Automated Mobile Attendance System (AMAS)

Aditi Dankar
Department of Electrical and Electronics Engineering
Manipal Institute of Technology
Manipal, India
aditidankar@gmail.com

Poornima Panduranga Kundapur
Department of Computer Applications
Manipal Institute of Technology
Manipal, India
poornima.girish@manipal.edu

***Abstract*— One of the most common academic processes that institutions/universities follow is that of maintaining student/staff attendance. However, it has been observed that the conventional method of taking students attendance on registers to confirm their physical presence is still prevalent. This method is time-consuming, inefficient and prone to human errors. In order to address the attendance issue, this paper proposes a simple user-friendly mobile application "Automated Mobile Attendance System" (AMAS). AMAS is interfaced with a website in the backend for data entry and report generation. The application is able to track students using GPS and Bluetooth beacons to confirm and verify their presence in classrooms. The application maintains a record of the absentees that is synced with the tables in a remote database server regularly. This application reduces the time required to take attendance, prevents the loss of data as well as provisions to edit incorrect responses. AMAS is developed using Android Studio 3.0.1 and is compatible with 4.4 (KitKat version). The mobile and web application integration is via WAMP server. The database used is MySQL.**

***Index Terms*—Automated Mobile Attendance System, Bluetooth Low Energy (BLE), IMEI, Geofencing, GPS, WAMP**

I. Introduction

Even in the era of computers and mobile devices, the most common method of taking attendance in classrooms in universities employs the use of pen and paper. The attendance system usually involves calling out the names of students in a classroom or by passing an attendance sheet around the class where each student is expected to sign the sheet to confirm their physical presence. While the first method is time-consuming and tedious, the second method can allow students to cheat on their attendance, where a student can sign for another student who is absent. There is also the risk of such attendance sheets being lost or misplaced.

These attendance systems also require extra work and time as the staff must manually enter the attendance of each student on a web application connected to an online database to store the data on the universities' online records. Thus, an automated system is required that would resolve these troubles, reduce the time and prevent the loss of data.

The Automated Mobile Attendance System (AMAS) is a portable, Android-based attendance system. This system requires the android phones of students and Bluetooth Low Energy (BLE) devices, or Bluetooth beacons, installed inside classrooms. The AMAS application detects the Bluetooth beacon, reads its unique ID and matches with the data stored in the 'classroom table' in the online database. If data is matched, and the student belongs to that class, they are marked present. The mobile system connected to an online database and is synced regularly to update the student attendance on the database. This system not only eliminates the need for taking attendance manually but also the need to update the universities' online records.

II. Related Work

a) : Sultana and Mouri, [1] talk about a smart location-based time and attendance tracking system. This is implemented on a smart-phone with Android as its OS. Any organization has a specific location that can be determined via GPS service of the phone. Subsequently, an employee can be located.

This application is made specifically for corporate organizations, is single-user and without many privileges given to the user. It is based on GPS location services only and is therefore not very accurate.

b) : The authors, Noor, Zaini and Hamzah [2], write about an Android application which on installation downloads a list of students from a web server designated for the purpose. The list of students thus obtained, is compared against the scan of each of the student ID cards for confirming and verifying the students presence by the device which acts as an RFID scanner. The camera on the device is used as a sensor to read the bar code given on a student's ID card. The attendance list is updated and uploaded online to a database.

c) : Woo Wing Hong [3] of UTAR, talks about the mobile application developed by the author to take attendance of students using a bar-code reader to read the bar-codes on the ID cards of students.

d) : In [4], the author Ratna Mehta of CSU develops a mobile application to take attendance of students using the phone's camera as an OCR (optical character recognition) to read the code on the student ID cards.

In [2] - [4], the applications employ the use of the device's camera as an RFID reader, a bar-code reader or an OCR scanner respectively on which the application is installed. These applications are a single user, only for the faculty's use, and semi-automated thus the staff must go around the classroom and scan each student's ID. This method is still time-consuming and tedious.

e) : The paper by Noguchi, Niibori and Kamada, [5] uses Bluetooth Low Energy technology. It comprises of a web application and a mobile application. The teacher uses the Web application to set the class ID and class name and feeds in the BLE IDs. The mobile application reads the BLE and scans the ID card of students which is sent to the server. The server checks for the corresponding name, ID and BLE ID and displays a list of classes taking place in the current room and nearby rooms. The students can select a class and mark their attendance.

The drawback of this system is that a student who is absent from the class can give his/her ID card to a fellow student and can thus cheat on their attendance.

f) : Another paper using the BLE technology [6] talks about an Android application which detects BLE when the students log-in to the app. The app sends the detected BLE data along with the time stamp to the server and if the data is found in the server, the attendance of the student is updated in the database.

While this system is significantly more automated than the previous systems, it still poses the problem of more than one student signing-in from the same app and thus making it possible for a student to sign-in with the credentials of another student who is absent.

g) : The paper [7] is written by Raghav Apoorv and Puja Mathur of DTU. The authors talk about college ID cards equipped with Bluetooth Low Energy sensors (Estimote Beacon Stickers) that can be read and recorded by the Android application developed for the staffs' mobile devices. The project uses an extra door counting mechanism, the Infrared Motion Analyzer (IRMA) to count the number of students inside the classroom and then compare with the attendance of the students taken to avoid false attendances.

This system is again single user and even though it is possible to compare the number of students inside the class to the number of people whose attendance is taken, it is difficult to find whose false attendance is taken.

h) : The paper [8] states the use of Browser/Server structure in designing a student information management system. The author explains system design principle, system plan and structure along with the functional module of information system as per their university student information management needs. There is a provision for an interactive student's platform to manage student information.

This application is a student information management system, it is also a web application and not an Android application.

III. Methodology

AMAS is an android based attendance management system which uses geofencing and Bluetooth Low Energy device to find the position of a student inside academic buildings and classrooms. The Android application uses a MySQL database as the back-end. The database stores the data with the help of an online application developed as a part of the project which acts as the sign-up for the faculty and students.

A. Web Application Design

The web application for the project is created using HTML, CSS and Vanilla JavaScript. It comprises the Welcome page which has two buttons to log in as a HOD or to register (same for both faculty and students). The Welcome page leads to the SelectUser page where the user can select between student and staff and requires them to enter their email ID. A verification code is sent to the mentioned email. The code sent is verified on the next screens separately for staff and student. If the verification code sent matches the code entered by the user the user is directed to the registration page, StaffRegistration or StudentRegistration, as selected by the user earlier.

The HOD is given special privileges. The HOD log in page leads to the Reports page where the HOD can view the reports of a certain class or student. The Report page consists of a dropdown list that with Subject, Student Name and Registration Number as the options. Selecting the Classroom makes the 'Date' and 'Section' textview fields visible, whereas selecting the Student Name makes the 'Enter Name' textview field visible and Registration Number makes the 'Enter Registration Number' field visible. The report is made visible to the user based on the selected criteria.

The web application also consists of several other pages to enter the data to the master tables of the database. These web pages consist of Department, Subject and Classroom. These web pages are not visible to the user, these are for the admin to make entries to the database easier.

B. Mobile Application Design and Implementation

The AMAS application comprises a select screen which lets the user select between HOD, Faculty and Student for different privileges provided to different users. The select screen leads to the login screen based on the preference. The login screen takes two parameters, login ID and password provided by the user during the sign-up. The login ID for students is the registration number and for the faculty is the faculty ID provided by the university. The next page is the navigation activity which shows the user-id, user-name retrieved from the database and the privilege of the user.

To mark the absentees, the staff needs to press the 'Close Class' button, this stores all details of all the students not present in the class as absent in the student_attendance table in the database. AMAS also allows a staff to manually enter the attendance of a student in case the student is unable to register his/her attendance through the application on their phone.

AMAS also has the provision for HOD and staff to send notifications through SMS and assign tasks through email. The email functionality requires an implicit intent [9] which uses a third party email application such as Gmail or Outlook, installed on the users mobile to handle the email request and send the email. All the fields required for the email, the receivers of the mail, the subject, the message including the sender's name, date and time is populated through AMAS, the email app is only required to send the email.

The SMS is sent through the AMAS and does not require an external application. Sending SMS requires SEND_SMS

permission in the Manifest file and uses the SmsManager class [10] for the different methods required to send an SMS. It uses the service provider available on the user's phone.

Figure 1 shows the class diagram of the developed system.

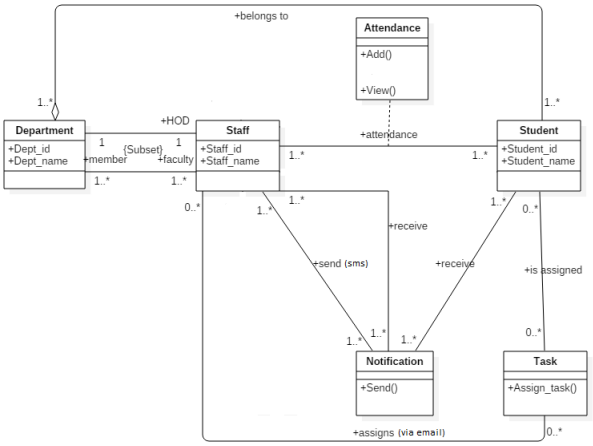

Fig. 1: Class Diagram

1) IMEI Number: IMEI or the International Mobile Equipment Identity is a unique number provided to SIM slot/s of every mobile device. This number is used for the identification of the device. The IMEI number of a mobile phone can be obtained by dialling '*#06#'.

The application requires READ_PHONE_STATE permission in the Manifest file and uses the method getSystemService(TELEPHONY_SERVICE), with the TELEPHONY_SERVICE as the parameter, of the TelephonyManager [11] class to read the IMEI number of a phone running the application.

In this project, the user needs to store the IMEI number of their phone that they will use for attendance in the database using the online application. When a student logs into the AMAS application, the IMEI number of the phone is read by the app and is cross-checked against the IMEI number stored for that user in the database. In case, the IMEI numbers are different the attendance is disabled for the student. This system ensures that one device is not used for marking the attendance of more than one students.

2) Geofencing: Geofences are virtual perimeters or boundaries around actual geographic areas implemented with the help of software or hardware. AMAS implements geofencing as virtual boundaries around each of the academic buildings in the college.

The geofencing API uses Google Play services and Google location services dependencies along with Firebase and Geofire dependencies in the Gradle file, and FINE_LOCATION, COARSE_LOCATION and INTERNET permissions in the Manifest file [12] [13]. The geofence LocateActivity.java implements OnMapReadyCallback, GoogleApiClient.ConnectionCallbacks, GoogleApiClient.OnConnectionFailedListener, LocationListener classes for the different methods required. The location services use GPS receiver inbuilt in android phones and WiFi or cell connectivity to find the location of an object, the student in this case. They follow asynchronous programming model to asynchronously request for location every time the location changes. The location services use Geofire service of Google Firebase, a no-SQL real-time database to store the current location of the student which gets updated every time a new location is requested.

The application uses the distance formula,

$$distance = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad (1)$$

to measures the distance between the student and the centre of the circle and compares it with the predefined radius of the geofence circle to check whether the student is inside the geofence boundary. If the student is outside the geofence, the attendance for the student is not enabled so that they cannot cheat on their attendance.

Figure 2 shows the geofences created around the different buildings of Manipal Institute of Technology, Manipal.

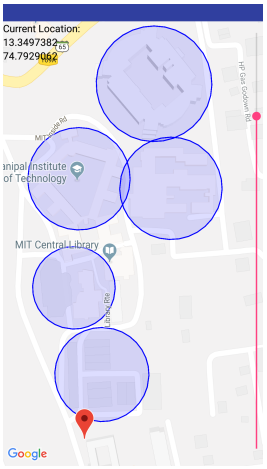

Fig. 2: Geofences

3) Bluetooth Low Energy Beacon: Bluetooth beacons are hardware devices that broadcast low energy signals that smartphones can pick up and listen for signals with mobile applications programmed for listening to such signals. Beacons are a cheap technology accessible to anyone with a smartphone (and Bluetooth services) [14].

The beacon hardware used in the project is configured to iBeacon protocol developed by Apple Inc [15]. iBeacon consists of a UUID (Universally Unique Identifier), a major identifier and a minor identifier which is read by the Android devices to identify the beacon. The UUID is a 16-byte hexadecimal number used to identify a large region, in this project, the UUID is used to define the entire college. The major value is a 2-byte hexadecimal number which identifies a sub-region inside the region defined by UUID. Each of the buildings in the college has a unique major ID. The minor value, again, is a 2-byte hexadecimal value which further subdivides the sub-regions, so each classroom in a building has a unique minor ID. The major and minor values together can identify

the location of a student inside the classroom of a particular building. If the proposed project is to be implemented on large scale, a Bluetooth beacon will be required for each classroom.

This project uses Android Beacon Library 2.13.1 [16], which is open-source beacon library for Android and can be configured to detect a wide variety of beacons. The beacon activity uses android-beacon-library dependency in the Gradle file, and BLUETOOTH and BLUETOOTH_ADMIN permissions in the Manifest file. The implementation of Bluetooth low energy requires BeconConsumer class of the Android Beacon Library for the different methods to read the beacon data, the UUID, major and minor values. The data is then compared with the major and minor values stored in the 'classroom table' of the database. If the major and minor values read by the application are same as the classroom the student is present in, the attendance button is enabled.

C. Database Design and Connectivity

AMAS uses the WampServer which is a software stack comprising MySQL as the database, Apache as the server and PHP as the scripting language.

1) Database Design: The database for AMAS consists of five master tables, viz. department, staff, student, subject and classroom, and five transaction tables or intermediate tables used to connect to the master table. The transaction tables are student_attendance, student_subject, subject_classroom, dept_sec_sem_student, and staff_attendance. The student_attendance table keeps track of the students' attendance. The student_subject table has the details of the different subjects taken by different students. The subject_classroom table stores the details of the different classrooms in which classes for different subjects take place, the faculty taking the class, the day of the week and the period.

Figure 1 shows the database schema of the developed system.

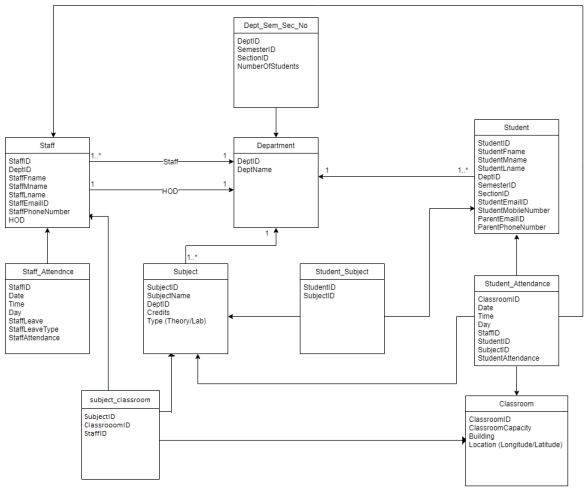

Fig. 3: Database Schema

2) Database Connectivity to Web Application: The web application uses PHP of the WAMP stack as the scripting language to read from and write data to the database in the form of queries. The web pages StudentRegistraion, StaffRegistration, Department, Subject and Classroom are used to store values directly into the corresponding five master tables available in the database. The PHP code takes data from each field in the HTML pages and uses the INSERT INTO query to insert the data into the respective fields in each table.

3) Database Connectivity to Mobile Application: AMAS also uses PHP to read from and write data to the database. The Android application makes use of the Retrofit library, an open source library, which acts as an HTTP client for Android [17]. Dependencies required in the Gradle file include retrofit and converter-gson. The application also requires a 'RetrofitApiClient' which contains the GsonBuilder method, 'RetrofitApiInterface' which acts as an interface to communicate with the PHP files and the 'RetrofitArrayData' which is a class to store the data fetched from the database. These classes and interface are called from inside the Activities that need to access the database.

D. Software Requirement

1) Developer:

a) ***Mobile****:*

- OS: Windows 7 or later
- Android Studio
- Mobile OS: Android 4.4 (KitKat version) or above
- Internet Connection

b) ***Web****:*

- OS: Windows 7 or later
- Back end: MySQL 5.7.1, PHP 7.0.26
- Front end: HTML, CSS, JavaScript
- Internet Connection

2) End User:

a) ***Mobile****:*

- Mobile OS: Android 4.4 (KitKat version) or above
- Internet Connection

b) ***Web****:*

- Browser: Mozilla Firefox, Google Chrome, Microsoft IE and Edge
- Internet Connection

IV. Tests and Results

The figures 4, 5 and 6 are parts of the web application developed for AMAS. The Select User page requires the user to select between staff or student and enter their email. The button Generate Code sends an OTP to the user's entered email which they need to verify on the next page, the Verification page. On successful verification, the student is taken to the Student Registration page. The web pages for the staff registration function in the same way.

Validation for each field's entry is provided in the registration web pages, so if a user forgets to enter any field, they are prompted to fill the particular field.

The figure 7 shows the data entered through the registration page.

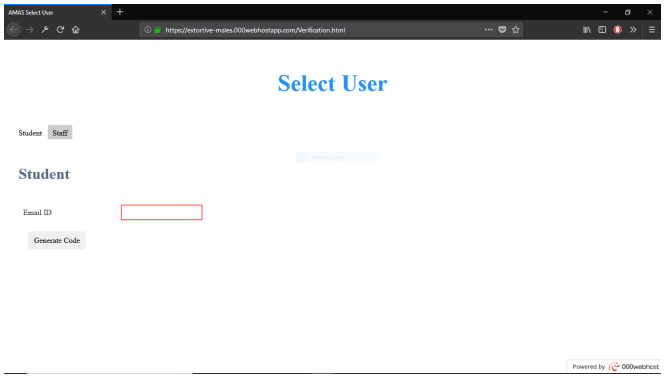

Fig. 4: Select User Webpage

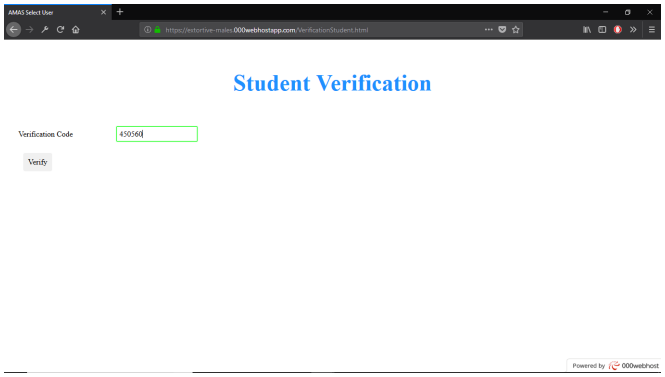

Fig. 5: Student Verification Webpage

The figures 8, 9 and 10 are parts of AMAS mobile application. The Select User page requires the user to select between HOD, staff or student. On selecting a specific category the user is taken to the respective log-in screen which is coded in three different Activities and PHP files to avoid mixing the results from the database table and allowing the users to enter with their respective privilege. On the log-in screen, the user is required to enter the user ID and the password created during the registration on the registration webpages. On successful login, the user is taken to the welcome page which shows the username, user ID and the privilege the user owns.

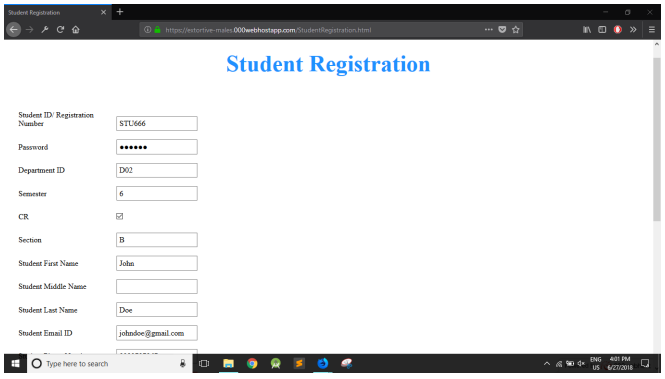

Fig. 6: Student Registration Webpage

STU12		D03	8	B	0	Shivangi	NULL	Misra	student12@gmail.com	9878990919
STU13		D02	4	A	0	Apoorv	NULL	Dankar	student13@gmail.com	9878998919
STU14		D01	4	NULL	0	Akanksha	NULL	Shilpi	student14@gmail.com	9870998919
STU15		D01	2	NULL	0	Kavya	NULL	Gupta	student15@gmail.com	9870998969
STU16		D05	4	A	0	Alok	NULL	Gupta	student16@gmail.com	9323678901
STU17		D05	6	A	1	Kriti		Kansal	kk@gmail.com	6786789897
STU18	123456	D01	4	A	1	Rakshith		Shetty	rk@gmail.com	4593845093
STU19	123456	D02	8	C	0	Adi		Ti	at@gmail.com	6734095621
STU666	123456	D02	6	B	1	John		Doe	johndoe@gmail.com	8989787867

Fig. 7: Database Entry

Figure 11 shows the warning displayed on the Welcome screen if a user logs in from a device which does not belong to them or has different IMEI number/s from what is entered in the database at the time of registration.

Figure 12 show the dialog box which pops up on successful registration of the attendance.

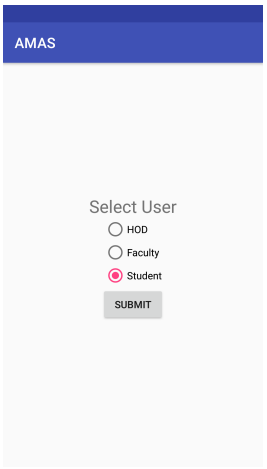

Fig. 8: Select User Screen

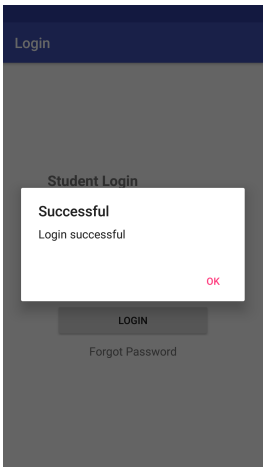

Fig. 9: Successful Student Login

V. Conclusion and Future Enhancements

A. Conclusion

The goal of the project was to design and develop an Android-based application which makes the process of taking

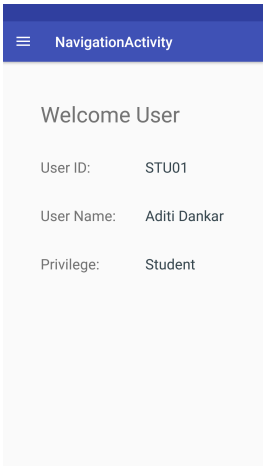

Fig. 10: Welcome Screen

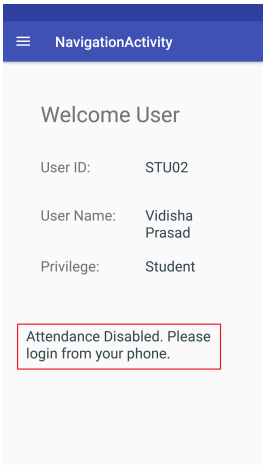

Fig. 11: Warning Displayed on Welcome Screen

attendance automated. The idea was to help both staff and students to implement attendance management system devoid of paper and pen approach. The objective of the project has been achieved via the development of AMAS. The services provided include attendance maintenance, notification SMS and task emails. Additionally, the application connects to a website and database server that stores and retrieves all records.

B. Future Enhancements

- Extending it to other Operating Systems (OS) like iOS and Windows
- Inclusion of faculty attendance
- Provision to generate and view reports on the mobile app
- Face recognition provided to the app to further eliminate the chances of false attendance

ACKNOWLEDGMENT

The authors would like to express their gratitude to the Electrical and Electronics Department and Department of Computer Applications, Manipal Institute of Technology, Manipal.

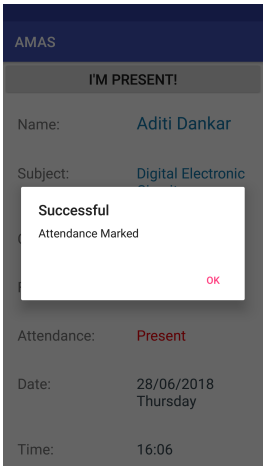

Fig. 12: Successful Attendance Registration Dialog Box

REFERENCES

[1] A. E. Shermin Sultana and I. J. Mouri, "A smart, loaction based time and attendance tracking system using android application," *International Journal of Computer Science, Engineering and Information Technology (IJCSEIT), Vol. 5,No.1, February 2015*, 2015.
[2] M. F. A. L. Siti Aisah Mohd Noor, Norliza Zaini and N. Hamzah, "Android-based attendance management system," *2015 IEEE Conference on Systems, Process and Control (ICSPC 2015), Malaysia*, 2015.
[3] W. W. Hong, "Student attendance recording system," 2014.
[4] R. Mehta, "Attendance tracking system android application," 2014.
[5] E. Z. Shota Noguchi, Michitoshi Niibori and M. Kamada, "Student attendance management system with bluetooth low energy beacon and android devices," *18th International Conference on Network-Based Information Systems*, 2015.
[6] L. K. Anil kumar B G, Bhagyalakshmi K C and G. K. H, "A bluetooth low energy based beacon system for smart short range surveillance," *IEEE International Conference On Recent Trends In Electronics Information Communication Technology*, 2016.
[7] R. Apoorv and P. Mathur, "Smart attendance management using bluetooth low energy and android," *IEEE Region 10 Conference (TENCON) Proceedings of the International Conference*, 2016.
[8] Q. C.-l. Jin Mei-shan and L. Jing, "The designment of student information management system based on b/s architecture," *IEEE Computer Society ,978-1-4799-3134-7/14/ 2014 IEEE*, 2014.
[9] "Intents and intent filters." [Online]. Available: https://developer.android.com/guide/components/intents-filters
[10] "Smsmanager." [Online]. Available: https://developer.android.com/reference/android/telephony/SmsManager
[11] "Telephonymanager." [Online]. Available: https://developer.android.com/reference/android/telephony/TelephonyManager
[12] "Geofence." [Online]. Available: https://developers.google.com/android/reference/com/google/android/gms/location/Geofence
[13] "Create and monitor geofences." [Online]. Available: https://developer.android.com/training/location/geofencing
[14] S. Roberts, "What is ble (bluetooth low energy)?" 2015. [Online]. Available: https://www.shoppertrak.com/article/what-is-bluetooth-low-energy-ble-bluetooth-smart/
[15] "ibeacon." [Online]. Available: https://developer.apple.com/ibeacon/Getting-Started-with-iBeacon.pdf
[16] "Android beacon library." [Online]. Available: https://altbeacon.github.io/android-beacon-library/documentation.html
[17] "Retrofit: A type-safe http client for android and java." [Online]. Available: http://square.github.io/retrofit/