Task 1: Installing Node-RED on a Raspberry Pi through Remote Login.

Step 1: Access Raspberry Pi via SSH

- Use **PuTTY** (Windows) or **Terminal** (macOS/Linux).
- Connect using:
- ssh pi@raspberrypi.local
- Enter the password when prompted.

Step 2: Update and Upgrade Raspberry Pi
sudo apt-get update && sudo apt-get upgrade -y
Step 3: Install Node-RED
Run the official installation script:
bash <(curl -sL https://raw.githubusercontent.com/node-red/linux-installers/master/deb/update-nodejs-and-nodered)
This script:

- Installs/updates Node.js and Node-RED.
- Installs necessary build tools.

Step 4: Enable Node-RED on Boot (Optional)
sudo systemctl enable nodered.service
Step 5: Start and Manage Node-RED

- Start Node-RED:
- node-red-start
- Stop Node-RED:
- node-red-stop
- Check logs:
- node-red-log

Step 6: Access Node-RED Interface
Open a web browser and visit:
http://<raspberrypi.local>:1880
Replace <raspberrypi.local> with your Raspberry Pi's IP or hostname.
Step 7: (Optional) Secure Node-RED

- Configure authentication and HTTPS in ~/.node-red/settings.js.

This completes the installation of Node-RED on a Raspberry Pi via remote login.

Task 2:- Create a simple Node-RED flow that takes input from an inject node and displays output in a debug node. Add a function node to modify the payload in the flow.

Step 1: Access the Node-RED Editor

- Open your web browser and navigate to http://localhost:1880 if running locally.

Step 2: Add Nodes to the Flow

- **Inject Node:** Drag an Inject node from the palette onto the workspace. This node allows manual triggering of the flow.
- **Function Node:** Drag a Function node onto the workspace to modify the payload.
- **Debug Node:** Drag a Debug node onto the workspace to display output in the debug sidebar.

Step 3: Connect the Nodes

- Wire the Inject node to the Function node.
- Wire the Function node to the Debug node.

Step 4: Configure the Inject Node

- Double-click on the Inject node to open its properties.
- Set the payload type (e.g., string or number). Example: Set a static string like "Hello, World!" or leave it as a timestamp.

Step 5: Configure the Function Node

- Double-click on the Function node to open its edit dialog.
- Enter JavaScript code to modify the payload:
- var fahrenheit = msg.payload;
- var celsius = (fahrenheit - 32) * (5 / 9);
- msg.payload = celsius;
- return msg;
- Click "Done" to save your changes.

Step 6: Deploy the Flow

- Click on the "Deploy" button in the top right corner of the editor to save and activate your flow.

Step 7: Test Your Flow

- With the Debug sidebar open, click on the button of the Inject node to trigger it.
- Observe the output in the Debug sidebar; it should show your modified payload.

Implement an HTTP request node to interact with an external API. In Node-RED Flow

1. Add a timestamp (inject) node
observe payload

change the repeat to "interval" every 5 seconds

2. Add a HTTP request node
method GET
URL: https://worldclockapi.com/

3. Add a HTML node
selector: span

4. Add a template node
Add group ex: (Home) Default
template code:

```
<html>
<body>
<h1>
<div ng-bind-html="msg.payload[1]"></div>
</h1>
</body>
</html>
```

5. Deploy

Configure security settings for Node-RED, including user authentication.

Basic Authentication Setup

1. Edit your settings.js file (typically located in ~/.node-red/settings.js).
2. Enable authentication by uncommenting and modifying the adminAuth section.
    - Generate a password hash using the node-red-admin command:
    - node-red-admin hash-pw

HTTPS Configuration

- To enable HTTPS, add the necessary configuration to your settings.js file.

Advanced Security Options

1. **User Permissions** (for multi-user environments).
2. **Disable the editor for public access:**
3. httpAdminRoot: false,
4. **Enable CORS restrictions** as needed.

Task 5: Build a basic IoT dashboard using the Node-RED dashboard nodes.

Step 1: Install the Node-RED Dashboard Module

1. Open the Node-RED editor in your browser (e.g., http://localhost:1880).
2. Go to the Menu (top-right corner) > Manage Palette > Install.
3. Search for node-red-dashboard and click **Install**.
4. Once installed, dashboard nodes (e.g., buttons, sliders, charts) will appear in the palette.

Step 2: Create a Dashboard Layout

1. Click on the **Dashboard** tab (top-right corner of the Node-RED editor).
2. Create a new **Tab** (a page in the dashboard):
    - Click the **+tab** button and name it (e.g., *"IoT Dashboard"*).
3. Create a **Group** within the tab:
    - Click the **+group** button and name it (e.g., *"Sensor Data"*).

Step 3: Build the Flow

1. Drag and drop the following nodes onto the workspace:
    - **Inject Node** to simulate data input.
    - **Function Node** to process or generate dynamic data.
    - **UI Gauge Node** to display sensor data on the dashboard.
    - **UI Chart Node** to visualize data trends over time.
2. Configure each node:

Inject Node:

Set it to inject a random number as payload (e.g., between 0 and 100).

1. Use the following
2. JavaScript code:
3. msg.payload = Math.floor(Math.random() * 101); // Random number between 0-100
4. return msg;

- **Function Node**:
    1. Add logic to process or format data if needed.
- **UI Gauge Node**:
    1. Double-click to configure.
    2. Assign it to the created **Tab/Group** (e.g., *"IoT Dashboard > Sensor Data"*).
    3. Set properties like **Label** (e.g., *"Temperature"*), **Range** (e.g., 0–100), and **Units** (e.g., *"°C"*).
- **UI Chart Node**:
    1. Assign it to the same **Tab/Group**.
    2. Configure it to display a time-series chart.

Integrate an MQTT node into a Node-RED flow and subscribe to a topic

1. Install and Configure Node-RED

• Ensure Node-RED is installed on your system. If not, install it using npm install -g --unsafe-perm node-red or Docker (docker run -it -p 1880:1880 --name mynodered nodered/node-red).

• Open the Node-RED editor in your browser (e.g., http://localhost:1880).

2. Add MQTT Nodes

• Drag an **MQTT In** node from the palette onto the workspace. This node will subscribe to a topic.

• Drag a **Debug** node onto the workspace to display messages in the debug sidebar.

3. Configure the MQTT Broker

• Double-click the **MQTT In** node to open its configuration.

• Click the pencil icon next to the **Server** field to add a new MQTT broker.

◦ Enter the broker address (e.g., mqtt://broker.hivemq.com) and port (default is 1883).

◦ Add credentials if required (username and password).

◦ Set QoS (Quality of Service) level as needed (e.g., 0, 1, or 2).

• Save the broker configuration.

4. Subscribe to a Topic

• In the **MQTT In** node, specify the topic you want to subscribe to (e.g., sensor/temperature).

• Save the configuration.

5. Connect Nodes

• Wire the **MQTT In** node to the **Debug** node. Optionally, add a **Function** node between them for data processing.

6. Deploy and Test

• Click **Deploy** in the top-right corner of Node-RED.

• Publish a message to the subscribed topic using an MQTT client or tool like **MQTTX** or mosquitto_pub.

• Check the **Debug** sidebar for incoming messages.

Implement SSL/TLS for secure communication in a Node-RED instance.

SSL/TLS in Node-RED

Overview

SSL (Secure Sockets Layer) and TLS (Transport Layer Security) are cryptographic protocols designed for secure communication over networks. TLS is the successor to SSL.

Key Purposes

1. **Encryption**: Prevents eavesdropping by scrambling data.
2. **Authentication**: Verifies the identity of communicating parties.
3. **Data Integrity**: Ensures data remains unaltered.

How SSL/TLS Works

1. **Handshake Process**:
    - Client connects to SSL-enabled server.
    - Server presents its digital certificate.
    - Client verifies the certificate.
    - Secure session keys are established.
2. **Symmetric Encryption**: Used for faster session encryption after the handshake.

Key Components

1. **Certificates**: Verify identity (issued by CAs, contain public key and identity information).
2. **Public/Private Key Pair**:
    - Public key encrypts data.
    - Private key decrypts data.
3. **Cipher Suites**: Define cryptographic algorithms.

Securing Node-RED with SSL/TLS

1. **Obtain SSL/TLS Certificates**
    - **Self-Signed (Testing)**:
    - openssl req -x509 -newkey rsa:4096 -keyout privkey.pem -out cert.pem -days 365 -nodes -subj "/CN=yourdomain.com"
    - **Let's Encrypt (Production)**:
    - sudo apt install certbot
    - sudo certbot certonly --standalone -d yourdomain.com
2. **Configure Node-RED for HTTPS**
    - Edit settings.js (typically in ~/.node-red/settings.js).
3. **Configure Security Headers**
    - Add security headers in settings.js.
4. **Verify Configuration**
    - Restart Node-RED and verify with:
    - curl -v -k https://localhost:1880
    - Test SSL configuration for public domains:
    - openssl s_client -connect yourdomain.com:443 -servername yourdomain.com | openssl x509 -noout -t

Include widgets for displaying sensor data and control buttons.

Step 1: Install Node-RED and Dashboard Nodes

1. Install Node-RED:
    - If Node-RED is not installed, install it on your Raspberry Pi:
    - sudo apt update
    - sudo apt install -y nodejs npm
    - sudo npm install -g --unsafe-perm node-red
    - Start Node-RED:
    - node-red-start
    - Access the Node-RED editor at: http://<Your_RPi_IP>:1880
2. Install Dashboard Nodes:
    - Go to Menu > Manage Palette > Install.
    - Search for node-red-dashboard and install it.

Step 2: Connect Sensors to Raspberry Pi

- Use a sensor like DHT11 or DHT22 for temperature and humidity readings.
- Connect the sensor to GPIO pins on the Raspberry Pi.
- Install the required Node-RED GPIO nodes:
- npm install node-red-node-pi-gpio
- npm install node-red-contrib-dht-sensor

Step 3: Create Dashboard Layout

1. Go to the Dashboard tab in the Node-RED editor.
2. Create a new Tab (e.g., "IoT Dashboard").
3. Add two groups:
    - Sensor Data: For displaying sensor readings.
    - Controls: For control buttons.

Step 4: Build the Flow

Drag and configure the following nodes:

Sensor Data Widgets

1. DHT Sensor Node:
    - Drag a DHT sensor node onto the workspace.
    - Configure it to read temperature and humidity from your connected sensor.
    - Set the GPIO pin where the sensor is connected.
2. UI Gauge Node:
    - Drag a Gauge node to display temperature readings.
    - Configure it with properties like:
        - Label: "Temperature"
        - Range: 0–50°C
        - Units: °C
3. UI Chart Node:
    - Drag a Chart node to visualize temperature trends over time.

Control Buttons Widgets

1. UI Button Node:
    - Drag a Button node onto the workspace.
    - Configure it to send commands (e.g., turn on/off an LED or relay).
2. Raspberry Pi GPIO Out Node:
    - Connect the Button node to a GPIO Out node to control devices like LEDs or relays.

Step 5: Deploy and Test

1. Click Deploy in the top-right corner of Node-RED.
2. Access your dashboard at: http://<Your_RPi_IP>:1880/ui
3. Test by observing sensor data updates and interacting with control buttons.
