# REMOTELY VIEW USER ACTIVITIES AND IMPOSE RULES AND PENALTIES IN A LOCAL AREA NETWORK ENVIRONMENT

Marisa Buctuanon, Roldan Debalucos and John Glenn Villalon

University of San Jose – Recoletos, Cebu City, Philippines

***ABSTRACT***

*Monitoring whether a student is opening other files other than the allowed file/s during an examination period is really a challenging task. The art of cheating in a written examination is also pertinent to a computer-based examination. Since the proctor is not at all time looking at the student's computer activities, students still have the chance to open restricted applications. In this hurdle, developing a system that will send to the proctor's machine the list and screenshots of the classified computer programs accessed in every student machine in real-time is beneficial. This system will detect the deceitful act through a warning message. If the act is not discontinued, the student machine will be penalized by either be locked, restarted, shutdown, sleep, or hibernate depending on the penalty set by the proctor. Furthermore, this system also records the student attendance during digital exams or as the students log-on to their machine.*

***KEYWORDS***

*Impose penalties to Machine, Monitor Restricted Files or Documents, LAN, Remote access, Computer Activities, Cheating Detection*

## 1. INTRODUCTION

E-assessment is already widely used to evaluate the student learning in which acquiring rich source of data to design a formative intervention to improve student learning is feasible [1]. However, digital cheating is also common to all, where students find ways to cheat using computer technology [2]. This kind of cheating could be done by accessing other websites, communicating with others via instant messaging tools or email during an exam, seeding test computers with answers, and bringing in non-exam disks containing solutions to the exam [3]. Additionally, since the students are already using digital assessment, it will be favorable to integrate attendance monitoring to the same system. Students who incurred a lot of truancies have fewer opportunities to learn the materials provided by the proctor and achieve better scores in exams [4]. Unfortunately, using conventional method of taking attendance is tedious, and cumbersome process since the proctor still needs to call the names of students and record them in paper or input them on their machines [5]. Thus, developing a system where monitoring the user activities in the machine and managing student's attendance in one system is advisable.

There are already studies related to monitoring and controlling network activities in real-time. One of them is the study of [6] where network activities represented in packets, such as URLs accessed, chat rooms visited, e-mails sent and received, instant messaging (IM) sessions, and among others are monitored. In this study, these network activities are monitored and controlled via Internet and these activities can be blocked or not based on the predefined rule set by the controlling mobile communications device(s). A similar system is also speculated which does not

DOI: 10.5121/ijcsit.2018.10607 

only monitor and capture network activities but also produces network activity reports at predetermined time [7]. Another computer activity monitoring and recording system is also conscripted where the computer user activities are being recorded and monitored by a supervisor [8]. There is a digital certificate that will allow the monitoring machine to monitor and record the activities according to the policies comprised in the certificate. Additionally, a system solely for monitoring an activity is also made available [9]. This system measures the activity rate and warns the machine that the activity reaches the predetermined level. An automatic method for cheating detection in online exams is also developed where different images of students are recorded. The images are processed and analyzed to compare with students' images at different times of exam [10]. With regards to the checking of attendance, there is a proposed framework where managing the students' attendance is done with the use of RFID technology [11]. Radio Frequency Identification (RFID) technology is equipped with circuits that gain power from radio waves emitted by readers in an environment and can be used as an information to a variety of purpose [12].

All the related studies have their own unique features and purposes. To enumerate them, the two systems designed in [6] and [7] allow the communication of network activities through internet. However in this paper, any communication during an examination should not be permitted. Also, in [8], it focuses on creating a digital trust between two computer machines used by the user and the supervisor which is not the focus of this paper. Furthermore, the third system is different from this paper since it checks only the time spent on an activity in the machine. Though a method for cheating detection is developed, it takes time to detect the dishonesty done by the students since the images retrieved still needs to be processed and analyzed only after the exam. The proposed framework of [11] only checks the students' attendance with the use of RFID.

In this paper, a system that enables the proctor to view the list of restricted files or apps accessed by the students during an examination period or a controlled session is made possible. The number of attempts in accessing these files or apps is recorded and will have corresponding points for deduction. The proctor sets first the number of warnings that determines the number of allowable attempts. If the student reaches the final warning, the student machine will be penalized. The penalty could either be lock, restart, shutdown, sleep, or hibernate depending on the value set by the proctor. Additionally, the system will not just check the apps accessed by the student but also, the students' presence. In this way, the proctor will no longer manually check their attendance.

## 2. REVIEW OF RELATED LITERATURE

Simple network management (SNMP) is a network management protocol that is used to collect status information and/or configuration information from devices connected to a network such as an intranet [13]. This information or data that is available on a device is defined in an abstract data structure known as a Management Information Base (MIB) [14]. This information can be parsed and filtered and be sent to the database server. It is now up to the server on what to do with the information. Additionally, it is necessary to have an ease in monitoring the different applications running in every client machine without manually initiating it. To do this, a task scheduler is required. This will initiate the running of scrips or programs to run in the background [15] Furthermore, to interact automatically to an operating system, a command line tool is needed. One of which is PowerShell. It performs administrative tasks on a machine just like having an administrative account where restrictions are nothing [16]. There are a lot of scripting languages that can work as a backend to access the database and connect to the browser. To name one is PHP. PHP or PHP: Hypertext Processor is a widely-used open source general-purpose scripting language that is suitable for web development [17].

### 2.1. Architectural Design

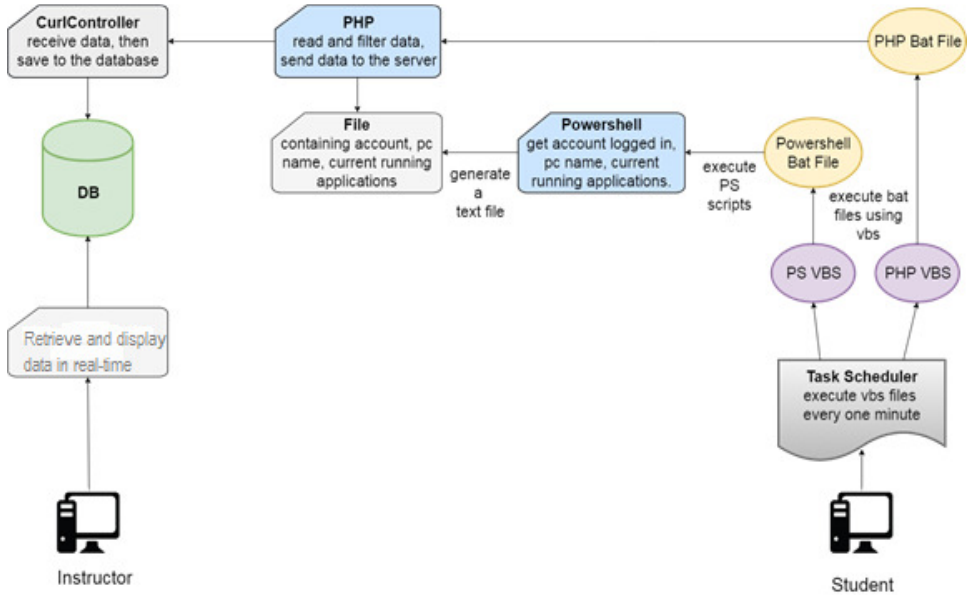


Figure 1. LookOut Architectural Design

LookOut is the name of the application introduced in this paper. Figure 1 depicts the interaction of the different modules in the LookOut system, from the client machine to the monitoring machine. These modules are installed in both entities where client has to have five modules which include the task scheduler, VB script, bat files, PHP script, and PowerShell. The task scheduler should be configured first to create an event or initiate the prerequisite scripts to trigger the next needed scripts. A one minute is set to execute the created event. The created event triggers the VB Script to generate bat files. These bat files are responsible for running the PowerShell and PHP scripts automatically. The PowerShell is designed to get the logon account, personal computer (PC), and the running apps that the user opened at the present moment. These data will be saved to a text file. The text file generated by the PowerShell will now be parsed by the PHP script. This script contains cURL that will pass the parsed data to the server. Furthermore, the server side has a Curl Controller module that checks the incoming data from the client side. These data will then be saved to the database to be consumed by the web application of the system.

## 3. METHODOLOGY

The system utilizes Apache HTTP Server to run the web services of system. This is run via XAMPP. These web services are developed using PHP. With the use of Laravel 5.3, the development follows a model view controller (MVC) architectural pattern. The web application is also developed using PHP. To implement the dynamic retrieval and display of data in the web application, JavaScript is used. On the other hand, the client side uses task scheduler to trigger the launching of the first scripts that will automatically call the other scripts in the client side. Furthermore, the PowerShell is used to retrieve the data needed by the system from the computer unit itself. These data will then be processed and passed to the server via PHP cURL.

### 3.1. Getting and Passing Installed Programs From an Active Machine to the Server

The system needs to get the installed programs of all the computer units connected to a distributed network environment to know which applications to allow during a certain exam session. The

student machine should have the scripts such as the VB and PHP scripts. Running these scripts automatically is initiated by the Task Scheduler. The student machines should be turned on to be able to send data to the server. In retrieving the installed programs in each machine, the PowerShell is configured to run the task list script every 1 minute. This script will retrieve all the installed programs of that machine and generates a text file to save the names of these applications. This file will then be parsed by PHP cURL and will be passed to the database server.

### 3.2. Recording the Student Attendance and Status

With the information retrieved after knowing the programs installed in every student machine, the sever in the proctor machine will also check which accounts have sent information. This determines the attendance of the student, whether they are absent, present, or late. Assigning an account in each student is already set by the proctor beforehand. It is not just the attendance of the student that can be generated out from the retrieved information, but also the number of violations committed with their corresponding screenshots. The proctor also sees the number of remaining warnings the student has and/or the penalty incurred by the student.

### 3.3. Getting and Checking Violations with Screenshots

The same with the retrieval of installed apps, the system uses PowerShell ISE to get the running applications in every workstation. Using the get-process script in PowerShell, the system retrieves the running applications and uses out-file script to generate a text file. Additionally, a screenshot is made to show proof that the said running applications are accessed by the students. This information will then be saved to the database server. After sending the data to the server, the system checks what applications are allowed at the current exam session. The exam session is set beforehand by the proctor. If there are running applications that are prohibited, the system increments the violations taken by the student. Applications that are already recorded during the last check and are still opened during the next round of checking will be considered as another violation.

### 3.4. Activating Warnings and Penalties

If the number of violations reaches the number of warnings, then penalties will be imposed. Penalties can be lock, restart, shutdown, sleep, or hibernate. This automatically happens after the script in the student machine has match the account logged on to it and the one warned or penalized by the server. The server writes the user account of the student to the warn script and the script in the student workstation checks whether the log in account exist in the warn script. If it exists, the warning message script installed in the student workstation will automatically execute. Just like in the implementation of executing a warning script, the penalty script in the student workstation will also check whether the log in account exist in the specific script in the server side. The server posts the user account of the student who reaches the limit of warning to the penalty script which will then be retrieved by the penalty script in the student workstation.

# 4. RESULTS AND DISCUSSION

## 4.1. User Interface

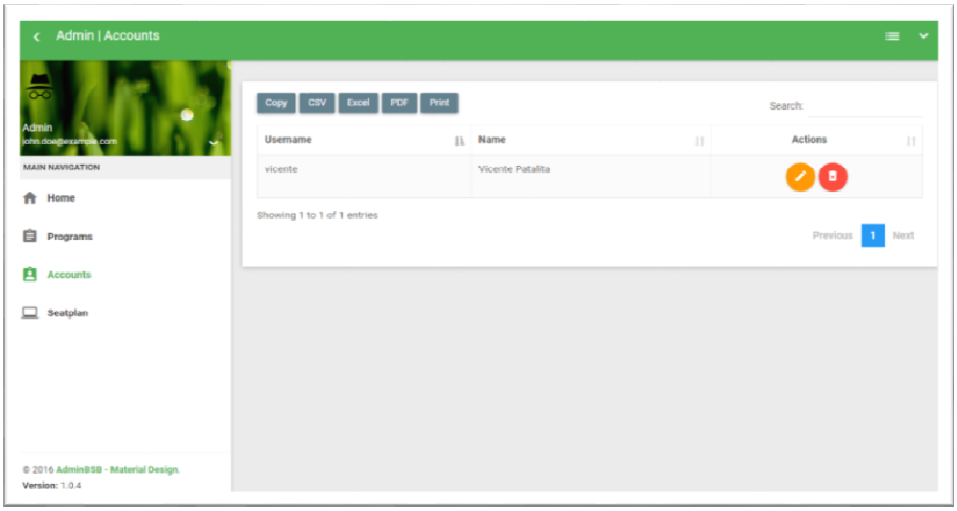


Figure 2. Proctor's Landing Page

Figure 2 is the interface where the proctor can upload a csv file with the list of students enrolled in a certain class with their assigned user account. The proctor can create multiple classes with their respective laboratory and the system identifies which student machine belongs to a certain laboratory. The proctor can also manage the accounts of the students here whether to add, edit, or delete them.

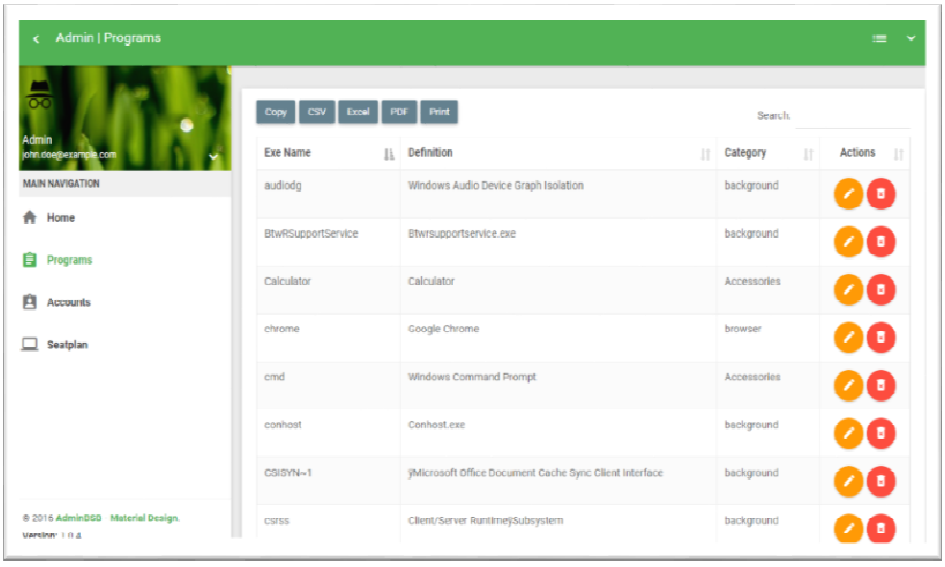


Figure 3. Manage Application Page

Figure 3 shows the list of unique applications retrieved from all the student workstations in a certain class or laboratory. In this paper, a class is associated to one laboratory. This is to retrench

the chance of students accessing a computer in another laboratory and be able to have an attendance.

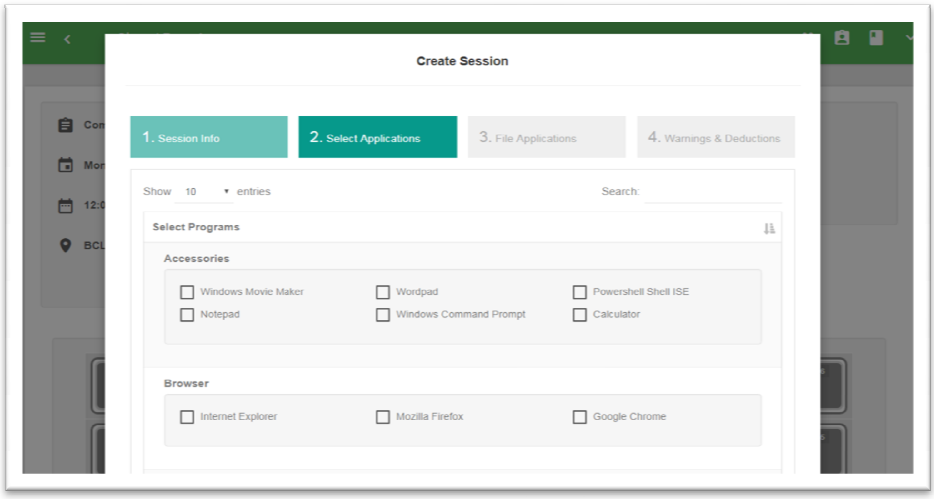


Figure 4. Create Session Page

Figure 4 is where the proctor can set configurations on the type of files restricted for a certain duration of time with their corresponding allowable attempts and penalties. First, the proctor needs to fill out the session info such as session name, date to be activated, start time and end time. This means that the checking will only happen during the set schedule. The next thing to do is to select the set of allowable applications, and this would mean that other than the selected ones, they are all restricted. The third step is to input the file names or URL of their allowed application. This is to ensure that even if the student is accessing the same type of application it should have the same name or URL as instructed. It is still a violation if student is accessing different file name or URL. The last step is to provide the number of warnings or allowable attempts, equivalent deduction for each warning and the penalty to be taken after reaching the final warning.

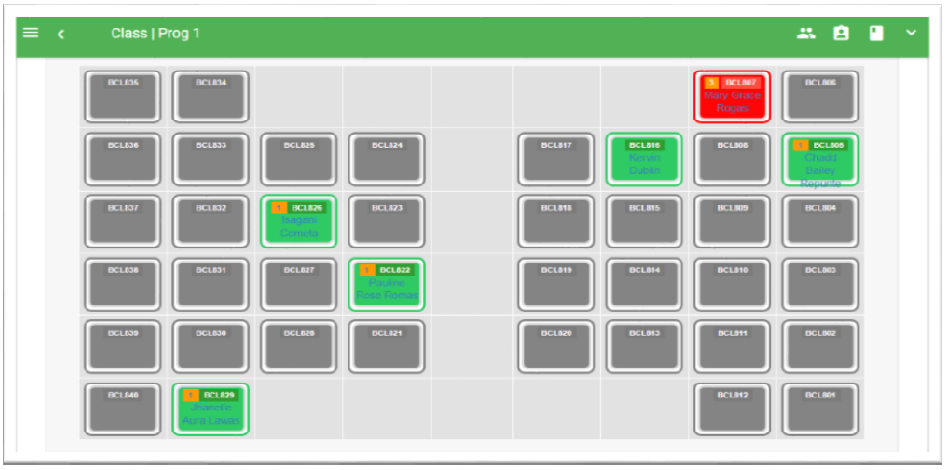


Figure 5. Seat Plan Page

Figure 5 is where the proctor can visually see a laboratory setup where the student is physically seated. This seat plan can be changed by dragging and dropping the individual square on the grid. This means that this setup can be modified in all other laboratories. However, this can also be used as a template in other laboratories. Visual cues are also shown to determine who is present, absent, late, committed a violation and/or penalized. As shown in Figure 5, the color green indicates that the workstation is active and this means that the student who is log on to the account is present. The color gray indicates that no one is using the workstation. The orange color highlights the number of violations committed by the student. And lastly, the red color indicates that the student is already penalized.

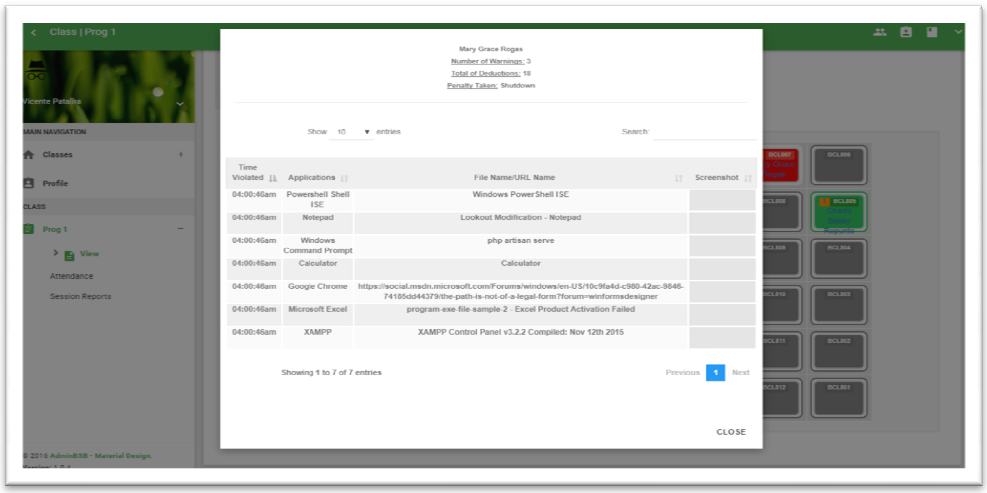


Figure 6. Student Machine Activity Log

When the proctor clicks any of the squares which represents a student in the seat plan, Figure 6 will be shown. This is where the proctor can see the log of all the violations done by the student during an exam. Each violation has a corresponding screenshot which can be shown upon clicking the rightmost part of the table.

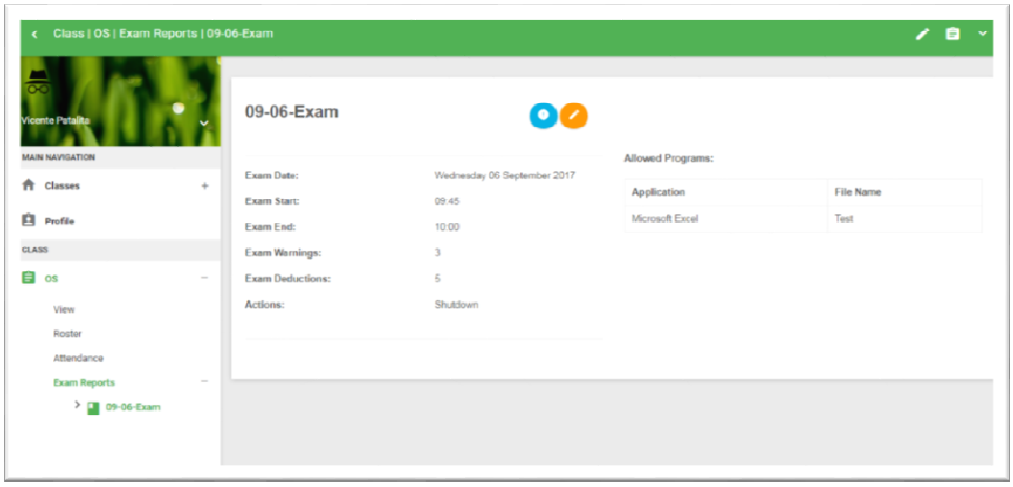


Figure 7. Session Report

Figure 7 shows an example of a session report where the proctor can see examination date, start time, end time, number of warnings, and equivalent point for deduction, penalty, and the allowed applications. The proctor can also see the list of students with their total points to be deducted from their examination score by clicking the view button (blue) just before the edit button (orange).

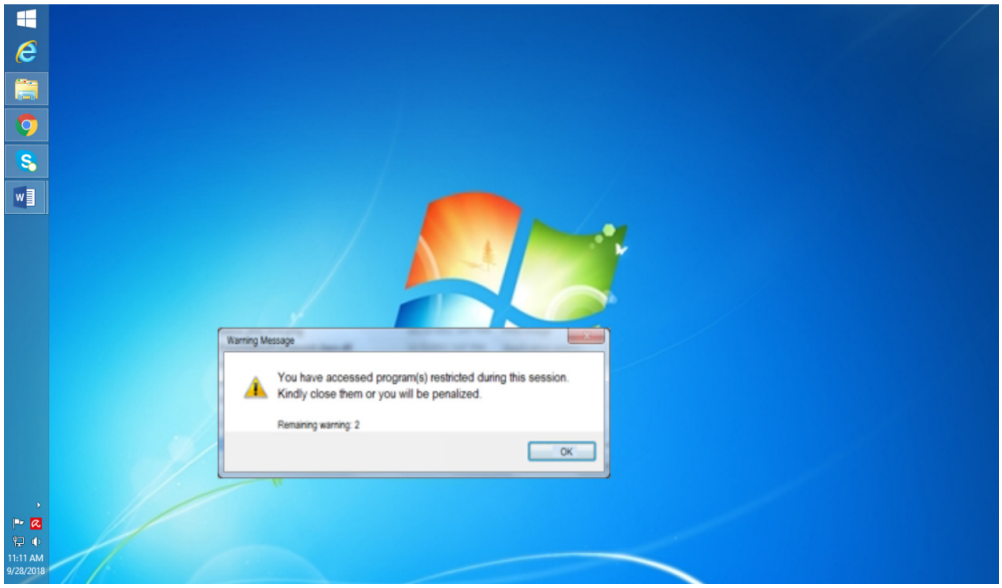


Figure 8. Student Machine Warning Message

Figure 8 is an example of a warning message taken from a student machine where the student committed a violation by opening a restricted file. It shows here that the student has 2 remaining attempts and if he will not close the restricted file or stop accessing this file he will eventually be penalized.

## 4.2. Performance Testing

Table 1. Generate files from the student machine.

| **Test Case 1: Generate files from the student machine** | **Average Time** |
|---|---|
| Retrieved current logged-in account | 30.48 seconds |
| Retrieved computer name | 1.03 seconds |
| Retrieved all the installed programs | 1.04 seconds |
| Retrieved the accessed URL | 1.1 seconds |
| Retrieved the running applications | 1.05 seconds |
| Generate screenshot image | 1.03 seconds |
| **Total average time:** | **5.96 seconds** |

The system is dry run first to check whether it can be deployed to multiple laboratories or not. In this case, one laboratory is identified to dry run the system. The processes in Table 1 only happen at the student machine. All the retrieved information are saved in a text file except for the screenshot. After saving, these files are then sent to the server. With the list of average time shown in Table 1, it tells that all the scripts responsible for each operation are running efficiently. Only the retrieval of the current logged-in account seems to be slow. This happens since the machine has just get started and there are some background processes done by the operating system.

Table 2. Script execution.

| **Test Case 2: Script Execution** | **Average Time** |
|---|---|
| Execute scripts using Task Scheduler | 10 seconds |
| Sending files from the student machine to the server | 13.33 seconds |
| Elapsed time in prompting warning message | 10 seconds |
| Elapsed time in penalizing the student machine | 10 seconds |
| **Total Average Time** | **10.83 seconds** |

Table 2 shows the result of the execution of task schedule, which happens at the very start before the processes in generating files are done, and all other scripts interacting with the server machine. The total average in executing these scripts are slower than generating files in Table 1 since these processes still need to transfer the data from one machine to another. Unlike these results, the processes in Table 1 are just retrieved and saved temporarily in the same machine.

## 4.3. Summary of Findings

In developing the LookOut system, there are interferences that need to have a workaround or a solution in order to deliver the system's requirements and purpose. One of these is the use of PowerShell ISE. This tool will make the accessing of information in the machine automated. By default, it has many restrictions. The first time executing a PowerShell script in the student machine gives a "Set-Execution Policy" error. To solve this, the "Set-Execution Policy Unrestricted" cmdlet code was run first in order to give authority to any script to run automatically in the machine, and which should be run as administrator. All the scripts created should be made directly from the PowerShell ISE. Otherwise, they will not be read. Additionally, the text files generated by the PowerShell have unnecessary binary codes or hex encoded string that needs to be parsed, and removed or replaced first before passing them to the server. To do this, php preg_replace was used. Furthermore, there are some incompatibility issues in some browsers in which the system has a problem retrieving the URL accessed by the student. Although, the system can identify the type of browser that is opened, the system needs to get the URL since the proctor can still let the students accessed URLs which are permitted by them. Only Google Chrome and Mozilla Firefox are the recommended browsers to use to make the feature work.

## 5. CONCLUSIONS

The overall objective of the study is to let the proctor impose rules and regulations on every student machine during an examination period. To be able to achieve this, different scripts were created and deployed to the student machine to retrieve computer name, installed programs, current user logged-in and screenshot. Batch file scripts are also created to be executed by PowerShell. Furthermore, PHP scripts are successfully coded to read and send generated text files from the student machine to the server. Action scripts for shut down, lock, restart, hibernate, sleep and warning message are also implemented properly. The system effectively use the Task Scheduler to automate the execution of visual basic scripts. The system has used C# language to retrieve current URLs in the browser. Integrating attendance monitoring to this automated method for cheating detection is also made possible in this study. Thus, this study will truly make a great impact to the academe. This system will help the instructors or professors to further boost productivity and focus on other tasks at hand.

## REFERENCES

[1] T. Vendlinksi and R. Stevens, "Assessing Student Problem-Solving Skills With Complex Computer-Based Tasks," The Journal of Technology, Learning and Assessment, vol. 1, no. 3, 2002.

[2] F. Rogers, "FACULTY PERCEPTIONS ABOUT E-CHEATING DURING," Journal of Computing Sciences in Colleges, vol. 22, no. 2, pp. 206-212, 2006.

[3] R. Baker and R. Papp, "Academic integrity violation in the digital realm," in Southern Association for Information Systems 2003 Annual Conference, 2003.

[4] J. L. EPSTEIN and S. B. SHELDON, "Present and Accounted for: Improving Student Attendance Through Family and Community Involvement," The Journal of Educational Research, vol. 95, no. 5, pp. 308-318, 2002.

[5] Shailendra, M. Singh, A. Khan, V. Singh, A. Patil and S. Wadar, "Attendance Management System," in Electronics and Communication Systems (ICECS), 2015 2nd International Conference on. IEEE, 2015.

[6] S. Chuang Li.United States Patent US 2004/0260801 A1, 2004.

[7] L. A. H. Tsung-Yen Dean Chang, S. Chuang Li and F. Bo Xiong.U.S. Patent 10/366,028, 2004.

[8] C. Zezhen Huang, "TRUSTED COMPUTER ACTIVITY MONITORING AND RECORDING SYSTEM AND METHOD". United States Patent US 2006/0041760 A1, 23 Feb 2006.

[9] B. J. Gould and S. D. Rudnik, "Computer Activity Monitoring System". United States Patent US006065138A, 16 May 2000.

[10] K. Jalali1 and F. Noorbehbahani, "An Automatic Method for Cheating Detection in Online Exams by Processing the Student`s Webcam Images," in 3rd Conference on Electrical and Computer Engineering Technology (E-Tech 2017), Tehran, Iran, 2017.

[11] P. Rajan, P. Nimisha and G. Mona, "Online Students‘ Attendance Monitoring System in Classroom Using Radio Frequency Identification Technology: A Proposed System Framework," International Journal of Emerging Technology and Advanced Engineering, vol. 2, no. 2, pp. 61-66, 2012.

[12] K. Finkenzeller, RFID Handbook: Radio-Frequency Identification Fundamentals and Applications, New York: Wiley, 2000.

[13] C. Tetsuro Motoyama and C. V. Avery Fong.United States Patent US 2002/0152292 A1, 2002.

[14] M. Roughan, "A Case Study of the Accuracy of SNMP Measurements," Journal of Electrical and Computer Engineering, 2010.

[15] T. L. Harris, -J. Peyton, J. R. Howell and J. R. Douceur.United States Patent US 7,716,249 B2, 2010.

[16] R. Siddaway, PowerShell and WMI, Manning, 2012.

[17] The PHP Group, "PHP," The PHP Group. [Online].

## Authors

**Marisa Buctuanon** Is an instructor in University of San Jose – Recoletos, Philippines teaching Artificial Intelligence, C# Programming, and all courses related to Computer Science. She is a member of Recoletos Educational Assistance for Deserving Students. She graduated Bachelor of Science in Computer Science major in Natural Language Processing, Cum laude. She finished her Master in Computer Science in Cebu Institute of Technology - University.

**Roldan Debalucos** Graduated Bachelor of Science in Information Technology in University of San Jose- Recoletos last March 2018.

**John Glenn Villalon**Graduated Bachelor of Science in Information Technology in University of San Jose- Recoletos last March 2018.