Instructions and guidelines:

1. This is a team-based task. You need to work with your team members to complete this task.
2. Use the given template, in MS-Word format, to complete the requirements of this task.
3. Upload your completed document file, in PDF format, on Moodle.
4. One submission is needed by each team.

|  |  |
| --- | --- |
| **Course Learning Outcomes (CLO)** | |
| CLO1 | Prepare the requirements, design, documentation and implementation of a software  system component taking quality assurance, economic, health, safety, legal, and marketing factors into account. |

**Task 3- Design Document**

Use the following template to prepare your project design document.

|  |  |
| --- | --- |
| **Task:** | **Design Document** |
| **Project Title:** | **Intelligent Autonomous Hoover as an IoT Solution** |
| **Team Members’ Names:** | 1. Nawaf Alzahem 2. Fahad Aldulaigan 3. Mohammad Abuhaimed 4. Khalid Abu Alsaud 5. Khalid Hali |
| **Supervisor Name:** | **Dr. George Violettas** |
| **Date:** | **4/12/2022** |

1. **System design**

# The system design is comprised of three distinct components. The Waspmote, which is the first component, can be broken into a number of modules based on various functions. The second component is the Waspmote Gateway, which serves as a coordinator in a wireless sensor network to monitor data transmissions. The third component is Meshlium, a Linux router that collects data sent by Waspmote nodes, stores it in local files or a local database, and sends it to the internet or stores it in an external database on the internet.

# System Decomposition

This section describes the major components of the system.

# Layers and Partitions

|  |  |
| --- | --- |
| **Unit** | **Description** |
| API interface | The Waspmote board can be immediately plugged with the basic sensors, such as the temperature sensor MCP9700A and the humidity sensor 808H5V5. Additionally, the corresponding particular functions have been created to gather fundamental environmental information, such as air temperature and humidity. |
| Admin Web page | Additionally, " Meshlium Manager" can be used with Meshlium to administer all MySQL databases. |
| website SDK | Open a new browser window or navigate to the , " Meshlium Manager" plugin in the Tools section and work there directly. Additionally The IP may change if you are not using Wi-Fi or if you have changed |
| triple sensor | Waspmote includes an integrated acceleration sensor that notifies the mote of changes in acceleration experienced along each of the three axes (X, Y, Z). This sensor's integration enables the measurement of acceleration along the three axes (X, Y, and Z), as well as X for gases, Y for temperature, and Z for humidity. |
| noise level sensor | The IEC 61672 standard for sound meters served as the basis for the design of the noise level sensor. with a precision of 0.5 dBA (1KHz) in particular, comparable to Class 2 type devices The value is the LeqA (Equivalent Continuous Sound Level, with A-weighting), which enables the determination of the average sound pressure level over a particular length of time. |
| NB-IoT network technology | Low power wide area (LPWA) technology built on standards was created to support a variety of new IoT gadgets and applications. In deep coverage, NB-IoT dramatically increases spectrum efficiency, system capacity, and user device power consumption. |
| cloud-based server MQTT broker | MQTT is a publish/subscribe messaging protocol that is incredibly lightweight, simple, and suitable for low-bandwidth, high-latency, or unreliable networks where battery life is crucial. Data from Waspmote can be kept inside these cloud platforms or any others that use MQTT due to its capabilities of delivery assurance and bandwidth minimization. These platforms include IBM and Carriots. |
| MySQL Database | A MySQL database is currently operational and used by Meshlium to store locally the data collected. The connection parameters can be seen in the "Local Data Base" page. |

# System Topology

The following is a high-level organization of the system:

Graphical user interface, application

Description automatically generated

# 1.3 Concurrency Identification

Several tiny embedded nodes with limited processing power that can link to one another via close-range wireless broadcasts make up wireless sensor networks, which are independent systems. In order to cover large geographic areas, these systems can build a multihop network employing dispersed routing techniques. By equipping nodes with specific sensors and transmitting their readings to the end users over the ad hoc network, this technology's main objective is to remotely monitor (possibly dangerous) settings. Due to its extensive applications in smart cities, monitoring of environment and health, and other areas.

# Hardware/Software Allocation

Hardware components like the RPL module, Wi-Fi module, Noise level sensor, and triple sensor are necessary for the wasp mote system in order for it to function properly and deliver the intended results.

# General System Performance

The wasp mote system avoids the relay nodes' queueing of incoming packets by operating below 30 ms (packet index from 160 to 180). With this approach, we see a packet loss rate that is almost zero. Because the audio source only transmits a packet once per 40 ms and the radio channel is congested for just a short period of time, there are extremely few intra-path interferences as a result.

# Input/Output Performance

There are 7 analog inputs on the Wasp Mote's sensor connection. The microcontroller is directly attached to each input. The 10-bit analog to digital converter used by the microcontroller (ADC). The inputs' reference voltage value is 0 V. (GND). The input voltage can be as high as 3.3 V. The function analogRead() retrieves input values. An integer number between 0 and 1023 bits, where 0 corresponds to 0 V and 1023 to 3.3 V, will be the value retrieved from this function.

# Processor Allocation

in shared sensor networks, where highly integrated wireless sensor systems are employed to support various applications, to achieve dynamic processor allocation. The architecture is built around a central controller that gathers the data from the sensor nodes, chooses dynamically which apps must be installed in each node at the same time, and then remotely updates the sensor nodes. As sensor nodes that use the ZigBee protocol to connect with the controller, wasp mote devices are employed. Experimental findings demonstrate the proposal's viability.

# Memory Allocation

The various components of the system will mostly use cloud-based storage, MQTT brokers, as well as SQL database solutions.

# Connectivity

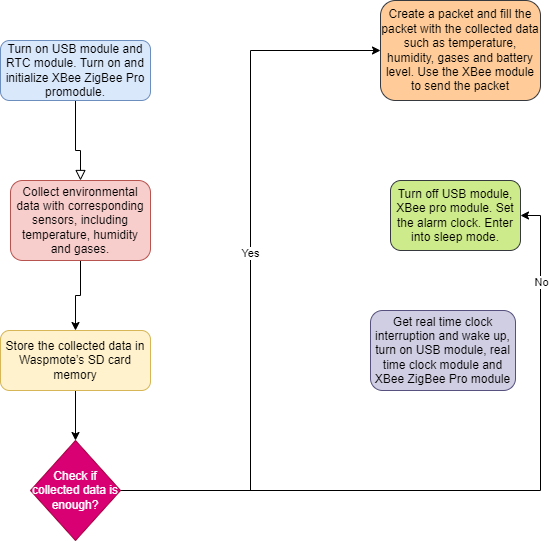
Without the requirement for a prior pairing procedure, sensor data can be communicated using Libelium's Waspmote BLE module. With a smartphone or tablet that bridges data to the cloud using the Wi-Fi or 3G radio accessible in the mobile device, BLE enables any device to connect to the Internet. It is possible to switch between any two of the waspmote radio protocol interfaces, which include Wi-Fi, ZigBee, 802.15.4, 3G, and GPRS, as needed.

# Network diagram Architecture:

**Diagram

Description automatically generated**

Design key program Flowchart:



* 1. **Data Management**

**DBMS ER Diagram:**

Diagram

Description automatically generated

# Software Control Implementation

The units in charge of handling control are described in this section along with how they operate.

# External Control Flow Between Subsystems

The web application will act as a focal point for organizing the control flow among the system's components.

# Concurrent Control

Because the system's components depend on one another heavily to complete their responsibilities, a solid internet connection is necessary for the system's components to successfully communicate and deliver the functionality for which it was created.

# Boundary Conditions

Boundary conditions are conditions that fall directly on, above, or below the bounds of input and output equivalence classes. The majority of the system design work is focused on steady-state behavior, but the system design phase also has to address the system's initiation and finalization (e.g., termination, failure). This is addressed by a brand-new set of use cases known as administrative use cases, Initialization, Termination, and Failure. In this section, a boundary value analysis is given before the overall testing strategy for the system. When creating tests, the outcome space is considered in addition to the input circumstances (the input space) (specifically, the output equivalence classes).

# Initialization

In the context of initializing the system and calibrating the hardware components, boundary testing for initialization is the procedure of testing between extreme ends or boundaries between partitions of the input data. We will also take into account any additional problems that could arise from the incorrect program installation.

# Termination

A very large number of test cases are broken down into smaller, more manageable portions using boundary analysis testing for system-led termination or user-led termination.

# Failure

To ensure quick and effective management of system problems, very clear rules on selecting test cases for system failure without sacrificing the efficiency of testing are required. Numerous factors, including but not limited to bugs, logical errors, hardware malfunction, and external issues, could lead to system failure.

# Packages

This section explains and provides examples of how packages interact at runtime through decomposition.

# Package Diagrams

The logical view of the architecture is demonstrated in the following package diagram.

Graphical user interface, application

Description automatically generated

# Software Package Documentation

|  |  |
| --- | --- |
| **Package** | **Purpose** |
| Network Layer | The data provided by the transport layer is to be routed by the network layer. |
| Application Layer | Different application types created for diverse sensing tasks are created and used at the application layer. |
| Transport Layer | When a sensor network application has to govern the data flow, the transport layer is used. |
| Physical Layer | The modulation-based transmission and reception mechanisms are defined by the physical layer. The power management plane, mobility management plane, and task management plane all keep an eye on the distribution of power, movement, and tasks. |
| Data link Layer | Due to the noisy environment and mobile sensor nodes, the data link layer's MAC protocol should be able to reduce the collision with a neighbor's broadcast. |