2011

Development And Evaluation Of A Multimedia Intensive Web-Based Exercise Motivation Tool

Rico Jamaine Kornegay
North Carolina Agricultural and Technical State University

DEVELOPMENT AND EVALUATION OF A MULTIMEDIA INTENSIVE WEB-BASED EXERCISE MOTIVATION TOOL

by

Rico Jamaine Kornegay

A thesis submitted to the graduate faculty
in partial fulfillment of the requirements for the degree of
MASTER OF SCIENCE

Department: Electrical & Computer Engineering
Major: Electrical Engineering
Major Professor: Dr. Corey Graves

North Carolina A&T State University
Greensboro, North Carolina
2011

School of Graduate Studies
North Carolina Agricultural and Technical State University

This is to certify that the Master's Dissertation of

Rico Jamaine Kornegay

has met the thesis requirements of
North Carolina Agricultural and Technical State University

Greensboro, North Carolina
2011

Approved by:

_____	_____
Dr. Corey Graves Major Professor	Dr. Alvernon Walker Committee Member
_____	_____
Dr. Christopher Doss Committee Member	Dr. John Kelly Department Chairperson

_____
Dr. Sanjiv Sarin
Dean of Graduate Studies

BIOGRAPHICAL SKETCH

Rico Jamaine Kornegay was born on August 22, 1982 in Goldsboro, North Carolina. He received the Associate of Science degree in Machining Technology from Wayne Community College in 2004 and a Bachelor of Science degree in Electrical Engineering from North Carolina Agricultural and Technical State University in 2009. He is a candidate for the M.S. degree in Electrical Engineering.

ACKNOWLEDGEMENTS

First I would like to thank my family. The constant love and support from my parents and my loving daughter gave me confidence and the drive to pursue my dreams.

I would like to thank Dr. Christopher Doss for giving me many programming tools and confidence in my undergraduate and graduate studies.

Thank you, Dr. Alvernon Walker for always having words of encouragement for me throughout my undergraduate and graduate studies to continue to strive for success.

I would like to express my gratitude to my advisor, Dr. Corey Graves, for cultivating my interest in the area of Computer Engineering, and from whom I took advanced programming courses in my undergraduate and graduate Electrical Engineering program. My interaction with him played a fundamental role in the development of this thesis.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE **PAGE**

LIST OF TABLES

TABLE **PAGE**

ABSTRACT

Kornegay, Rico Jamaine. DEVELOPMENT AND EVALUATION OF A MULTIMEDIA INTENSIVE WEB-BASED EXERCISE MOTIVATION TOOL. **(Major Advisor: Corey Graves),** North Carolina Agricultural and Technical State University

A Multimedia-Intensive Web-Based Exercise Motivation Tool named WEMURCT (Web-Based Multi-User Randomized Circuit Training) was developed to utilize modern technology and web-based mediums to address the lack of motivation that an individual has when it comes to exercising regularly. This project was implemented using state-of-art rich internet application (RIA) technology. Popular RIA technologies that are optimized for creating multimedia-intensive applications are Flash by Adobe and Silverlight by Microsoft. In using Silverlight 4.0, we took advantage of multimedia data being embedded in a web-based application, which will be running in a web browser, to provide audio and visual instructions for the user to follow and guide him or her through each workout circuit.

The WEMURCT runs on a personal computer in the privacy the user's own home, and is user friendly. The user is required to set up some initial parameters for which his/her workout will be based upon, including the following: warm up time, the number of times to switch exercises, and the average time in which the user takes a break between each exercise. Each exercise session will randomly generate a different workout exercise sequence, and the times the user will do each of exercise will be random as well. WEMURCT will guarantee that the user will not get the same workout every time. In

addition to that, it will give the user a chance to work various muscle groups and give him/her an opportunity to have a well balanced workout, and will possibly keep the user from getting bored while doing his/her workout, which could lead the user to quit exercising altogether.

Comparisons were done between the WEMURCT system and to another similar system, respectively. Results indicate that the WEMURCT system is the better system for many reasons. One reason is that WEMURCT is more user-friendly than similar system. Another reason is that WEMURCT takes advantage of the use of multimedia prompts to guide the user through the entirety of the workout circuit. Moreover, with WEMURCT, there is a rest period between each randomly chosen exercise, which enables a smooth transition. Also, the user is guaranteed a well balance workout.

Chapter 1

INTRODUCTION

1.1 Motivation

In our modern-day lives there has been a strong and increasing emphasis put on physical fitness and maintaining personal health because of various health issues that exists in our society. Regular exercise will reduce the risk of mortal disease such as heart disease, cancer, diabetes and prevents or minimize high blood pressure, arthritis, osteoporosis, stroke, and depression. Despite the known benefits of regular exercise, most adults do not participate in levels of physical activity that are sufficient to improve health [1]. In addition, fewer than 20 % of people with heart disease participate in exercise-based cardiac rehabilitation programs, highlighting the vast underutilization of these services, especially in the elderly and women [1]. Doctors and studies show some similar facts. One study found that the average 65-year-old can expect an additional 12.7 years of healthy life-meaning he/she will live disability-free if he/she continues a balanced diet until the age 77.7. Highly active 65-years-olds, however, have an additional 5.7 years of healthy life expectancy - they will remain disability free until age 83.4 [2]. Another study found that increasing physical activity after age 50 can add years to one's life [2]. In the study, individuals with and without cardiovascular disease were compared by the amount of physical activity they did. Men who were moderately active added 1.3 years to their lives and those who were highly active added 3.7 years. Women who were moderately active added 1.1 years and those who were highly active added 3.2 years. In

addition, people who exercised more also lived more years free of cardiovascular disease. While moderate exercise increases life expectancy, highly active people more than doubled the benefits [2].

A person today doesn't exercise as regularly as he/she would like to because of time being a constraint. For example, many people stay long hours at work, and are tired after a long day. A day may include office work, meetings, and family errands, making it tough to find the time to exercise. Additionally, a person might also lack the confidence within himself/herself to exercise regularly, not having anyone to encourage him. Even though there are several methods of exercising, exercising is not a hobby for most people. Also, most people need a structured workout plan to go by to accomplish the goals and results they would like to get to over a period of time.

The structure of a person's workout plan is important because it helps the person keep track of his/her accomplishments and goals. For example, a person's goal is to lose 50lbs over a course of a year. After establishing his/her workout plan, he/she keeps track of his/her improvements each week to see how far he/she has to go to reach his/her goal. Having a structured workout routine can help a person balance his/her daily workout schedule.

1.2 Traditional Solution

There are several traditional solutions to working out regularly. One solution is to have a self-imposed workout plan. In a self-imposed workout plan, a person might write a personal schedule out weekly or bi-weekly for how he will work-out different muscle

groups accordingly to the days of the week. Another solution is to find the extra motivation. Motivation can come from friends, family, and exercise groups. This way an individual will have someone to encourage him/her to continue exercising and to see personal satisfaction with his/her results [3]. An example of a work-out group would be a person that is a member of a local YMCA, who decides to take a class such as aerobics or cycling. His/her instructor and his/her peers will encourage him/her to continue exercising during the whole duration of the class. Another solution that some other people may chose is to invest in a personal trainer who will give them guidance and instructions about what to exercise in a given workout session [1]. In addition to that, the trainer may advise the individual what workout station he/she should go to, for how long he should be at each work station, and how long he/she should take a break between each rotation; this type of working out is called circuit training.

In [4], circuit training is described as a mix of strength training and endurance training. In a circuit training workout one does a group, or circuit, of exercises with little or no rest in between exercises. Usually, one circuit consists of 1 to 14 exercises. Each exercise is performed for a set number of repetitions or period of time before moving to the next exercise. For example, a person might do jumping-jacks for 2 minutes, rest 20 seconds, and then do lunges with each leg for a minute followed by other exercises a different time intervals. Depending on a person's fitness level and experience, he or she might do one circuit or several circuits during each workout as seen in Figure 1.

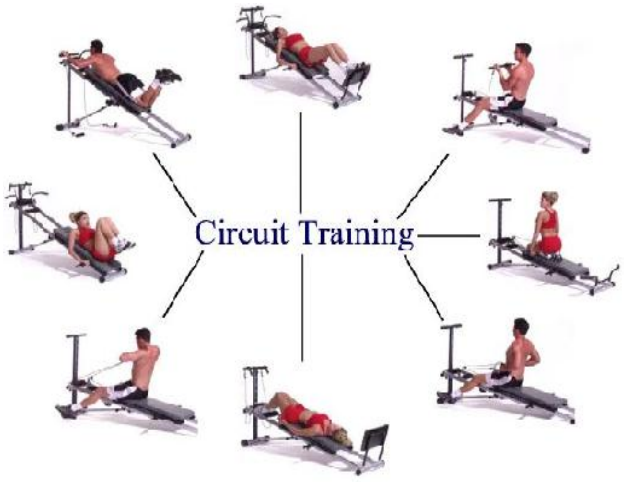

Figure 1: Circuit Training Example

Some of the benefits of circuit training include: (1) being able to exercise different muscle groups to achieve a total-body workout(2) building strength and endurance and (3) being able to gain motivation from other people on the circuit. Since there are a variety of exercises to be performed, a circuit training workout can be made harder or as easier as the person changes the amount of effort he exhibits, and the length of exercise intervals [4].

Problems with some traditional workout solutions are boredom, privacy and convenience. Some people don't like to go to the local gym to workout because sometimes workout routines are repetitive and they start to lose interest in the goal at hand. Another problem is privacy. Some people are self conscious and don't like to work-out in public gyms so they like to work-out in the privacy of their own home. Also, a popular reason that people choose to train at home is because of the convenience. In today's hectic world, many find it hard to find the time to squeeze in trips back and forth to their gym within gym hours. A home gym or workout space at home allows you 24

hour access to your exercise equipments. In contrast, having a workout space at home may initially be a large investment. However, over the long term the savings can be substantial over a gym membership.

1.3 Technology-Based Solution

There are many technology-based solutions to motivate people to work-out regularly at home. One solution is watching DVDs or Blue ray videos. For example P90 X by Tony Horton, and Insanity, by Shaun T, are two great video series to motivate people to work-out regularly. Another solution is electronic work-out equipment. Some examples are the elliptical machines, treadmills, and bicycles-all which commonly have digital interfaces. In the game console fitness market, there are two major companies Nintendo and Microsoft, which both accommodate the combination of entertainment and fitness. These two companies have developed the Wii video game console and X-box Kinect video game console, respectively [5]. Both video game consoles have fitness games such as Wii Fitness Plus, and Zumba, which has yoga, strength training, and tone exercises, to entertain and to motivate people to exercise frequently. Also, there are smart device based workout applications. In [4], the EMURCT system was proposed as a smart device based workout application that involved circuit training. In using the EMURCT system, the workout circuit is randomly selected during each workout session. Another smart device based workout application uses an iPhone platform along with behavioral cueing, music, and performance feedback to provide motivation during interval training exercise session [3].

In the proposed system, a multimedia rich web application is developed that can be accessible through the use of any personal computer. It takes advantage of Silverlight 4.0 technology, which is optimized for multi-media intensive web applications. The proposed application has a graphical user interface that uses multimedia prompts such as video and audio to randomly guide the user through his workout circuit to motivate him and demonstrate to him how to do each exercise the appropriate way.

The remainder of this document is organized as follows. Chapter 2 discusses the research and background for general concepts for the system. Chapter 3 explains the overview, evolving idea, and implementation of the system. Chapter 4 describes the comparison analysis of a similar system. Chapter 5 draws the conclusion and future research and implementation of the system.

Chapter 2

Workout Motivation Technology

2.1 Broad Technology Categories of Interest

Pervasive Computing and Multimedia are both relevant technologies that greatly impact workout motivation technologies. Workout motivation technologies consist of workout videos, electronic workout equipment, video games, PC applications, smart device applications, and web-based applications.

2.1.1 Pervasive Computing

Pervasive computing, also referred to as ubiquitous computing, deals with the idea of making "computing power" available in any place at any time in a uniform way so that it may be exploited for meeting the challenges faced by society [6]. There are several aspects of pervasive computing which includes: context awareness, middleware, wearable computing, web-based computing, and smart spaces.

Context awareness is one of many integral parts of pervasive computing. Context awareness refers to an application's ability to use information about the circumstances under which, it (the application), the device, or the user is currently in, to make some type of decision on how it (the application) should operate [4]. An example of context awareness is when a person uses thumb recognition software to log on his/her personal computer to gain personal access.

Middleware is another integral part of pervasive computing. Middleware is computer software that connects different applications on one or more machines to one to

another [7]. This technology evolved to provide for interoperability in support of the move from mainframes to coherent distributed architectures, which are most often used to support and simplify complex distributed applications. In other words, middleware is like glue that ties everything together in a system. It utilizes web servers, application servers, and similar tools that support application development and delivery such as using various libraries of functions in multiple computer languages like Java and C++ [8]. Middleware is especially integral to modern information technology based on XML, SOAP, Web services, and service-oriented architecture [9].

Another facet of pervasive computing is wearable computing. Wearable computing refers to a person being able to wear computing devices on the body. This technology can be very useful when the user is engaged with the physical environment, but the application still requires some type of computational support [4]. This could refer to any device as small as computers that can be interwoven in clothing or smart devices such as PDAs, and smartphones, which can be considered wearable computing because of the vast operations they can perform at anytime or anyplace [4].

A Smart Space, also referred to as an intelligent environment, is a physical world that richly involves the use of sensors, actuators, displays and computational elements embedded seamlessly into everyday objects, that are connected through a continuous network. In simpler terms, a smart space is an environment that consists of a collection of various smart devices and smart objects that we used on daily basis [4].

Web-based computing is an environment that consists of clients networked over the Internet or intranet [10]. An application in this environment consists of code on the

servers distributed to clients containing a browser, such as Mozilla FireFox or Microsoft's Internet Explorer as the default web browser. The browser completely defines the user interface [10]. The proposed research includes web-based computing and focuses on using the web browser to completely define the user interface of the system.

2.1.2 Pervasive Computing and Multimedia

Multimedia may be the largest revolution of our recent history of the computer industry. Those who entered the field decades ago used computers primarily to perform arithmetic calculations at high speed. As the field of pervasive computing evolved, they began to see a computers data manipulation capability as perhaps far more important. Thanks to the new accommodation of the web-based intensive software called Silverlight, images, graphics, animations, sounds and videos can be used to make an application "come alive" like in a movie theater. Silverlight includes strong multimedia support, including new state of the art high-definition video streaming. Microsoft also provides a service called Silverlight Streaming for distributing multimedia-intensive Silverlight applications [11].

Multimedia programming offers many new challenges. The field of pervasive computing is already enormous and growing rapidly. Most new computers sold today are "multimedia ready," with CD-RW and DVD drives, audio boards and special video capabilities. Today's desktop and laptop computers are so powerful that they can store and play DVD-quality sound and video. Further advances are enabled in the programmable multimedia capabilities available through programming languages. One thing that has been learned is to plan for the "impossible." In computer and

communications fields, the “impossible” has repeatedly become reality [11]. Pervasive computing and multimedia have gone hand in hand in many ways. One way is that there are some cheap and popular pervasive and multimedia outputs devices such as the flat screen LCD display, audio speakers, PC monitor, and PC sound cards. Another example is the use of MMS messaging on cellular phones, which are pervasive computing devices.

2.2 Workout Videos

Advances in multimedia have also had a great impact on the home video industry. The home video exercise industry has also grown and evolved over many decades from VHS (Video Home System) to DVDs/ Blue Ray discs. Some examples of state of the art workout videos are P90 X by Tony Horton, and Insanity, by Shaun T. Both DVD video series provide state of the art concepts for physical fitness and also dieting. In the P90 X video series, a person who views these video must go through exercise circuits. The first circuit includes easy moves that work to tone, firm and strengthen your entire body. The second circuit involves innovative cardio moves, kickboxing, Pilates, and more ways to burn fat and shed inches faster. The third circuit concentrates mainly on tighten up your core by doing crunches and ab exercises. In the Insanity DVD video series one must follow the elite nutrition plan, follow the fitness guide, and follow the insanity Calendar to organize and keep track of his/her monthly progress, set workout goals, and keep him/her motivated. Plus, the Fit Test Tracker tracks your results as you watch your body transform before your eyes.

There are many advantages in using workout videos to regularly exercise. One advantage is that you can workout without the need for expensive fitness equipment. Another advantage is that you can workout in home instead of going to the gym. Also, the person in each video will guide through each exercise during the entirety of the workout session. Furthermore they are so affordable to the public.

One limitation in using workout videos to exercise regularly is that they may become damaged because of scratches and other defects on them over a period of time, causing them not to be as effective as they were previously. Another problem is that they may be easily misplaced.

2.3 Electronic Workout Equipment

Electronic workout equipment is another technology solution for helping people to exercise regularly. We use state-of-the-art electronic workout equipment in gyms as well as in our household environment. Some examples of electronic workout equipment are the treadmill, exercise bicycles, stair master, and elliptical machines. In Figure 2, is an electronic exercise bike, and in Figure 3 is electronic exercise bike user interface. Also in Figure 4 is an elliptical machine, and in Figure 5 is the elliptical machine user interface.

Figure 2: Electronic Exercise Bike

Figure 3: Electronic Exercise Bike User Interface

Figure 4: Elliptical Machine

Figure 5: Elliptical Machine with User Interface

Most electronic workout equipment use an electronic user interfaces. An electronic user interface allows a user to customize the equipment for his/her workout. For example, while using the electronic equipment the user would enter relevant personal data (such weight, gender, and age), physiological data (such as weight, resting heart rate, and peak heart rate), and proficiency level (such as beginner, or experienced) [12].

There are some advantages in using electronic workout equipment. One advantage is that the user can customize his or her workout by using the electronic user interface. Another advantage is that he/she can keep track of how much time he/she has exercised and how many calories he/she has burned over that period of time.

There are many limitations in using electronic workout equipment to exercise regularly. One limitation of electronic workout machines is that most of them are very expensive. Another limitation is the place of electronic workout equipment. Electronic equipment is now found most commonly in gyms, and gym memberships are not affordable for some people. One other limitation is that people can't go to gyms because of times constraint. Another limitation is that individual exercise programs currently are

preinstalled in individual commercial machines produced by different manufacturers. Upgrading those preinstalled programs is very difficult. Moreover, those programs lack variation, so they cannot respond to changes in physical activity during exercise [13].

2.4 Workout Video Games

As the world of technology is forever changing and continuously evolving each day, so is the world of video game consoles. They are growing and evolving as well. The two main video games consoles that are evolving are the Japanese Nintendo Wii Fit and Microsoft X-Box Kinect. They both provide fitness video games to motivate people to exercise regularly as well as for group and individual entertainment. Some examples of the popular fitness games on Nintendo Wii Fit are Zumba Fitness, Gold's Gym Dance Workout, Gold's Gym Cardio Workout, and EA Sports Active More Workout Bundle. To control the games on the Nintendo Wii Fit the user uses a controller called the nunchuk and a Dreamgear fitness board. Some examples of popular fitness games for X-Box Kinect are Your Shape: Fitness Evolved, Zumba Fitness, and the Biggest Loser Ultimate Workout. In contrast, the X-Box Kinect uses motion detection and is based around a webcam-style add-on peripheral for the Xbox 360 console. It enables users to control and interact with the Xbox 360 without the need to touch a game controller, through a natural user interface using gestures and spoken commands.

One limitation of the workout video game consoles are they are not as affordable to the consumers until they go on sale around holidays, or until they depreciate over many years. Furthermore, the fitness games for both game consoles average around 45

dollars per game. In Figure 6, there is a picture Wii Fitness with user interface and in Figure 7 is a picture X-Box Kinect with user interface.

Figure 6: Nintendo Wii Fitness with User Interface

Figure 7: X-Box Kinect with User Interface

2.5 Workout Applications for Personal Computers

There are several workout applications for personal computers. One workout application is called the Weightmania Pro. The Weightmania Pro is a complete fitness

and wellness tracking journal for individuals and groups. It allows users to track 117 things related to workouts, nutrition, weight loss, diabetes and medical info. It supports cardio and weight training, accommodates any workout or diet plan, and tracks multiple people. It allows the users to visualize their progress, judge the effectiveness of diets and workouts, and reach goals sooner [14]. Another workout application for both the desktop and laptop is called the HealthWizard software by Microfit [15].The HealthWizard software is a seamlessly integrated suite of four software programs that analyze physical health and fitness, lifestyle behaviors and current health status. This popular software is designed for health and fitness centers serving a full range of clients from young children to senior citizens [15]. In authors in [12] developed a personalized music system called IM4Sports for individual exercising, although running is the primary target. The IM4Sports system requires an installation stage to personalize the system using a personal computer. The system allows the user to alter or add exercises definitions and to indicate their likes and dislikes of music in terms of songs, artist, or genre with respect to the exercises [12]. Multiple users can use the system, although each has an individual profile. Users can always come back to the installation stage to reset their data or change their music collection.

There are some advantages in using personal computer workout applications. One advantage is that personal computers today are inexpensive. Another advantage is that it allows the user to take advantage of the use of pervasive and multimedia devices.

One limitation of workout applications for the personal computers is that they do not have in-browser capability. Another limitation is that the user must install the

software on each computer he/she wants to use it on. This can limit the location at which one can workout.

2.6 Smart Device Applications

Smart device applications are applications run-on portable devices such as PDAs and smart phones. The development of most applications is done in various computer languages such as C#(C-sharp), Java, and C++. There are some exercise applications for PDAs and smart phones.

For example, the authors in [3] provide motivation and guidance in interval training exercise sessions with a behavioral cueing system developed for the iPhone platform. In Figure 8- is a picture of an iPhone with a Fitness application. Based on many general characteristics, competitive group exercise methods incorporated into their system may keep the users' interest and motivate them to participate in interval training more frequently than before. The abilities of having many social network networks, MMS messaging to friends, and the capability to upload his/her personal rankings on a website can be very effective in motivating the multiple users of the system. By closely observing and analyzing a user, user groups, and exercise context the system will adapt to his/her physical activity by selecting an appropriate song to motivate him/her during interval training.

Figure 8: iPhone with 3g and Fitness Application

Another example of a smart device exercise application is Electronic Multi-User Randomized Circuit Training application. The authors in [4] used circuit training, in which each exercise is randomly selected, without replacement, and multiple repetitions of that exercise are performed over a period of time entered by users before moving to the next random exercise in the circuit. A graphical user interface (GUI) was designed and customized for the user to use just like a digital interface is used for electronic workout equipment. In Figure 9-is a picture of the GUI. The idea is that in randomizing a workout session will keep the users in suspense and it will give them a chance to work various muscle groups, and possibly keep them from getting bored with their workout. Furthermore, audio prompts guide, instruct, and motivate the users throughout each exercise during the workout session.

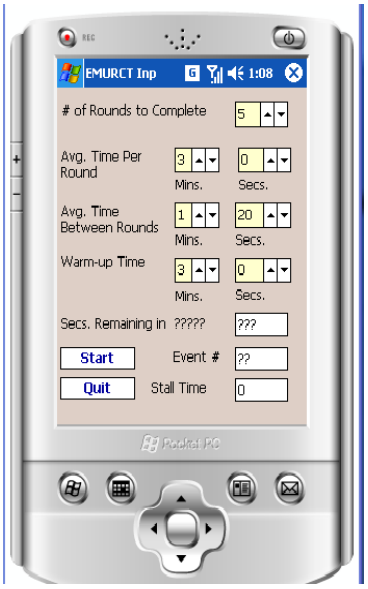

Figure 9: PDA with EMURCT Graphical User Interface

There are some advantages in using smart device such as iPhone and a PDA to incorporate an exercise application. One advantage is that it is capable of handling various formats of multimedia to guide and motivate the users through each workout session [3]. Another advantage is using a Safari web browser with 3g, and both Bluetooth and Wi-Fi capability, allows the user access to the Internet almost anywhere to upload personal rankings and to build many social networks to motivate multiple users to exercise regularly. Furthermore, it is more affordable than traditional exercise equipment such as the treadmill and exercise bikes. In addition, iPhone has a built in accelerometer, a light and a proximity sensor. There is no restriction on the quantity and choice of music that the user can listen to, since the streaming music data is stored in the web database and additional data storage is not required. Moreover, the streaming music data is selected by the music recommendation system. Thus, the probability that the music

corresponds to the user's taste is higher than randomly selected music. By using a 3-axis accelerometer on the iPhone the physical activity of the user is detected. There are some disadvantages of using smart device for incorporating a exercise application. One disadvantage is that the application must be launched from a web browser. Another disadvantage is that it requires an internet connection. Furthermore, it may be a burden to carry the smart device around during each workout session.

2.7 Web-based Applications

A web- based application refers to an application that uses the internet to provide its service to its clients and users. Furthermore, a web browser is used to access the service of the application. There are a limited amount of web-based exercise applications found as research projects but there are many web-based fitness log-in applications that are available.

One popular web-based fitness log-in application is called Fitclick. In using web-based log-in application like Fitclick he/she logs into free weight loss programs, diet plans and workout routines. When he/she uses these resources he/she may use a web-based calorie counter and food journal, and track his/her exercise program with a workout tracker. He/she can create and share diets and workouts, join weight loss groups and more [16].

For instance, the authors in [13] created a web-based exercise solution and designed a four-tier client/server web-based enterprise architecture based on J2EE (Java 2 Enterprise Edition) to develop a continuously supported machine-based exercise system

to motivate people to exercise regularly. The application extended a health promotion model that considers four types of players: users, exercise trainers, manufacturers, and exercise program creators. In using the J2EE platform, it included a client tier, a presentation tier, a service tier, and a database tier. The client tier represents the clients accessing the health promotion services. The presentation tier is responsible for all client interaction. This tier contains Java Servlets and JavaServer Pages for producing user interface elements. The service tier encapsulates the health promotion system's business logic to process request from clients while interacting with the backend database systems. The database tier represents database systems to provide and store health promotion data, such as individual information for the players, exercise programs, and exercise results. Also, in [13] Java Web Start is used, which is a new technology that is associated with using a web browser. Java Web Start handles the entire process for the exercise program without any user interaction, except for the initial single mouse click on the webpage to run the application.

There are some advantages in using the system previously described. One advantage is that with the a web browser, he/she is able to review his/her exercise records, select the exercise program, check exercise progress, and contact the exercise trainer[13]. Moreover, the system is user friendly in a sense because any exercise program can by access by the user with just a simple single mouse click on the webpage. The disadvantages of the system are that any exercise programs must be launched from the web, and exercise program operations may differ widely depending on the operating system, and requires an internet connection. Since there is a web-based log-in application

like Fitclick, then a web-based electronic multi-user randomized circuit training system can be designed and implemented in a similar fashion.

Chapter 3

DESIGN AND IMPLEMENTATION OF THE WEB-BASED ELECTRONIC MULTI-USER RANDOMIZED CIRCUIT TRAINING SYSTEM

3.1 Overview of Design

The proposed system is unique in that it is the only web-based exercise system that involves multiuser randomized circuit training. The system incorporates visual and audio instructions for the user to follow and that guide him or her through each workout circuit. Moreover, it has special features such full screen, skip, pause, and quit.

The system was developed by using Rich Internet Application (RIA) technology. Two Popular RIAs which are optimized for creating multimedia-intensive applications are Flash by Adobe and Silverlight by Microsoft. Flash manipulates vector and raster graphics to provide animation of text, drawings, and still images for applications that will be ran in the web browser. Silverlight is combination of different technologies into a single development platform that allows developers to select tools and the programming language they want to use. Silverlight integrates seamlessly with your existing JavaScript and ASP.NET AJAX code to complement functionality which has been already created. The advantages of using Silverlight in oppose to Flash are (1) the ability to select tools and many programming languages to create an application, which computations can be easily implemented (2) popular developmental environment, and (3) has extreme speed [17]. Microsoft visual C# (C sharp) and the web-based application builder Silverlight were used as the code base for this application. It was chosen because of ease of

implementation. Microsoft Visual Studio 2010.NET framework was the developmental tool that was used. In Figure 10, is a screenshot of the Visual Studio development environment.

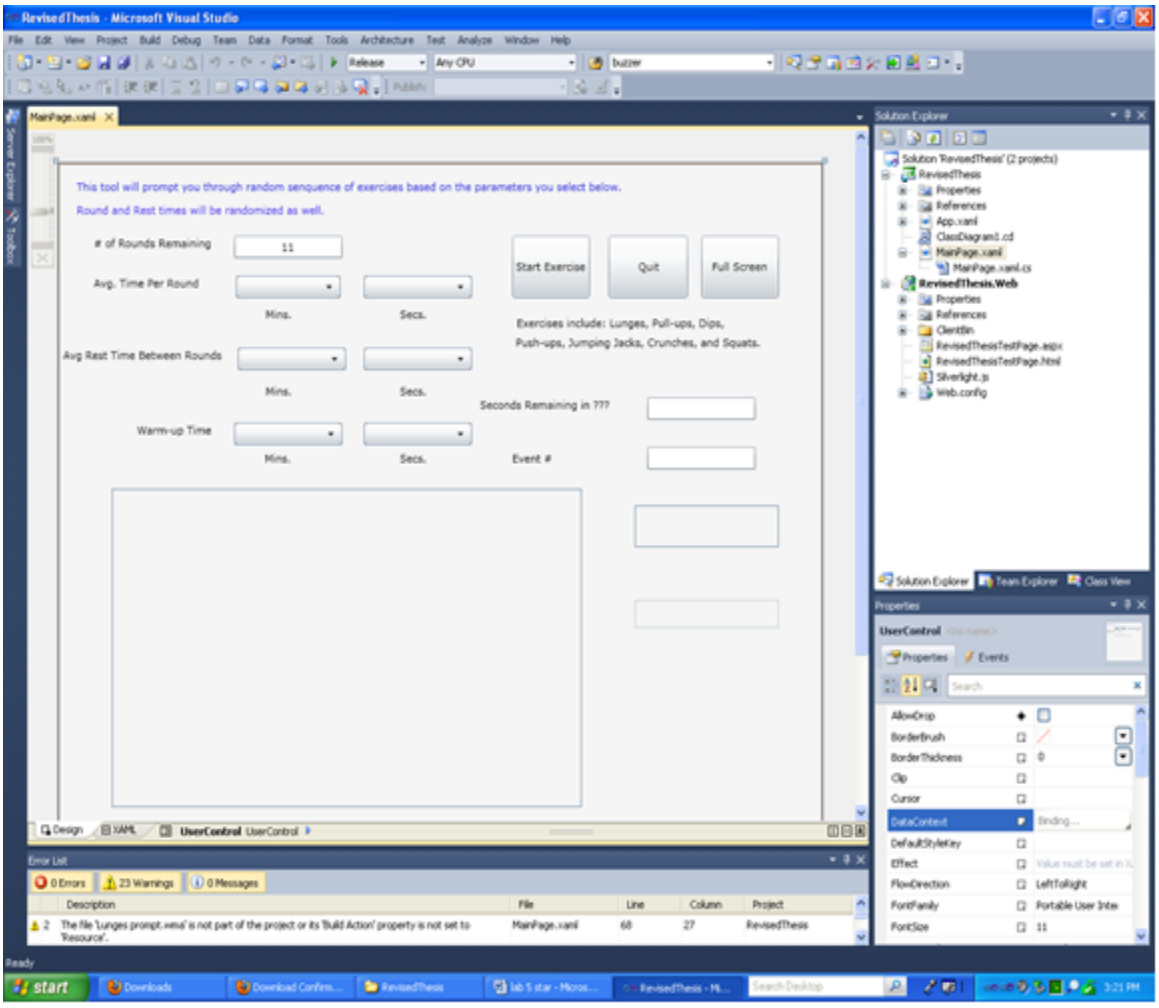

Figure 10: Screenshot of Visual Studio Development Environment

The design of this system evolved from the pervious implementation of a system called EMURCT (Electronic Multi-User Randomized Circuit Training) application. The name of this system is WEMURCT (Web-Based Electronic Multi-User Randomized Circuit Training), which is the web-based version of the EMURCT. WEMURCT is a system that has visual and audio instructive styles for the user to follow and guide him/her through each workout circuit.

3.2 WEMURCT Design Details

As previously mentioned the design of this system was evolved from the pervious implementation of a system called EMURCT as it shown in Figure 9 in chapter 2. The name of this system is WEMURCT (Web-Based Multiuser Randomized Circuit Training). The system's graphical user interface (GUI), in Figure 11, shows is a textbox that the user will enter the number of exercises he/she would like to do during the entire workout session. In addition to that, there are six combo list boxes to enter the number of minutes and seconds the user would like to warm-up before exercising, how long he would like to do each exercise, and rest time between each of the exercises. Also, there is a "Start Exercise" Button to start the application. It also includes a "Quit" button to refresh the application if the user wants to stop the application or to start over again. It also includes a video frame which allows the user to view each one of the instructive exercise videos as they are randomly selected after the "Start Exercise" button is pressed. It also includes a "Full Screen" button, which gives the user the option to view the instructive videos in a bigger frame shown in Figure 12. Additionally, there are two other

text boxes. One text box will display the countdown for warming up time, rest time, and the round time. The second text box will display the name of the random exercise that is currently being demonstrated in the video frame. In addition, there two other special features. One feature is a “Skip” button to skip to next exercise in the random sequence, and another feature is a “Pause” button that pause the application. Both the “Skip” button and “Pause” button are also accessible during Full Screen mode.

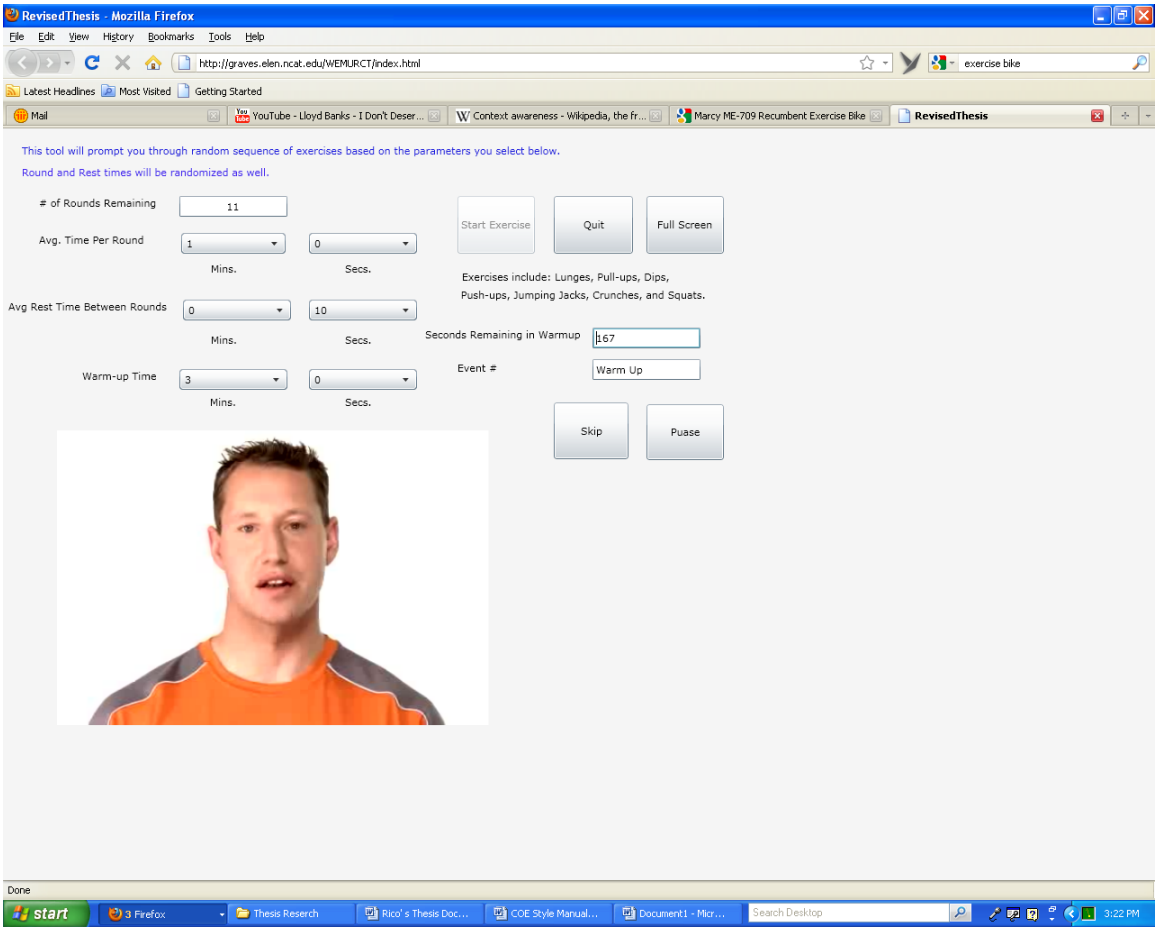

Figure 11: Graphical User Interface (GUI)

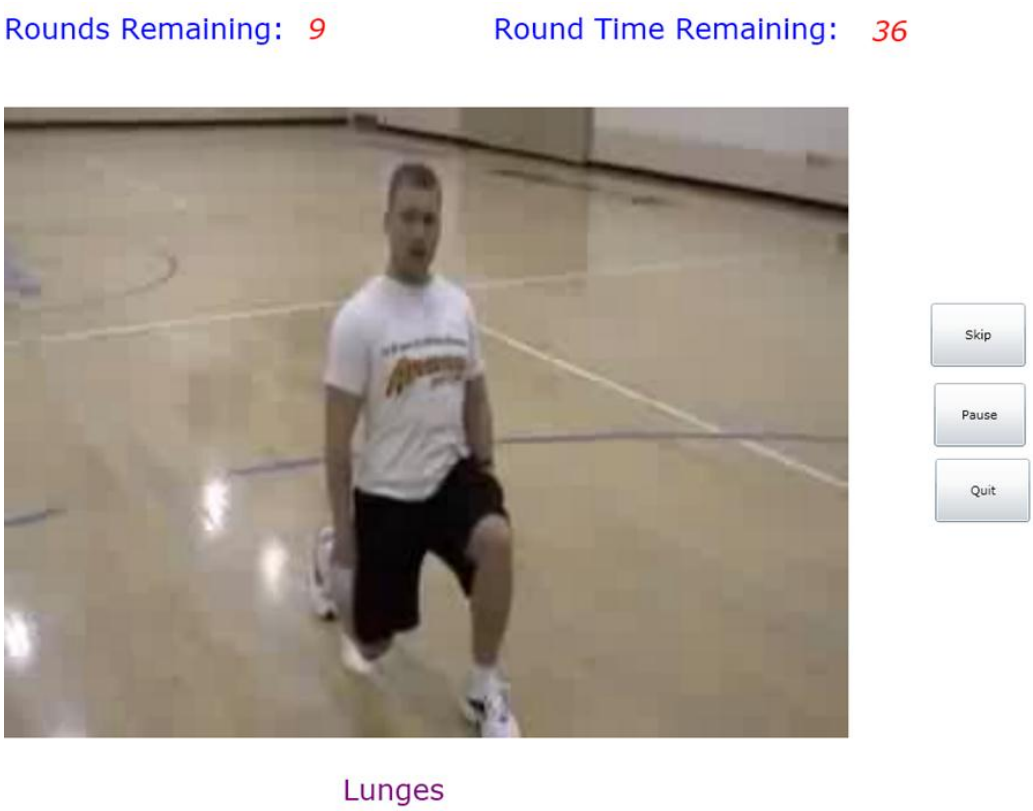

Figure 12: Full-Screen of Application

The overall system implementation consists of two important files. One file of the system is the XAML file. The XAML file contains all of the GUI elements that are created and used in WEMURCT application such combo list boxes, buttons, video frames, audio frames, labels, textblocks, and grids. Furthermore, the XAML contains the properties and event handlers for each of the GUI elements. XAML enables a workflow where separate parties can work on the GUI and the logic of an application, using potentially different tools [18]. The XAML file for WEMURCT application is located in Appendix A of this document.

The second file of the overall system and most important was the .cs file. The .cs file contains the Microsoft visual C# (C sharp) code in which the graphical user interface interacts with the XAML file created with Microsoft Visual Studio 2010.NET developmental tool. The .cs file does the following: (1) time conversion for each time entered by the user, (2) plays each randomly selected audio and video for each exercise, and (3) plays music to motivate the user during each rest interval. The .cs file for WEMURCT is located in Appendix B of this document.

3.3 Implementation

Once the user enters the appropriate web address in the web browser URL field to launch the application, the WEMURCT webpage is loaded up and a voice prompt will prompt the user to enter his/her personal parameters. The personal parameters include the following: (1) the number of rounds remaining (number of desired exercises) (2) the average round time (3) the average rest time between rounds and (4) warm-up time before exercising. The round time and rest time are both randomized-meaning that whatever average time the user enters may be slightly overshot or undershot during each round and rest period. After the user enters his personal parameters, the user will hit the "Start Exercise" button and then the random workout prompts will begin. Specifically, a voice will prompt the user to begin warm-up and video will play and demonstrate a warm-up routine to follow. Next, the user will hear a voice prompt tell him/her to rest and to get ready for the next exercise. Motivation music will be playing during this time. After the rest time has expired, a chime will sound, signaling the beginning of the round,

and then a random exercise will be selected for the user to do during the round. During the round time, a video of the exercise selected will play and demonstrate how to do the exercise the appropriate way. After the round time has expired, a buzzer will sound, signaling that it is the end of round for the exercise randomly selected. Then, the cycle of events, random rest and random exercise will continue until the numbers of rounds remaining are zero. The exercises that are included in this randomized circuit training are lunges, pull-ups, squats, push-ups, crunches, dips, and jumping-jacks Figure 13, shows a flow chart of the WEMURCT web-based application.

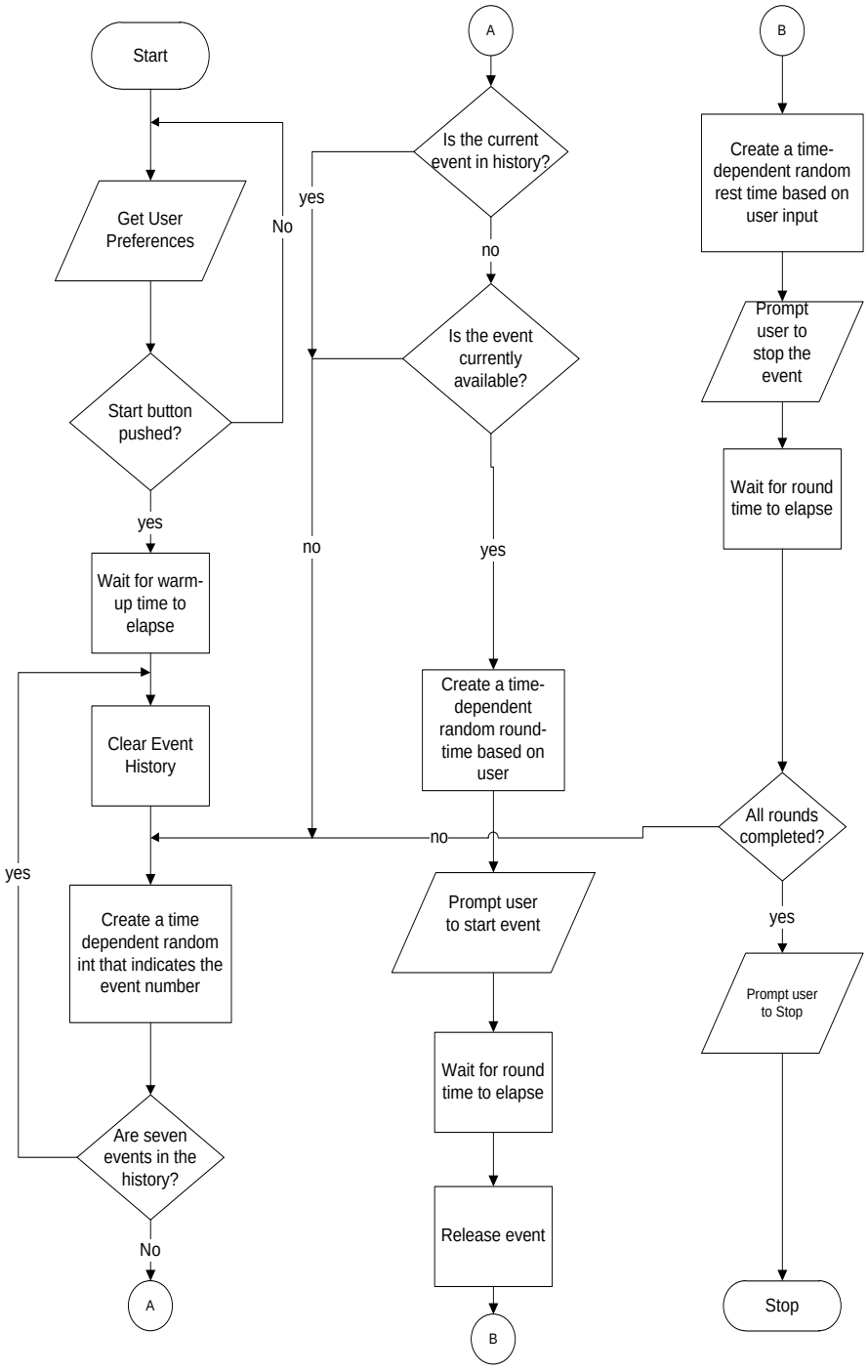

Figure 13: Flow chart of WEMURCT

Chapter 4

COMPARSION AND ANALYSIS

4.1 Discussion of Similar Application

In [19] there is a system with some similarities to the WEMURCT system. It is implemented as Microsoft Excel spreadsheet macro. The authors refer to it simply as Randomizer 2. One similarity is that they both promote circuit training as a regular form of exercise. In addition, both systems use the concept of randomizing the order of the sequence of exercises that are chosen for the user, with hope that the user's mental and physical intensity level will become more focused on and interested in performing the next exercise to come. Moreover, both systems can be ran in the privacy of user's own home, on a personal computer. However, WEMURCT is web-based, where as Randomizer 2 is not. Also in contrast to WEMURCT, Randomizer 2 performs sequence randomization without replacement. This is to say that Randomizer 2 randomly generates an exercise that can be repeated multiple times before all other exercises have been exhausted and that WEMURCT randomly generates an exercise, that can not repeated until all other exercises have been exhausted. In the Randomizer 2 system, if an exercise has a higher weighted value then it is likely to occur more frequently than all other exercises during the same random sequence [19]. In Figure 14 is a picture of the Randomizer 2 user interface.

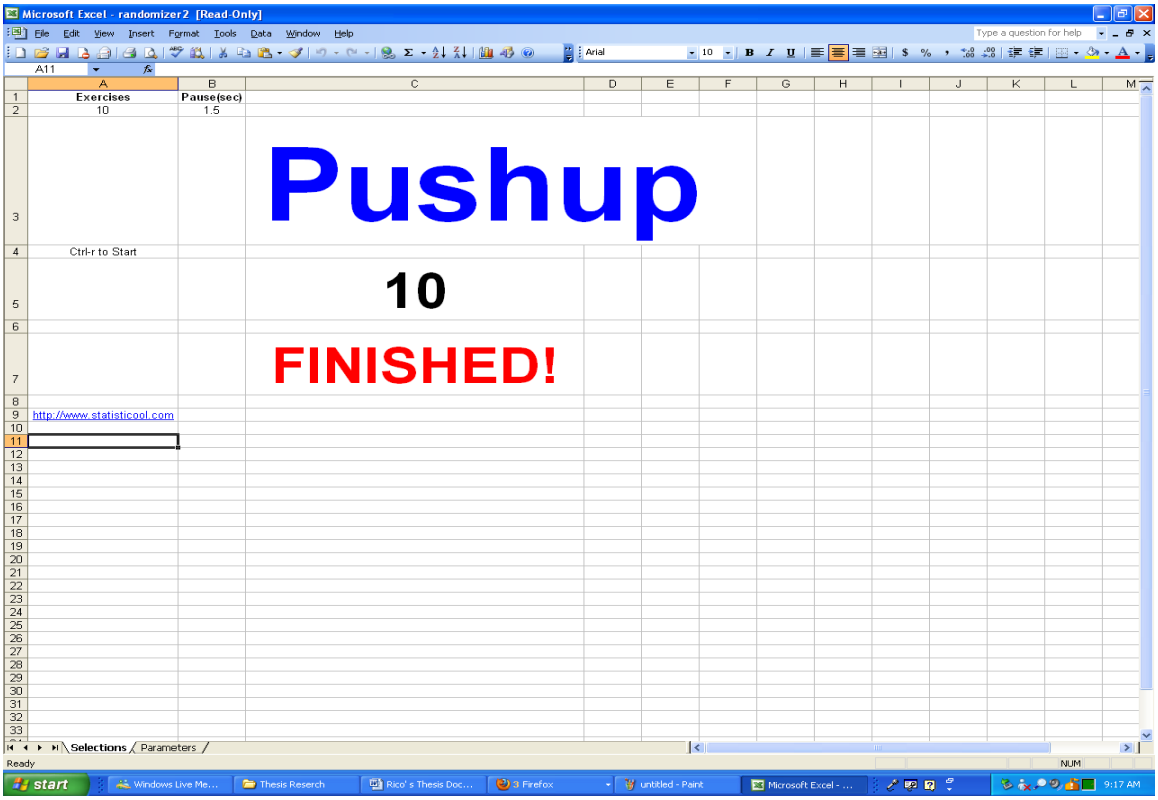

Figure 14: Spreadsheet User Interface

4.2 Experimental Comparison of Systems

As previously mentioned, the Randomizer 2 system uses a Microsoft Excel spreadsheet as its user interface and the WEMURCT system uses a Silverlight-enabled webpage as its user interface. With WEMURCT, once the user enters the desired parameters and pressed the “Start Exercise” button, he/she hears audio and sees video prompts to guide him/her through each exercise of the application. With Randomizer 2, however, there aren’t any audio and video prompts to guide and show the user the appropriate way to do each exercise in the randomized sequence.

To utilize Randomizer 2, the user must first enable the macros of the spreadsheet and hold Ctrl + r before using the application. The Randomizer 2 allows the user to enter the number of rounds and the constant time per round. On the contrary, Randomizer 2 displays text to guide the user through each exercise of the random sequence. In the WEMURCT system time per exercise and the rest time between each of the exercises are randomized. On the other hand, in the Randomizer 2 system there is no rest time between each randomly chosen exercise, therefore there's isn't a smooth transition from exercise to exercise. In both systems the numbers of minutes are converted into equivalent number of seconds that is desired by the user. Table 1 summarizes the functional differences between the two systems.

Table 1. Comparison of Similar Features

	WEMURCT	**Randomizer 2**
Randomized Exercises	Without Replacement	With Replacement
Multimedia Prompts	Audio, Video, and Text	Text with Large font
Application Start Sequence	1) Start Browser 2) Enter URL 3) Select Parameters (text boxes and pull down boxes) 4) Press "Start Exercise" button	1) Start MS Excel spreadsheet 2) Open Randomizer 2 file 3) Enable the Macros 4) Enter Parameters 5) Press Ctrl+r
Exercise Transition	User defined rest time between exercise	No rest time between exercise
User Interface	Webpage	Spreadsheet
Time per Exercise	Randomized base on user preference	Constant

4.3 Results

According to Table 1, the WEMURCT system randomizes each exercise without replacement, but on the other the Randomizer 2 also has randomizes each exercise, but uses replacement. Furthermore, both have prompts, however the WEMURCT system has video, audio, and text multimedia, and the Randomizer 2 only has text with large fonts. Moreover, for the Application start sequence for the WEMURCT system has one step less to start the application than Randomizer 2. As for the exercise transition, with the WEMURCT system, the user can defined the rest time between exercises, however, the Randomizer 2 does not have a resting period between exercises. Furthermore, the time per exercise for the WEMURCT is randomized on per-round bases, based on the user input. However for the Randomizer 2, the time per exercise is constant for each round base on user input. Of special interest is that, the WEMURCT system is a web page and the Randomizer 2 is a spreadsheet.

The special features that the WEMURCT system does have and the Randomizer 2 does not have are the following: (1) Full Screen capability for instructive videos, (2) Audio to guide and motivate the user, (3) Ability to skip another exercise, (4) Ability to Pause the application, (5) Ability Quit/ Restart the application to start over, and (6) An instructive warm-up video.

Chapter 5

CONCLUSION

5.1 Final Product

The final product of this research is a multi-media intensive web-based motivational system which runs in a personal computer web browser, which serves as the user interface. The application will randomly assign exercises so that the user will never know what to anticipate next. The system keeps the user in suspense and also encourages the user to continue to workout. In addition to that, the application can be used in a house-hold setting, and can be used privately. It also incorporates visual and audio instruction styles.

5.2 Summary of Results

Comparisons were done between the WEMURCT system and the Randomizer 2 system, respectively. Results indicate that the WEMURCT system is the better system for many reasons. One reason is that WEMURCT is more user-friendly than Randomizer 2. Another reason is that WEMURCT takes advantage of the use of multimedia prompts to guide the user through the entirety of the workout circuit. Moreover, with WEMURCT, there is a rest period between each randomly chosen exercise, which enables a smooth transition. Also, the user is guaranteed a well balance workout circuit for each workout session. Table 2 shows comparison of each the pervasive and multimedia aspects of each of the workout motivation technologies previously discussed and WEMURCT. Out of all

the workout motivation technologies, the two that best meet most of the desired pervasive and multimedia aspects are WEMURCT and video games.

Table 2. Comparison of Pervasive/Multimedia Aspects

Workout Motivation Technology	Accessibility	Context Awareness	Seamless	High Quality Multimedia	Various Multimedia Types/ Customize media
Videos	Poor	N/A	Fair	Good	Fair/Good
Electronic Equipment	Poor	Good	Good	Poor	Poor/ Poor
Video Games	Poor	Good	Fair	Good	Good/ Good
PC-Based	Poor	N/A	Fair	Good	Fair/Good
Smart Device	Fair	Poor	Fair/Good	Fair	Good/Fair
Web-Based Exercise Logging	Fair	Poor	Poor	Poor	Poor/ Poor
WEMURCT	Fair	N/A	Good	Good	Good/ Good

5.3 Future Research and Implementation

Future research may include implementing this system on a smart phone. Additionally, the multimedia-intensive web-based motivation system may be improved by customizing the system even more, so that user can perform the normal exercises presented in the application already or just chose an arm related workout, just a leg related workout, or just cardiovascular type workout. Furthermore, randomization with replacement and weighting ideas used in Randomizer 2 may be incorporated into

WEMURCT in the future. Finally, research on utilizing camera and microphone input to monitor physical activity can be done, in order to add some level of context awareness.

REFERENCES

[1] Jang, S. J., Park, S. R., Jang, Y.G, Park, J. E., Yoon, Y. R., & Park, S. H. "Automated Individual Prescription of Exercise with an XML-based Expert System," IEEE Engineering in Medicine and Biology 27th Annual Conference,1-4 September 2005. (pp. 882-885). Shanghai, China: IEEE 2005.

[2] Stibich, Mark, Ph.D. "Exercise, Longevity and Life Expectancy", (2007) Web. 3 March 30. <http://www.longevity.about.com/od/lifelongfitness/a/exercise_life_e.htm.>

[3] Suh,Myung-kung, Lee, Kyujoong, Naphapetian, Ani, Sarrafzadeh, Majid (2009). "Interval Training Guidance with Music and Wireless Group Exercise Motivations",(pp110-116).

[4] Graves, C.A., Muldrew, (2009). "Using Wi-Fi Fluctuation To Estimate Activity Level for Users for Computer Assisted Exercise Motivation System", Vol 1.(pp2-2).

[5] Chai, Chunlei, Wang, Mingming, (2010). "Neck Entertainment Fitness System based on Face Recognition", (pp.792).

[6] Reddy, Y.V. "Pervasive Computing: Implications, Opportunities and Challenges for the Society". Pervasive Computing and Applications, 2006 1st International Symposium, (pp. 5 – 5) 3-5 Aug 2006

[7] Ibrahim, Noha. (2009) "Orthogonal Classification of Middleware Technologies",(pp46)

[8] Dorow, Kevin. "Flexible Fault Tolerance In Configurable Middleware for Embedded System." (December 2002) Web. 6 April 2011 < http://moab.eecs.wsu.edu/~bakken/562-files/KevinDorow-Thesis.pdf>

[9] "Middleware" Wikipedia. n.d. Web. 6 April 2011 <http://en.wikipedia.org/wiki/Middleware>

[10] Marco, Lou. "Enterprise System Journal" HighBeam® Research. 01 May 2000. Web. 11 April 2011.<http://www.highbeam.com/doc/1G1-77930530.html>

[11] Deitel, P.J., and Deitel, H.M.: Visual C# 2008 How to Program. (pp.1233).

[12] Wijnalda, Gertjan, Pauws, Steffen, Vignoli, Fabio, and Stuckenschmidt, Heiner.(2005)"A Personalized Music System for Motivation in Sports Performance"

[13] Wang, Zhihui, and Kiryu, Tohru. (2006) "A Java-Based Enterprise System Architecture for Implementing a Continuously Supported and Entirely Web-Based Exercise Solution"

[14] Edward A. Greenwood, Inc. Web. 14 April 2011.<http://www.weightmania.com>

[15] MicoFit Health and Fitness Systems. Web. 14 April 2011. < http:// www.microfit.com>

[16] Genesant Technologies, Inc. n.d. Web. 20 April 2011<www.fitclick.com>

[17] "Flash Vs Silverlight" Web Development Tools and Articles. 24 July 2010. Web. 22 April 2011. <http://www.xibl.com/web-development/flash-vs-silverlight/>

[18] "XAML Overview" Microsoft. n.d. Web. 24 April 2011 <http://msdn.microsoft.com/en-us/library/ms752059.aspx>

[19] Smith, Justin "Weighted Exercise" (2006) Web. 26 April 2011 <http://www.statisticool.com/weightedexercise.htm>

APPENDIX A.
XAML FILE

```
<UserControl x:Class="RevisedThesis.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d"
    d:DesignHeight="850" d:DesignWidth="850" AllowDrop="False"
xmlns:sdk="http://schemas.microsoft.com/winfx/2006/xaml/presentation/sdk">



    <!-- Main grid -->

    <Grid>
        <Grid x:Name="LayoutRoot" Background="WhiteSmoke"
HorizontalAlignment="Stretch" AllowDrop="False">

            <Grid.RowDefinitions>
                <RowDefinition Height="426*"  />
                <RowDefinition Height="424*"/>

            </Grid.RowDefinitions>
            <Grid.ColumnDefinitions>
                <ColumnDefinition Width="Auto"  />
                <ColumnDefinition Width="*" />
            </Grid.ColumnDefinitions>

            <!-- Text Blocks -->

            <TextBlock Height="23" HorizontalAlignment="Left" Margin="40,75,0,0"
x:Name="textBlock1" Text="# of Rounds Remaining" VerticalAlignment="Top" />
            <TextBlock Height="23" HorizontalAlignment="Left" Margin="40,116,0,0"
x:Name="textBlock2" Text="Avg. Time Per Round" VerticalAlignment="Top" />
            <TextBlock Height="23" HorizontalAlignment="Left" Margin="6,190,0,0"
x:Name="textBlock3" Text="Avg Rest Time Between Rounds" VerticalAlignment="Top" />
            <TextBlock Height="23" HorizontalAlignment="Left" Margin="88,267,0,0"
x:Name="textBlock4" Text="Warm-up Time" VerticalAlignment="Top" />
            <TextBlock HorizontalAlignment="Left" Margin="59,241,0,0"
x:Name="textBlock5" Text="Seconds Remaining in ???" Width="180" Grid.Column="1"
Height="31" VerticalAlignment="Top" />
            <TextBlock Margin="95,296,0,0" x:Name="textBlock6" Text="Event #"
Grid.Column="1" Height="23" VerticalAlignment="Top" HorizontalAlignment="Left"
Width="45" />
            <TextBlock Height="23" HorizontalAlignment="Left" Margin="231,148,0,0"
x:Name="textBlock8" Text="Mins." VerticalAlignment="Top" />
            <TextBlock Height="23" HorizontalAlignment="Left" Margin="231,227,0,0"
x:Name="textBlock10" Text="Mins." VerticalAlignment="Top" />
            <TextBlock Height="23" HorizontalAlignment="Left" Margin="380,227,0,0"
x:Name="textBlock11" Text="Secs." VerticalAlignment="Top" />
```

```
            <TextBlock Height="23" HorizontalAlignment="Left" Margin="230,296,0,0"
x:Name="textBlock12" Text="Mins." VerticalAlignment="Top" />
            <TextBlock Height="23" HorizontalAlignment="Left" Margin="380,296,0,0"
x:Name="textBlock13" Text="Secs." VerticalAlignment="Top" />
            <TextBlock Height="23" HorizontalAlignment="Left" Margin="380,148,0,0"
Name="textBlock9" Text="Secs." VerticalAlignment="Top" />
            <TextBlock   Height="23" HorizontalAlignment="Right"
Margin="0,352,181,0" x:Name="FinalTime" Text="  " VerticalAlignment="Top"
Width="31" />


            <!-- Buttons -->

            <Button Content="Full Screen" Height="64" HorizontalAlignment="Left"
Margin="305,75,0,0" Name="Full" VerticalAlignment="Top" Width="86"
Click="Full_Click" Grid.Column="1" />
            <Button Content="Quit" Height="64" HorizontalAlignment="Left"
Margin="202,75,0,0" x:Name="Quit" VerticalAlignment="Top" Width="88"
Click="Quit_Click" Grid.Column="1" />
            <Button Content="Start Exercise" Height="64"
HorizontalAlignment="Left" Margin="95,75,0,0" x:Name="Start"
VerticalAlignment="Top" Width="86" Click="Start_Click" Grid.Column="1" />

            <!-- Text Boxes -->
            <TextBox  Height="23" MaxLength="3" HorizontalAlignment="Left"
Margin="245,241,0,0" x:Name="Totaltime" VerticalAlignment="Top" Width="120"
KeyDown="Totaltime_KeyDown" Grid.Column="1" />
            <TextBox MaxLength="1" Margin="245,292,0,0" x:Name="RecEvent"
Grid.Column="1" Height="23" VerticalAlignment="Top" HorizontalAlignment="Left"
Width="120" />
            <TextBox  MaxLength="2" Height="23" HorizontalAlignment="Left"
Margin="196,75,0,0" Name="RoundsC" VerticalAlignment="Top" Width="120"
TextAlignment="Center" KeyDown="RoundsC_KeyDown" Text="11" />
```

```
<!-- Media prompts for each Exercise -->
            <MediaElement Source="Welcome.wma" Height="29"
HorizontalAlignment="Left" Margin="231,23,0,0" Name="audio1"
VerticalAlignment="Top" Width="160" AutoPlay="True" Grid.Column="1" Grid.Row="1"
/>
            <MediaElement Source="Lunges prompt.wma" Height="43"
HorizontalAlignment="Left" Margin="231,352,0,0"   Name="audio2"
MediaEnded="audio2_MediaEnded"  VerticalAlignment="Top" Width="160"
        AutoPlay="False" Grid.Column="1" />
            <MediaElement Source="Jumping jack prompt.wma" Height="43"
HorizontalAlignment="Left" Margin="231,352,0,0" Name="audio3"
MediaEnded="audio3_MediaEnded"  VerticalAlignment="Top" Width="160"
AutoPlay="False" Grid.Column="1" />
```

```
            <MediaElement Source="push-up prompt.wma" Height="43"
HorizontalAlignment="Left" Margin="231,352,0,0" Name="audio4"
MediaEnded="audio4_MediaEnded" VerticalAlignment="Top" Width="160"
AutoPlay="False" Grid.Column="1" />
            <MediaElement Source="Squat prompt.wma" Height="43"
HorizontalAlignment="Left" Margin="231,352,0,0" Name="audio5"
MediaEnded="audio5_MediaEnded"  VerticalAlignment="Top" Width="160"
AutoPlay="False" Grid.Column="1" />
            <MediaElement Source="pull-up prompt.wma" Height="43"
HorizontalAlignment="Left" Margin="231,352,0,0" Name="audio6"
MediaEnded="audio6_MediaEnded" VerticalAlignment="Top" Width="160"
AutoPlay="False" Grid.Column="1" />
            <MediaElement Source="Crunches prompt.wma" Height="43"
HorizontalAlignment="Left" Margin="231,352,0,0" Name="audio7"
MediaEnded="audio7_MediaEnded"   VerticalAlignment="Top" Width="160"
AutoPlay="False" Grid.Column="1" />
            <MediaElement Source="Dips prompt.wma" Height="43"
HorizontalAlignment="Left" Margin="231,352,0,0" Name="audio8"
MediaEnded="audio8_MediaEnded"  VerticalAlignment="Top" Width="160"
AutoPlay="False" Grid.Column="1" />
            <MediaElement Source="Chime start event.wma" Height="43"
HorizontalAlignment="Left" Margin="231,352,0,0" Name="audio9"
MediaEnded="audio9_MediaEnded"  VerticalAlignment="Top" Width="160"
AutoPlay="False" Grid.Column="1" />
            <MediaElement Source="Buzzer.wma" Height="43"
HorizontalAlignment="Left" Margin="231,352,0,0" Name="audio10"
MediaEnded="audio10_MediaEnded"  VerticalAlignment="Top" Width="160"
AutoPlay="False" Grid.Column="1" />
            <MediaElement Source="Cheering.wma" Height="43"
HorizontalAlignment="Left" Margin="231,352,0,0" Name="audio11"
MediaEnded="audio11_MediaEnded"  VerticalAlignment="Top" Width="160"
AutoPlay="False" Grid.Column="1" />

            <MediaElement Source="Rest prompt.wma" Height="43"
HorizontalAlignment="Left" Margin="231,352,0,0" Name="audio12"
MediaEnded="audio12_MediaEnded"  VerticalAlignment="Top" Width="160"
AutoPlay="False" Volume="1" Grid.Column="1" />
            <MediaElement Source="WarmupPromptStar.wma" Height="43"
HorizontalAlignment="Left" Margin="231,352,0,0" Name="audio13"
VerticalAlignment="Top" Width="160" AutoPlay="False" Volume="1" Grid.Column="1" />
            <MediaElement Source="EndingPrompt.wma" Height="43"
HorizontalAlignment="Left" Margin="231,352,0,0" Name="audio14"
VerticalAlignment="Top" Width="160" AutoPlay="False" Volume="1" Grid.Column="1" />
            <MediaElement Source="RestMusic.wma" Height="43"
HorizontalAlignment="Left" Margin="231,352,0,0" Name="audio15"
MediaEnded="audio15_MediaEnded"  VerticalAlignment="Top" Width="160"
AutoPlay="False" Volume="1" Grid.Column="1" />
```

```
            <!-- Media videos for each Exercise -->
            <MediaElement x:Name="video1" Source="The Right Way to Do Lunges
Star.wmv" Height="327" Margin="60,335,0,0" AutoPlay="False"
VerticalAlignment="Top" MediaEnded="video1_MediaEnded" Grid.ColumnSpan="2"
Stretch="Uniform" HorizontalAlignment="Left" Width="522" Grid.RowSpan="2"
Volume="2" />
            <MediaElement x:Name="video2" Source="Proper+Crunch+Technique++.wmv"
Height="327" Margin="60,335,0,0" AutoPlay="False" VerticalAlignment="Top"
MediaEnded="video2_MediaEnded" Grid.ColumnSpan="2" Stretch="Uniform"
HorizontalAlignment="Left" Width="522" Grid.RowSpan="2" Volume="1" />
            <MediaElement x:Name="video3"
Source="Basic+Exercise+Plans++How+to+Do+Jumping+Jacks++.wmv" Height="327"
Margin="60,335,0,0" AutoPlay="False" VerticalAlignment="Top"
MediaEnded="video3_MediaEnded" Grid.ColumnSpan="2" Stretch="Uniform"
HorizontalAlignment="Left" Width="522" Grid.RowSpan="2" Volume="1" />
            <MediaElement x:Name="video4" Source="Perfect Pull Up - How To Do Pull
Ups  .wmv" Height="327" Margin="60,335,0,0" AutoPlay="False"
VerticalAlignment="Top" MediaEnded="video4_MediaEnded" Grid.ColumnSpan="2"
Stretch="Uniform" HorizontalAlignment="Left" Width="522" Grid.RowSpan="2"
Volume="1" />
            <MediaElement x:Name="video5"
Source="The_Right_Way_To_Do_Squats__.wmv" Height="327" Margin="60,335,0,0"
AutoPlay="False" VerticalAlignment="Top" MediaEnded="video5_MediaEnded"
Grid.ColumnSpan="2" Stretch="Uniform" HorizontalAlignment="Left" Width="522"
Grid.RowSpan="2" Volume="1" />
            <MediaElement x:Name="video6"
Source="Personal_Training_Workout_Tips__Drills__Bench_Dip_.wmv" Height="327"
Margin="60,335,0,0" AutoPlay="False" VerticalAlignment="Top"
MediaEnded="video6_MediaEnded" Grid.ColumnSpan="2" Stretch="Uniform"
HorizontalAlignment="Left" Width="522" Grid.RowSpan="2" Volume="1" />
            <MediaElement x:Name="video7"
Source="How_to_Do_a_Push_Up_Correctly__(2).wmv" Height="327" Margin="60,335,0,0"
AutoPlay="False" VerticalAlignment="Top" MediaEnded="video7_MediaEnded"
Grid.ColumnSpan="2" Stretch="Uniform" HorizontalAlignment="Left" Width="522"
Grid.RowSpan="2" Volume="1" />
            <MediaElement x:Name="video8" Source="General Exercise Warm-Up Video
.wmv" Height="327" Margin="60,335,0,0" AutoPlay="False" VerticalAlignment="Top"
MediaEnded="video8_MediaEnded" Grid.ColumnSpan="2" Stretch="Uniform"
HorizontalAlignment="Left" Width="522" Grid.RowSpan="2" Volume="1" />



            <!-- Combo Boxes for user time input -->

            <ComboBox Height="23" HorizontalAlignment="Left" Margin="198,116,0,0"
Name="RoundMin" VerticalAlignment="Top" Width="116" ItemsSource="{Binding}"
IsEnabled="True" MaxDropDownHeight="100">
                <ComboBoxItem Content="1" />
                <ComboBoxItem Content="2" />
                <ComboBoxItem Content="3" />
                <ComboBoxItem Content="4" />
                <ComboBoxItem Content="5" />
                <ComboBoxItem Content="6" />
                <ComboBoxItem Content="7" />
```

```
                <ComboBoxItem Content="8" />
                <ComboBoxItem Content="9" />
            </ComboBox>
            <ComboBox Height="23" HorizontalAlignment="Left" Margin="340,116,0,0"
Name="RoundSec" VerticalAlignment="Top" Width="120" MaxDropDownHeight="100"
Grid.ColumnSpan="2">
                <ComboBoxItem Content="0" />
                <ComboBoxItem Content="1" />
                <ComboBoxItem Content="2" />
                <ComboBoxItem Content="3" />
                <ComboBoxItem Content="4" />
                <ComboBoxItem Content="5" />
                <ComboBoxItem Content="6" />
                <ComboBoxItem Content="7" />
                <ComboBoxItem Content="8" />
                <ComboBoxItem Content="9" />
                <ComboBoxItem Content="10" />
                <ComboBoxItem Content="11" />
                <ComboBoxItem Content="12" />
                <ComboBoxItem Content="13" />
                <ComboBoxItem Content="14" />
                <ComboBoxItem Content="15" />
                <ComboBoxItem Content="16" />
                <ComboBoxItem Content="17" />
                <ComboBoxItem Content="18" />
                <ComboBoxItem Content="19" />
                <ComboBoxItem Content="20" />
                <ComboBoxItem Content="21" />
                <ComboBoxItem Content="22" />
                <ComboBoxItem Content="23" />
                <ComboBoxItem Content="24" />
                <ComboBoxItem Content="25" />
                <ComboBoxItem Content="26" />
                <ComboBoxItem Content="27" />
                <ComboBoxItem Content="28" />
                <ComboBoxItem Content="29" />
                <ComboBoxItem Content="30" />
                <ComboBoxItem Content="31" />
                <ComboBoxItem Content="32" />
                <ComboBoxItem Content="33" />
                <ComboBoxItem Content="34" />
                <ComboBoxItem Content="35" />
                <ComboBoxItem Content="36" />
                <ComboBoxItem Content="37" />
                <ComboBoxItem Content="38" />
                <ComboBoxItem Content="39" />
                <ComboBoxItem Content="40" />
                <ComboBoxItem Content="41" />
                <ComboBoxItem Content="42" />
                <ComboBoxItem Content="43" />
                <ComboBoxItem Content="44" />
                <ComboBoxItem Content="45" />
                <ComboBoxItem Content="46" />
                <ComboBoxItem Content="47" />
                <ComboBoxItem Content="48" />
```

```
                <ComboBoxItem Content="49" />
                <ComboBoxItem Content="50" />
                <ComboBoxItem Content="51" />
                <ComboBoxItem Content="52" />
                <ComboBoxItem Content="53" />
                <ComboBoxItem Content="54" />
                <ComboBoxItem Content="55" />
                <ComboBoxItem Content="56" />
                <ComboBoxItem Content="57" />
                <ComboBoxItem Content="58" />
                <ComboBoxItem Content="59" />
            </ComboBox>
            <ComboBox Height="23" HorizontalAlignment="Left" Margin="200,190,0,0"
Name="RestMin" VerticalAlignment="Top" Width="120" MaxDropDownHeight="100">
                <ComboBoxItem Content="0" />
                <ComboBoxItem Content="1" />
                <ComboBoxItem Content="2" />
                <ComboBoxItem Content="3" />
                <ComboBoxItem Content="4" />
                <ComboBoxItem Content="5" />
                <ComboBoxItem Content="6" />
                <ComboBoxItem Content="7" />
                <ComboBoxItem Content="8" />
                <ComboBoxItem Content="9" />
            </ComboBox>
            <ComboBox Height="23" HorizontalAlignment="Left" Margin="340,190,0,0"
Name="RestSec" VerticalAlignment="Top" Width="120" MaxDropDownHeight="100"
Grid.ColumnSpan="2">
                <ComboBoxItem Content="0" />
                <ComboBoxItem Content="1" />
                <ComboBoxItem Content="2" />
                <ComboBoxItem Content="3" />
                <ComboBoxItem Content="4" />
                <ComboBoxItem Content="5" />
                <ComboBoxItem Content="6" />
                <ComboBoxItem Content="7" />
                <ComboBoxItem Content="8" />
                <ComboBoxItem Content="9" />
                <ComboBoxItem Content="10" />
                <ComboBoxItem Content="11" />
                <ComboBoxItem Content="12" />
                <ComboBoxItem Content="13" />
                <ComboBoxItem Content="14" />
                <ComboBoxItem Content="15" />
                <ComboBoxItem Content="16" />
                <ComboBoxItem Content="17" />
                <ComboBoxItem Content="18" />
                <ComboBoxItem Content="19" />
                <ComboBoxItem Content="20" />
                <ComboBoxItem Content="21" />
                <ComboBoxItem Content="22" />
                <ComboBoxItem Content="23" />
                <ComboBoxItem Content="24" />
                <ComboBoxItem Content="25" />
                <ComboBoxItem Content="26" />
```

```
                <ComboBoxItem Content="27" />
                <ComboBoxItem Content="28" />
                <ComboBoxItem Content="29" />
                <ComboBoxItem Content="30" />
                <ComboBoxItem Content="31" />
                <ComboBoxItem Content="32" />
                <ComboBoxItem Content="33" />
                <ComboBoxItem Content="34" />
                <ComboBoxItem Content="35" />
                <ComboBoxItem Content="36" />
                <ComboBoxItem Content="37" />
                <ComboBoxItem Content="38" />
                <ComboBoxItem Content="39" />
                <ComboBoxItem Content="40" />
                <ComboBoxItem Content="41" />
                <ComboBoxItem Content="42" />
                <ComboBoxItem Content="43" />
                <ComboBoxItem Content="44" />
                <ComboBoxItem Content="45" />
                <ComboBoxItem Content="46" />
                <ComboBoxItem Content="47" />
                <ComboBoxItem Content="48" />
                <ComboBoxItem Content="49" />
                <ComboBoxItem Content="50" />
                <ComboBoxItem Content="51" />
                <ComboBoxItem Content="52" />
                <ComboBoxItem Content="53" />
                <ComboBoxItem Content="54" />
                <ComboBoxItem Content="55" />
                <ComboBoxItem Content="56" />
                <ComboBoxItem Content="57" />
                <ComboBoxItem Content="58" />
                <ComboBoxItem Content="59" />
            </ComboBox>
            <ComboBox Height="23" HorizontalAlignment="Left" Margin="196,267,0,0"
Name="WarmMin" VerticalAlignment="Top" Width="120" MaxDropDownHeight="100">
                <ComboBoxItem Content="0" />
                <ComboBoxItem Content="1" />
                <ComboBoxItem Content="2" />
                <ComboBoxItem Content="3" />
                <ComboBoxItem Content="4" />
                <ComboBoxItem Content="5" />
                <ComboBoxItem Content="6" />
                <ComboBoxItem Content="7" />
                <ComboBoxItem Content="8" />
                <ComboBoxItem Content="9" />
            </ComboBox>
            <ComboBox Height="23" HorizontalAlignment="Left" Margin="340,267,0,0"
Name="WarmSec" VerticalAlignment="Top" Width="120" MaxDropDownHeight="100"
Grid.ColumnSpan="2">
                <ComboBoxItem Content="0" />
                <ComboBoxItem Content="1" />
                <ComboBoxItem Content="2" />
                <ComboBoxItem Content="3" />
                <ComboBoxItem Content="4" />
```

```
<ComboBoxItem Content="5" />
<ComboBoxItem Content="6" />
<ComboBoxItem Content="7" />
<ComboBoxItem Content="8" />
<ComboBoxItem Content="9" />
<ComboBoxItem Content="10" />
<ComboBoxItem Content="11" />
<ComboBoxItem Content="12" />
<ComboBoxItem Content="13" />
<ComboBoxItem Content="14" />
<ComboBoxItem Content="15" />
<ComboBoxItem Content="16" />
<ComboBoxItem Content="17" />
<ComboBoxItem Content="18" />
<ComboBoxItem Content="19" />
<ComboBoxItem Content="20" />
<ComboBoxItem Content="21" />
<ComboBoxItem Content="22" />
<ComboBoxItem Content="23" />
<ComboBoxItem Content="24" />
<ComboBoxItem Content="25" />
<ComboBoxItem Content="26" />
<ComboBoxItem Content="27" />
<ComboBoxItem Content="28" />
<ComboBoxItem Content="29" />
<ComboBoxItem Content="30" />
<ComboBoxItem Content="31" />
<ComboBoxItem Content="32" />
<ComboBoxItem Content="33" />
<ComboBoxItem Content="34" />
<ComboBoxItem Content="35" />
<ComboBoxItem Content="36" />
<ComboBoxItem Content="37" />
<ComboBoxItem Content="38" />
<ComboBoxItem Content="39" />
<ComboBoxItem Content="40" />
<ComboBoxItem Content="41" />
<ComboBoxItem Content="42" />
<ComboBoxItem Content="43" />
<ComboBoxItem Content="44" />
<ComboBoxItem Content="45" />
<ComboBoxItem Content="46" />
<ComboBoxItem Content="47" />
<ComboBoxItem Content="48" />
<ComboBoxItem Content="49" />
<ComboBoxItem Content="50" />
<ComboBoxItem Content="51" />
<ComboBoxItem Content="52" />
<ComboBoxItem Content="53" />
<ComboBoxItem Content="54" />
<ComboBoxItem Content="55" />
<ComboBoxItem Content="56" />
<ComboBoxItem Content="57" />
<ComboBoxItem Content="58" />
<ComboBoxItem Content="59" />
```

```
            </ComboBox>

           <!-- Labels instruction and Label for exercises-->

            <sdk:Label Height="28" HorizontalAlignment="Left" Margin="100,157,0,0"
Name="label1" VerticalAlignment="Top" Width="237" Grid.Column="1"
Content="Exercises include: Lunges, Pull-ups, Dips," />
            <sdk:Label Height="28" HorizontalAlignment="Left" Margin="95,177,0,0"
Name="label2" VerticalAlignment="Top" Width="280" Grid.Column="1" Content=" Push-
ups, Jumping Jacks, Crunches, and Squats." />
            <sdk:Label Height="28" HorizontalAlignment="Left" Margin="21,17,0,0"
Name="label3" VerticalAlignment="Top" Width="608" Content="This tool will prompt
you through random senquence of exercises based on the parameters you select
below. " Foreground="#FF1D1DDE" Grid.ColumnSpan="2" />
            <sdk:Label Height="28" HorizontalAlignment="Left" Margin="21,41,0,0"
Name="label4" VerticalAlignment="Top" Width="284" Content="Round and Rest times
will be randomized as well." Foreground="#FF4425EF" />
        </Grid>
<!-- end of Main Grid-->

<!-- Begin of Full screen Grid-->
        <Grid x:Name="fsUI" Visibility="Collapsed" Margin="0">
            <Grid.ColumnDefinitions>
                <ColumnDefinition Width="837*" />
                <ColumnDefinition Width="0*" />
            </Grid.ColumnDefinitions>

<!--  Full screen for all videos-->
            <Rectangle x:Name="rectVideoBrush" Stretch="Uniform"
Visibility="Collapsed" Margin="0" >

                <Rectangle.Fill>

                    <VideoBrush x:Name="mediaBrush" SourceName="video1"
Stretch="Uniform" />

                </Rectangle.Fill>

            </Rectangle>

            <Rectangle x:Name="rectVideoBrush2" Stretch="Uniform"
Visibility="Collapsed" Margin="0">

                <Rectangle.Fill>

                    <VideoBrush x:Name="mediaBrush2" SourceName="video2"
Stretch="Uniform"  />

                </Rectangle.Fill>

            </Rectangle>

            <Rectangle x:Name="rectVideoBrush3" Stretch="Uniform"
Visibility="Collapsed" Margin="0" >
```

```
                <Rectangle.Fill>

                    <VideoBrush x:Name="mediaBrush3" SourceName="video3"
Stretch="Uniform"/>

                </Rectangle.Fill>

            </Rectangle>

            <Rectangle x:Name="rectVideoBrush4" Stretch="Uniform"
Visibility="Collapsed" Margin="0" >

                <Rectangle.Fill>

                    <VideoBrush x:Name="mediaBrush4" SourceName="video4"
Stretch="Uniform"/>

                </Rectangle.Fill>

            </Rectangle>

            <Rectangle x:Name="rectVideoBrush5" Stretch="Uniform"
Visibility="Collapsed" Margin="0">

                <Rectangle.Fill>

                    <VideoBrush x:Name="mediaBrush5" SourceName="video5"
Stretch="Uniform"/>

                </Rectangle.Fill>

            </Rectangle>

            <Rectangle x:Name="rectVideoBrush6" Stretch="Uniform"
Visibility="Collapsed" Margin="0" >

                <Rectangle.Fill>

                    <VideoBrush x:Name="mediaBrush6" SourceName="video6"
Stretch="Uniform"/>

                </Rectangle.Fill>

            </Rectangle>

            <Rectangle x:Name="rectVideoBrush7" Stretch="Uniform"
Visibility="Collapsed" Margin="0">

                <Rectangle.Fill>

                    <VideoBrush x:Name="mediaBrush7" SourceName="video7"
Stretch="Uniform"/>

                </Rectangle.Fill>
```

```
            </Rectangle>


            <Rectangle x:Name="rectVideoBrush8" Stretch="Uniform"
Visibility="Collapsed" Margin="0">

                <Rectangle.Fill>

                    <VideoBrush x:Name="mediaBrush8" SourceName="video8"
Stretch="Uniform"/>

                </Rectangle.Fill>

            </Rectangle>
<!-- end of Full screen videos-->

        </Grid>
    </Grid>
<!--end of Full screen Grid-->
</UserControl>
```

APPENDIX B.
C Sharp (.CS) File

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Xml;
using System.Net;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Threading;
using System.Threading;
using System.Windows.Browser;
using System.Runtime.InteropServices;
using System.Windows.Media;
using System.Windows.Media.Animation;
using System.Windows.Shapes;
using System.Text.RegularExpressions;
using System.IO;
using System.Reflection;
using System.ComponentModel;
using System.Windows.Interop;



namespace RevisedThesis
{


   public partial class MainPage : UserControl
   {

      // Global Variables
      int rounds; // number of rounds
      int roundMin;
      int roundSec;
      int round1;
      int round2;
      int TotalRoundTime;// Total time for each round
      int rest1;
```

```
        int rest2;
        int restMin;
        int restSec;
        int TotalRestTime; // rest time between exercises
        int we;
        int we2;
        int warm1;
        int warm2;
        int warmup; // warm-up time at the start of routine
        int counter;
        int number;
        Random rand;
        Random drand;


        Thread RunThread;// Thread for Exercise Motivational Tool




        public MainPage()
        {
            InitializeComponent();



            Application.Current.Host.Content.FullScreenChanged += new
EventHandler(Content_FullScreenChanged);
            RoundMin.SelectedIndex = 0;
            RoundSec.SelectedIndex = 0;
            RestMin.SelectedIndex = 0;
            RestSec.SelectedIndex = 10;
            WarmMin.SelectedIndex = 0;
            WarmSec.SelectedIndex = 15;

        }






        private void Start_Click(object sender, RoutedEventArgs e)// Start Exercise
Button
        {
            Start.IsEnabled = false;




            drand = new Random();
```

```
            // Warm up time conversion
            we = Convert.ToInt16(WarmMin.SelectionBoxItem);


            // Convert.ToInt16(WarmMin);
            we2 = Convert.ToInt16(WarmSec.SelectionBoxItem);



            warm1 = we * 60 ;
            warm2 = we2 ;
            warmup = Convert.ToInt32(warm1 + warm2);




            // rest time conversion
            restMin = Convert.ToInt16(RestMin.SelectionBoxItem);
            restSec = Convert.ToInt16(RestSec.SelectionBoxItem);
            rest1 = restMin * 60;
            rest2 = restSec ;
            TotalRestTime = Convert.ToInt32(rest1 + rest2);




            // Round time conversion
            roundMin = Convert.ToInt16(RoundMin.SelectionBoxItem);
            roundSec = Convert.ToInt16(RoundSec.SelectionBoxItem);
            round1 = roundMin * 60;
            round2 = roundSec;
            TotalRoundTime = Convert.ToInt32(round1 + round2);


            // Round Conversion
            rounds = Convert.ToInt16(RoundsC.Text);
            if ((TotalRoundTime >= 60) && (TotalRestTime >= 10))
            {

                RunThread = new Thread(new ThreadStart(ButtonExercise));// Start the
new thread
                RunThread.Start();

            }
            else
                Start.IsEnabled = true;


        }
```

```
      private void Quit_Click(object sender, System.Windows.RoutedEventArgs e)//
Quit Button
      {
         Quit.IsEnabled = false;
         System.Windows.Browser.HtmlPage.Document.Submit();// Refresh the webpage
to restart


      }



      private void ButtonExercise()// Thread that executes the workout session
      {

         int[] mynumber = new int[7];




         audio13.Dispatcher.BeginInvoke(delegate()// Warmup prompt
         {

            audio13.Play();



         });

         Thread.Sleep(2000);

         video8.Dispatcher.BeginInvoke(delegate()// Warmup video
         {

            video8.Play();



         });



         while (warmup > 0) // Warmup Time
         {



            Thread.Sleep(1000);
```

```
            warmup--;

            Totaltime.Dispatcher.BeginInvoke(delegate()
            {
               textBlock5.Text = "Seconds Remaining in Warmup";
               Totaltime.Text = warmup.ToString();


               if (Application.Current.Host.Content.IsFullScreen) // Full screen
for warmup video
               {
                  rectVideoBrush8.Visibility = System.Windows.Visibility.Visible;

               }

            });



         }



         video8.Dispatcher.BeginInvoke(delegate()// Stops Warmup video after warm-
up time has expired
         {


            video8.Stop();



         });


         Thread.Sleep(2000);




         while (rounds >= 1)
         {

            int localRestTime = TotalRestTime - 10;
            int localRoundTime = TotalRoundTime - 60;
           drand.NextDouble();
            localRestTime = 10 + (int)(localRestTime * 2 * drand.NextDouble()); //
Randomizes Total RestTime
            drand.NextDouble();
            localRoundTime = 60 + (int)(localRoundTime * 2 * drand.NextDouble());
// Randomizes Total RoundTime

            RoundsC.Dispatcher.BeginInvoke(delegate()// Updates Round Remaining
            {
```

```
   rounds--;
   RoundsC.Text = rounds.ToString();

});



rand = new Random();// Random generator


number =  rand.Next(0, 7);



audio12.Dispatcher.BeginInvoke(delegate()// Rest Prompt
{


   audio12.Play();



});



Thread.Sleep(2500);
audio15.Dispatcher.BeginInvoke(delegate()// Rest Music
{


   audio15.Play();



});









while (localRestTime > 0)// Rest Time between each exercise
{


   Thread.Sleep(1000);
   localRestTime--;

   Totaltime.Dispatcher.BeginInvoke(delegate()
   {
      textBlock5.Text = "Seconds Remaining in RestTime";
```

```
                Totaltime.Text = localRestTime.ToString();
            });



        }

        audio15.Dispatcher.BeginInvoke(delegate()// Stops RestMusic
        {


            audio15.Stop();



        });



        audio9.Dispatcher.BeginInvoke(delegate()// Chimes Start each round
        {


            audio9.Play();



        });
        Thread.Sleep(2000);


        //Determining which video to play for each exercise in circuit
training
        if (counter == 7)
        {
            mynumber = new int[7];
            counter = 0;

        }


       // number = rand.Next(7);


        while (Array.IndexOf(mynumber, number) != -1) // Checks to see if a
event has already been done
        {

            number = rand.Next(0,7)+1;  // does another event  that has not
been chosen

        }
        mynumber[counter] = number;  // the array the stores the current
events completed
```

```
            counter++;

            switch (number)
            {
               case 1:audio2.Dispatcher.BeginInvoke(delegate()
               {

                  audio2.Play();// Lunges

                  audio3.Opacity = 0;
                  audio4.Opacity = 0;
                  audio5.Opacity = 0;
                  audio6.Opacity = 0;
                  audio7.Opacity = 0;
                  audio8.Opacity = 0;
               });

                  Thread.Sleep(2000);

                  video1.Dispatcher.BeginInvoke(delegate()
               {

                  video1.Play();

                  video2.Opacity = 0;
                  video3.Opacity = 0;
                  video4.Opacity = 0;
                  video5.Opacity = 0;
                  video6.Opacity = 0;
                  video7.Opacity = 0;
                  video8.Opacity = 0;
                  if (Application.Current.Host.Content.IsFullScreen)
                  {
                      rectVideoBrush.Visibility =
System.Windows.Visibility.Visible;
                      rectVideoBrush8.Visibility =
System.Windows.Visibility.Collapsed;
                  }

               });
                  break;

               case 2: audio3.Dispatcher.BeginInvoke(delegate()
               {

                  audio3.Play();  // Jumping jacks

                  audio2.Opacity = 0;

                  audio4.Opacity = 0;
                  audio5.Opacity = 0;
                  audio6.Opacity = 0;
                  audio7.Opacity = 0;
                  audio8.Opacity = 0;
               });
```

```
                    Thread.Sleep(2000);

                    video3.Dispatcher.BeginInvoke(delegate()
                {


                    video3.Play();

                    video2.Opacity = 0;
                    video4.Opacity = 0;
                    video5.Opacity = 0;
                    video6.Opacity = 0;
                    video7.Opacity = 0;
                    video8.Opacity = 0;
                    if (Application.Current.Host.Content.IsFullScreen)
                    {
                        rectVideoBrush3.Visibility =
System.Windows.Visibility.Visible;
                        rectVideoBrush8.Visibility =
System.Windows.Visibility.Collapsed;
                    }

                });

                    break;

                case 3:audio4.Dispatcher.BeginInvoke(delegate()
                {




                    audio4.Play(); // push ups

                    audio2.Opacity = 0;
                    audio3.Opacity = 0;

                    audio5.Opacity = 0;
                    audio6.Opacity = 0;
                    audio7.Opacity = 0;
                    audio8.Opacity = 0;

                });

                    Thread.Sleep(2000);

                    video7.Dispatcher.BeginInvoke(delegate()
                {



                    video7.Play();

                    video2.Opacity = 0;
                    video3.Opacity = 0;
```

```
                    video4.Opacity = 0;
                    video5.Opacity = 0;
                    video6.Opacity = 0;
                    video8.Opacity = 0;
                    if (Application.Current.Host.Content.IsFullScreen)
                    {
                        rectVideoBrush7.Visibility =
System.Windows.Visibility.Visible;
                        rectVideoBrush8.Visibility =
System.Windows.Visibility.Collapsed;
                    }



                });

                    break;

                case 4: audio5.Dispatcher.BeginInvoke(delegate(){


                    audio5.Play();  //squats

                    audio2.Opacity = 0;
                    audio3.Opacity = 0;
                    audio4.Opacity = 0;

                    audio6.Opacity = 0;
                    audio7.Opacity = 0;
                    audio8.Opacity = 0;
                });

                    Thread.Sleep(2000);
                    video5.Dispatcher.BeginInvoke(delegate()
                    {


                    video5.Play();

                    video2.Opacity = 0;
                    video3.Opacity = 0;
                    video4.Opacity = 0;
                    video6.Opacity = 0;
                    video7.Opacity = 0;
                    video8.Opacity = 0;
                    if (Application.Current.Host.Content.IsFullScreen)
                    {
                        rectVideoBrush5.Visibility =
System.Windows.Visibility.Visible;
                        rectVideoBrush8.Visibility =
System.Windows.Visibility.Collapsed;
                    }
```

```
                });

                   break;

                case 5: audio6.Dispatcher.BeginInvoke(delegate()
                {



                   audio6.Play();// pull up

                   audio2.Opacity = 0;
                   audio3.Opacity = 0;
                   audio4.Opacity = 0;
                   audio5.Opacity = 0;

                     });

                   Thread.Sleep(2000);

                   video4.Dispatcher.BeginInvoke(delegate()
                {


                   audio7.Opacity = 0;
                   audio8.Opacity = 0;
                   video4.Play();

                   video2.Opacity = 0;
                   video3.Opacity = 0;
                   video5.Opacity = 0;
                   video6.Opacity = 0;
                   video7.Opacity = 0;
                   video8.Opacity = 0;
                   if (Application.Current.Host.Content.IsFullScreen)
                   {
                       rectVideoBrush4.Visibility =
System.Windows.Visibility.Visible;
                       rectVideoBrush8.Visibility =
System.Windows.Visibility.Collapsed;
                   }

                });
                   break;
                case 6: audio7.Dispatcher.BeginInvoke(delegate()
                {



                   audio7.Play(); // crunches

                   audio2.Opacity = 0;
                   audio3.Opacity = 0;
                   audio4.Opacity = 0;
```

```
                    audio5.Opacity = 0;
                    audio6.Opacity = 0;

                    audio8.Opacity = 0;

                });

                    Thread.Sleep(2000);
                    video2.Dispatcher.BeginInvoke(delegate()
                {


                    video2.Play();

                    video3.Opacity = 0;
                    video4.Opacity = 0;
                    video5.Opacity = 0;
                    video6.Opacity = 0;
                    video7.Opacity = 0;
                    video8.Opacity = 0;
                    if (Application.Current.Host.Content.IsFullScreen)
                    {
                        rectVideoBrush2.Visibility =
System.Windows.Visibility.Visible;
                        rectVideoBrush8.Visibility =
System.Windows.Visibility.Collapsed;
                    }



                });
                    break;

                case 7:audio8.Dispatcher.BeginInvoke(delegate()
                {

                    audio8.Play();  // DIPS

                    audio2.Opacity = 0;
                    audio3.Opacity = 0;
                    audio4.Opacity = 0;
                    audio5.Opacity = 0;
                    audio6.Opacity = 0;
                    audio7.Opacity = 0;

                });

                    Thread.Sleep(2000);

                    video6.Dispatcher.BeginInvoke(delegate()
                {



                    video6.Play();
```

```
                    video2.Opacity = 0;
                    video3.Opacity = 0;
                    video4.Opacity = 0;
                    video5.Opacity = 0;
                    video7.Opacity = 0;
                    video8.Opacity = 0;
                    if (Application.Current.Host.Content.IsFullScreen)
                    {
                        rectVideoBrush6.Visibility =
System.Windows.Visibility.Visible;
                        rectVideoBrush8.Visibility =
System.Windows.Visibility.Collapsed;
                    }

                });
                    break;



            }



            // Up dates the event number chosen
            RecEvent.Dispatcher.BeginInvoke(delegate()
            {

                RecEvent.Text = Convert.ToString(number);



            });




            while (localRoundTime > 0)
            {



                Thread.Sleep(1000);
                localRoundTime--;

                Totaltime.Dispatcher.BeginInvoke(delegate()
                {
                    textBlock5.Text = "Seconds Remaining in Round";
                    Totaltime.Text = localRoundTime.ToString();


                });
```

```
            }

            audio10.Dispatcher.BeginInvoke(delegate()// RestPrompt
            {


               audio10.Play();



            });

            Thread.Sleep(1500);

            // Stops the video and exercise perviously done by user
            switch (number)
            {
               case 1:
                  video1.Dispatcher.BeginInvoke(delegate()
                  {

                     video1.Stop();
                     if (Application.Current.Host.Content.IsFullScreen)
                     {
                         rectVideoBrush.Visibility =
System.Windows.Visibility.Collapsed;
                         rectVideoBrush8.Visibility =
System.Windows.Visibility.Collapsed;

                     }
                     video2.Opacity = 1;
                     video3.Opacity = 1;
                     video4.Opacity = 1;
                     video5.Opacity = 1;
                     video6.Opacity = 1;
                     video7.Opacity = 1;
                     video8.Opacity = 1;
                  });
                  break;

               case 2: video3.Dispatcher.BeginInvoke(delegate()
               {
                    if (Application.Current.Host.Content.IsFullScreen)
                    {
                        rectVideoBrush3.Visibility =
System.Windows.Visibility.Collapsed;
                         rectVideoBrush8.Visibility =
System.Windows.Visibility.Collapsed;
                    }
                  video1.Opacity = 1;
                  video2.Opacity = 1;
                  video3.Stop();
                  video4.Opacity = 1;
                  video5.Opacity = 1;
```

```
                    video6.Opacity = 1;
                    video7.Opacity = 1;
                    video8.Opacity = 1;
                });

                    break;

                case 3: video7.Dispatcher.BeginInvoke(delegate()
                {

                    if (Application.Current.Host.Content.IsFullScreen)
                    {
                        rectVideoBrush7.Visibility =
System.Windows.Visibility.Collapsed;
                        rectVideoBrush8.Visibility =
System.Windows.Visibility.Collapsed;
                    }
                    video1.Opacity = 1;
                    video2.Opacity = 1;
                    video3.Opacity = 1;
                    video4.Opacity = 1;
                    video5.Opacity = 1;
                    video6.Opacity = 1;
                    video7.Stop();
                    video8.Opacity = 1;



                });

                    break;

                case 4: video5.Dispatcher.BeginInvoke(delegate()
                {

                    if (Application.Current.Host.Content.IsFullScreen)
                    {
                        rectVideoBrush5.Visibility =
System.Windows.Visibility.Collapsed;
                        rectVideoBrush8.Visibility =
System.Windows.Visibility.Collapsed;
                    }
                    video1.Opacity = 1;
                    video2.Opacity = 1;
                    video3.Opacity = 1;
                    video4.Opacity = 1;
                    video5.Stop();
                    video6.Opacity = 1;
                    video7.Opacity = 1;
                    video8.Opacity = 1;



                });
```

```
                    break;

                case 5: video4.Dispatcher.BeginInvoke(delegate()
                {

                    if (Application.Current.Host.Content.IsFullScreen)
                    {
                        rectVideoBrush4.Visibility =
System.Windows.Visibility.Collapsed;
                        rectVideoBrush8.Visibility =
System.Windows.Visibility.Collapsed;
                    }
                    video1.Opacity = 1;
                   video2.Opacity = 1;
                   video3.Opacity = 1;
                   video4.Stop();
                   video5.Opacity = 1;
                   video6.Opacity = 1;
                   video7.Opacity = 1;
                   video8.Opacity = 1;


                });
                    break;
                case 6: video2.Dispatcher.BeginInvoke(delegate()
                {


                    if (Application.Current.Host.Content.IsFullScreen)
                    {
                        rectVideoBrush2.Visibility =
System.Windows.Visibility.Collapsed;
                        rectVideoBrush8.Visibility =
System.Windows.Visibility.Collapsed;
                    }
                   video1.Opacity = 1;
                   video2.Stop();
                   video3.Opacity = 1;
                   video4.Opacity = 1;
                   video5.Opacity = 1;
                   video6.Opacity = 1;
                   video7.Opacity = 1;
                   video8.Opacity = 1;

                });
                    break;

                case 7: video6.Dispatcher.BeginInvoke(delegate()
                {

                    if (Application.Current.Host.Content.IsFullScreen)
                    {
                        rectVideoBrush6.Visibility =
System.Windows.Visibility.Collapsed;
```

```
                        rectVideoBrush8.Visibility =
System.Windows.Visibility.Collapsed;
                    }
                    video1.Opacity = 1;
                   video2.Opacity = 1;
                   video3.Opacity = 1;
                   video4.Opacity = 1;
                   video5.Opacity = 1;
                   video6.Stop();
                   video7.Opacity = 1;
                   video8.Opacity = 1;
               });
                   break;

            }



            audio12.Dispatcher.BeginInvoke(delegate()// Buzzer sound at end of
each round
            {


               audio12.Play();



            });
            Thread.Sleep(1500);

         }// END OF round

         audio10.Dispatcher.BeginInvoke(delegate()// Buzzer sounds after workout
session
         {


            audio10.Play();



         });


           Thread.Sleep(1500);
         audio14.Dispatcher.BeginInvoke(delegate()// Ending prompt after the
workout session
         {

            audio14.Play();



         });
         Thread.Sleep(1500);
```

```
            audio11.Dispatcher.BeginInvoke(delegate()// Cheering prompt after workout
routine
            {


               audio11.Play();



            });


            Start.IsEnabled = true;
            Quit.IsEnabled = true;

         }//End of New thread called ButtonExercise



         // To only accept numeric values in each of the Textboxes
         private void RoundsC_KeyDown(object sender, KeyEventArgs e)
         {

            if (e.Key == Key.Tab)
            {
            }
            else
            {

               var thisKeyStr = "";
               if (e.PlatformKeyCode == 190 || e.PlatformKeyCode == 110)
               {
                  thisKeyStr = ".";
               }
               else
               {
                  thisKeyStr = e.Key.ToString().Replace("D", "").Replace("NumPad",
"");
               }
               var s = (sender as TextBox).Text + thisKeyStr;
               var rStr = "^[0-9]+[.]?[0-9]*$";
               var r = new Regex(rStr, RegexOptions.IgnoreCase);
               e.Handled = !r.IsMatch(s);

            }

         }




         private void Totaltime_KeyDown(object sender, KeyEventArgs e)
         {
            if (e.Key == Key.Tab)
```

```
            {
            }
            else
            {

                var thisKeyStr = "";
                if (e.PlatformKeyCode == 190 || e.PlatformKeyCode == 110)
                {
                    thisKeyStr = ".";
                }
                else
                {
                    thisKeyStr = e.Key.ToString().Replace("D", "").Replace("NumPad",
""");
                }
                var s = (sender as TextBox).Text + thisKeyStr;
                var rStr = "^[0-9]+[.]?[0-9]*$";
                var r = new Regex(rStr, RegexOptions.IgnoreCase);
                e.Handled = !r.IsMatch(s);
            }
        }



        // To Replay the video and do exercise over again during a round


        private void video1_MediaEnded(object sender, RoutedEventArgs e)
        {

            video1.Stop();
            video1.Play();


        }



        private void video2_MediaEnded(object sender, RoutedEventArgs e)
        {

            video2.Stop();
            video2.Play();
        }


        private void video3_MediaEnded(object sender, RoutedEventArgs e)
        {

            video3.Stop();
            video3.Play();
        }
```

```
private void video4_MediaEnded(object sender, RoutedEventArgs e)
{

   video4.Stop();
   video4.Play();
}



private void video5_MediaEnded(object sender, RoutedEventArgs e)
{

   video5.Stop();
   video5.Play();
}


private void video6_MediaEnded(object sender, RoutedEventArgs e)
{

   video6.Stop();
   video6.Play();
}

private void video7_MediaEnded(object sender, RoutedEventArgs e)
{

   video7.Stop();
   video7.Play();
}

// To Stop audio at the end of each round

private void audio2_MediaEnded(object sender, RoutedEventArgs e)
{
   audio2.Stop();

}

private void audio3_MediaEnded(object sender, RoutedEventArgs e)
{
   audio3.Stop();
}

private void audio4_MediaEnded(object sender, RoutedEventArgs e)
{
   audio4.Stop();
}

private void audio5_MediaEnded(object sender, RoutedEventArgs e)
{
   audio5.Stop();
}
```

```
        private void audio6_MediaEnded(object sender, RoutedEventArgs e)
        {
            audio6.Stop();
        }

        private void audio7_MediaEnded(object sender, RoutedEventArgs e)
        {
            audio7.Stop();
        }

        private void audio8_MediaEnded(object sender, RoutedEventArgs e)
        {
            audio8.Stop();
        }

        private void audio9_MediaEnded(object sender, RoutedEventArgs e)
        {
            audio9.Stop();
        }

        private void audio10_MediaEnded(object sender, RoutedEventArgs e)
        {
            audio10.Stop();
        }

        private void audio11_MediaEnded(object sender, RoutedEventArgs e)
        {
            audio11.Stop();
        }

        private void audio12_MediaEnded(object sender, RoutedEventArgs e)
        {
            audio12.Stop();
        }


        private void Full_Click(object sender, RoutedEventArgs e) // Full Screen
Button
        {
            Application.Current.Host.Content.IsFullScreen = true;




        }


        // Contents of the Full Screen that is determined by the random number

        void Content_FullScreenChanged(object sender, EventArgs e)
 {
    switch (number)
    {
```

```
    case 1:

       {

          rectVideoBrush.Visibility = System.Windows.Visibility.Visible;
          rectVideoBrush2.Visibility = System.Windows.Visibility.Collapsed;
          rectVideoBrush3.Visibility = System.Windows.Visibility.Collapsed;
          rectVideoBrush4.Visibility = System.Windows.Visibility.Collapsed;
          rectVideoBrush5.Visibility = System.Windows.Visibility.Collapsed;
          rectVideoBrush6.Visibility = System.Windows.Visibility.Collapsed;
          rectVideoBrush7.Visibility = System.Windows.Visibility.Collapsed;
          rectVideoBrush8.Visibility = System.Windows.Visibility.Collapsed;

       }
       break;

    case 2:
    {
          rectVideoBrush.Visibility = System.Windows.Visibility.Collapsed;
          rectVideoBrush2.Visibility = System.Windows.Visibility.Collapsed;
          rectVideoBrush3.Visibility = System.Windows.Visibility.Visible;
          rectVideoBrush4.Visibility = System.Windows.Visibility.Collapsed;
          rectVideoBrush5.Visibility = System.Windows.Visibility.Collapsed;
          rectVideoBrush6.Visibility = System.Windows.Visibility.Collapsed;
          rectVideoBrush7.Visibility = System.Windows.Visibility.Collapsed;
          rectVideoBrush8.Visibility = System.Windows.Visibility.Collapsed;
       }



       break;

    case 3:
    {


          rectVideoBrush.Visibility = System.Windows.Visibility.Collapsed;
          rectVideoBrush2.Visibility = System.Windows.Visibility.Collapsed;
          rectVideoBrush3.Visibility = System.Windows.Visibility.Collapsed;
          rectVideoBrush4.Visibility = System.Windows.Visibility.Collapsed;
          rectVideoBrush5.Visibility = System.Windows.Visibility.Collapsed;
          rectVideoBrush6.Visibility = System.Windows.Visibility.Collapsed;
          rectVideoBrush7.Visibility = System.Windows.Visibility.Visible;
          rectVideoBrush8.Visibility = System.Windows.Visibility.Collapsed;

       }







       break;

case 4:
```

```
{

      rectVideoBrush.Visibility = System.Windows.Visibility.Collapsed;
      rectVideoBrush2.Visibility = System.Windows.Visibility.Collapsed;
      rectVideoBrush3.Visibility = System.Windows.Visibility.Collapsed;
      rectVideoBrush4.Visibility = System.Windows.Visibility.Collapsed;
      rectVideoBrush5.Visibility = System.Windows.Visibility.Visible;
      rectVideoBrush6.Visibility = System.Windows.Visibility.Collapsed;
      rectVideoBrush7.Visibility = System.Windows.Visibility.Collapsed;
      rectVideoBrush8.Visibility = System.Windows.Visibility.Collapsed;

   }




   break;

case 5:
{

      rectVideoBrush.Visibility = System.Windows.Visibility.Collapsed;
      rectVideoBrush2.Visibility = System.Windows.Visibility.Collapsed;
      rectVideoBrush3.Visibility = System.Windows.Visibility.Collapsed;
      rectVideoBrush4.Visibility = System.Windows.Visibility.Visible;
      rectVideoBrush5.Visibility = System.Windows.Visibility.Collapsed;
      rectVideoBrush6.Visibility = System.Windows.Visibility.Collapsed;
      rectVideoBrush7.Visibility = System.Windows.Visibility.Collapsed;
      rectVideoBrush8.Visibility = System.Windows.Visibility.Collapsed;



}
   break;
case 6:
{

      rectVideoBrush.Visibility = System.Windows.Visibility.Collapsed;
      rectVideoBrush2.Visibility = System.Windows.Visibility.Visible;
      rectVideoBrush3.Visibility = System.Windows.Visibility.Collapsed;
      rectVideoBrush4.Visibility = System.Windows.Visibility.Collapsed;
      rectVideoBrush5.Visibility = System.Windows.Visibility.Collapsed;
      rectVideoBrush6.Visibility = System.Windows.Visibility.Collapsed;
      rectVideoBrush7.Visibility = System.Windows.Visibility.Collapsed;
      rectVideoBrush8.Visibility = System.Windows.Visibility.Collapsed;



}
   break;

case 7:
{
```

```
                rectVideoBrush.Visibility = System.Windows.Visibility.Collapsed;
                rectVideoBrush2.Visibility = System.Windows.Visibility.Collapsed;
                rectVideoBrush3.Visibility = System.Windows.Visibility.Collapsed;
                rectVideoBrush4.Visibility = System.Windows.Visibility.Collapsed;
                rectVideoBrush5.Visibility = System.Windows.Visibility.Collapsed;
                rectVideoBrush6.Visibility = System.Windows.Visibility.Visible;
                rectVideoBrush7.Visibility = System.Windows.Visibility.Collapsed;
                rectVideoBrush8.Visibility = System.Windows.Visibility.Collapsed;


            }
                break;

        }



            if (Application.Current.Host.Content.IsFullScreen)
            {
                LayoutRoot.Visibility = Visibility.Collapsed;
                fsUI.Visibility = Visibility.Visible;



            }



            else
            {
                LayoutRoot.Visibility = Visibility.Visible;
                fsUI.Visibility = Visibility.Collapsed;

            }
        }



        private void audio15_MediaEnded(object sender, RoutedEventArgs e) // Stops
and replays the rest music
        {
            audio15.Stop();
            audio15.Play();
        }

        private void video8_MediaEnded(object sender, RoutedEventArgs e)// Stops and
replays the warmup video
        {
            video8.Stop();
            video8.Play();
        }


    }
}
```