AN ADAPTIVE AND SMART SYSTEM FOR PARENTAL CONTROL ON DIGITAL GAMES

Clark Ren[1], Yu Sun[2] and Fangyan Zhang[3]

[1]Northwood high school, Irvine, CA 92620
[2]Department of Computer Science, California State Polytechnic University, Pomona, CA, 91768
[3]ASML, San Jose, CA, 95131

ABSTRACT

As more and more students get access to computers to aid them in their studies, they also gain access to machines that can play games, which can negatively affect a student's academic performance. However, it is also debated that playing video games could also positively affect a student's academic performance. In order to address both sides of the argument, we can create an app that limits the amount of time a student has to play games while not completely removing the ability for students to play games.

KEYWORDS

Parental Control, Smart System, Digital Games, Web Service

1. INTRODUCTION

As video games grow to be more popular, students will start dedicating more of their time to playing video games [1][2][3]. The debate on whether this is positive or negative has been raging since video games have come out with valid arguments from both sides [4][5]. On one side, games could stimulate a student's mind and be a platform for social interaction, but it could also cause a student to perform poorer academically. Many parents who do not want to see their child perform poorly academically will completely ban video games, which eliminates the possibility for any positive outcomes to come from a child playing games [6]. However, doing the opposite and letting a child play to his or her heart's content will lead to the child neglecting schoolwork and performing worse in school. The real problem linking games and bad grades is the frequency and duration of time spent on video games. If a student was to spend excessive amounts of time playing video games or playing games the day before an important test instead of studying, the student is not going to perform in school as well as he or she could have if the student spent his or her time studying instead of playing. If a student were to occasionally play a few games only during times where there are no upcoming tests, he or she would be able to play some video games without being completely distracted by games around upcoming tests, which eliminates the risk of bad grades playing right before a test brings. However, not all students have enough self-discipline to achieve this, so an alternate method must be used to limit the time a student can use for gaming. One solution that can limit the game time of a student is through the use of an app that allows parents and a student's grades to regulate the time available for a student to play games can be used to control the duration and frequency of a student's gaming [7][8].

2. MOTIVATION AND CHALLENGES

Many students in the modern day require a computer for school so they can do their homework, use a word processing tool to write up an assignment, do research, and do many other things. However, when given a computer, many students will use it to play video games. Although it is widely debated that video games aren't necessarily a bad thing for children, excessive time in front of a game almost certainly is. By using an app that can limit the amount of time a child can play games and automatically close a game when the time limit has ended, a student can spend some time playing games, while still being able to use a computer to do school related tasks. However, a few problems still remain as to how an app that would do this would work [9].

When coding an app that can detect if a student is playing games, we are faced with a few problems. One problem is how the app can determine a game is a game. To the computer, a game is just another process running on the computer, so there isn't a way to determine if a running process is or isn't a game. Because our app has to determine if a process a game in order to close it, this problem must be overcome. Another problem is how the amount of time a student can use their computer to play games is determined. The easiest way to accomplish this is to give a set amount of time for every student. However, even though this accomplishes the goal of setting a time for each student, this method does not reflect their performance in academics. In order to set a student's time based on his or her academic performance, another method must be used.

3. SOLUTIONS

Figure 1 is an overview of the blocker solution. It has 3 components: 1) The desktop application that monitors and blocks the game applications; 2) A web-based system that enables teachers to input grade and push the blocking configurations to the database [10][11]; and 3) A mobile system [12][13] to allow parents remotely monitor and control the blocking status.

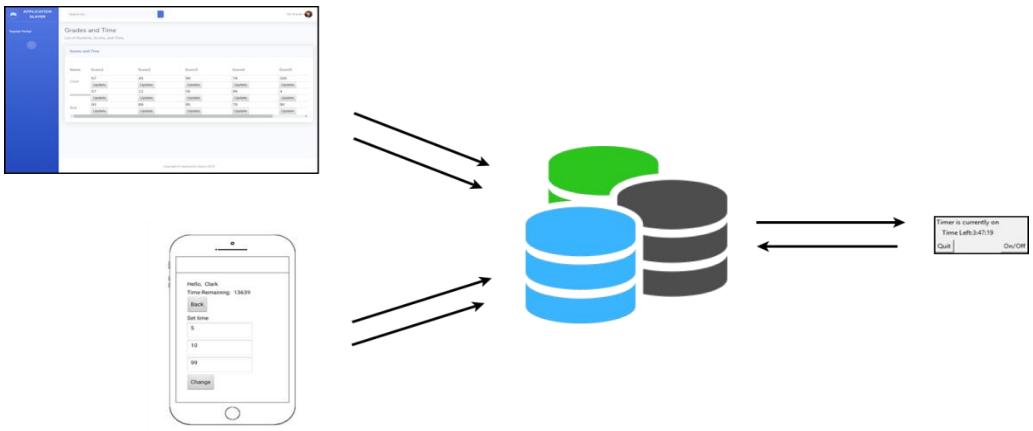
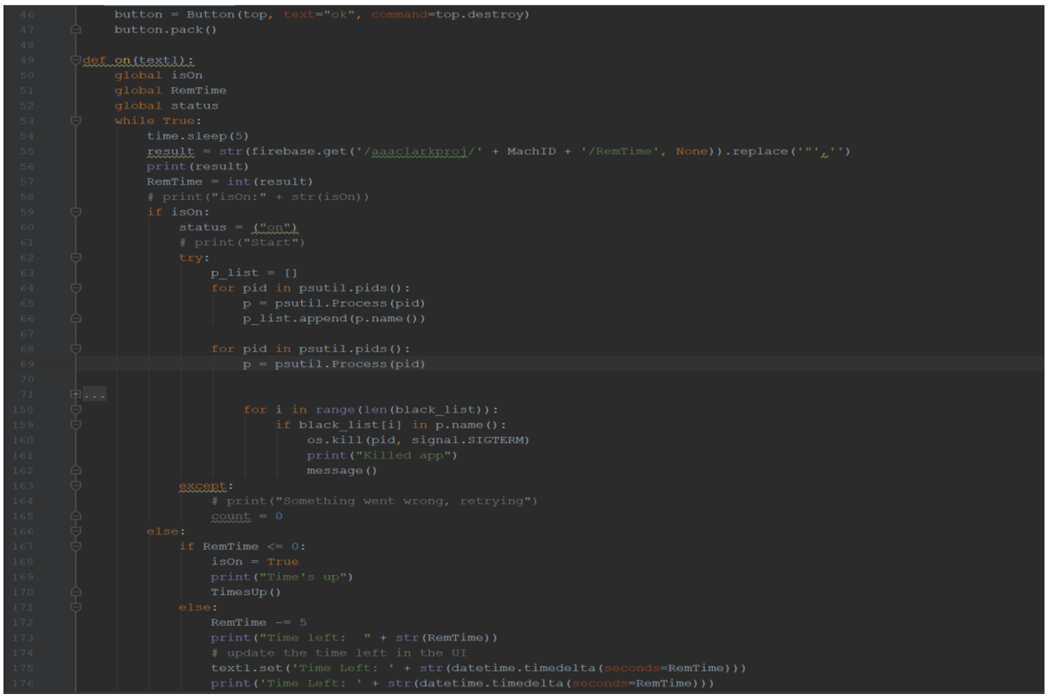


Figure 1. An Overview of the System

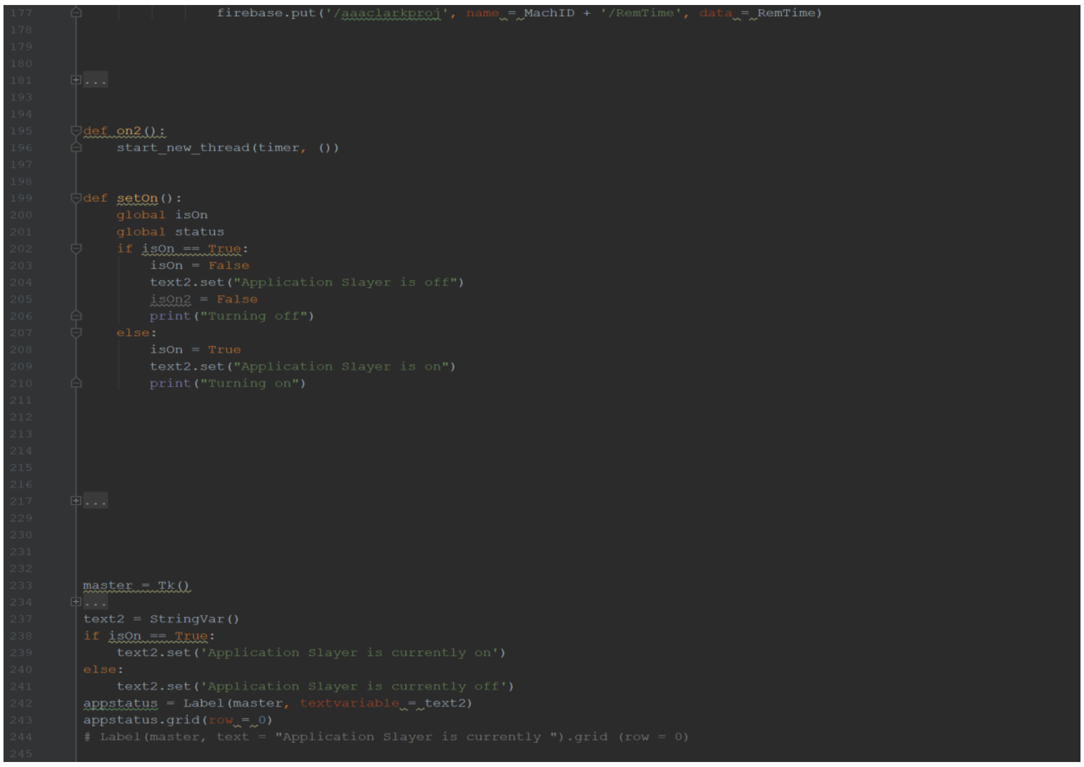
In order to detect if a process that is running on a student's computer is a game, we can store the process names for many games on a database and stop the game from running if the process name for a game is detected running on a computer. Additionally, in order to set the amount of time a student has access to playing games, we can make an app for parents to control the available time a student has to play games and a portal for teachers, where grades can be input in order to determine the time a student has for games based on their performance in school [14][15].



```
    button = Button(top, text="ok", command=top.destroy)
    button.pack()

def on(text1):
    global isOn
    global RemTime
    global status
    while True:
        time.sleep(5)
        result = str(firebase.get('/aaaclarkproj/' + MachID + '/RemTime', None)).replace('"','')
        print(result)
        RemTime = int(result)
        # print("isOn:" + str(isOn))
        if isOn:
            status = ("on")
            # print("Start")
            try:
                p_list = []
                for pid in psutil.pids():
                    p = psutil.Process(pid)
                    p_list.append(p.name())

                for pid in psutil.pids():
                    p = psutil.Process(pid)

...
                    for i in range(len(black_list)):
                        if black_list[i] in p.name():
                            os.kill(pid, signal.SIGTERM)
                            print("Killed app")
                            message()
            except:
                # print("Something went wrong, retrying")
                count = 0
        else:
            if RemTime <= 0:
                isOn = True
                print("Time's up")
                TimesUp()
            else:
                RemTime -= 5
                print("Time left:  " + str(RemTime))
                # update the time left in the UI
                text1.set('Time Left: ' + str(datetime.timedelta(seconds=RemTime)))
                print('Time Left: ' + str(datetime.timedelta(seconds=RemTime)))
```

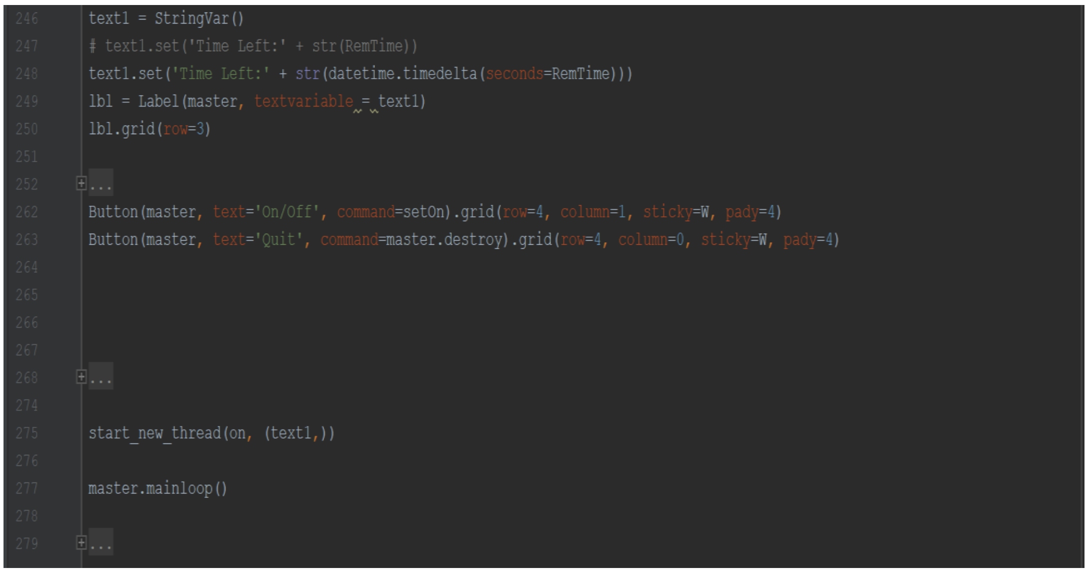
Figure 2. The Desktop Application of the Blocker Program used by Parents

Figure 2 shows the desktop UI of the blocker program that runs on the computers to monitor. Figure 3 shows an excerpt of the code that implements the detection and monitor engine, while Figure 4 shows the code that implements the desktop UI component.



```
                firebase.put('/aaaclarkproj', name = MachID + '/RemTime', data = RemTime)



...


def on2():
    start_new_thread(timer, ())


def setOn():
    global isOn
    global status
    if isOn == True:
        isOn = False
        text2.set("Application Slayer is off")
        isOn2 = False
        print("Turning off")
    else:
        isOn = True
        text2.set("Application Slayer is on")
        print("Turning on")







...




master = Tk()
...
text2 = StringVar()
if isOn == True:
    text2.set('Application Slayer is currently on')
else:
    text2.set('Application Slayer is currently off')
appstatus = Label(master, textvariable = text2)
appstatus.grid(row = 0)
# Label(master, text = "Application Slayer is currently ").grid (row = 0)

```

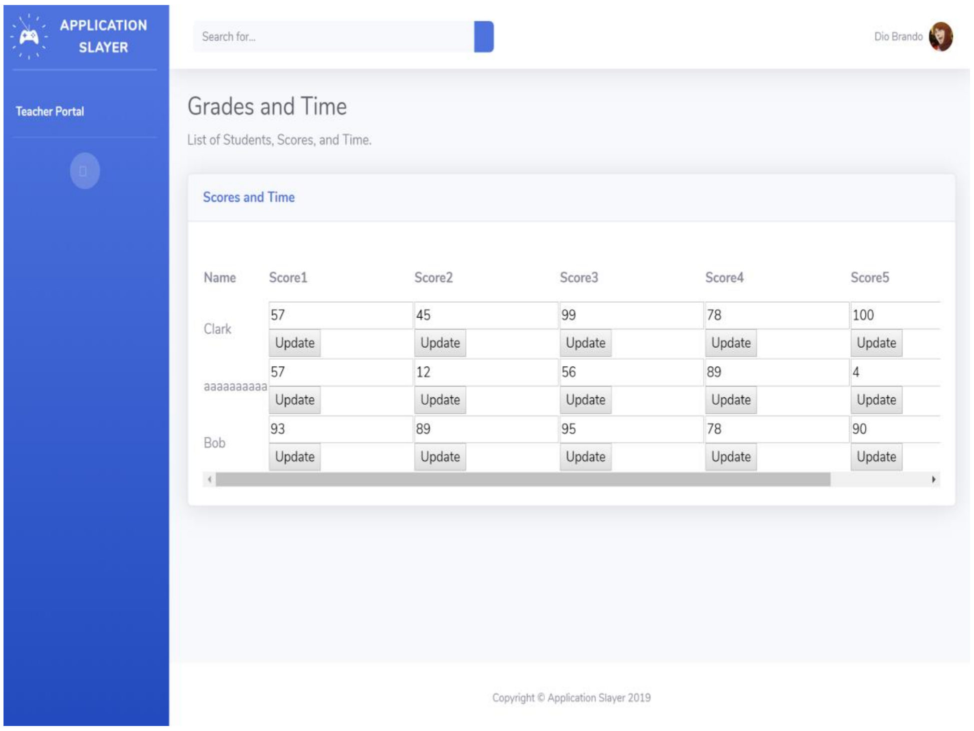
Figure 3. A Code Excerpt of the System Process Detection and Blocking



```
text1 = StringVar()
# text1.set('Time Left:' + str(RemTime))
text1.set('Time Left:' + str(datetime.timedelta(seconds=RemTime)))
lbl = Label(master, textvariable = text1)
lbl.grid(row=3)

...
Button(master, text='On/Off', command=setOn).grid(row=4, column=1, sticky=W, pady=4)
Button(master, text='Quit', command=master.destroy).grid(row=4, column=0, sticky=W, pady=4)




...

start_new_thread(on, (text1,))

master.mainloop()

...
```

Figure 4. An Excerpt of the Code that Implements the Desktop UI System

The app works by checking the processes running on the computer against a database of the process names of certain games. When a game is detected, the app will automatically terminate the process. In order to set a time limit on a student, we keep the time remaining for every student stored on a database, which can change depending on the scores, which are input into a portal by a teacher, or they can be changed by a parent. When the timer is turned on, the app stops terminating processes to let a student play their games, while counting down from the time remaining. When the time remaining becomes zero, the timer stops and the app starts terminating processes again.



Grades and Time

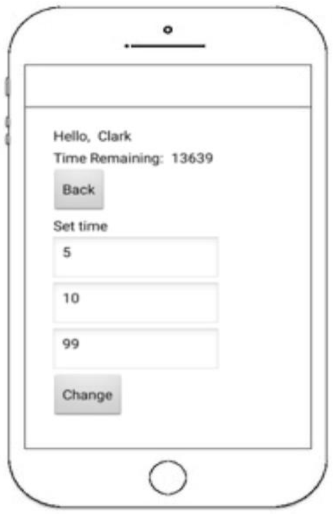
List of Students, Scores, and Time.

Scores and Time

Name	Score1	Score2	Score3	Score4	Score5
Clark	57	45	99	78	100
aaaaaaaaaa	57	12	56	89	4
Bob	93	89	95	78	90

Copyright © Application Slayer 2019

Figure 5. The Web UI of the Blocker Program used by Teachers


Hello, Clark

Time Remaining: 13639

Set time

5

10

99

Figure 6. The Mobile UI of the Blocker Program used by Parents

4. RELATED WORK

There are many existing parental control apps out there such as Windows Live Family Safety (), Norton Online Family (), or Family Shield. What makes our app stand out is that rather than completely locking a student out of their computer after a time limit has ended or only blocking apps outside of the Windows Store, like Windows Live Family Safety, or only blocking and monitoring traffic to websites, like Norton Online Family or Family Shield by OpenDNS, our app blocks games from running on a computer rather than blocking websites and monitoring internet traffic, unlike Norton Online Family, Family Shield and Windows Live Family Safety, a student is free to continue using his or her computer to study after the time limit has finished.

5. Conclusion and Future Work

In this paper, we have created an app that blocks games from running on a computer after a period of time determined by a student's academic performance, which stands out from existing parental control apps that already exist. To address the problems, we encountered while designing the app, we created a system where a game's process name is checked with a database full of the process names of various games and we developed a system where a student's academic performance affects the amount of time is available to them for playing games. However, some problems still remain. Some games may not be in our database, so a student can play them as much as they want. To combat this, artificial intelligence can be used to track a student's behavior on their computer and if the behaviour is suspicious, it can add the process name of the game into the database, making it unplayable with the timer off.

References

1) Baumrind, D. (1966). Effects of authoritative parental control on child behavior. *Child development*, 887-907.

2) Baumrind, D. (1968). Authoritarian vs. authoritative parental control. *Adolescence*, *3*(11), 255.

3) Chao, R. K. (1994). Beyond parental control and authoritarian parenting style: Understanding Chinese parenting through the cultural notion of training. *Child development*, *65*(4), 1111-1119.

4) Gros, B. (2007). Digital games in education: The design of games-based learning environments. *Journal of research on technology in education*, *40*(1), 23-38.

5) Ritterfeld, U., Cody, M., & Vorderer, P. (Eds.). (2009). *Serious games: Mechanisms and effects*. Routledge.

6) Rutter, J., & Bryce, J. (Eds.). (2006). *Understanding digital games*. Sage.

7) Khang, H., Kim, J. K., & Kim, Y. (2013). Self-traits and motivations as antecedents of digital media flow and addiction: The Internet, mobile phones, and video games. Computers in Human Behavior, 29(6), 2416-2424.

8) Choi, D., & Kim, J. (2004). Why people continue to play online games: In search of critical design factors to increase customer loyalty to online contents. CyberPsychology & behavior, 7(1), 11-24.

9) Liu, E. Z. F. (2011). Avoiding internet addiction when integrating digital games into teaching. Social Behavior and Personality: an international journal, 39(10), 1325-1335.

10) Moroney, L., Moroney, & Anglin. (2017). Definitive Guide to Firebase. Apress

11) Moroney, L. (2017). Firebase Cloud Messaging. In The Definitive Guide to Firebase (pp. 163-188). Apress, Berkeley, CA.

12) Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. Journal of machine learning research, 12(Oct), 2825-2830.

13) Oliphant, T. E. (2007). Python for scientific computing. Computing in Science & Engineering, 9(3), 10-20.

14) Pokress, S. C., & Veiga, J. J. D. (2013). MIT App Inventor: Enabling personal mobile computing. arXiv preprint arXiv:1310.2830.

15) Gray, J., Abelson, H., Wolber, D., & Friend, M. (2012, March). Teaching CS principles with app inventor. In Proceedings of the 50th Annual Southeast Regional Conference (pp. 405-406). ACM.