Presence: An Integrated Mobile Solution for Truancy Detection using RFID and Cloud-based Notification Services

Daniel Angelo B. Go
Carl Joshua M. Deiparine
Jovelyn C. Cuizon
Bradford L. Canonigo
University of San Jose - Recoletos
Cebu City, Philippines

Abstract
This study aims to use mobile and cloud-based notification services to detect truancy in students and notify the concerned party in a real-time basis. Attendance logs are captured by radio-frequency identification (RFID) scanners at the entrance and exit gates of the campus. Students' identification cards are embedded with a chip that uses radio frequency to communicate data to a reader. The mobile application pulls data from the database through a web service to reflect whether or not the student gets inside the campus. The presence of the students in their respective classes is further checked by the faculty through a user interface. Parents and sponsors can check, at real-time, the presence of the students in the campus as well as in their classes. To handle fresh new information in a timely fashion, the system implements cloud-based notification services called Google Cloud Messaging for Android (GCM). Parents get immediate notification about the students' missed classes, violations, and class suspensions.

Keywords: Attendance; Mobile application; Android; RFID; GCM; Push notification

Introduction
Truancy, or a students' skipping of classes without parental knowledge or consent has always been a dilemma not only to the parents and guardians but also to the school personnel. Parents and sponsors only become aware when it is already too late to intervene; when students are already failing, and a significant amount of money have already been wasted. While this is essentially an unsolvable problem, several schemes have been placed, which aims at improving the attendance of the students, and ultimately provide a better quality of education to everyone. The use of mobile technology proves to be an effective tool to keep everyone stays informed anytime, anywhere.

Mobile devices such as smart phones have become increasingly available to people regardless of their social status. Attendance systems have evolved from manual roll calls to mobile systems. Numerous works on attendance monitoring had been conducted using a wide variety of technologies. Patel, et al. (2012) used ultra high frequency (UHF) readers to automatically scan all the students’ RFID tags in the classroom and record the attendance of the class. The study of Patela and Jainb (2013) introduced a Near-field Communication (NFC) supported mobile attendance system to keep track of
an employee’s time-in and time-out. (Ayu & Ahmad, 2014) and (Cho & Kim, 2015) also used an NFC-based attendance checking by simply touching an attendance poster, tag or the lecturer’s NFC-based mobile device in the class. Olantunbosun et al. utilized the versatility of RFID in developing a course attendance recording system by letting the students swipe their IDs over RFID readers located at the lecture room doors (Arulogun, Olantunbosun, & Fakolujo, 2013). Masalha et al. implemented the students’ attendance system in which the students scan the QR code posted within the classroom to confirm their attendance in the class (Masalha & Hirzallah, 2014). Akter et al. used an IoT-based system which combines proximity (RFID) and biometric (fingerprint) devices connected to an Arduino microcontroller to reduce the complexity of manual attendance tracking (Akter, Akhi, Farin, Khondoker, & Saklayen, 2018).

While these systems proved to be effective tools to collect attendance logs, the lack of on-time notification made the concerned parties unaware of the rampant cases of unauthorized absenteeism. The key requirement for an effective attendance monitoring application is mobility and ubiquity, being able to respond at real-time, independent of the location and time constraints (Fernando, Loke, & Rahayu, 2013; Chetan, Gautam, Dinesh, & Mathew, 2014). A mechanism that goes beyond a mere data entry must be in place. One way to provide timely updates to stakeholders is to use a push notification technology (Warren, Meads, Srirama, Weerasinghe, & Paniagua, 2014). Push messaging is a communication method in which a connected client is informed of an event in a remote system by receiving notifications from the system (Schwarz, Dutson, Steele, & To, 2013). Mobile push services provide efficient and flexible information disseminations to numerous users in the environment, that support user mobility. (Podnar, Hauswirth, & Jazayeri, 2014)

Presence is a real-time mobile truancy detection and notification system. Real-time attendance logs of students through the RFID campus entry system, including the time of entry and exit from the school premises are captured and are checked against actual student subject time schedules. The system implements a push notification technology called Google Cloud Messaging for Android (GCM) (Google, n.d.), a service for client/server communication, to update the trackers in a real-time basis. The real-time notification system will enable the schools, parents, and sponsors to identify whether or not the student is attending classes, thereby minimizing, if not eliminating truancy, and driving student achievement.

**Materials and Methods**

As shown in Figure 1, the entire process started with the data capture of student attendance log using the RFID scanner situated in the campus entry and exit gates. An application server was implemented which managed the sending of data to registered mobile applications through web service call and processing other client requests. The application server provided a set of web services, which performed backend processing such as data manipulation and storage. A scheduled task or cronjob was run to periodically extract attendance logs from the login/logout application to populate the database. The mobile application sent data requests and received the result set to and from the application server through web services. GCM pushed the notification from the server to the registered mobile applications.

Presence is designed to cater to four interdependent actors as follows:

1. **Student** - A student can view the subjects enrolled, announcements and attendance logs.
2. **Tracker** - A tracker receives notification about the student’s attendance, tardiness
and violations. A referral key to signify permission from the student is required to receive an on-time notification.

3. Faculty - A faculty member can view faculty loads and class rosters and can verify the attendance. The chairperson and dean can approve faculty requests and can post college-wide class suspensions and college announcements.

4. Student Affairs Office (SAO) - The SAO records confiscated IDs that are due to students’ violations.

User Interface

All applications used a different color scheme or color theme, but implemented the same style so that the design was consistent for all the apps. For the student app, the color theme was Navy Blue. For the tracker app, the color theme was Orange. The faculty app was Teal. Presence used the Holo Light Dark Action Bar Theme. Overall, the project followed the Android design principles for a better look and a much better user experience.

Student Mobile Application

The student could use the application to do the registration process, to view the class list, to log the attendance and to enable or disable the tracking feature. Figure 2 is a screenshot of the student’s class list.

The student mobile application requested for the user’s list of subjects enrolled. When the user logged-in in or resumed from a previous use of the app, the app would automatically send a request to the application server. The request was validated by the server, and if it succeeded, it would respond with the information encrypted and in a JSON format. Encryption was done through MCrypt. The mobile application received this encrypted data, then decrypted it. Only the application server and the mobile app knew the encryption key so that the data sent would be kept secured. Afterwards, the JSON data was translated to GSON Models, and then lastly, it was displayed in a ListView. The new data was then added or updated to an offline database using the DataManagement for Android.

Tracker Mobile Application

A tracker could add a student that he or she wanted to to monitor by providing a referral key validated by the server. The
tracker would get a push notification about the dependent’s attendance log and class warnings. In instances when classes were suspened, the application server got all the GCM registration IDs of the students enrolled in that subject. Then, it updated the status of that class for that day, indicating that there was no class for that day. The server proceeded in sending a request to the GCM connection server, where it would send a message to all devices with the corresponding registration ID. However, if the device was not online, the message was queued until it was online. From the raw data, the application would make a notification and would alert the student. A delay in sending the data was expected, as the GCM server still needed to establish a connection to the device.

Faculty Mobile App

The faculty used the faculty mobile app to check the attendance of the students in the class. The app would indicate whether or not the student was in the school premises. The faculty could set class suspensions due to faculty seminars and meetings. On the other hand, the chairman and the dean were given special privileges to set and to notify stakeholders about various college-wide activities and to approve requests from the faculty.

SAO Mobile App

The Student Affairs Office (SAO) played a vital role in the effective implementation of the system. When a student’s ID was
confiscated for making a violation, the student could no longer log in or log out using the campus entry system. The SAO mobile application could be used to manually override the attendance log indicating the student’s presence in the campus. A push notification would be sent to the trackers about the violation.

**Offline Accessibility**

To create an offline accessibility of the application while internet connectivity was not yet established, the system used the Android SharedPreferences and the DataManagement library. DataManagement would store all the necessary classes from the JSON data fetched from the server into a local SQLite database. Meanwhile, a shared preference would keep note whether the data had been written into the local database. The app checks whether there was an internet connection. If there was none, then the DataManager object would be instantiated. The data stored from a previous session could then be taken and translated, then the adapter could be set using the data from the local database.

**Implementation of GCM**

The system utilized the push notification services of Google Cloud Messaging (GCM) which included a third-party application server and a client application (Google, n.d.). To implement Google
Cloud Messaging, two classes were made namely, GCMBroadcastReceiver and GCMIntentService. Also, some specific properties must be placed in the AndroidManifest file. This took care of creating and managing a partial wake lock for the application. It passed off the work of processing the GCM message to the GCMIntentService while ensuring that the device would not sleep in the transition. Receiving the GCM message must be put in a BroadcastReceiver so that the server would continuously run in the background, so that the device could get notifications even when the application was not open. The GCMIntentService would get the message from the GCMBroadcastReceiver, and would handle it accordingly by making an Android notification from it. cURL (client URL) was used to allow transfer of data across websites such as API interaction and oAuth. To contact the GCM server, a request must be made to https://android.googleapis.com/gcm/send using cURL with a header containing a content-type and the authorization key.

Software Testing

In order to assess the degree to which the product satisfies the software specification, Presence was tested based on product quality model (ISO/IEC 25010) quality attributes: functional suitability, performance efficiency and compatibility.

Functional Suitability

The system is deemed as a success if:

1. All functional specification for each actor (Students, Trackers, and Faculty app) are successfully implemented.

2. Push notifications powered by GCM are integrated and enabled in the system.

3. Offline capabilities are available

In order to validate whether or not the functional requirements specified were met, functional testing was conducted. Test case task scenarios were given to actors to perform. Actual and expected results were recorded and tabulated. Table 1 shows a sample test case scenario to test the functionality of the mobile application.

Table 2 shows a sample test case scenario to determine correctness in the implementation of the backend web services.

The test results revealed that functional suitability of the system was at an acceptable level. The system was able to execute all the necessary functionalities. The testing involved 10 test case scenarios for the mobile side and
Table 1. Backend test case for student get information

<table>
<thead>
<tr>
<th>Test Case No.</th>
<th>FTC4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test Title</td>
<td>Check Attendance</td>
</tr>
<tr>
<td>Test Data</td>
<td>Offer_code, date, student_id, status</td>
</tr>
<tr>
<td>Test Summary</td>
<td>Checks attendance of student for a subject and a certain date</td>
</tr>
</tbody>
</table>
| Test Steps    | 1. Swipe left from Subject Information  
                2. Click ‘Check Attendance’ button in ActionBar  
                3. Pick a date  
                4. Tap on a student to change status  
                5. Click Submit button  
                6. Confirm selection, click Submit |
| Expected Result | 1. Shows student list  
                     2. Shows DatePicker dialog  
                     3. Shows student list and its current status  
                     4. Status changes  
                     5. Shows the Dialog containing attendance record summary  
                     6. Updates the student attendance in database |
| Actual Result  | 1. Shows the student list  
                     2. Shows the DatePicker dialog  
                     3. Shows the student list and its current status  
                     4. Status changes  
                     5. Shows the Dialog containing the attendance record summary  
                     6. Updates the student attendance in database |
| Status         | Passed |

11 for the backend web services. All test cases passed the test based on the expected result.

It was found that it was much easier to use 3rd party libraries such as Android-Query, DataManagement, and PanesLayout library because they do the heavy lifting for the developer while keeping them customizable at the same time. Additionally, 3rd party libraries minimize the lines of code needed for a certain function. For example, inserting a record into an SQLite database only took one line when using the DataManagement library. AQuery also provided a very efficient way to call out web services and to get their response, as opposed to the traditional AsyncTask method in Android.

Table 2. Test case for check attendance

<table>
<thead>
<tr>
<th>Test Case No.</th>
<th>B-STC1 (Backend-Student Case)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test Title</td>
<td>Fetch the Student Information</td>
</tr>
<tr>
<td>Test Data</td>
<td>Account_id, student_id</td>
</tr>
<tr>
<td>Test Summary</td>
<td>Retrieves the student information from the database</td>
</tr>
</tbody>
</table>
| Test Steps    | 1. Query from ‘STUDENT’ table where student_id = given  
                     2. Query from the ‘ACCOUNT’ table where account_id = given  
                     3. Encrypt information  
                     4. Post response in JSON |
| Expected Result | 1. Successful query, list all information about the student  
                     2. Successful query, list all account information about the student  
                     3. Encryption using MCrypt successful  
                     4. Response in JSON successful |
| Actual Result  | 1. Successful query, list all information about the student  
                     2. Successful query, list all the account information about the student  
                     3. Encryption using MCrypt was successful  
                     4. Response in JSON was successful |
| Status         | Passed |

Performance

The timeliness of getting the notification was crucial in this application in order not to cause misinformation to the part of the tracker. The push notification feature was tested to determine the speed and responsiveness of the system on actual run. In sending and receiving push notifications using Google Cloud Messaging services, a significant delay was observed during the first attempt to get the message. When the server was triggered to send a notification to a device for a recent session, it took almost 5-10 minutes before the device received the notification. A considerable improvement in the response time was observed for subsequent requests, which may range from a 1-minute delay up to below 5 seconds with a stable internet connection.
Compatibility

The system was tested on different mobile devices to check product compatibilities. The system could run on Android platform, specifically versions 3.0 and above, such as Honeycomb, Ice Cream Sandwich, Jelly Bean and Kit Kat. This meant that the students, parents, or faculty members who had phones running in a different OS such as IOS or Windows, or had Android devices running on before 3.0 such as Froyo or Gingerbread could not make use of the system. Methods implemented in the previous version were already deprecated while new API methods in the newer versions provided greater ease in programming.

Conclusion

The study aimed at developing a mobile application to provide timely notifications of the students' attendance and violations in order to prevent truancy and cutting of classes, to give an alternative and a more convenient way of checking attendance, and ultimately to give a better quality of education to all. Students would be informed about their subjects, attendance records, class suspensions from their classes, and announcements from faculty members. Trackers, people who followed the progress of a student, would receive information about the students. Faculty members could make class suspension announcements, and could check the class attendance. All of these could be done using the Android-powered smart phone or tablet. With a good implementation of the application and cooperation of all its stakeholders, Presence would be a useful application for academic environment in all levels.

The cooperation of the different stakeholders would be vital in the successful implementation of the system. Logging in and out of the university using the scanner must be strictly imposed in all university entrances and exits, to be able to utilize the system’s full potential. If the university would fail to impose these rules, then the system would be rendered useless.

Recommendation

To further improve the existing capability of the system, the researchers recommend incorporating SMS-based notification to accommodate trackers who may not have smart phones, and possibly to use non-proximity identification technologies in order to determine attendance without being close to a scanning device. Moreover, GPS tracking capability may be integrated to give the tracker an idea about the whereabouts of the students being tracked if they are not within the campus. However, all of these need to be studied further due to ethical and privacy implication these might bring to the students.

References Cited


