A Novel Approach to an Automated Attendance System

Ali Alzahrani

[†]Faculty of Computer Science, Islamic University of Madinah, Madinah, 170 Saudi Arabia

Summary
Radio frequency identification (RFID) systems have been utilized immensely as a result of recent technological advances. The RFID system is widely considered as the main technology to realize a ubiquitous computing environment, but the features of RFID systems and the constraints of RFID devices may result in various privacy problems. The major challenge in RFID technology is providing benefits while protecting from the threat of frauds. To address this problem, we developed a solution for combining two-factor authentication technology (RFID and Fingerprint) in order to ensure automatic identification for an attendance system. Human fingerprints are rich in details, and they are one of the most popular and accurate biometric technologies. Fingerprint image analysis using automatic identification technology has been developed to an extent in which it can be used in a university to maintain an attendance system, and it is far better in terms of cost and time than the manual method.

Key words:
RFID-based Attendance, Students' Attendance, Fingerprint-based Attendance, Two-Factor Authentication Technology for Students' Attendance

1. Introduction

Managing students' attendance records in an institution is an essential but tedious task. Authenticating students' identity consumes a number of resources, including time and paper. To automate the attendance process and develop an online mechanism for it, single- or two-factor authentication techniques are used [6].
Authentication, in general, is the process in which a user's authenticity is verified as he/she attempts to access a resource [1]. Authentication may be done through various techniques [2]. The following three main factors characterize authentication techniques:

(i). Knowledge: something that is known to the user (e.g., a PIN or a password) [3]
(ii). Possession: something owned or possessed by the user (e.g., radio frequency identification [RFID], a USB token, a smart card) [4]
(iii). Characteristic: something intrinsic to the user (e.g., biometric features, such as fingerprints or eye patterns) [5]

In this paper, we designed a system that takes attendance through RFID and fingerprint software; the system provides valuable information that is useful for both students and administrators.

RFID [7], [8] is based on radio communication for tagging and identification of objects. It consists of two blocks: RFID transceivers or readers and RFID transponders or tags. An RFID tag consists of a small integrated circuit for storing information and an antenna for communication. A basic RFID system is based on wireless communication between a reader and a tag. RFID readers can read information stored in no line-of-sight RFID tags in their vicinity and communicate it to the central database system through a wired or wireless interface [9].
Fingerprint reader software captures the thumb impression of a student through a thumb scanner and authenticates the information with the impression already registered. We developed a fingerprint recognition application to allow logging of student attendance in lectures. This application provides specific services to students, such as showing student information, calculating the percentage of absence and sending messages to a student's phone.
RFID and fingerprints are the best and fastest methods for identification. They are secure to use, unique for every person, and do not change in one's lifetime. Aside from these, the implementation of RFID and a fingerprint recognition system is cheap, easy, and accurate [10]. RFID and fingerprints enforce access control policy.
Using RFID and fingerprints as two-factor authentication in an attendance system helps ensure students' commitment to attend their classes regularly in universities. In Saudi Arabia, students' attendance is usually taken manually using attendance sheets circulated by course instructors in a class. The method is appropriate for classes with a size of 30 to 60. However, when it comes to checking the attendance of a larger number of students, the aforementioned method becomes inappropriate. Taking attendance during presentations and lectures in a conference hall or room by calling roll numbers and names is laborious and time consuming. The method is also less accurate. There is a greater possibility of missing information and having unauthorized proxies; for example, a student can request his/her friend to mark him/her present in class even if he/she is absent. This often happens among students, as there are really those students who are less interested in the course and just want to fulfill the 80% attendance requirement so that they can be allowed to take the final examination at the end of semester. It is also difficult for a single instructor to monitor all students and record their attendance accurately and efficiently.

Manuscript received November 5, 2018
Manuscript revised November 20, 2018

As per attendance policies of universities and institutions, a course instructor needs to monitor the attendance of individual students for the entire semester. For those students who fail to meet the 80% attendance requirement in class, proper notice is issued to them to inform them regarding their absence and the possible actions to be taken by the institution.

To help instructors take students' attendance and save time, an automated system that records students' attendance accurately is required. We propose the use of this system, which is implemented through RFID devices that need to be provided per class in the faculty. The system will record the attendance of students in class when the class begins and ends. This is to ensure that the students attend the whole class.

In addition, we investigated RFID and key biometric technologies in the market today in terms of development standards, implementation, performance issues, and social impact. We collected the right requirements relevant to the fingerprint attendance system and created a database containing the records of all students. We implemented the system and installed it, and then we evaluated its efficiency during a semester.

The rest of the paper is organized as follows. Section 2 provides a brief background of the various strategies used to ensure students' attendance. Section 3 presents our proposed solution, which is supported and explained by a survey and methodology in Sections 4 and 5. In Section 5 and, we provide our analysis and implementation details. The paper concludes in Section 6.

2. Background

A project by Navons [11] provides a solution to identify students by using a fingerprint reader and comparing the fingerprint with the information stored in the database; if the data match, then the student will be marked as present. The administrator can generate reports on the subject or date and print them out. With this project, a student can become more regular in attending his/her classes, and his/her friends cannot help mark his/her attendance, as everyone has a unique fingerprint. There is no need to maintain an attendance sheet, as the attendance is electronically stored in a database. The system helps faculty with attendance checking. Nevertheless, although this system seems professional, there is still an issue with it: taking attendance with fingerprints for more than 30 students consumes much time. Suppose that each student will need 7 seconds to take his/her attendance, so that would be 3.5 minutes for all students in the class, and as per our survey, the faculty needs 1–3 minutes to take the attendance of 30 students. This system can be a good choice for classes with less than 30 students [11].

Another attendance keeping system has been implemented with RFID [12]. This system has been developed for a school to monitor students' attendance. Each student has a specific student ID, and when he/she reaches the school, the system takes his/her attendance and sends a message to his/her parents informing them about the arrival of their child. Therefore, the system also helps parents know about their child, such as the exact time the student reaches school and the time he/she leaves school. The obvious advantages of this system are that it involves web-based reporting and takes students' attendance quickly. The system is particularly useful for elementary schools, so parents can track their children and know exactly when they enter and leave school. The only disadvantage with this system is that every class requires a fingerprint reader to access the system. Furthermore, this system is not the best choice to be used in universities where attendance needs to be checked per class. Students can also cheat the system easily by giving their card to their friends and asking them to mark them present in class [12].

3. Motivation

A review of previous works will show that a system that takes attendance through RFID cards and random fingerprints check is required in universities. With this system, the collection of attendance will take less time. It is difficult to take attendance manually, and in some cases, students sign the attendance for others. It is also a waste of resources and lecture time.

If an RFID system is applied in Saudi universities, then the accuracy of taking students' attendance would increase, and students will become more committed to attending their classes. Subsequently, the quality of students that universities produce will also improve.

To proceed with the development of the automated attendance system (AAS), a questionnaire was designed. This questionnaire essentially captures all the details required by instructors during the taking of attendance. The standard questions are outlined below:

Survey Questionnaire:

- Do you normally take the attendance of students?
- Have you missed taking students' attendance before?
- If yes, state the reasons.
- How long do you spend to check the attendance of a class of 30 students?
- Have you made a mistake before while you are taking attendance?
- Do you take attendance directly to the academic e-portal or on paper?
- If no, when do you upload it to the system? (i.e., right away after the lecture, by the end of the day,

or by the end of the week)

- What do you think about taking attendance using fingerprint technology?
- What do you think about taking attendance using RFID technology?

Responses to the aforementioned questions were documented. Responses to question 1 are presented in Figure 1. In response to the question "How long do you spend to check attendance, 73% responded with 1 to 3 minutes, whereas 27% responded with 4 minutes and more. For the question "Have you made a mistake before while you are taking attendance?", 70% replied affirmatively, and the remaining 30% replied negatively. In response to the question "Do you take attendance directly to the website or on paper?", 73% responded with yes, whereas 27% responded with no. For the sake of readers' convenience, we do not document other responses.

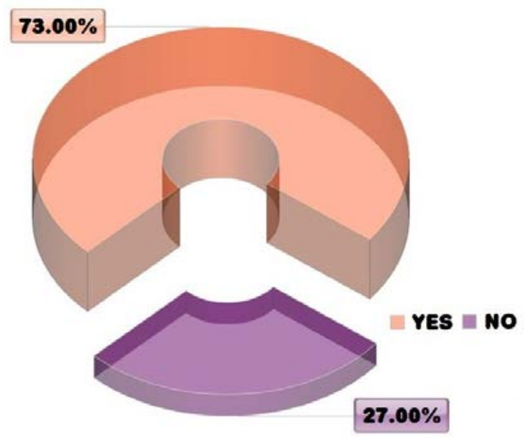


Fig. 1 Have you missed taking students' attendance before?

Based on the above survey, a system that takes attendance by using RFID cards and random fingerprint checks is required. With this system, the checking of attendance will take less time. It is difficult to take attendance manually, and in some cases, students sign the attendance for others. Manual checking is also a waste of resources and lecture time.

The manual attendance procedure obviously has limitations where students sign the attendance for others. The immediate need for an efficient technique to authenticate and monitor attendance is recognized because of the aforementioned problems. We can come up with a technique that is better in terms of quality, efficiency, and manageability (flexibility), as shown in Fig. 2, characteristics that are currently present in two-factor authentication systems. Our motivation is described as follows:

(i). Student attendance issues can be addressed at the proper time by keeping track of student attendance through a runtime tracking and monitoring system.

(ii). Our system has the ability to analyze all events related to the attendance of students through a history behavior of attendance. As a result, students can receive the assistance they need.

(iii). The attendance behavior of students can be controlled with attendance policies (Section 4.1.4) is to be consulted). The attendance system acquires information on students' attendance state during specific attendance activities through our system. The authentication system will then be able to better evaluate the attendance behavior of students, as these students will be monitored according to a policy at runtime.

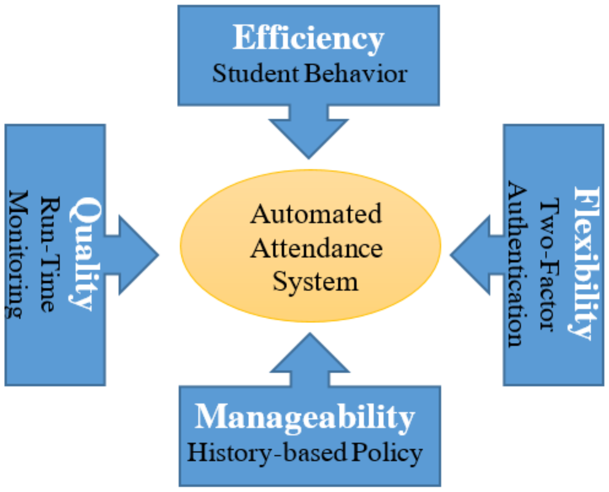


Fig. 2 Motivation and challenges of our

4. Automated Attendance System (AAS)

4.1 AAS Architecture

The proposed architecture of our system is depicted in Figure 3. The figure shows a conceptual model that defines the structure, history behavior of attendance. The RFID cards issued to student are placed alongside the RFID reader, which outputs information to the checker, along with the data sent by the fingerprint reader. The attendance records through the RFID and the fingerprints are stored and updated in the database, and an output is displayed on the screen.

As shown in Figure 3, we divided our architecture into three layers: the authentication layer, the runtime validation layer, and the action layer.

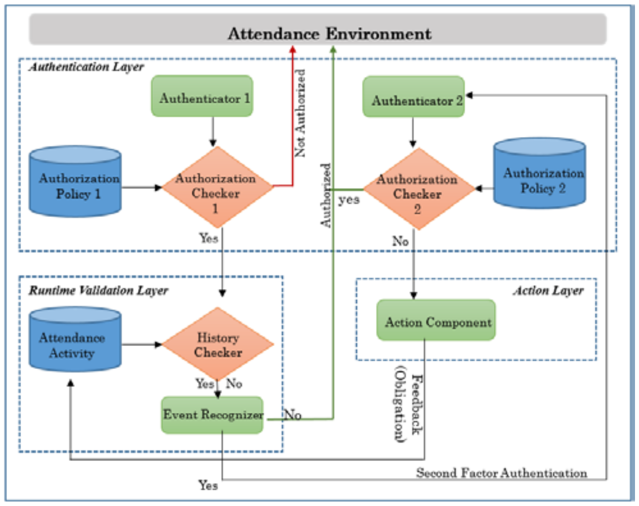


Fig. 3 System architecture.

4.1.1. Authentication Layer

This layer aims to authenticate students. It has two components: authenticator 1 and authenticator 2. The RFID card issued to student are placed alongside the RFID reader (authenticator 1), which outputs information to authorization checker 1. The fingerprint reader (authenticator 2) outputs information to authorization checker 2. Both authorization checkers enforce the authorization policy to verify students' identities. If a student is authenticated by authorization checker 1, our system can recognize his/her previous attendance activities in the runtime validation layer to determine if he/she has had bad behavior. Then, the fingerprint reader (authenticator 2) can authenticate this student, or it randomly chooses some students to do so. Afterwards, authorization checker 1 obligates and updates the attendance activity policy database, and the output is displayed on the screen.

4.1.2. Runtime Validation Layer

Our system will observe if the given policies are being followed, as evident in the students' attendance behavior, through the validation of these policies at runtime in this layer. This process is highly significant in successfully checking the attendance behavior of students. There are a number of requirements related to attendance activities in a policy. Furthermore, this process is completed through the utilization of a runtime validation technique that works in accordance with the AnaTempura validation toolkit [12]. Safety and timing properties are validated through this tool at runtime (see Section 4.2 for a further discussion of this tool). As Figure 3 illustrates, this layer has two components, and they are described as follows:

(i). History Checker: The History Checker requires the events in the previous attendance activity state of the student and the Attendance Activities repository. The decision on whether the requirements of the policy are met by the attendance behavior of the student is made by the History Checker. History-based events from the attendance activity repository are used to check behavior. These events are considered by the History Checker, which checks whether students have had undesirable attendance activities.

(ii). Event Recognizer: The Event Recognizer controls the workflow in which students need to be identified again by the second authentication factor (fingerprint) based on the result of the History Checker. The results of the History Checker are taken into consideration for the recognition of an event that is a snapshot of states describing a particular student's attendance behavior. Afterwards, the Event Recognizer components get the recognized events; if the student has not had a bad history and is not chosen for random re-authentication by authenticator 2, he/she is authorized. Otherwise, the Event Recognizer requires that the student be re-authenticated by authenticator 2.

4.1.3. Action Layer

There is contact between the authentication layer, the runtime validation layer, and the action layer. In the action layer, when authorization checker 2 determines the policy violation of a student, the action component comes into play. This component processes the information sent by authorization checker 2. Afterwards, to prevent possible issues that could emerge and affect the progress of the student, it determines what needs to be done for this student. The actions could tell the teacher about the present state of the student via e-mail or send warning messages and obligate (update) the Attendance Activities repository in the runtime validation layer only if the student is not authorized by the second authentication factor (fingerprint).

4.1.4. Attendance Activity Policies

It was also previously discussed that a number of requirements form a policy. In relation to attendance activities, ensuring good attendance behavior is the objective of these rules. Authorization is used for expressing rules, such as obligation rules, in a large number of policy languages.

4.1.4.1. Rule Structure

Suppose that the set of every subject, object, and action in the system is represented by S, O, and A, respectively. Students are represented by subjects, attendance activity by objects, and the actions taken by subjects on objects by actions. Please see [13] for the concrete syntax and semantics. This section provides some of these specifications. Nevertheless, the occurrence of events in the system history is directly verified by the policy rules in place of the state of the system. Done(s; o; a) represents that an action has been successfully executed on object o by subject s, whereas an access control decision is represented by author(s, o, a), for the policy language utilized. We do not focus on the real semantics of the rule, which are applied in security access control in policy-based management systems [13]. Instead, controlling the attendance activities of students is our main concern in this study.

4.1.4.2. Authorization

Whether the access control is positive or negative can be asserted by the authorization rule. Authorization is denied with negative rules, whereas it is allowed with positive rules. The condition for granting or denying an access request can be seen from both positive and negative rules. Therefore, the condition must be system based and behavioral, in which the historical behavior of a student determines a condition, for the environmental policies to be expressed by the definition of the authorization rules.
Examples: Three of the cases are presented as follows:

a) Let a student ID (SID1234) be the subject and the course section (section1234) be the activity (object); enter should be the action taken:

```
True → autho (SID1234; section1234;
```

b) Let a student ID (SID1234) be the subject and the course section (section1234) be the object o; enter should be the action taken, where the date is the 1st of October:

```
Date (01/10) → autho (SID1234; section1234; enter)
```

c) Let a student ID (SID1234) be the subject and the course section (section1234) be the object o; enter should be the action taken, where the authorization here has always not been denial for him/her previously:

```
■Done (SID1234; section1234; enter) →
autho (SID1234; section1234; enter) //for
RFID authentication
```

The next policy is enforced at the beginning of a semester and when there is a possibility of misinformation and unauthorized proxy, such as when a student asks his/her friend to mark him/her as present, but he/she is really absent. The system requires the student to authenticate also via fingerprint.

```
◊denied(SID1234; section1234R; enter) →
autho(SID1234; section1234F; enter) //for
fingerprint authentication
```

4.1.4.3. Obligation Rules

An action that takes place when a subject has performed a particular action is stated by obligation rules [13].
In general, the obligated action can be taken when the student is authorized by RFID (1st factor) and not authorized by fingerprint (2nd factor). We expressed the behavioral policies by defining the obligation rules.
Example: Let a student ID (SID1234) be the subject and the course section (section1234) be the object o; enter should be the action taken. Accordingly, the action that the student performed needs to be reported to the lecturer (Lec1234).

```
Done (SID1234; section1234RFID; start)
Denied (SID1234; section1234FINGRPRINT; start) ^
→ Oblige (sys, Lec1234;    //for obligation
```

4.2 Technology: Runtime Validation Toolkit (AnaTempura)

Interval temporal logic (ITL) refers to a flexible means of presenting propositional and first-order reasoning regarding the time periods present in descriptions of hardware and software systems. ITL is different from the majority of other temporal logics in that it can deal with sequential and parallel compositions, and it provides robust and extensible specification and proof methods that can be used to explain properties pertaining to safety (ensuring that nothing bad takes place), liveliness, and timeliness [12]. It is not only possible to express time constraints; rather, in a slightly adjusted version of ITL, the most significant programming constructs can also be considered as formulas. It is also possible to use TLA (Lamport, 1994) or event calculus [14] as alternative formulas [14]. Furthermore, there is an executable subset of ITL, Tempura, which suggests that accessing an interpreter for attendance tasks is easy, if it can be expressed in this subset. Tempura has been used effectively in earlier studies [15] for runtime validation of functional (safety) and real-time properties. Learning policies are essentially safety properties; therefore, they can be expressed in Tempura.
The tool used to verify the runtime of timing and safety properties is known as AnaTempura. Assertion is used in this technique to determine if a system fulfills the safety conditions given in the ITL [15]. The Event Recognizer

constituent of our system architecture describes this assertion by sending a series of events (attendance activity states), such as the values of variables, when the software is running. An ITL property is related to a group of state sequences (intervals), so runtime validation determines if the sequence of events presented by our Event Recognizer belongs to the group of rules related to the safety and timeliness properties presented by the attendance activity policy that we are seeking to assess. This membership check is carried out by the Tempura interpreter [15]. Therefore, AnaTempura can be used to check the runtime of attendance activity policy for the events.

5. AAS Design and Implementation

This section explains the system architecture and design of our proposed system within a small simplified scenario that illustrates the use of our attendance activity policy. We present the design of the system and provide the registration phase, the authentication phase, the monitoring process and policy enforcement algorithm, and the implementation in the following subsection.

5.1 Registration Phase

The sequence diagram of the admin side is depicted in Figure 4, in which the processes of the admin when using our system are to add a subject, take attendance, add new data to the system, add new students, and add new classrooms.

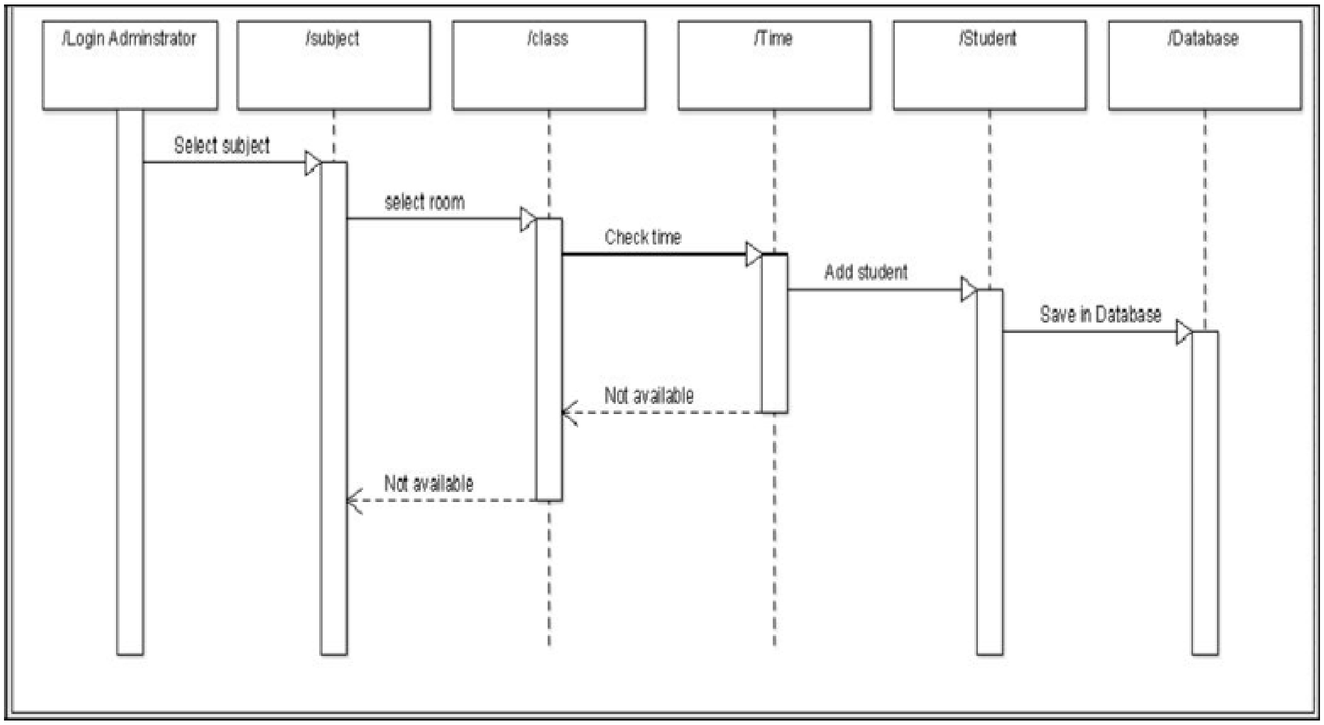


Fig. 4 Sequence diagram of lecture registration.

5.2 Authentication Phase

We present a small simplified scenario that illustrates the use of our history-based policy for attendance activities. In this scenario, we focus on attendance activities, as attendance is required in each lecture.
Scenario: The student uses his/her RFID to log in. For the purpose of authorization, the student identity is sent by the RFID reader to the checker. After authorization is received, the current state is recorded by the system in the database. After login, if the student is found to have an untrustworthy history (attendance cheating), the history of the cheating state is determined by the system; therefore, it is also important to use the fingerprint of the student for authorization. If the student fingerprint is authenticated by the checker, this means that the student is present in the database. If this is not the case, the student will be absent.
For the attendance activity, this situation can be formalized as a policy (Policy1) as follows:
Here, student S and attendance activity aRFID belong to the section attendance. aRFID signifies the attendance activity

that should be carried out by the RFID technology. Nonetheless, if the TimeUnit in the final state of the interval is below 15, signifying the number of sessions for a single section and always over the states, the earlier session has been authorized by the RFID technology, then in the subsequent state, the student is permitted to use the RFID technology for the attendance activity $\mathrm{aRFID_i}$.

$$Policy1 = \begin{pmatrix} \mathrm{fin}\ \ (\mathrm{TimeUnit} < 15) \wedge \\ \mathrm{Student}(\mathrm{S}, \mathrm{Section}) \wedge \\ \mathrm{attendance}(\mathrm{aRFID_n}, \mathrm{Section}) \\ \wedge_{i=0}^{n-1} \blacksquare \mathrm{done}(\mathrm{S}, \mathrm{aRFID_i}\,, \mathrm{use}) \end{pmatrix} \rightarrow (\mathrm{autho}^{+}(\mathrm{S}, \mathrm{aRFID_n}, \mathrm{use})$$

Here, student S and attendance activity aRFID belong to the section attendance. aRFID signifies the attendance activity that should be carried out by the RFID technology. Nonetheless, if the TimeUnit in the final state of the interval is below 15, signifying the number of sessions for a single section and sometimes, over the states, the earlier session has been unauthorized by the RFID technology, then in the subsequent state, the student is permitted to use the fingerprint technology for the attendance activity $\mathrm{aFINGERPRINT_i}$.

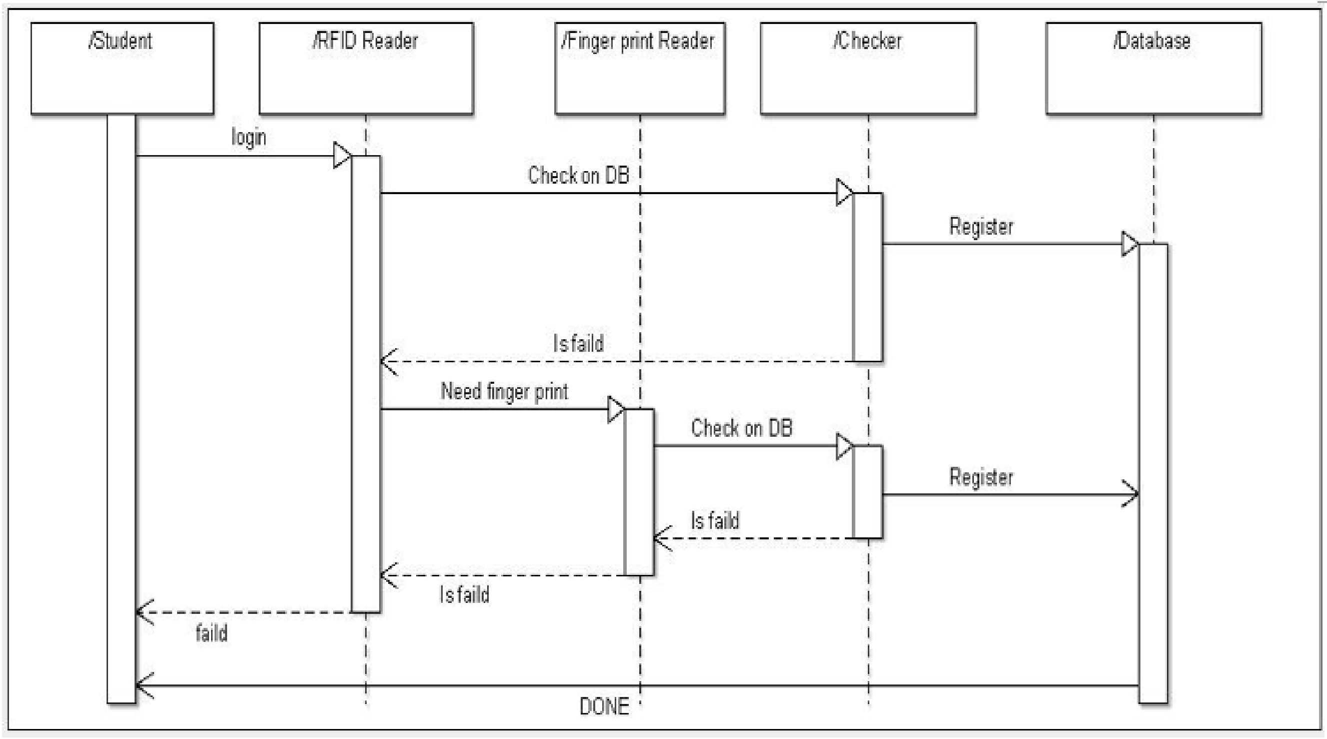


Fig. 5 Sequence diagram authentication phase.

$$Policy2 = \begin{pmatrix} \mathrm{fin}\ \ (\mathrm{TimeUnit} < 15) \wedge \\ \mathrm{Student}(\mathrm{S}, \mathrm{Section}) \wedge \\ \mathrm{attendance}(\mathrm{aRFID_n}, \mathrm{Section}) \\ \wedge_{i=0}^{n-1} \lozenge\ \mathrm{done}(\mathrm{S}, \mathrm{aRFID_i}\,, \mathrm{use}) \end{pmatrix} \rightarrow (\mathrm{autho}^{-}(\mathrm{S}, \mathrm{aRFID_n}, \mathrm{use})\ \wedge\ (\mathrm{autho}^{+}(\mathrm{S}, \mathrm{aFINGERPRINT_n}, \mathrm{use})$$

Figure 5 shows the sequence diagram of the authentication phase, depicting the procedures followed by students using the RFID and fingerprint technologies. The (Policy2) shows the formalization of these requirements.

5.3 Monitoring and Policy Enforcement Algorithm Phase

The monitoring procedure and the enforcement of attendance activity policies in the runtime validation layer for the scenario of the attendance activity given earlier can be seen in the algorithm. The basis of this algorithm is the state transition diagram shown in Figure 5. The algorithm essentially seeks to monitor the events taking place in the attendance activity from the preliminary session until the activity concludes. The algorithm also aims to implement an applicable policy on the basis of the existing state of the attendance activity.

The way the monitoring procedure for the attendance activity shifts from one state to another can be seen in algorithm (1) given below. The global variables used in the entire process are initially described. The main state of the process is then determined, which is the active state. The variable state refers to the existing state of the attendance activity in the monitoring procedure. There are substates of the active state (Active = {Selecting A, Doing A, Done A, Not Done A}, {Aborted, Cheated}), and its foremost state is inactive.

In addition, the functions given below in algorithm (1) are used to carry out the monitoring process and the implementation of policies, where the input parameter is the existing chosen attendance (Attendance[n]).

```
1.  Algorithm 1: Check Quiz Activity
2.  Require: Attendance[n], NoOfAttendance, AState, SelectedA,
3.  Ensure: n ∈ NoOfAttendance
4.  Active is the main State!, Inactive is the Initial state!
5.  for AState = Active do
6.      (SelectingA, DoingA, DoneA, NotDoneA, AbortedA,CheatedA) //are substates!
7.      NotDoneA is Initial Substate!
8.      for AState = SelectingA do
9.          (GettingPermit, Allowed, Denied) // areSubstates!
10.         GettingPermit is the Initial Substate!
11.     end for
12. end for
13. repeat
14.   SelectedA = CurrentAttendance()
15.   if n = SelectedA then
16.     AState = Active
17.     CheckPermission(Attendance[n]) // Moving to SelectingA state!
18.     CheckDoingA(Attendance[n]) // Moving to DoingA state or CheatedA state!
19.     CheckAuthentication(Attendance[n]) // Moving to DoneA !
20.     AState = Inactive // Final State in this process!
21.   end if
22. until AState = Inactive
```

Algorithm (2) is the sub-algorithm of the main algorithm (algorithm 1), depicting the procedure of the function CheckPermission(). The purpose of this function is to determine if a student can get access to the existing attendance. The inputs for this function are the applicable policy from the policies available and the events related to the policy attributes. Therefore, in this function, the policy will be enforced when the student is permitted to access the existing attendance and do the attendance, or not allowed access, which would mean that the monitoring process has concluded by giving the status of Aborted to the attendance.

```
1.  Algorithm 2: CheckPermission(attendance[n])
2.  Require: Attendance policy for permission
3.  Ensure: Events correspond to the policy attributes.
4.    if AState[attendance[n]] = CheatedA ∧ DoneA ∧ AbortedA
5.   then
6.     AState[attendance[n]] ⟵ SelectingA
7.     Apermission[attendance[n]] ⟵ GettingPermit // Moving to GettingPermit state!
8.     GetAttendancePolicy(attendance[n]) // Find applicable Policy from the repository!
9.   return PolicyX
10.  GetAEvents(attendance[n])
11.  return SequenceOfEvents!
12.  CheckPolicyV.S.Events
13.  if Policy is Satisfied then
14.  Apermission[attendance[n]] ⟵ Allowed
15.  AState[attendance[n]] ⟵ DoingA // Moving to DoingA state!
16.  else
17.  APermission[attendance[n]] ⟵ Denied
18.  AState[attendance[n]] ⟵ Aborted
19.  end if
20.  end if
```

The process of the function CheckAuthentication() is denoted by the sub-algorithm (Algorithm 3), the purpose of which is to determine the authentication state of the existing attendance. An applicable policy for this case will be enforced, regardless of whether the present attendance is carried out or not. At times, the system selects certain students randomly to perform repeated authentication through the fingerprint reader in order to determine attendance cheating.

First Scenario: The policy prevents the students from performing the attendance through just the RFID. If a student does not use the fingerprint, as well, for the attendance, then he/she will be considered a cheater and will not be allowed to do the next attendance by using RFID.

```
1.  Algorithm 3: CheckAuthentication
```

```
    (attendance[n])
2.  Require: Attendance policy for the authentication state
3.  Ensure: Events correspond to the policy attributes.
4.  if AState[attendance[n]] = DoingA then
5.  GetAttendance Policy(attendance[n]) // Find two-factor authentication (FRID and Fingerprint) Policy from the repository!
6.  return PolicyY
7.  GetAEvents(attendance[n])
8.  return SequanceOfEvents!
9.  CheckPolicyV.S.Events
10. if Policy is Satisfied then
11. AState[attendance[n]] ← Done // Move to the Done state!
12. else
13. AState[attendance[n]] ← Cheated // Obligate the database
14. end if
15. else
16. AState[attendance[ n] ← Idle
17. end if
```

$$Policy3 =$$
$$\begin{pmatrix} \text{fin } (\text{TimeUnit} < 15) \wedge \\ \text{Student}(\text{S}, \text{Section}) \wedge \\ \text{attendance}(\text{aRFID}_{\text{n}}, \text{Section}) \\ \wedge_{\text{i}=0}^{\text{n}-1} \ \blacksquare \text{done}(\text{S}, \text{aRFID}_{\text{i}}, \text{use}) \end{pmatrix} \rightarrow (\text{autho}^{+}(\text{S}, \text{aRFID}_{\text{n}}, \text{use}) \ \wedge \ (\text{autho}^{+}(\text{S}, \text{aFINGERPRINT}_{\text{n}}, \text{use})$$

$$Policy4 =$$
$$\begin{pmatrix} \blacksquare \text{done } (\text{S}, \text{aRFID}_{\text{n}}, \text{use}) \wedge \\ \Diamond \text{ done } (\text{S}, \text{aFINGERPRINT}_{\text{n}}, use) \end{pmatrix} \rightarrow (\text{Oblig } (\text{Sys}, \text{aDATABASE}, \text{notify})$$

Second Scenario: This is similar to the first scenario, but sometimes, the system randomly chooses some students to authenticate again via the fingerprint reader in order to detect attendance cheating. The fingerprint checker authorizes a student to make him/her present on a database, but if not authorized, the student will be marked absent.

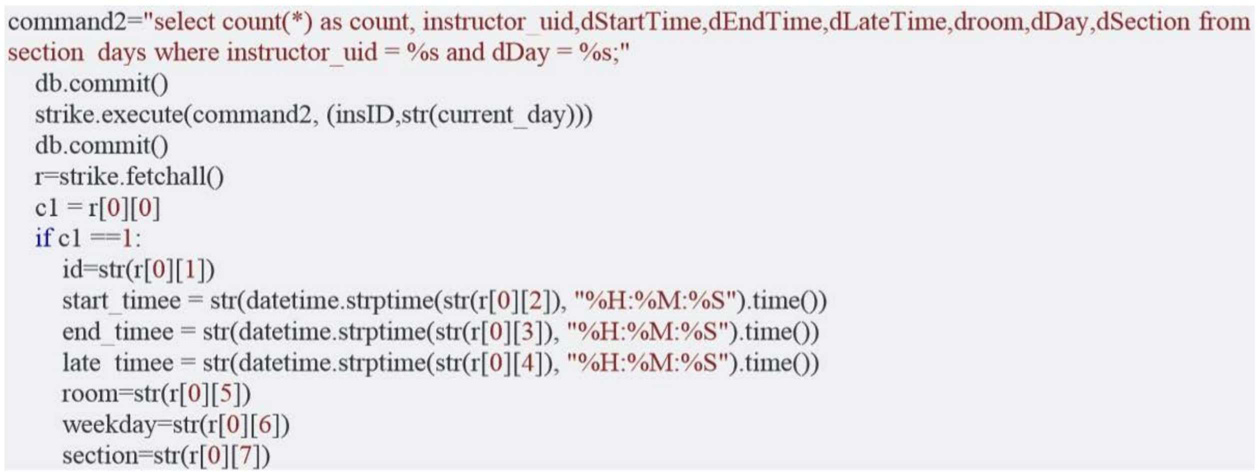
The above policy (Policy3) ensures that a student does not participate in attendance cheating. The cheating will be identified by the monitoring system on the basis of performing attendance by using both RFID and fingerprint technologies simultaneously. After this, an obligation policy (Policy4) will be used to inform the attendance database on when the student is indulging in any improper behavior so that the student's state can be recorded immediately to provide fingerprint authentication for every state next time.

The sequence of activities from students' end is shown in Figure 5. After login by the RFID, the system chooses some students to authenticate again via the fingerprint reader. The fingerprint checker authorizes a student to make him/her present on a database, but if not authorized, the student will be marked absent.

5.4 Implementation

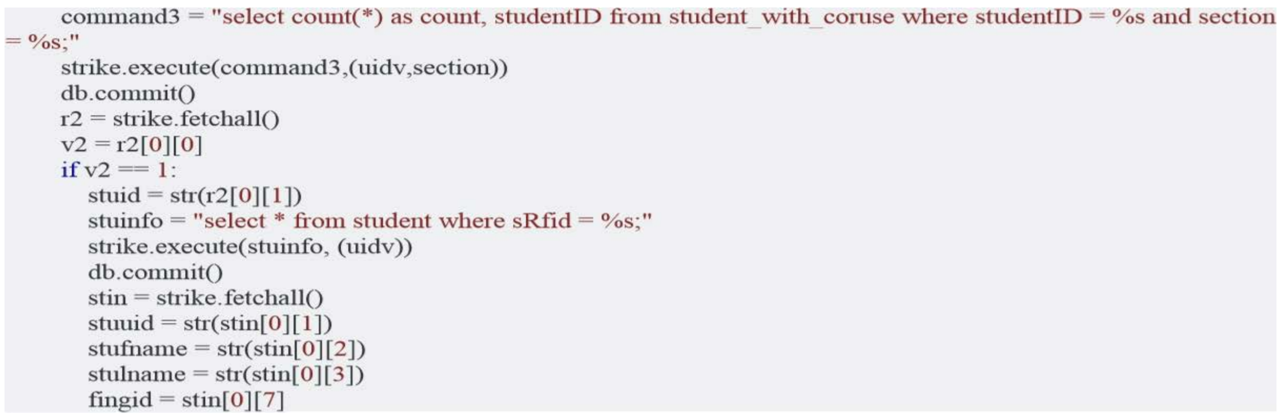
The proposed solution is implemented in Python, and Raspberry Pi devices are utilized to implement the system. All the required libraries that we use in the script, such as RFID, Fingerprint, LCD, and the database, are designed. We define the connection for the LCD screen and check if the fingerprint is connected. We define the new values and determine the current time, current date, current day, and scan for the RFID cards. We store the card number into uidv and make the connection with the database. In Figure 6, we present the sectional information related to an instructor.

In Figure 7, whether a student is in the same section is checked. If he/she is in the same section, we obtain his/her information, such as his/her name, fingerprint, and ID. We also check if the student already took his/her attendance, and store the value in a variable. Similarly, we check if the student has a class. If he/she has a particular class on a particular day, we display the welcome massage for him/her on the screen through the following lines of codes:



```
command2="select count(*) as count, instructor_uid,dStartTime,dEndTime,dLateTime,droom,dDay,dSection from
section_days where instructor_uid = %s and dDay = %s;"
   db.commit()
   strike.execute(command2, (insID,str(current_day)))
   db.commit()
   r=strike.fetchall()
   c1 = r[0][0]
   if c1 ==1:
      id=str(r[0][1])
      start_timee = str(datetime.strptime(str(r[0][2]), "%H:%M:%S").time())
      end_timee = str(datetime.strptime(str(r[0][3]), "%H:%M:%S").time())
      late_timee = str(datetime.strptime(str(r[0][4]), "%H:%M:%S").time())
      room=str(r[0][5])
      weekday=str(r[0][6])
      section=str(r[0][7])
```

Fig. 6 Query to check a section’s information for instructor.



```
      command3 = "select count(*) as count, studentID from student_with_coruse where studentID = %s and section
= %s;"
      strike.execute(command3,(uidv,section))
      db.commit()
      r2 = strike.fetchall()
      v2 = r2[0][0]
      if v2 == 1:
         stuid = str(r2[0][1])
         stuinfo = "select * from student where sRfid = %s;"
         strike.execute(stuinfo, (uidv))
         db.commit()
         stin = strike.fetchall()
         stuuid = str(stin[0][1])
         stufname = str(stin[0][2])
         stulname = str(stin[0][3])
         fingid = stin[0][7]
```

Fig. 7 Query to check a section’s information for student.

if uidv == stuid: ¥newline
print "The ID of the student is matched"
print "Welcome " + stufname + " " + stulname
lcd.clear()
lcd.write¥_string("Welcome " + stufname)
time.sleep(1)
lcd.clear()
rannum = random.randint(0,1)
current_timee =
datetime.now().strftime("¥%H:¥%M:¥%S")
if current_day == weekday:
print "Today, you have a class!" And generate a random number from 0 to 1.

We monitor attendance in case the same student repeats the use of a device for a second-time attendance entry through the following lines of code:
if v1 == 1:
print "You already took your attendance"
lcd.write_string("Already took your attendance")
time.sleep(3)
lcd.clear()
lcd.write_string("Swipe to take your attendance")

The aforementioned code displays the following information on the screen: Already took your attendance.
Similarly, our proposed system checks the current time if it is equal to or larger than the late time for the section and if it is less than the end time. If the random number is 1, it asks for the fingerprint, and if it is not matched, the student is marked absent, and the information is displayed on the screen. If the fingerprint is the same, the system marks the student late and shows that on the LCD. If the random number is 0, the student is marked late. The system also considers those situations in which students attends the class for a few minutes only, that is, they come in very late when the class is about to end. In this situation, students are marked absent.
Likewise, our proposed system has fully automated features to create a new function for inserting student cards and fingerprints into the database. We do not provide here all the codes for the purpose of maintaining clarity.

6. Conclusion

Handling the attendance records of students in any institution, for example, in a university, is very difficult, as it takes up much time and involves substantial paperwork. It is also not possible for a lecturer to supervise all students in the classroom and precisely and efficiently mark the attendance of all students.

This is why installing a system to maintain the attendance records of students more accurately without the need to manually check attendance is imperative. In addition, a system has been developed in this study to automate all attendance-related work and make it online. This system uses RFID and fingerprint software to take attendance, and it provides valuable information that is useful for both students and administrators. In this study, we provided a solution to combine two-factor authentication technology (RFID and Fingerprint) in order to maintain automatic identification for an attendance system. Human fingerprints are rich in details, and they are one of the most popular and accurate biometric technologies. Fingerprint image analysis using automatic identification technology has been developed to the extent that it can be used in a university to maintain an attendance system, and it is far better in terms of cost and time than the manual method.

References

[1] L. F. Cranor and S. Garfinkel, Security and Usability: Designing Secure Systems that People Can Use. CA: O'Reilly Media, Inc., 2005, pp. 1–740.

[2] E. De Cristofaro, H. Du, J. Freudiger, and G. Norcie, "A comparative usability study of two-factor authentication," arXiv preprint , vol. 1, Sept 2013.

[3] C. A. Fidas, A. G. Voyiatzis, and N. M. Avouris, "When security meets usability: A user-centric approach on a crossroads priority problem," in the 2010 14th Panhellenic Conference on Informatics, 10-12 Sept, Tripoli Greece, IEEE, 2010. pp. 112–117.

[4] C. L. Paul, E. Morse, A. Zhang, Y.-Y. Choong, and M. Theofanos, "A field study of user behavior and perceptions in smartcard authentication," in the IFIP Conference on Human-Computer Interaction, 5-9 Sept, Berlin, Heidelberg. Springer Berlin Heidelberg, 2011. pp. 1–17.

[5] L. O'Gorman, "Comparing passwords, tokens, and biometrics for user authentication," Proceedings of the IEEE, vol. 91, pp. 2021–2040, Dec 2003.

[6] C. B. Chew, M. Mahinderjit-Singh, K. C. Wei, T. W. Sheng, M. H. Husin, and N. H. A. H. Malim, "Sensors-enabled smart attendance systems using nfc and rfid technologies," Int. J. New Comput. Archit. Appl, vol. 5, pp. 19–29, Feb 2015.

[7] C. Gilbert-Rolfe, "Radio-frequency identification," Routledge, 2017, pp. 81–90.

[8] F. M. M. Al-Naima and H. A. Ameen, "Design of an rfid based students/employee attendance system," Majlesi Journal of Electrical Engineering, vol. 10, p. 23-33, March 2016.

[9] N. Irfan, M. C. Yagoub, and K. Hettak, "Design of a microstrip-line-fed inset patch antenna for rfid applications," International Journal of Engineering and Technology, vol. 4, p. 558, Oct 2012.

[10] C. Saraswat and A. Kumar, "An efficient automatic attendance system using fingerprint verification technique," International Journal of Computer Science and Engineering, vol. 2, pp. 264–269, Feb 2010.

[11] "Nevon solutions, student attendance with fingerprint reader," Sep 2015. [Online]. Available: http://nevonprojects.com/student-attendance-with-fingerprint-reader. [Accessed: Sept. 30, 2018].

[12] "Original rfid smart gate reader: Rfid walk through smart gate reader," 2017. [Online]. Available: http://www.childsafetyindia.com/home2.html/. [Accessed: Sept. 30, 2018].

[13] A. Cau and B. Moszkowski, "Interval Temporal Logic," 2006. [Online]. Available: https://www.cse.dmu.ac.uk/STRL/ITL/index.html. [Accessed: Sept. 30, 2018].

[14] H. Janicke, "The development of secure multi-agent systems," PhD thesis, De Montfort University, Leicester England, 2007.

[15] R. Kowalski and M. Sergot, "A logic-based calculus of events," New Gen. Comput., vol. 4, pp. 67–95, Dec 1986.

[16] S. Zhou, H. Zedan, and A. Cau, "Run-time analysis of time-critical systems," Elsevier North-Holland, Inc., vol. 51, pp. 331–345, May 2005.

Ali Alzahrani is an Assistant Professor at Islamic University of Madinah. He is Vice-Dean of Computer and Information System. His research interests include computer security, distributed system, IoT and blockchain.