E3276 (Internet of Things)

February 2022

Question 1

a. List any SIX (6) advantages of the Iot systems.

Ans:- Here are six advantages of IoT systems:-

1. Improved Efficiency :- IoT systems can automate tasks, collect data in real-time, and provide insights that improve operational efficiency. For example, predictive maintenance in industrial IoT can reduce downtime by anticipating equipment failures.

2. Enhanced Decision Making :- With access to vast amounts of data collected by IoT devices, businesses can make data-driven decisions faster and more accurately. This can lead to optimized processes, better resource allocation, and improved outcomes.

3. Cost Savings :- By automating processes, reducing manual intervention, and optimizing resource usage, IoT systems can lead to cost savings in various areas such as energy consumption, maintenance, and logistics.

4. Increased Productivity :- IoT devices can monitor equipment performance, track inventory, and streamline workflows, leading to increased productivity. For example, in agriculture, IoT sensors can monitor soil conditions and automate irrigation, leading to higher crop yields.

5. Enhanced Customer Experience :- IoT enables businesses to provide personalized and proactive services to customers. For example, IoT-connected wearable devices can track health metrics and provide personalized recommendations for fitness and healthcare.

6. Safety and Security :- IoT systems can enhance safety and security by monitoring environments, detecting anomalies, and providing alerts in real-time. For example, smart home security systems can detect intrusions and notify homeowners immediately.

b. Describe the IOT level 2 system.

Ans:- In a Level-2 IoT system with a single node performing sensing, actuation, and local analysis, the architecture typically consists of the following components and characteristics:-

1. Single Node :- This node, often referred to as an IoT device or sensor node, is responsible for collecting data from sensors, performing local analysis, and controlling actuators based on the analysis results. It may have limited computational resources compared to higher-level nodes or servers.

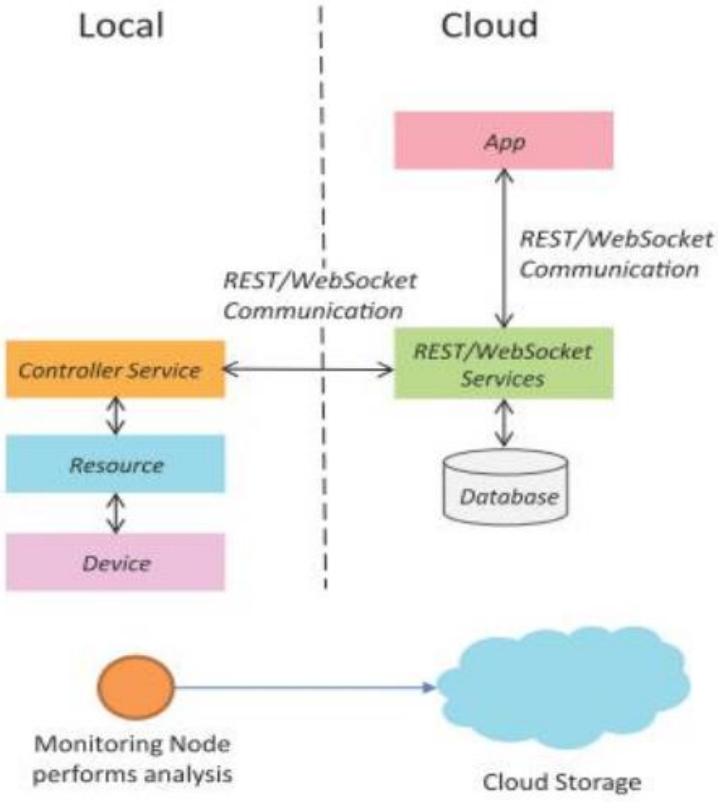
2. Sensing and Actuation :- The node collects data from various sensors, such as temperature sensors, humidity sensors, motion sensors, etc. It also controls actuators, such as motors, valves, or switches, to perform actions based on the analysis of sensor data.

3. Local Analysis :- The node processes the collected sensor data locally using algorithms or models preloaded onto the device. The analysis may involve basic data filtering, aggregation, or simple decision-making based on predefined rules. Since the analysis is performed locally, it reduces the need for frequent data transmission to external servers, saving bandwidth and latency.

4. Cloud Storage :- The processed data or relevant insights are transmitted to the cloud for storage. Cloud storage offers scalability, reliability, and accessibility, enabling efficient storage of large volumes of data generated by IoT devices over time. It also facilitates data sharing and integration with other cloud-based applications or services.

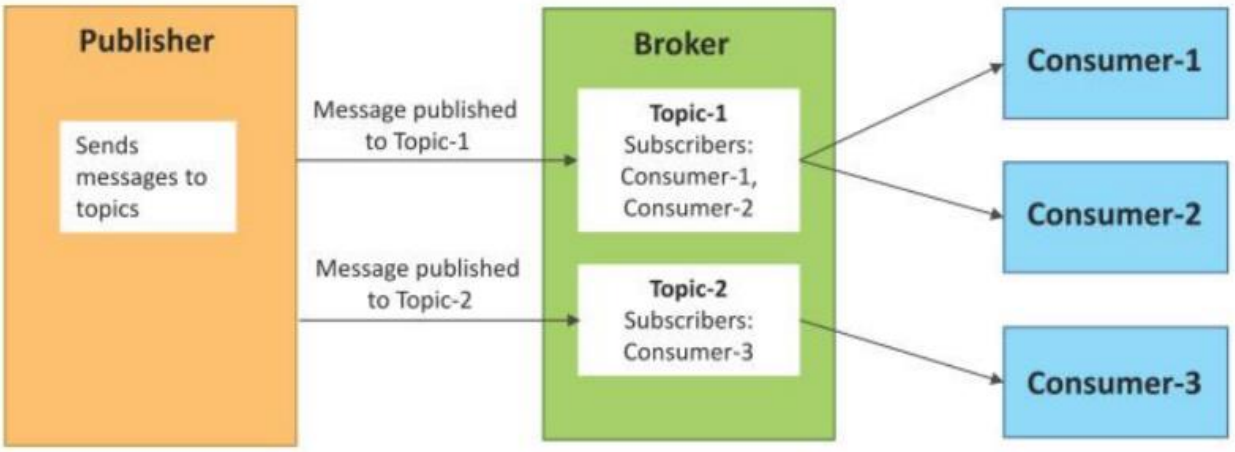
5. Cloud-Based Application :- The application layer, hosted in the cloud, interacts with the stored data to provide visualization, monitoring, control, and further analysis capabilities to users or other connected systems. It leverages the stored data to derive insights, generate reports, trigger alerts, or facilitate decision-making processes.

6. Suitability for Big Data :- Level-2 IoT systems are suitable for solutions dealing with large volumes of data (big data), as they can efficiently collect, process, and store data both locally and in the cloud. However, the primary analysis requirement is not computationally intensive and can be handled by the limited computational resources available at the node itself, reducing the need for powerful processing capabilities in the cloud.



C. With an appropriate diagram, explain the Publish-Subscribe communication model in the lot system.

Ans :- In the Publish-Subscribe communication model in an IoT system, there are three main entities: publishers, subscribers, and a broker.



Publishers :- These are the devices or sensors that generate data. They publish data to the broker at predefined intervals. In an IoT context, sensors can be considered as publishers. Data is published by publishers as topics, which represent specific types of information or events.

Broker :- The broker is an intermediary entity that receives and maintains the published messages from the publishers. It manages different topics to which subscribers can subscribe. The broker ensures that messages from publishers are efficiently delivered to the appropriate subscribers. It acts as a central hub for message routing and distribution. In most cases, the broker is implemented as a server.

Subscribers :- Subscribers are the consumers of the data published by publishers. They subscribe to specific topics of interest maintained by the broker. Subscribers receive data relevant to the topics they have subscribed to. In the context of IoT applications, subscribers are typically the IoT applications or systems through which users interact. A subscriber can subscribe to one or more topics managed by the broker, depending on its requirements.

Question 2

a. Explain any THREE (3) types of IoT sensors.

Ans :- A sensor is a device that detects physical changes or conditions in its environment and converts them into electrical, optical, or digital signals for processing and analysis.

1. Temperature Sensors :- Temperature sensors measure the ambient temperature of their surroundings. They can provide real-time temperature data, allowing for monitoring and control of temperature-sensitive environments.

Applications:- Temperature sensors are widely used in industries such as agriculture for monitoring crop conditions, in HVAC systems for climate control, in food storage facilities to ensure proper storage conditions, and in wearable devices for monitoring body temperature.

2. Gas Sensors :- Gas sensors detect the presence and concentration of various gases in the environment. They are essential for monitoring air quality, detecting hazardous gases, and ensuring safety in industrial and domestic settings.

Applications :- Gas sensors are used in industries such as manufacturing for detecting leaks of toxic or flammable gases, in environmental monitoring for detecting pollutants in the air, in smart homes for detecting carbon monoxide and natural gas leaks, and in healthcare for monitoring patient gas levels.

3. Soil Moisture Sensors :- Soil moisture sensors measure the moisture content of the soil. They provide data on soil moisture levels, which is crucial for efficient irrigation management and optimizing plant growth.

Applications :- Soil moisture sensors find applications in agriculture for optimizing irrigation schedules and preventing overwatering or underwatering of crops, in landscaping for maintaining healthy gardens, in golf courses for efficient turf management, and in environmental monitoring for assessing soil health and erosion control.

b. Describe any THREE (3) architectures difference between the M2M system and the IOT system.

Ans:- Architectural Differences Between M2M and IoT Systems :-

M2M System	**IOT System**
M2M systems typically have a centralized architecture where devices communicate directly with a central server or control unit. Communication is usually point-to-point between devices and the central server	IoT systems often have a decentralized architecture where devices communicate with each other through a network without the need for a central control unit. Communication can be peer-to-peer or through intermediary devices such as gateways or cloud services.
M2M systems may lack scalability and interoperability as they are often designed for specific applications with limited device compatibility.	IoT systems are designed to be highly scalable and interoperable, supporting a wide range of devices and applications. They utilize standardized protocols and communication technologies to ensure seamless integration and interoperability.
M2M systems typically focus on data collection and basic control functions, with limited capabilities for data processing and analytics at the device level.	IoT systems leverage advanced data processing and analytics capabilities, often at the edge or in the cloud, to derive actionable insights from the collected data. This enables intelligent decision-making and automation based on real-time data analysis.

c. Explain the importance to setup the M2M gateway in the M2M area network.

Ans :- Setting up an M2M gateway in an M2M area network is important for several reasons:-

Protocol Translation :- The gateway facilitates communication between devices using different protocols by translating data from one protocol to another, ensuring interoperability.

Data Aggregation :- The gateway collects data from multiple devices within the network, aggregates it, and sends it to the central server or cloud platform for further processing and analysis.

Security :- The gateway serves as a security checkpoint, implementing security measures such as encryption and authentication to protect data transmitted between devices and the central server.

Local Processing :- The gateway may perform local processing of data, enabling faster response times and reducing reliance on the central server for processing tasks.

Connectivity :- The gateway provides connectivity to external networks such as cellular or Wi-Fi networks, enabling remote monitoring and control of devices within the M2M network.

d. State the differences between IoT and IIoT?

Ans :- The Internet of Things (IoT) and the Industrial Internet of Things (IIoT) share similarities but also have distinct differences:

Internet of Things (IoT)	Industrial Internet of Things (IIoT)
IoT encompasses a wide range of applications beyond industry, including smart homes, healthcare, agriculture, and consumer electronics.	IIoT specifically focuses on industrial applications such as manufacturing, transportation, energy, and utilities.
IoT use cases often involve consumer-centric applications such as home automation, wearable devices, and smart appliances.	IIoT use cases revolve around optimizing industrial processes, improving operational efficiency, and enabling predictive maintenance in industrial settings.
IoT systems may handle smaller data volumes from consumer devices and applications.	IIoT systems typically deal with larger volumes of data generated by industrial sensors and machinery, often in complex environments with strict reliability and safety requirements.
IoT standards may vary more widely across different applications and industries.	IIoT emphasizes interoperability and adherence to industry standards to ensure compatibility between different systems and devices.
IoT systems may have varying levels of security and reliability depending on the application and deployment environment.	IIoT places a strong emphasis on security and reliability due to the critical nature of industrial operations.

Question 3

a. You are equipped with a Node-MCU ESP8266 hardware development board. This board has a built-in WiFi library and SPI library support. You can assume:

- **HomeWifi as your Wi-Fi AP SSID**
- **123456789 as your Wi-Fi authentication password**
- **10 seconds waiting for the establishment of a connection**

Include the serial monitor program with a baud rate of 9600, to display your connection status in the serial monitor interface.

Write a simple program to connect your board to your home Wi-Fi Access Point (AP).

Ans :-

```
#include <ESP8266WiFi.h>

const char* ssid = "HomeWifi";
const char* password = "123456789";

void setup() {
  Serial.begin(9600);

  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);

  int attempts = 0;
  while (WiFi.status() != WL_CONNECTED && attempts < 10) {
    delay(1000);
    Serial.print(".");
    attempts++;
  }

  if (WiFi.status() == WL_CONNECTED) {
    Serial.println("");
    Serial.println("WiFi connected");
    Serial.println("IP address: " + WiFi.localIP().toString());
  } else {
    Serial.println("");
    Serial.println("Failed to connect to WiFi");
  }
}

void loop() {

}
```

b. Discuss how Software Defined Networks (SDN) can revolutionize Iot.

Ans :- Software Defined Networking (SDN) can revolutionize IoT by offering greater flexibility, scalability, and control over network infrastructure.

Here's how SDN can impact IoT:-

1. Centralized Management :- SDN enables centralized management of network resources through a centralized controller. In an IoT environment with a vast number of devices, SDN provides a single point of control for configuring, monitoring, and managing the network, leading to improved efficiency and security.

2. Dynamic Network Adaptation :- SDN allows for dynamic adaptation of network configurations based on changing IoT requirements. This flexibility is crucial in IoT deployments where devices may join or leave the network frequently or where network traffic patterns fluctuate unpredictably.

3. Traffic Prioritization and Quality of Service (QoS) :- SDN enables traffic prioritization and QoS enforcement, ensuring that critical IoT data packets receive priority treatment over non-critical traffic. This is essential for real-time IoT applications such as industrial automation and healthcare monitoring.

4. Optimized Routing and Load Balancing :- SDN facilitates optimized routing and load balancing across the IoT network, improving network performance and reliability. By dynamically adjusting routing paths based on real-time conditions, SDN helps prevent congestion and bottlenecks in IoT deployments.

5. Enhanced Security :- SDN enhances IoT security by providing granular control over network access and traffic flows. Security policies can be centrally enforced and dynamically updated to address evolving threats and vulnerabilities in IoT environments.

6. Integration with Cloud and Edge Computing :- SDN seamlessly integrates with cloud and edge computing platforms, enabling efficient data processing and analysis closer to IoT devices. This reduces latency and bandwidth usage by offloading processing tasks from centralized servers to distributed edge nodes.

Overall, SDN offers a scalable and adaptable networking paradigm that aligns well with the requirements of IoT deployments, making it a key enabler for realizing the full potential of IoT technologies.

c. Describe the real-time database and its application in IoT using firebase real-time database.

Ans :- A real-time database is a database system that provides instant updates to data changes and ensures that the most recent data is always available to users or applications in real time. Firebase Realtime Database is a cloud-hosted database provided by Google as part of the Firebase platform.

Here's how it can be applied in IoT:-

1. Real-Time Data Synchronization :- Firebase Realtime Database allows IoT devices to synchronize data in real time. As IoT devices generate data, it is immediately stored and updated in the database, enabling real-time monitoring and analysis of IoT data streams.

2. Offline Support :- Firebase Realtime Database offers offline support, allowing IoT devices to continue storing and updating data locally even when they are disconnected from the network. Once the connection is restored, the data is synchronized with the cloud database seamlessly.

3. Scalability and Performance :- Firebase Realtime Database automatically scales to accommodate growing IoT data volumes and provides low-latency access to data. This ensures that IoT applications can handle large-scale deployments and process data with minimal delay.

4. Security Rules :- Firebase Realtime Database allows administrators to define security rules to control access to data based on user authentication and authorization. This ensures that IoT data remains secure and only authorized users or devices can access or modify it.

5. Integration with Firebase Services :- Firebase Realtime Database seamlessly integrates with other Firebase services such as authentication, cloud messaging, and analytics. This enables developers to build comprehensive IoT applications with features such as user authentication, push notifications, and data analytics.

Overall, Firebase Realtime Database provides a reliable and scalable backend infrastructure for IoT applications, allowing developers to focus on building innovative IoT solutions without worrying about managing database infrastructure.

Question 4

a. Explain THREE(3) most common lot design challenges.

Ans :- Three common lot design challenges are:

1. Space Utilization :- One challenge in lot design is maximizing the efficient use of available space. This involves ensuring that the layout of buildings, parking areas, and green spaces is optimized to accommodate the desired functionalities while adhering to zoning regulations and considering factors such as traffic flow and pedestrian access.

2. Accessibility and Circulation :- Designing a lot that provides easy access and circulation for vehicles and pedestrians can be challenging, especially in densely populated areas or sites with irregular shapes. Ensuring proper traffic flow, parking availability, and safe pedestrian pathways requires careful planning and design.

3. Stormwater Management :- Managing stormwater runoff is another significant challenge in lot design, particularly in urban areas where impervious surfaces such as pavement and buildings prevent natural absorption. Implementing effective stormwater management techniques, such as permeable paving, rain gardens, and detention basins, is essential to mitigate flooding and prevent water pollution.

b. List any FOUR(4) security challenges in IoT.

Ans :- Four security challenges in IoT are:-

1. Data Privacy :- IoT devices collect vast amounts of data, often including sensitive information about individuals or organizations. Ensuring the privacy of this data, both in transit and at rest, presents a significant challenge. Unauthorized access to IoT data can lead to identity theft, financial fraud, or other privacy breaches.

2. Device Authentication and Authorization :- IoT devices are often deployed in large numbers across diverse environments, making it challenging to manage and secure them effectively. Ensuring that only authorized devices can access networks and services, and that they communicate securely, requires robust authentication and authorization mechanisms.

3. Firmware and Software Vulnerabilities :- Many IoT devices have limited processing power and memory, making them susceptible to security vulnerabilities in their firmware and software. Ensuring timely updates and patches to address known vulnerabilities is crucial to prevent exploitation by attackers.

4. Network Security :- IoT devices typically communicate over wireless networks, which can be vulnerable to interception, eavesdropping, and unauthorized access. Securing IoT networks against attacks such as man-in-the-middle attacks, denial-of-service attacks, and packet sniffing requires implementing encryption, access controls, and other security measures.

c.

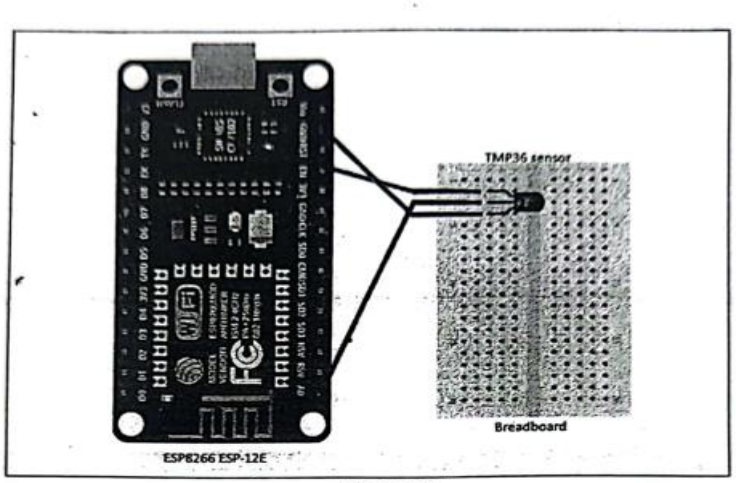


Figure 1

You are asking to design a simple temperature monitoring system using the Node-MCU ESP8266 development board. The hardware setup is as in Figure 1.The sensor used in this system is the TMP36 analog temperature sensor. (The TMP36 sense pin connect to pin A0, Supply pin connects to 3.3V supply pin and the GND pin is connected to GND pin of ESP8266)

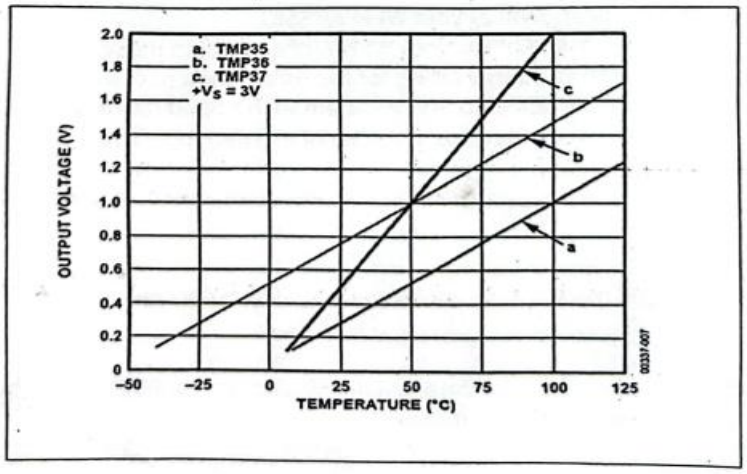


Figure 2

The operation of this sensor has been described in figure 2. The sensitivity of this TMP36 sensor is defined as 10mV/°C and the offset voltage is 500mV.

Write a program for this system with captured temperature data display in the IDE Serial Monitor interface.

Ans :-

```
#include <ESP8266WiFi.h>

const int sensorPin = A0;

void setup() {
  Serial.begin(9600);

  WiFi.begin("your_network_name", "your_network_password");

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}

void loop() {
  int sensorValue = analogRead(sensorPin);

  float voltage = sensorValue * (3.3 / 1024.0);

  float temperatureC = (voltage - 0.5) * 100.0;

  Serial.print("Temperature: ");
  Serial.print(temperatureC);
  Serial.println(" °C");

  delay(1000);
}
```

Question 5

a. A lots of Development platforms can be chosen for the development of Iot systems. There are Operating System based boards such as variance of Raspberry Pi development board and Non-Operating System based boards such as Arduino, Node-MCU and Microbit. Explain any THREE (3) differences between these two types of development boards.

Ans :- Three key differences between Operating System (OS) based boards like Raspberry Pi and Non-Operating System (Non-OS) based boards like Arduino, Node-MCU, and Microbit:-

1. Operating System Presence

OS-based boards like Raspberry Pi run a full-fledged operating system like Linux, which enables them to handle multitasking, run multiple applications simultaneously, and execute complex tasks.

Non-OS based boards like Arduino, Node-MCU, and Microbit do not run a traditional operating system. Instead, they typically run a single program that is uploaded to the board. They are more suited for dedicated, single-purpose tasks.

2. Processing Power and Resources

OS-based boards generally have more processing power, memory, and storage compared to Non-OS based boards. This allows them to handle more computationally intensive tasks, run complex algorithms, and support a wider range of applications and peripherals.

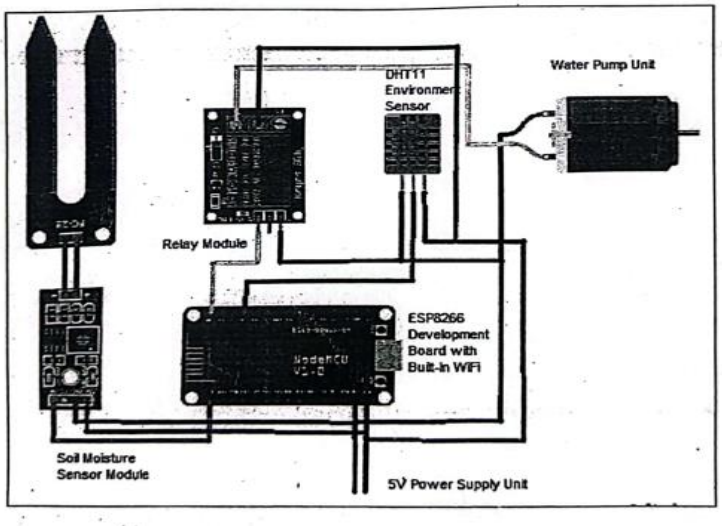
Non-OS based boards usually have limited processing power, memory, and storage. They are designed for simpler tasks and applications where real-time performance or low power consumption is critical.

3. Development Complexity and Flexibility

Developing software for **OS-based boards** often requires knowledge of higher-level programming languages like Python, C/C++, or Java, as well as familiarity with operating system concepts such as file systems, process management, and networking.

Non-OS based boards are typically programmed using simpler languages like C/C++ or Arduino IDE's simplified version of C/C++. The development process is usually more straight forward and requires less overhead in terms of understanding complex operating system internals.

b.



An IoT based smart irrigation system has been designed as figure 2. The system was designed based on the Node MCU ESP 8266 development board. This board has feature with an integrated built-in WiFi module. Identify the following components of this system:

i. Sensors

ii. Actuator

iii. Communication module

Ans:- In the provided IoT-based smart irrigation system:

i. Sensors

DHT11 Environment sensor :- This sensor is used to measure temperature and humidity levels in the environment.

Soil moisture sensor module :- This sensor is used to measure the moisture content in the soil, providing information about soil moisture levels.

ii. Actuator

Water pump unit :- This is the actuator in the system. It controls the flow of water and is responsible for irrigating the soil based on the data received from the sensors.

iii. Communication Module

ESP8266 Development board with built-in WiFi :- This serves as the communication module in the system. The ESP8266 board integrates a WiFi module, allowing it to connect to the internet and communicate with other devices or servers. It enables the system to send sensor data to a remote server for analysis or receive instructions for controlling the water pump unit over the internet.

c. Blynk is a special development tool to create interactive GUI for mobile users to access IoT devices. Explain how this tool helps the development process of an IoT system.

Ans :- Blynk is indeed a powerful tool for developing IoT systems, particularly when it comes to creating interactive graphical user interfaces (GUIs) for mobile devices.

Blynk helps in the development process of an IoT system :-

Easy Interface Design :- Blynk provides a drag-and-drop interface for designing GUIs without the need for extensive programming knowledge. Users can simply choose from a variety of widgets such as buttons, sliders, gauges, graphs, and displays, and arrange them on a virtual canvas to create a custom interface tailored to their specific IoT application.

Real-Time Interaction :- With Blynk, users can create interfaces that allow real-time interaction with IoT devices. This means that users can remotely monitor sensors, control actuators, and receive live updates from their IoT devices directly on their mobile devices, enhancing the overall user experience and convenience.

Cloud Connectivity :- Blynk offers cloud connectivity, allowing IoT devices to communicate with the Blynk Cloud platform. This enables seamless integration with various IoT hardware platforms and eliminates the need for users to set up their own servers or manage complex networking configurations.

Support for Multiple Platforms :- Blynk supports a wide range of hardware platforms and communication protocols, making it highly versatile and compatible with different types of IoT devices. Whether users are working with Arduino, Raspberry Pi, ESP8266, or other popular IoT platforms, they can easily integrate Blynk into their projects.

Extensive Library of Widgets and Features :- Blynk provides a rich library of widgets and features that users can leverage to enhance the functionality of their IoT applications. From push notifications and email alerts to data logging and analytics, Blynk offers a comprehensive set of tools for building sophisticated IoT solutions.

Community Support and Documentation :- Blynk has a large and active community of developers and users who contribute to its ecosystem by sharing projects, providing support, and creating tutorials and documentation. This makes it easier for newcomers to get started with Blynk and troubleshoot any issues they encounter during the development process.

Overall, Blynk streamlines the development process of IoT systems by providing an intuitive interface for designing interactive mobile GUIs, seamless cloud connectivity, and support for a wide range of hardware platforms and features. It empowers developers to quickly prototype, deploy, and iterate on their IoT projects, ultimately accelerating the pace of innovation in the IoT space.

d. Write a program for your Node-MCU ESP8266, to prepare for connection with the Blynk Cloud server. Blynk command is supported by BlynkSimpleEsp8266.h library. You can assume that ABCD is the authentication code to connect to the Blynk Cloud server.

Ans:-

```
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

char ssid[] = "YourWiFiSSID";
char pass[] = "YourWiFiPassword";

char auth[] = "ABCD";

void setup() {
  Serial.begin(115200);

  Serial.println("Connecting to WiFi...");
  WiFi.begin(ssid, pass);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nWiFi connected!");

  Serial.println("Connecting to Blynk Cloud server...");
  Blynk.begin(auth, ssid, pass);
}

void loop() {
  Blynk.run();
}
```