Smart Attendance System

1Mahantesh S 2 Mahesh H 3Ramchandra K 4Shivanand K 5Rohini H

BVBCET, Vidyanagar, Hubli - 580031

[1mahantesh.salimath130@gmail.com](mailto:1mahantesh.salimath130@gmail.com), [2mahesh.hegde129@gmail.com](mailto:2mahesh.hegde129@gmail.com), [3ramchandraak@gmail.com](mailto:3ramchandraak@gmail.com), 4[shivanandk58@gmail.com](mailto:shivanandk58@gmail.com), [5rohini\_sh@bvb.edu](mailto:5rohini_sh@bvb.edu)

**ABSTRACT:**

In today’s world, IoT is playing a paramount role in consumer products, . Many devices are becoming smart by connecting to the internet. Attendance monitoring, especially in schools and colleges, is one area where the usage of smart devices is limited. Attendance plays a very important role in modern education system. To ensure that students attend their classes on a regular basis, Educational institutions are taking several measures. But the existing systems have flaws in them, they are not entirely accurate, for example, the Professor may fail to recognize any proxy attendances and it is also an overhead on the Professor to update the attendance of each and every student on website. Taking attendance in class as well as updating it onto the website is time consuming and extra burden. If the attendance is not updated on time, the parents are unaware of their ward's status. This problem has been overlooked for a long time. The following paper aims to provide a novel solution to the problem. A simple handheld device with a biometric sensor, and a wireless radio is used to record attendance and send it to a remote server.

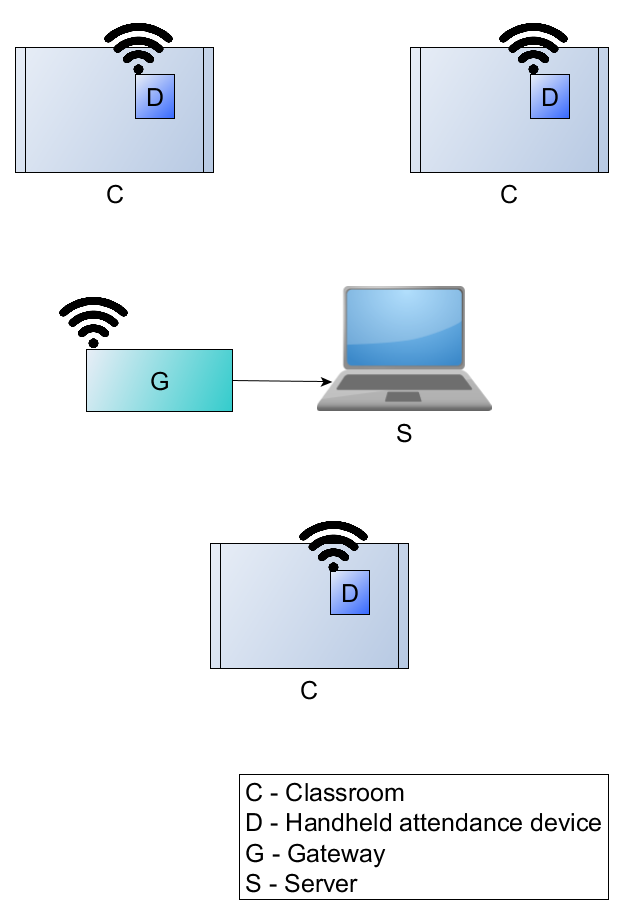
**Keywords:** Biometric attendance, Mobile attendance monitoring, Internet of Things.

1. **INTRODUCTION**

Traditionally, in schools and colleges, the attendance of each student is taken manually by the teacher. The teacher notes down roll numbers of students present for that class and updates it on to the website later. On an average 5-10 minutes are spent in taking attendance of each class. The above steps are to be carried out by each teacher for every class he/she engages. The attendance data of students is not updated onto the website everyday but, once in a week or in a fortnight.

In various organizations, automated attendance systems are used which are based on electronic tags, barcode badges, magnetic stripe cards, biometrics etc. These systems are static and are mount at specific places. Every individual has to approach the module to cast his attendance. Thus, these systems cannot be employed in schools and colleges as they involve taking attendance of a large number of students on an hourly basis. We aim to address the issues in existing attendance monitoring system by designing Smart Attendance System. It is a portable biometrics based attendance system which updates the attendance directly onto the website without teacher’s intervention. The paper first explains the system block diagram, then proceeds with the implementation details for the employed techniques and finally concludes with the discussion of obtained results.

1. **METHODOLOGY**



**Figure 1.1: Block diagram for Smart Attendance System**

A typical use case of Smart Attendance System in a school/college environment is shown in figure 1.1. Handheld devices are used in different classrooms to record the attendance and a server updates the same into a database.

**2.1 Block description:**

Smart Attendance System comprises of:

1. Multiple Handheld devices
2. Gateway device
3. Server

**2.1.1 Handheld device**:

Each handheld device is made up of a fingerprint module, a microcontroller and a Zigbee transceiver. This device is circulated throughout the class to register attendance.

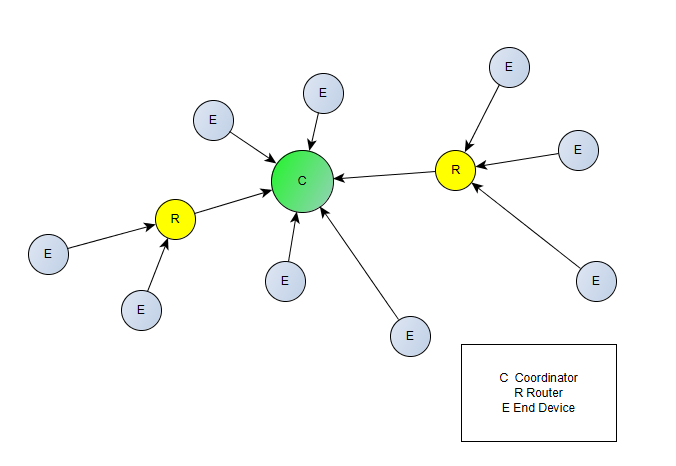
1. Fingerprint module:

The fingerprint module is a biometric device used for student identification. It comes with a storage to save user's fingerprints and an on-board processor to identify them later on. Depending on the size of class, modules of different capacity can be used. The module attaches a unique ID for each fingerprint stored and on identification, returns the same ID. It sends the ID of user on a serial interface, which is interpreted by the microcontroller.

1. Zigbee:

For the wireless communication between handheld device and the server, Zigbee-enabled devices are used. Xbee is one such device which implements the Zigbee standard.

In a Zigbee network, the nodes can be classified as coordinator, router, and end device. In brief, a coordinator is responsible for forming the Zigbee sensor network, the router is responsible to route data between different end nodes in the network, end devices cannot route data they are either connected to a router or the coordinator and are designed in a way to consume low power.



Communication using Xbee is possible in two modes, AT (Transparent) mode or API (Application programming Interface) mode. AT mode is limited to fixed point-to-point communication between two Xbee devices. In API mode it is possible to send and receive data from any Xbee device in the network. Additionally, other information can also be exchanged such as knowing the status of remote IO and ADC and also control them, feedback of the packet reception etc.

In this project, the Xbee present at the server end is configured as the coordinator. It acts as a gateway device between the sensor network and the remote server PC. know the address of the remote Xbee which sent the data to it. The Xbee in the handheld device is configured as an end device. Both coordinator and end device are configured with API mode.

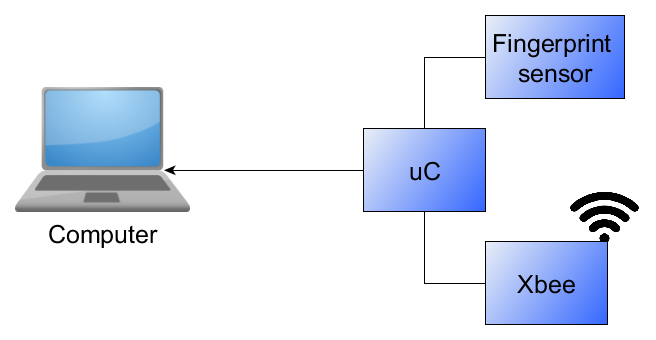
1. Microcontroller

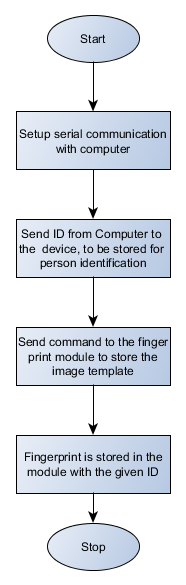
Both fingerprint module and Zigbee device are interfaced to a microcontroller. The microcontroller receives information from the fingerprint module, interprets it and then intelligently transmits the required data to gateway device via Xbee. ATMega328P was used as the controller whereas any microcontroller which supports 2 UARTs can be used for this application.

1. Server:

The server is serially connected to the gateway device (Xbee coordinator) which provides the data received from various handheld devices. The server decodes it, and updates the central attendance database.

1. **IMPLEMENTATION**  
   1. **Fingerprint Enrollment:**

****  
**Figure 3.1: Enrollment setup**

The flowchart in figure 3.2 shows the enrollment procedure of students’ fingerprints. The fingerprint module is connected to the microcontroller and they communicate with each other using serial communication. The controller is also initialized for serial communication with a computer. A unique identification number (Roll number) is sent via computer to the controller serially. This identifier is used by the controller to command the fingerprint module to store the thumb impression template at the ID specified. Thus each fingerprint stored is associated with a corresponding ID. For the enrollment of each fingerprint, two images are taken and the information from both the images is used to generate a template which is then stored in the module. This template is used for search operations and whenever a match is found, the corresponding ID is sent serially to the microcontroller.  


**Figure 3.2: Enrollment flow**

Figure 3.2 explains the series of actions performed by the controller during the enrollment of a new fingerprint.

* 1. **Updating attendance**

The first step in taking the attendance is Subject Selection. The professor selects the relevant subject using a button interface. As he presses the button, subjects pertinent to the class are iterated and are displayed on the LCD. He then confirms it by asserting his fingerprint on the sensor. The device is then passed onto the students so that they can register their attendance. After all the students have recorded their attendance, the professor closes the session by asserting his fingerprint again. Now, the device frames the attendance data and sends it to the server. The figure 3.3 depicts the flow explained above.

1. Frame format:
2. Minimizing payload:
3. Timeout handling:
   * 1. Communication between handheld device and gateway

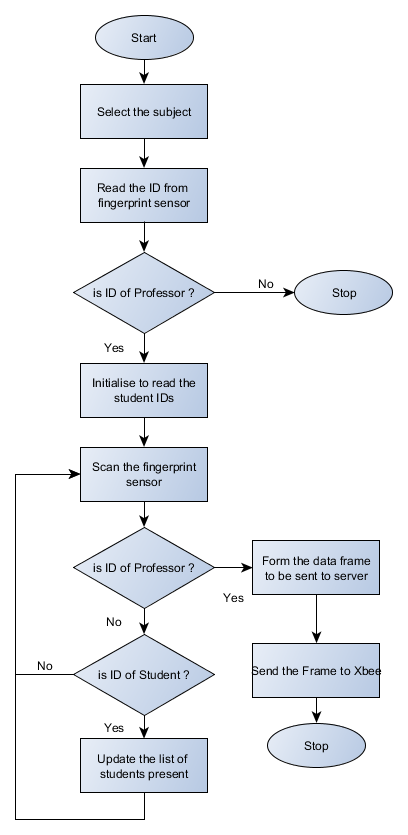


Figure 3.3: Fingerprint identification

**3.2 Data transmission**

Sending the recorded attendance data is done through Xbee. Xbee while transmitting data can consume up to 215mA (Link), which can be considered as overwhelming for a hand held battery backed device. Hence we need to use Xbee efficiently to save power. The power consumed by Xbee increases with every byte of data transmitted. It approximately takes 32 microseconds ([link](https://www.researchgate.net/publication/271431151_Data_fusion_for_reducing_power_consumption_in_Arduino-Xbee_wireless_sensor_network_platform)) for transmission of 1 byte of data. Thus, it is evident that by reducing the size of data, we can reduce the transmission time, hence power. Various compression mechanisms can be employed to reduce the payload. However, the resource limitations of microcontroller have to be considered while selecting the compression technique.

Taking into account all these factors, data is compressed using the following 3 steps:

1. Packing of attendance data:

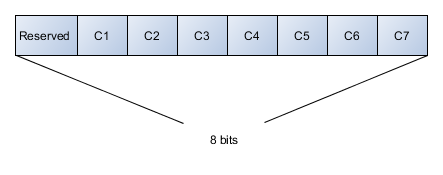
Attendance of each person for 7 classes is packed into 1 byte. C1, C2 ... represent the attendance, in order, for the classes conducted. A bit is set if the student is present, else unset. MSB is reserved for facilitating “[Run-length encoding](https://en.wikipedia.org/wiki/Run-length_encoding)”.  


Figure 3.4: Packing of attendance data

1. Maximum occurrence byte encoding:

It is very likely that attendance of multiple students is same. To encode such repetitive data, the following technique is used.

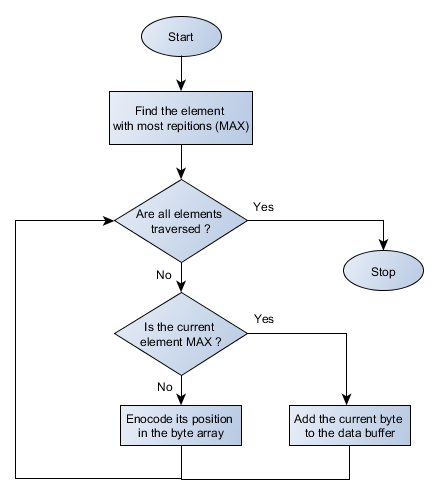


Figure 3.5: Maximum occurrence byte flowchart  
Ex: Consider the data:

0x32, 0x5f, 0x4d, 0x1, 0x1, 0x1, 0x3c, 0x3c  
Here, the element with maximum repetitions is 0x01. Following iterations show the values of byte array and data buffer -

|  |  |
| --- | --- |
| Byte array | Data buffer |
| 0x00 | 0x32 |
| 0x00 | 0x32, 0x5f |
| 0x00 | 0x32, 0x5f, 0x4d |
| 0x10 | 0x32, 0x5f, 0x4d |
| 0x18 | 0x32, 0x5f, 0x4d |
| 0x1c | 0x32, 0x5f, 0x4d |
| 0x1c | 0x32, 0x5f, 0x4d, 0x3c |
| 0x1c | 0x32, 0x5f, 0x4d, 0x3c, 0x3c |

The byte array will be a part of meta data. The data buffer will be passed to RLE algorithm for further encoding.

1. Run-length encoding:

Run-length encoding (RLE) is a very simple form of lossless data compression in which runs of data (that is, sequences in which the same data value occurs in many consecutive data elements) are stored as a single data value and count, rather than as the original run.

PackBits is a commonly used format in RLE. Here, PackBits format is employed with some modifications. In standard PackBits format, both signed and unsigned integers can be encoded, here, we are limiting it from 0 to 127. By introduction of this limitation, the overhead data of delimiters is reduced.

Ex: if the data is -   
0x1, 0x32, 0x5f, 0x4d, 0x1, 0x1, 0x1, 0x3c, 0x3c, 0x3c, 0x3c, 0x4d, 0x5c, 0x5c, 0x5c,   
0x5c, 0x5c, 0x5c, 0x3d, 0x7c, 0x7c, 0x7c  
Length of data – 22 bytes

Encoded data –   
0x1, 0x32, 0x5f, 0x4d, 0xfd, 0x1, 0xfc, 0x3c, 0x4d, 0xfa, 0x5c, 0x3d, 0xfd, 0x7c

Length of encoded data – 14 bytes

**Frame format**

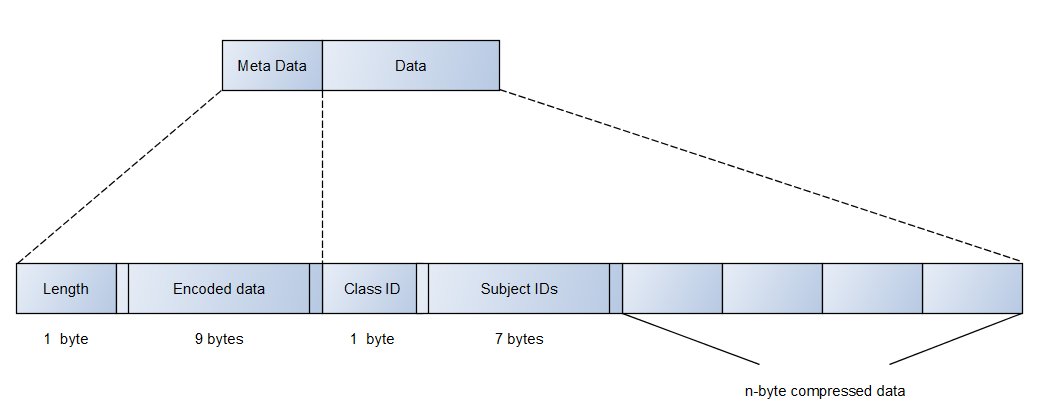


Figure 3.6: Frame format

As shown in the figure, the frame consists of: Meta data and payload.  
  
Meta data is made up of:

1. Length: Size of frame in bytes.
2. Encoded data: This is generated during - Maximum occurrence byte encoding

Payload is made up of:

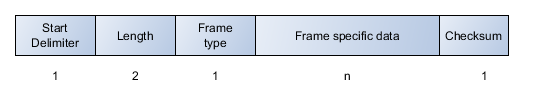
1. Class ID: a unique ID assigned to each class.
2. Subject IDs: Unique subject IDs for classes conducted in order.
3. n-byte compressed data: This is the data obtained after Run-length encoding.

The above method of encoding was tried with multiple samples of attendance data and we found that:

1. Data was compressed to an average of 70-75% of the actual data. Thus, both number of transmissions and power consumed were reduced.

Xbee frame transmission

All Xbees are configured in API mode. Data needs to be sent in API frame format from controller to the end device serially in order to transmit it to coordinator (gateway). The figure 3. Shows the generic [API frame format](https://docs.digi.com/display/XBeeZigBeeMeshKit/Frame+structure).



For transmission, Frame type is configured to – Transmit Request Frame (0x10). Frame specific data has 3 important fields for Transmit Request Frame:

1. 64-bit Destination address – This address should correspond to coordinator’s address.
2. Options – 0x40 - Use the extended transmission timeout for this destination. On setting this option, Xbee retries transmission multiple times, in case of failure.
3. RF Data - This will be compressed attendance data obtained after RLE. It can be up to 255 bytes.

**4 RESULTS**

The proposed Smart Attendance System was implemented using ATMega328 as controller, Xbee Tx and Rx, R305 fingerprint sensor, and a PC running a localhost server. Smart handheld devices were used in multiple classrooms and the attendance data was continuously updated in the localhost server.

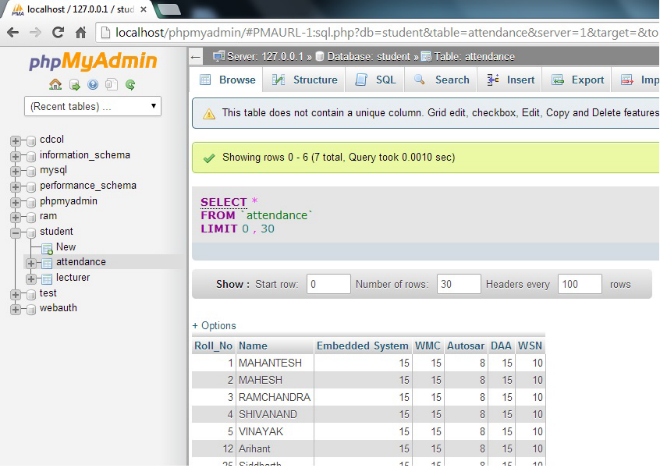
****

Figure 4.1 shows database of student names and attendances for each course. This data is fetched using php, to create a HTML page for display.

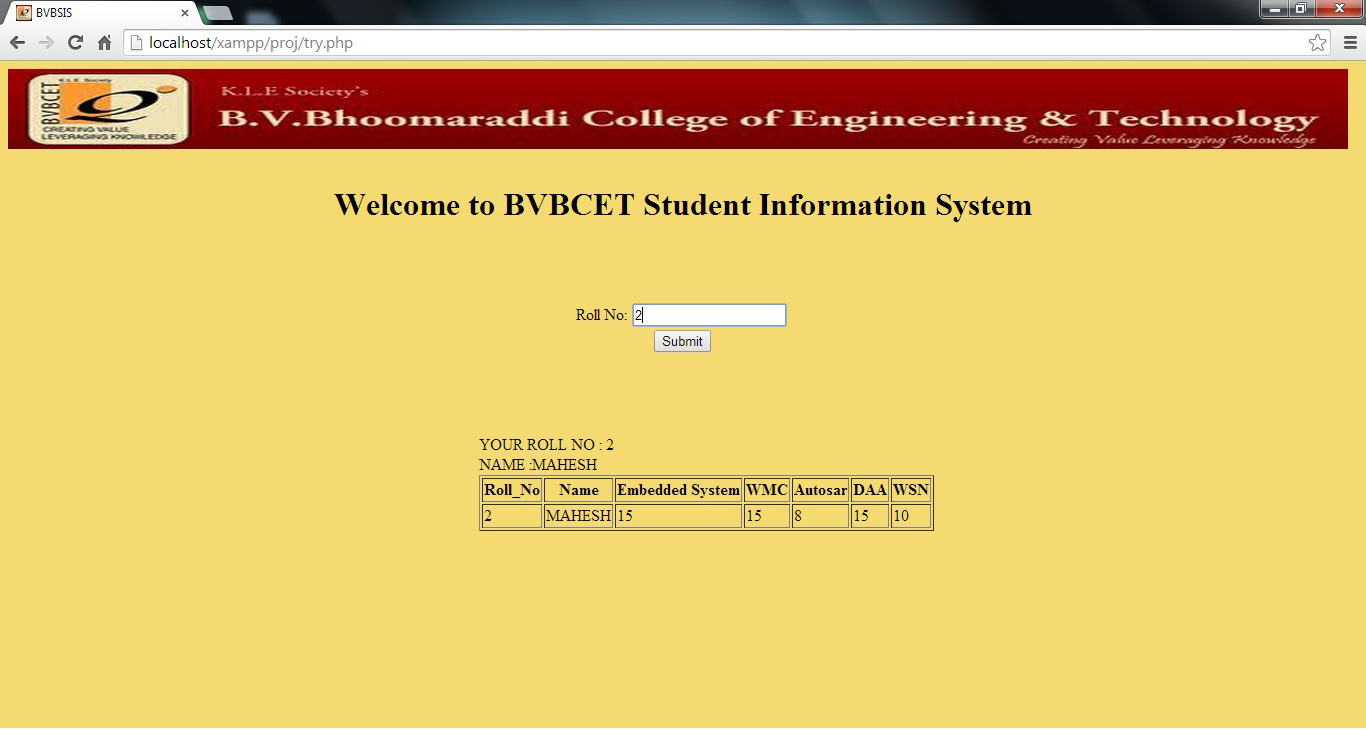


Figure shows a webpage containing the attendance data of a student.



Figure shows the completed portable handheld module used for the Smart Attendance System.

**5 CONCLUSION**

Through this project we were able to model a simple and fool proof attendance system which monitors students’ attendance by using biometrics and automatically updates it to the school/college website without teacher’s intervention and thus helping to solve the problems existing in attendance monitoring in schools and colleges.

**6 FUTURE WORK**

This system can also be used for attendance monitoring not only in schools/colleges but also in various scenarios like corporate offices, government offices, clubs etc.

A standalone module could be developed using GPRS technology which will eliminate the requirement of a local web server.

A central database containing information about all the fingerprints can be maintained which can eliminate the restriction of using a unique module to every class.

Better compression algorithms can be used by using advanced controller.

**REFERENCE**

1. “ An introduction to non-os programming”, http://www.friendlyarm.net/forum/topic/611. .
2. “Development board arm 1810”, http://www.ti.com/product/arm810.
3. “ Stellaris ware graphics library standalone package”, http://www.ti.com/tools/sw.grl.