the control logic in a **centralized software controller.**

Benefits of SDN

1. Centralized Control

-Single SDN controller manages the entire network



5. Reduced Cost

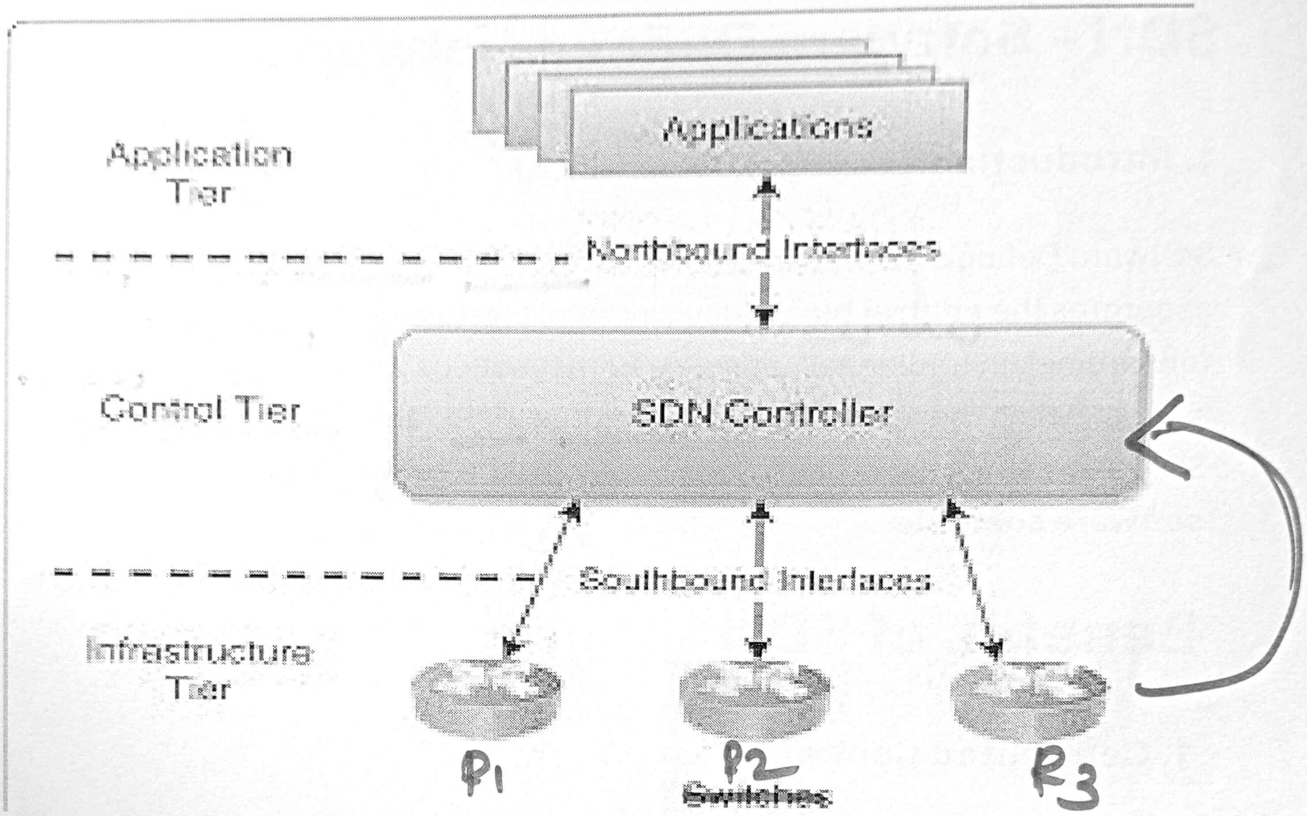
- Router

6. Better Security

- Controller can detect attacks and update rules instantly.

SDN Architecture

SDN Arc.



i) Data Plane (Infrastructure Layer / Forwarding Layer)

The Data Plane consists of all the **physical and virtual network devices** such as:

- Switches

- Routers
  SDN switches in the data plane are **dumb but fast**. Their only job is to forward packets according to rules learned from the controller

- Perform actions like **forward, drop, modify, or send to controller**.
- Collect statistics and send them back to the controller.

(ii) Control Layer (Control Plane / SDN Controller)

This is the **central layer** and acts as the **brain of the entire SDN architecture**. The SDN controller is a software platform located logically in the center of the network and manages all forwarding devices.

Examples of Popular Controllers:

Cisco APIC

It coordinates between the applications and the physical devices.

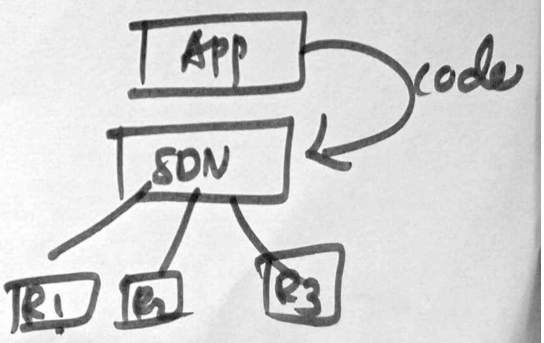
(iii) Application Layer (Top Layer)

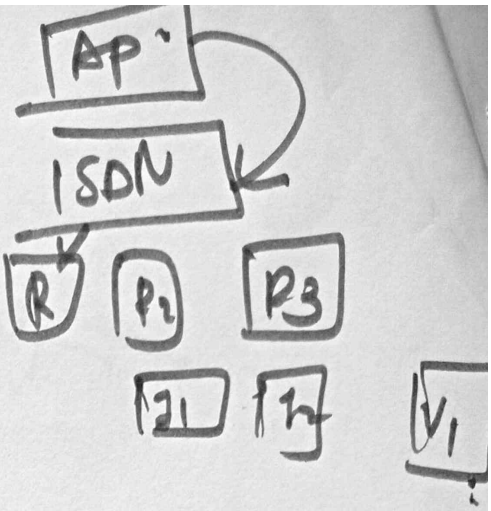
The Application Layer is the topmost layer in the SDN architecture.

Contains various **network applications, services, and business logic**.

In this layer, network engineers do not need to configure each switch or router manually. Instead,

they **write software programs** that tell the SDN controller what the network should do





The applications Layer tell,

giving priority to video traffic,

blocking certain IP addresses,

detecting intrusions.