# **Royal University of Phnom Penh**

## **Faculty of Engineering**

### **Department of Information Technology Engineering**

**Report of Automator**

1. **Introduction:**

It’s very convenient to have a schedule, so, people could plan and execute tasks more accurately. We can eliminate everyday tasks’ redundancies by making a schedule and try to understand what are our habits and plans. So, that’s why this **Automator** comes in handy.

1. **Objective:**

The main goal is to help people with their habits by allowing them to execute certain tasks at a specific time. Currently, there’s this mode called “Do not disturb” (DND) or “Focus mode” (Windows 10) that allow users to turn off all notification sounds and notification banners. However, this is still a problem as accident could happen. For instance, you have your Do DND turned on scheduled within your work times. You could still get sounds because you trusted that DND mode to suppress all notification sounds from your computer but your master volume will still allow your computer to blast some sound from, for example, video ads on the internet. One of the other objectives is that, people could execute/open a specific software at a certain schedule. Imagine this, you came back home and be ready to finish your work from your workplace and you set the **Automator** to launch your favorite music player, your software for work environment with browser at the same time. You could set **Automator** to launch/behave like that everyday which will save your time and probably be a reminder that it’s time to do something. Last but not least, advance users could execute scripts at a certain schedule. Please also take note that when I mentioned “schedule” I was mentioning about making a schedule when a specific time (Time-based schedule) is met or at a specific battery level is met(Battery-based schedule).

1. **Result:**

Currently, this **Automator** is close from what the objective is. MacOS users could fully use the time-based schedule while most UNIX system could use time-based schedule with executing script and set a power state. Microsoft windows will not be supported at the moment due to time and resource constraints within this development. So far, these are the current features:

* 1. **Set master output volume:**

This feature will set the master output volume to a specific level ranging from 0% to 100%.

* 1. **Set system power state:**

This feature currently supports powering off and rebooting the system.

* 1. **Launch executable software:**

This will launch .app extensions application on MacOS and .exe on Windows.

* 1. **Execute scripts:**

This will execute .sh scripts on UNIX-based system and .bat file on MS Windows.

Currently, tasks are using the bash shell to execute certain tasks. On Apple MacOS, there’s this utility called “osascript” that uses JavaScript OSA to interact with the MacOS system. Some of the other feature like executing scripts and .app could be executed by calling the “./FILE\_PATH” command. The current project is only a demo to what features are going to be on the final development.

1. **Conclusion:**

Making an automation software to do something for you is a very convenient and time saving considering everyone are busy focusing most of the time on their work, not everybody will bother or even care about the number next to the volume icon on their computer or give a sense about asking themselves “Did I turned off my computer?”