**Hardware and Software Documentation for Smart Waste Bin Management Project**

**Hardware Requirements**

1. NodeMCU ESP8266

* Microcontroller for connecting to Wi-Fi and managing sensor data.

2. HC-SR04 Ultrasonic Sensor

* Used to measure the distance to the waste level inside the dustbin.

3. Connections

* VCC (HC-SR04) → 3V3 (NodeMCU):
  + Supplies power to the ultrasonic sensor. The NodeMCU’s 3.3V pin provides sufficient power for the HC-SR04 sensor to operate efficiently.
* GND (HC-SR04) → GND (NodeMCU):
  + Completes the electrical circuit by providing a common ground between the NodeMCU and the ultrasonic sensor, ensuring reliable data transmission.
* Trig (HC-SR04) → D5 (GPIO 14) (NodeMCU):
  + The Trig pin is used to send a signal to the HC-SR04 to emit an ultrasonic pulse. GPIO 14 is configured as an output pin on the NodeMCU for this purpose.
* Echo (HC-SR04) → D6 (GPIO 12) (NodeMCU):
  + The Echo pin receives the reflected ultrasonic signal and sends it back to the NodeMCU. GPIO 12 is configured as an input pin to measure the duration of the signal.

4. Dustbin

* Standard height of 20 cm (adjusted to 18 cm in the software for error margin).

5. Power Supply

* A USB cable or a power adapter to power the NodeMCU.

**Software Requirements**

1. Arduino IDE

* To upload the code to the NodeMCU.

2. Node.js

* Backend server environment.

3. MongoDB Atlas

* Cloud database for storing waste management data.

4. Dependencies

* Node.js modules: `express`, `mongoose`

**Hardware Setup**

1. Assembling the Ultrasonic Sensor

* Attach the HC-SR04 to the top inside edge of the dustbin.
* Ensure the sensor faces downward to measure the distance to the waste.

2. Wiring Connections

* Connect the sensor pins as described in the connections section.

3. NodeMCU

* Ensure the NodeMCU is connected to a stable Wi-Fi network with credentials:

```

const char\* ssid = "Dhyey\_phone";

const char\* password = "13072003";

```

**Cloud Provider**

1. MongoDB Atlas

* Provides secure and scalable cloud storage for waste management data.
* Steps to set up:
  + Create an account at [MongoDB Atlas](https://www.mongodb.com/cloud/atlas).
  + Set up a cluster and get the connection string.
  + Replace the placeholder in `server.js`:
  + Whitelist your IP for secure access.

2. Hosting Server

* The backend server can be hosted on any platform supporting Node.js (e.g., AWS EC2, Heroku).

**Testing the System**

1. Testing NodeMCU

* Upload the Arduino code and monitor serial output for distance and filled capacity.

2. Testing Backend

* Use tools like Postman to send test data to `/api/waste-data`.
* Verify the data is stored correctly in MongoDB.

**Future Enhancements**

* Integration with real-time dashboards for waste management monitoring.
* Notifications for timely waste collection.