Technical Document

Unit 14 – Event Driven Programming

By: Chester Descallar

Contents

[Brief 2](#_Toc468789640)

[Hardware Costs 2](#_Toc468789641)

[Software Costs 2](#_Toc468789642)

[Development Cost 3](#_Toc468789643)

[Introduction 3](#_Toc468789644)

[Tools Used 3](#_Toc468789645)

[Programming Techniques Used 5](#_Toc468789646)

[Data Handling 5](#_Toc468789647)

[Entry Validation 6](#_Toc468789648)

[Functions and Event Handlers 6](#_Toc468789649)

[External Data 6](#_Toc468789650)

[Software Structures 7](#_Toc468789651)

[Sequence 7](#_Toc468789652)

[Selection 8](#_Toc468789653)

[Iteration 9](#_Toc468789654)

[Efficiency 10](#_Toc468789655)

[Robustness 10](#_Toc468789656)

[Print Screen Evidence of Media Player 11](#_Toc468789657)

[Code listing 16](#_Toc468789658)

[Home Page Code 16](#_Toc468789659)

[Play Page Code 17](#_Toc468789660)

[Setting Page Code 18](#_Toc468789661)

[Help Page Code 19](#_Toc468789662)

[Mobile Media Player Test Plan 20](#_Toc468789663)

[Test Results Analysis 25](#_Toc468789664)

[Evaluation 28](#_Toc468789665)

# Brief

The scenario given was to create a media player for the company called ‘Universal Media’ which is suitable to work on a mobile phone. This is to enables the user to listen to songs whenever they want and on the go. Along with the program, a design portfolio and evidence of the program in a report was also required to explain the contents of the media player.

Overall, the scenario took 64 hours to do the 3 assignments given. The first assignment was to research about the programming language that the program will be written on – this was Visual Studio using VB.NET and to write a report about the different tools and techniques that can be used. I started my research on the September 13th and finished by 19th of September; this accounted to 7 hours in 3 working days. The second assignment was then started on the 27th of September – this task was to create the design portfolio of the media player which was to include screen designs, pseudocode, requirement specifications, etc. This was done in 14 hours within 6 working days finishing the second assignment on the 10th of October. The final assignment, was the longest of them all as this required me to implement the media player, test it, and make a documentation about all of the things that I have done. The final assignment took 47 hours in total. I started the implementation stage on the 18th of October and finished it on the 23rd of November along with the implementation stage, I also did testing and improved the program – totalling to 36 hours of programming, testing and improving the program. The report then took 7 hours between the 24th of November and the 1st of December to finish all of the scenario that was needed by the company.

## Hardware Costs

This is the specification of the machine that was used to create the program:

Processor: Intel Core i5-650 - £109

Graphics Card: £100

RAM: 3.0GB DDR3 - £18

This is just the basic cost of the machine without the peripherals, however the overall price might cost around £400-£500

## Software Costs

Microsoft Office 2010 - £79

Microsoft Windows 7 Professional - £129

Microsoft Visual Studio 2013 - £310

The software cost of the project would be around £518.

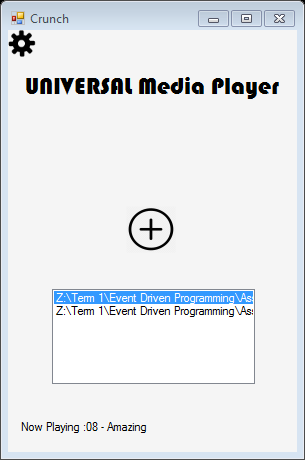
## Development Cost

An average salary for a software engineer is around £35 an hour meaning that it would cost around £2240 (35\*64) to do this assignment.

# Introduction

In my program, I have used different tools and techniques such as data handling, conditionals and looping. This is to make the most efficient program that the user can use as well as for other programmers to read and follow the code. Visual Studio, the programming language that I am writing on also gives me different kinds of pre-built tools allowing me to use them and create the most efficient code that I can possibly make.

# Tools Used



*Picture Box*

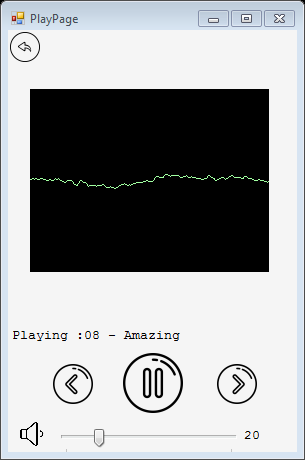


*Label*

*Picture box*

*List box*

*Label*

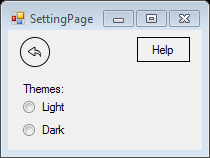


*Picture box*

*Windows Media Player*

*Label*

*Picture Boxes*



*Picture Boxes*

*Button*

*Label*

*Radio Button*

When creating my program, I have used different kinds of tools:

**Labels** – I have used labels in various parts in my form this is because labels allow me to have a text that is unchangeable by the user – therefore it allowed me to create titles and sentences for example the title ‘Universal Media player’ or to show the song that is currently playing in the media player.

**Text Box** – I have used a text box in the homepage of my media player, this is to enable the user to search for the song that they would like. That is because a text box allows the user to either enter or change that text which I can use to find the song that the user has searched in the list box.

**Button –** I have used a button in the setting page of media player, this is the ‘Help Button’ that sends the user to the Help Page and assist them when using the media player. I have used a regular button as it allows me to write text in it which is what I need (a button with a help text)

**List Box - I have also used** a list box to store the music that is added to the media player this is because you don’t know how many the songs the user will store so using a list box allows the user to store as many as they want. A list box also displays all the song to the user and waits until a song is clicked to do a command within when an item is selected in this media player’s case to play the selected song.

**Radio Button** - I have used a radio button in my program to allow the user to choose the genre. Having an option buttons allows me to display multiple choices from such as the ‘Dark theme’ or the ‘Light theme’ but the user can choose only one is the perfect tool when choosing a theme as you can only choose one.

**Picture Box** – I have used picture box in in all the forms in my media player. In the home page of my media player, I have used it for the settings button. In the play page, it is used for play/pause buttons as well as the next and previous and the volume indicator. Picture boxes acts as a visual container for other controls that is because once clicked it triggers a command which is helpful for making buttons for a media player such as the next button; as when you click on it skips to the next song.

**Timer** – I have used a timer in my media player so that I can have a scrolling text to display to the user the song that is currently playing. When the timer starts, lines of code runs which makes the label keep scrolling to the side.

**Track bar –** In my media player, track bars are used as a tool to change the volume of the media player acting as a volume slider. The minimum value is set to 0 and the maximum is set to 100 so that the user can freely slide the track bar and choose how loud they want their media player to be. This gives the user more freedom and control of the loudness of their song that is played and to have that accurate volume instead of clicking on the side buttons which might not give the desired volume that the user wants.

**Open File Dialog –** I have also used ‘openfiledialog’ in my media player which is in my home page. I have used open file dialog to allow me to import files from any directory so that I am able to play music within my media player.

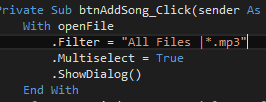
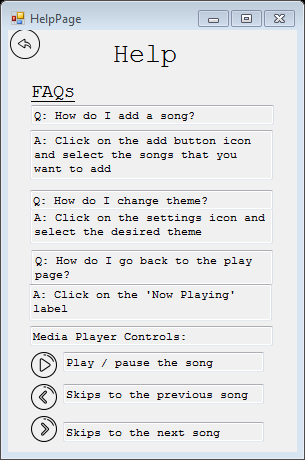
**Windows Media Player –** The window media player was what allowed me to play music in my media player and this is what my entire program was based around. I removed the user interface of the windows media player only showing the visualizer and I added my own user controls such as the play/pause button etc. instead – this allowed me to make my program a lot quicker saving me a lot of time as well as making my ode more efficient.

# Programming Techniques Used

## Data Handling

Data handling is needed for the program to control what is inputted and outputted for example:

There are different problems when a user inputs data such as inputting the wrong option as they have not followed instructions.

In my media player, I have ensured that the user’s input is controlled by filtering out any files that exist in the user’s chosen folder apart from .mp3 or music files. This is because the program that I am creating is a media player therefore it should only accept media files – however if the user selects a non-media file then it shouldn’t fail or cause the program to crash e.g. a file extension of .jpeg or .png is not accepted by the media player.

The help page is full of constants which aids the user when using the media player

This is the filter that I used to only mp3 files when the directory is opened

In Visual Basic, picture boxes can be used to show the user controls that contains a piece of code. For example, within my code, there is a setting button. This setting button acts as an output control for the user as it is just sits at them main menu to be clicked by the user. Thus, when the user clicks on the setting button a pop up of the setting form appears.

Constants can also be used to output information to the user which cannot be changed when the program is running. They can be used to guide the user of the program to a certain path, controlling the data inputted by the users and preventing errors. For example, in my program, in my media player I have also added a help button which answers the user’s most basic questions and it also shows what each button does like the play button, this way the user won’t expect a button to a different thing – for example they might expect a button to fast forward instead of skipping to the next song.

## Entry Validation

Entry validation is important and useful to ensure that a program is efficient and robust – this is because data that is being inputted and stored is validated by the program before the program crashes. For example, .mp3 files are also stored in the ‘music’ list box to prevent errors from occurring this is because if there are other files then it might cause the program to crash or the media player might not work.

## Functions and Event Handlers

Functions are lines of codes that accomplish a specific task, they receive data and return a result to the user in which they need to be assigned to a variable. I used these functions as they allowed me to find errors easily which allowed me to test certain parts of the program without doing anything to the rest of the code and it also allowed me to easily understand my program.

In Visual Studio event handlers are already pre-made and is built-in the program – this allowed me to easily write a code in each event handler saving time making the whole process a lot quicker.

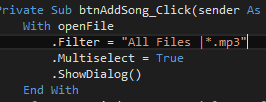
Clicking on an object in Visual Studio brings up an event handlers where you can write your code in.

## External Data

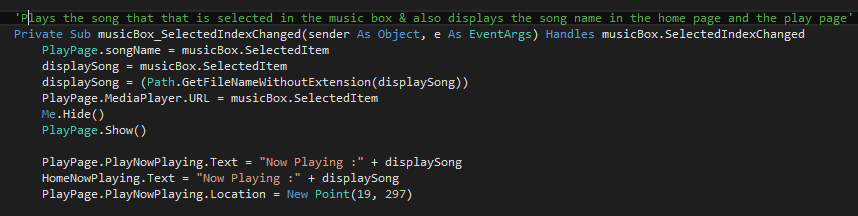
External data can be handled in different ways such as accessing it a file from outside the program or in Visual Studio’s using the ‘open file dialog’ to access a file externally as it is a built-in function and so you can play it in the media player easily.

# Software Structures

## Sequence

Sequence is a process where the code in a program are executed on line at a time. In my program, I have decided to use one of the software structures – sequence. Sequence is shown in home page of my media player via an add button. Clicking the add button opens a file dialog where the user can locate and choose the song/s they want to add.

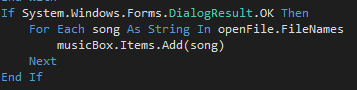
This is an example of a sequence – the user clicks on the button opening a file dialog and allowing them to choose a song

Another example of sequence in my program is when the user chooses and clicks on a song in the list box. When the user clicks on the song, the selected item is also assigned into a variable which is linked to the windows media player, the user is then sent to the play page of the media player as well as playing the song. Labels are also displayed to the user to tell them what song is currently playing and the labels appear in both the home page and the play page.

This is another example of sequence being used in my program – this sequence is initiated upon the user clicking on an item within the music box.

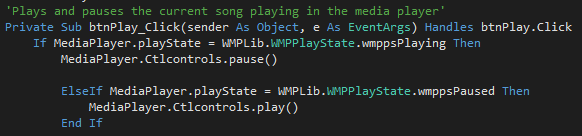
Using selection doesn’t give the user options when using the program but limits what they can do within the program preventing them from ‘breaking’ the program. Additionally, the user wouldn’t have to worry about any confusion as the program sends them to a single direction only which is opening the file dialog and adding the songs.

## Selection

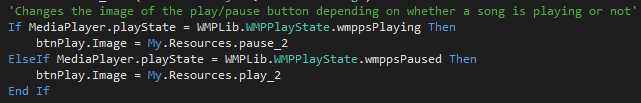
Selection or conditionals is used to evaluate whether a condition met depending on the result, to allow one or more statements / codes to run, usually the condition is an expression that uses a comparison operator to compare one value or variable with another. Throughout my program selection is commonly used. Firstly, I have used selection in the home page – when the open file dialog is opened the program checks whether the user has chosen a song or not – if not then the program doesn’t allow the user to enter that ‘file' . In this part of the program, selection is used to verify what the user is inputting making the overall program more robust and less likely to crash and produce errors.

This code is an example of selection in the home page which checks whether the user has chosen a file or not.

Moreover, selection is mostly used in the play page. For example, in the play button – the play button contains ‘IFs’ and ‘ELIFs’ statement where if music is playing then pressing the button causes it to pause. Similarly, an else if statement is used so that if the music is paused then the media player played. Using selection in this part of the program makes it more efficient, that is because I would only need one object and trigger to play and pause the button instead of having two buttons that plays or pauses the song which can take up more memory. This also provides the user more options to choose from using just a single object.

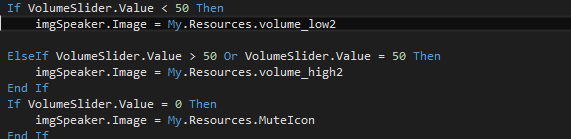


This is the code used for the play/pause button – it checks the state of the media player and does one of the code when the button is clicked depending on the media player’s state.



This code is also part of the play/pause button but coded in the ‘Timer’ – this changes the image of the button depending on the state e.g. if it is playing then the pause image is show.

Additionally, another selection in the play page is within the volume slider. The selection used in this part of the program is not so much to change the volume but to change the volume icon instead that might help the user visually on how loud the media player is. An IF statement is used so that if the volume is less than 50 then the icon changes so it will only have one sound wave coming out of the speaker. An ELIF statement is then used so that if the volume is 50 or above then the icon changes so it will have three sound waves coming out of the speaker. ELIF is also used so that if the volume is zero then the mute icon appears instead. Using conditional statements give the user has more options to choose from as it set conditions for which code to run for what they have inputted allowing data to be handled a lot easier.

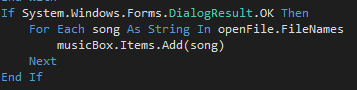


This is the part of the code that changes the image of the picture box depending on the value of the track bar.

## Iteration

Iteration or looping is used in my program to continually repeat certain blocks of codes until a condition is met. A "For" Loop is used to repeat blocks of codes a certain number of times depending on what the programmer set it to be. ‘While’ loops are used to repeat certain block of codes an unknown number of times or until a condition is met. Looping makes a code more efficient as instead of repeating the code several times I only need a single ‘while’ line of code to repeat a series of code. It was also quick and easy preventing a programmer from being repetitive.

In my program, I have used a ‘For’ loop to add the list box. Using a ‘For’ loop makes it a lot easier and quicker for the user to use the media player as it allows them select multiple songs and add them in the list box in one open dialog. This is because the ‘For’ loop enables the program to add all the chosen songs in one go instead of the user repeatedly click on the add button several times to add the songs that they want.



This code is an example of iteration – every file that is selected by user in the open file dialog gets added to the list box instead of the user adding each file one by one.

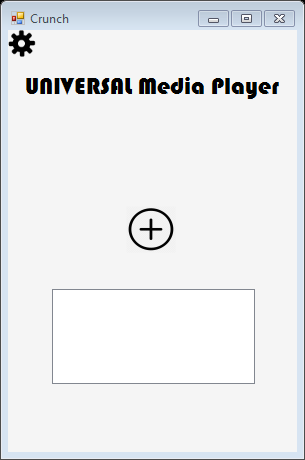
## Efficiency

Efficiency is a vital part of programming as it allows several things such as debugging is easier, files can be smaller and takes up less memory space and it allows programmer to collaborate easily making them understand what the code is all about. I have created a quite efficient program as Visual Studio has built-in event handlers or functions that made the process so much quicker and it allowed me to test the program and find bugs quicker. These event handlers also prevented me from repeating lines of code.

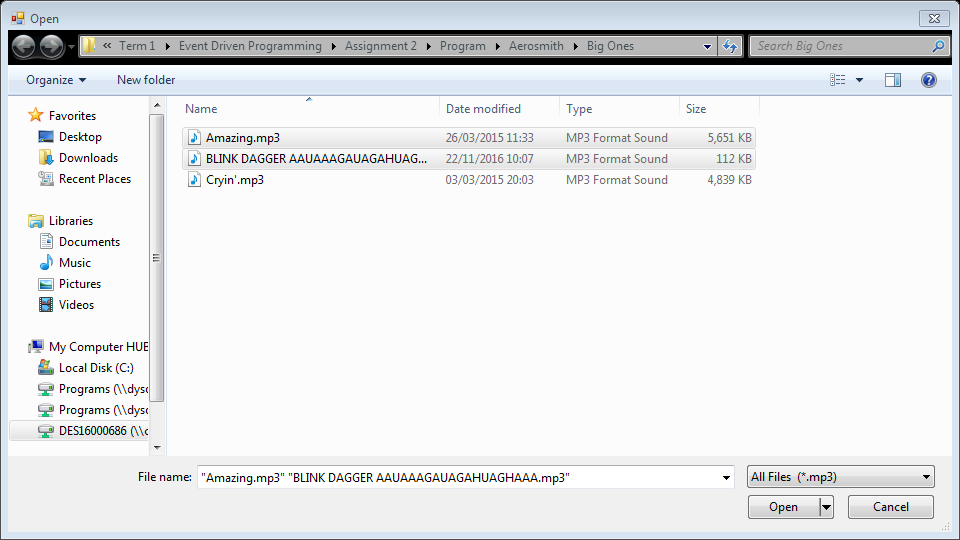
## Robustness

To make my program robust, I have made sure that the program doesn’t end or causes a problem to the user unexpected; instead the user redirected somewhere else where the problem is solved appropriately. This is done by using Else If statements. So that the program does not simply crash or restart but instead handles the error a different way to enable the user to use the program (the media player). For example, else if the media player is playing then pause the song.

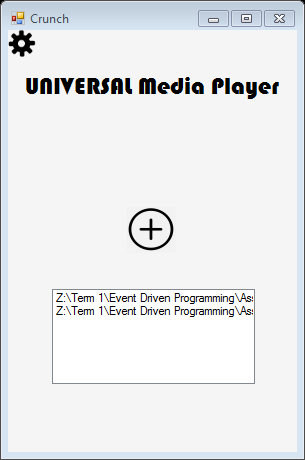
# Print Screen Evidence of Media Player



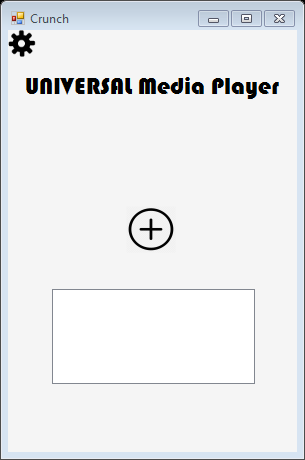
In my media player, I have created a homepage but is missing some of the features in my screen designs such as the search box and the genre box. This is because of time issues of the way you code them – this prevents me from meeting two of my user requirements. However, the setting button is still there and also added an add button and a list box which instead will contain all the music that the user can click on.

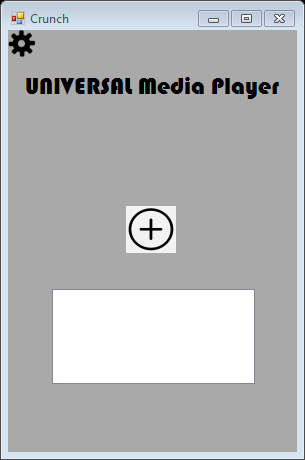
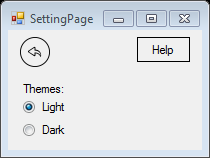
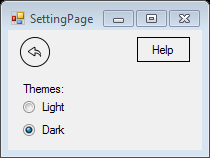


When I click on the add button, the open file dialog appears that allows the user to choose the song that they want to listen to.



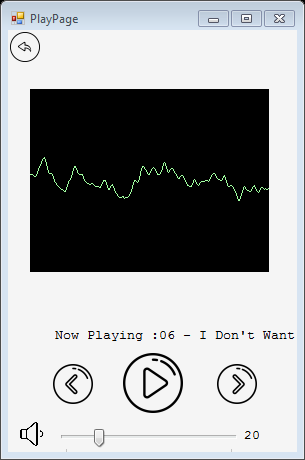
Moreover, when they decide to add that song it gets added to the list box which will allow them to click on the song and play it for them. Upon clicking on the song that they desire, they are sent to the play page. (see below)





When I click on the setting button the user is sent to the setting page that allows them to change their theme from ‘light’ or ‘dark’. Choosing the light option changes the media player to a lighter theme and choosing the dark option changes it to a darker theme. The back button in the setting page makes the page hide itself.

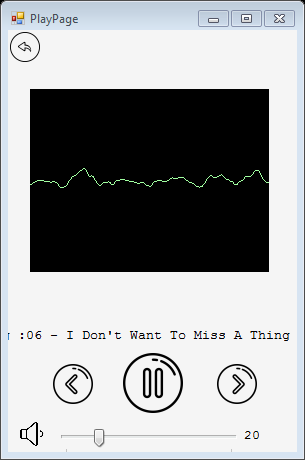
The setting page is also where the ‘Help’ button is situated that helps the user use the media player properly.



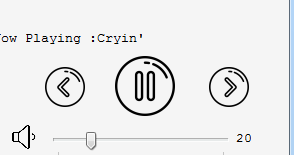
I have created a play page for my media player - the play page contains a visualiser, play and pause button, next and previous buttons and also a volume slider that the user can use.

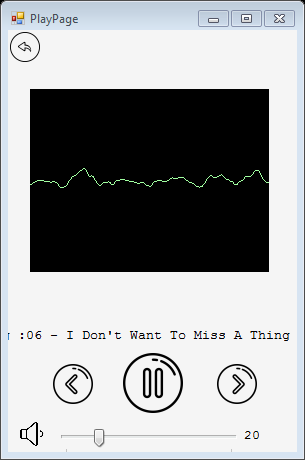
As the song that the user has chosen is played, the visualiser starts to work, going in beat of the music. The scrolling text also starts moving telling the user the current song that is playing.

In my media player, there is a play/pause button allows the user to play or pause the music whenever they choose to do so. The image on the button changes depending on the play state of the media player. If a song is currently playing then the pause button shown, but if the song is paused then the play button is shown.



There are also next and previous buttons that allows the user to change the current song. Clicking on next button skips the current song to the next item on the list box and automatically plays that song. Similarly, the previous button skips to the song to the previous item on the list box.



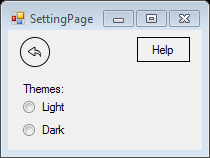


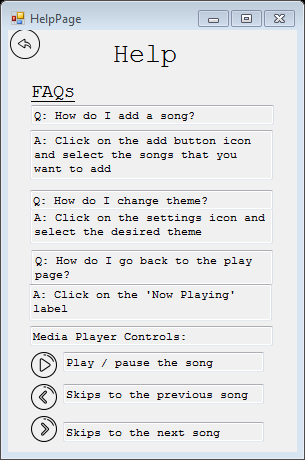
In my play page, I have also added a volume slider which allows the user to change the volume by sliding the bar for precision instead of clicking on the side volume buttons on the phone. A value of how loud the media player is, is also shown using a label.

Additionally, the picture box linked to the volume slider also changes depending on its value, it displays a mute image if it is at 0, low volume if it is between 1-49 and a high volume if it is 50 or above.









This is the on screen help and what pops up when the help button on the setting page is clicked on the setting page. In this page, the frequently asked questions about the media player are answered and it also tells them the basic controls when using the media player

# Code listing

## Home Page Code

Imports System.IO

Public Class HomePage

'Variable where the song is assigned so that it is displayed to the user

Public displaySong As String

'Sets the default theme of the media player to the 'light' theme

Private Sub HomePage\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

Me.BackColor = Color.WhiteSmoke

End Sub

'Loads the setting page & makes sure that no option is chosen between the themes

Private Sub btnSettings\_Click(sender As Object, e As EventArgs) Handles btnSettings.Click

SettingPage.Show()

SettingPage.HiddenButton.Checked = True

End Sub

'Opens a file directory and allows the user to add songs to the list box

Private Sub btnAddSong\_Click(sender As Object, e As EventArgs) Handles btnAddSong.Click

With openFile

.Filter = "All Files |\*.mp3"

.Multiselect = True

.ShowDialog()

End With

If System.Windows.Forms.DialogResult.OK Then

For Each song As String In openFile.FileNames

musicBox.Items.Add(song)

Next

End If

End Sub

'Plays the song that that is selected in the music box & also displays the song name in the home page and the play page

Private Sub musicBox\_SelectedIndexChanged(sender As Object, e As EventArgs) Handles musicBox.SelectedIndexChanged

PlayPage.songName = musicBox.SelectedItem

displaySong = musicBox.SelectedItem

displaySong = (Path.GetFileNameWithoutExtension(displaySong))

PlayPage.MediaPlayer.URL = musicBox.SelectedItem

Me.Hide()

PlayPage.Show()

PlayPage.PlayNowPlaying.Text = "Now Playing :" + displaySong

HomeNowPlaying.Text = "Now Playing :" + displaySong

PlayPage.PlayNowPlaying.Location = New Point(19, 297)

End Sub

'Sends user back to play page on click

Private Sub HomeNowPlaying\_Click(sender As Object, e As EventArgs) Handles HomeNowPlaying.Click

Me.Hide()

PlayPage.Show()

End Sub

End Class

## Play Page Code

Public Class PlayPage

Public songName As String

'Plays and pauses the current song playing in the media player

Private Sub btnPlay\_Click(sender As Object, e As EventArgs) Handles btnPlay.Click

If MediaPlayer.playState = WMPLib.WMPPlayState.wmppsPlaying Then

MediaPlayer.Ctlcontrols.pause()

ElseIf MediaPlayer.playState = WMPLib.WMPPlayState.wmppsPaused Then

MediaPlayer.Ctlcontrols.play()

End If

End Sub

'Skips to the previous song (previous item in the list box) and also loops the song around to the last song if it is the first item in the list box

Private Sub btnPrevious\_Click(sender As Object, e As EventArgs) Handles btnPrevious.Click

If HomePage.musicBox.SelectedIndex < HomePage.musicBox.Items.Count Then

HomePage.musicBox.SelectedIndex = HomePage.musicBox.SelectedIndex - 1

MediaPlayer.URL = songName

MediaPlayer.Ctlcontrols.play()

End If

If HomePage.musicBox.SelectedIndex < HomePage.musicBox.TopIndex Then

HomePage.musicBox.SelectedIndex = HomePage.musicBox.Items.Count - 1

End If

End Sub

'Skips to the next song (nextitem in the list box) and also loops the song around to the first song if it is the last item in the list box

Private Sub btnNext\_Click(sender As Object, e As EventArgs) Handles btnNext.Click

If HomePage.musicBox.SelectedIndex < HomePage.musicBox.Items.Count - 1 Then

HomePage.musicBox.SelectedIndex = HomePage.musicBox.SelectedIndex + 1

MediaPlayer.URL = songName

MediaPlayer.Ctlcontrols.play()

ElseIf HomePage.musicBox.SelectedIndex = HomePage.musicBox.Items.Count - 1 Then

HomePage.musicBox.SelectedIndex = HomePage.musicBox.TopIndex

End If

End Sub

'Chanes the value of the media player as well is displaying the value

Private Sub VolumeSlider\_Scroll(sender As Object, e As EventArgs) Handles VolumeSlider.Scroll

MediaPlayer.settings.volume = VolumeSlider.Value

txtVolume.Text = VolumeSlider.Value

'Changes the image of the picture box depending on the value of the trackbar

If VolumeSlider.Value < 50 Then

imgSpeaker.Image = My.Resources.volume\_low2

ElseIf VolumeSlider.Value > 50 Or VolumeSlider.Value = 50 Then

imgSpeaker.Image = My.Resources.volume\_high2

End If

If VolumeSlider.Value = 0 Then

imgSpeaker.Image = My.Resources.MuteIcon

End If

End Sub

'Goes back to the home page'

Private Sub btnBack\_Click(sender As Object, e As EventArgs) Handles btnBack.Click

Me.Hide()

HomePage.Show()

End Sub

Private Sub Timer\_Tick(sender As Object, e As EventArgs) Handles Timer.Tick

'Changes the image of the play/pause button depending on whether a song is playing or not'

If MediaPlayer.playState = WMPLib.WMPPlayState.wmppsPlaying Then

btnPlay.Image = My.Resources.pause\_2

ElseIf MediaPlayer.playState = WMPLib.WMPPlayState.wmppsPaused Then

btnPlay.Image = My.Resources.play\_2

End If

'Makes the label into a scrolling text that keeps going around'

PlayNowPlaying.Left = PlayNowPlaying.Left - 3

If PlayNowPlaying.Left < 0 - PlayNowPlaying.Width Then

PlayNowPlaying.Left = 575

End If

'Goes to the next song as soon as the current track is finished'

If MediaPlayer.playState = WMPLib.WMPPlayState.wmppsStopped Then

If HomePage.musicBox.SelectedIndex = HomePage.musicBox.Items.Count - 1 Then

lblEndPlaylist.Show()

PlayNowPlaying.Hide()

End If

HomePage.musicBox.SelectedIndex = HomePage.musicBox.SelectedIndex + 1

MediaPlayer.URL = HomePage.musicBox.SelectedItem

MediaPlayer.Ctlcontrols.play()

End If

End Sub

'Starts the timer which'

Private Sub PlayPage\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

Timer.Start()

HomePage.Timer1.Start()

lblEndPlaylist.Hide()

End Sub

End Class

## Setting Page Code

Public Class SettingPage

'Changes the media player's theme to light

Private Sub optLight\_CheckedChanged(sender As Object, e As EventArgs) Handles optLight.CheckedChanged

HomePage.BackColor = Color.WhiteSmoke

PlayPage.BackColor = Color.WhiteSmoke

End Sub

'Changes the media player's theme to dark

Private Sub optDark\_CheckedChanged(sender As Object, e As EventArgs) Handles optDark.CheckedChanged

HomePage.BackColor = Color.DarkGray

PlayPage.BackColor = Color.DarkGray

End Sub

'Opens the help page and hides the setting page

Private Sub btnHelp\_Click(sender As Object, e As EventArgs) Handles btnHelp.Click

HelpPage.Show()

Me.Hide()

End Sub

'Hides the setting page

Private Sub btnBack\_Click(sender As Object, e As EventArgs) Handles settingsbtnBack.Click

Me.Hide()

End Sub

End Class

## Help Page Code

Public Class HelpPage

'Sends the user back to the home page

Private Sub btnBack\_Click(sender As Object, e As EventArgs) Handles btnBack.Click

Me.Hide()

End Sub

End Class

# Mobile Media Player Test Plan

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Number** | **Description** | **Test Data** | **Expected Outcome** | **Actual Outcome** | **Action / Notes** |
| 1 | Open setting page | Click on the setting button | When I click on the setting button, a pop up must appear that allows me to choose between a light theme and a dark theme | When I clicked on the setting button, a setting page popped up showing the themes that you could choose as well as a help button that the user can click. |  |
| 2 | Choosing between two different themes | Choose ‘light theme’ | When I choose the light option it should change the layout of the media player to a lighter background, | Upon the setting page popping up, when I choose the ‘light’ option the layout of the media player changed to a lighter theme. |  |
| 3 | Choosing between two different themes | choose ‘dark theme | Similarly when I choose the ‘dark theme’ it should also switch it to a media player with a lighter background | When I chose the ‘dark’ theme, the layout of the media player has also changed into a darker theme, e.g. the background turned grey instead of white. | Dark theme still needs improvement with how appealing it looks to the user |
| 4 | Show help page | Click on the help button on the setting page | When I click on the help button, a pop up help page must appear which displays answers of the most common questions. | When I clicked on the help button, the help page popped up; containing the explanation of the basic controls and the most common questions asked. |  |
| 5 | Adding song | Click on the add button and add an mp3 file called ‘Snow’ | When I click on the mp3 in the open file dialog and press ‘Open’, the music that I have chosen should be added to the music (list) box | When I clicked on the mp3 file, it added the song to the music box which allowed me to click on the song and play it. |  |
| 6 | Adding song | Click on the add button and try to find a different file that is not .mp3 such as .png or .doc | When I add a png or doc file it shouldn’t be accepted since it is a media player. | When I clicked on the add button only .mp3 files are shown. | This was a better solution as it made the program more robust. |
| 7 | Showing the genre list | Click on the drop down list | When I click on the genre drop down box, it should show me a list of genre – the list should contain a genre of: classical, country, folk, electronic, jazz, reggae, rock, anime. | When I clicked on the genre drop down box, it showed me list of all the genre which are: classical, country, folk, electronic, jazz, reggae, rock, anime. |  |
| 8 | Showing the user the songs within the genre | Choose the ‘Rock’ genre | When I choose the ‘rock’ genre, it should send me to a list of songs that are all ‘Rock’ | I created a list box for my genre and added all the genre types that the user can choose. However due to time limitations, this functionality of the media player was not implemented and was left out. |  |
| 9 | Searching for a song using the search bar | Search for a song called ‘Snow’ | When I search for the song ‘Snow’ it should show a list of all the songs and artist that has Snow in it where I can click to play the music. | I initially made this text box so that when the user search for a specific word (‘Play’ is what I used) it sends them to the play page – this is the logic that I was using for the search bar. However due to time limitations, this functionality of the media player was not implemented and was left out. |  |
| 10 | Playing the music by clicking it on the music box | Choose a song on the music box | When I click on the ‘Snow’ option or any song, it should play the music. | When I clicked on ‘Snow’ or any song in the music box it just sent me to the play page and I had to click the play button to actually play it. | I have changed it so that when the user clicks on the song in the list box, then they are sent to the play page and the song automatically plays. |
| 11 | Display an image or make sure that the visualiser is moving | Play a song | When I play a song, the visualiser should go with the beat of the song | When the song was playing, the visualiser was in the beat of the song showing me that it is playing. |  |
| 12 | Pausing the music that is currently playing | Click on the play/pause button | If the song is playing, when I click on the play/pause button, it should pause the song to the current time and vice versa. | Initially, clicking on the pause/play button resets the whole song instead of just pausing it. However I have changed it so that when the song was currently playing and I clicked on the play/pause button, then the song was paused to the current time. Alternatively, when the song was paused and I clicked on the button again; then the song resumed to the current time. | When the current song ended without clicking on the next button it didn’t automatically play the next song. |
| 13 | Skip to the next song | Click on the next button | When I click on the next button, it should skip the current song and play the song that is next on the list | When I clicked on the next button, it skips the current song and plays the song that is next on the music box.  However, if the current song is the last item on the music box then it doesn’t allow it to skip. | I have changed it so that if the music is the last item on the list box, clicking on the next button sends it to the first item of the music box so it is on a loop. |
| 14 | Go back to the previous song | Click on the previous button | When I click on the next button, it should skip the current song and play the previous song on the list. | When I clicked on the previous button, it skips the current song and plays the previous song on the music box.  However, like the next button, if the current song is the first item on the music box then it doesn’t allow it to skip | I have changed it so that if the music is the first item on the list box, clicking on the previous button sends it to the last item of the music box so it is on a loop. |
| 15 | Increase the volume of the media player | Increase the volume slider | When I increase the volume slider the music that is being played should also go louder | When I increased the volume slider the media player played the music louder. | I have also added a label in my media player so that the value of the volume slider is being displayed |
| 16 | Decrease the volume of the media player | Decrease the volume slider | When I decrease the volume slider the music that is being played should go quieter | When I decreased the volume slider the media player played the music quieter. | I have also added a label in my media player so that the value of the volume slider is being displayed |
| 17 | Increase volume to highest value | Increase volume slider highest possible value | When I increase the volume slider the highest possible value should be 100 | When I increased the volume slider, the highest possible value was 100 |  |
| 18 | Decrease volume to lowest value | Decrease volume slider lowest possible value | When I decrease the volume slider the lowest possible value should be 0 – muting the song | When I decreased the volume slider, the lowest possible value was 0 and muted the song |  |

# Test Results Analysis

**Test 1 – Opening setting page**

In this test, I expected for the setting page to pop up upon clicking on the setting button; when I debugged my program the actual outcome was what was expected which allowed the user to choose a theme. In the setting page, I have also added the help button, therefore clicking on the setting button then clicking on the help button sends the user to the help page.

**Test 2 & 3 – Choosing between two different themes**

The actual outcome of the second test and third test was also what I put in the expected outcome of my test plan and got normal results. When I debugged my program, choosing the light option made the theme of the media player and choosing the dark option made the theme of the media player darker. However, the darker theme needs improvement such as a better darker background and suitable font colours that blends in with the background – this is because the current dark theme doesn’t look pleasing and appealing.

**Test 4 – Show help page**

The fourth test in my plan was to click on the help button and have a help page pop up; as expected, the actual outcome was the same – the help page appeared which what I thought would have been the most frequently asked questions and I have also explained to the user the basic controls of the media player which aids them when using it.

**Test 5 & 6 – Adding songs**

Test 5 was a test that used normal data – an mp3 file was being used and it was accepted by the media player and played the songs and it passed the test without problems. Test 6 was a test that should have used abnormal data; for example .png files or .doc files – which shouldn’t get accepted by a media player. However, these kinds of files can’t be added to my media player as I have made it more robust by only showing the user mp3 and filtering out any other files when opening

**Test 7 – Showing the genre list**

In test 7, I expected that the list box that I created will show me a list of all the different genre (classical, country, folk, electronic, jazz, reggae, rock, anime). When I debugged my program, the actual outcome of the test was the same as the expected outcome without any difference.

**Test 8 – Show the user the songs within the genre**

I created a list box for my genre and added all the genre types that the user can choose. However due to time limitations, this functionality of the media player was not implemented and was left out.

**Test 9 – Searching for a song using the search bar**

I initially made this text box so that when the user search for a specific word (‘Play’ is what I used) it sends them to the play page – this is the logic that I was using for the search bar. However due to time limitations, this functionality of the media player was not implemented and was left out.

**Test 10 – Playing the music by clicking it on the music box**

In this test, initially, the actual outcome was different to the expected outcome. When I debugged my program, upon clicking on a song in the music box it sent me to the play page but didn’t automatically play the song instead, I had to click on the play button to play the song. However, I have changed my program so that when the user clicks on the song then the user is sent to the play page and the song is automatically played.

**Test 11 – Display an image or make sure that the visualiser is moving**

When I chose a song on the list box and played it, the visualiser started moving as it was expected and there weren’t any differences in my expected outcome and my actual outcome.

**Test 12 – Pausing the music that is currently playing and vice versa**

During the first stages of my testing, when I clicked on the play/pause button instead of just pausing the track it reset the whole song, this wasn’t what I expected. However I have changed it so that when I click on the play/pause button and a song is currently playing then it pauses a song. Alternatively, when the song is paused and I click on the button again then it resumes the current song. The image of the button also changes depending on the state of the media player if it is playing then the pause image shows, and if it is paused then a play image shows.

In my media player, when the song ended, it also didn’t automatically played the next song. However, I have also resolved the problem and now in my program as soon as the track ends then the media player the next song in the list.

**Test 13 – Skip to the next song**

When I first coded the next button I used the pre-built windows media player control – this wasn’t what I expected in this test as the line of code that I used fast forwarded the song instead of skipping to the next song. However, I have amended my code and the next button now skips the current song and plays the song that is next on the list box.

Additionally, when I did my tests, I have found out that if the song is the last item on the list box then clicking on the next button doesn’t do anything –it doesn’t skip to anything. However, I have changed it so that if the music is the last item on the list box, clicking on the next button sends it to the first item of the music box so that it is on a loop.

**Test 14 – Go back to the previous song**

When I first coded the previous button I used the pre-built windows media player control – this wasn’t what I expected in this test as the line of code that I used rewound the song instead of going back to the previous song. However, I have amended my code and the previous button now skips the current song and plays the song that is next on the list box.

Additionally, when I did my tests, I have found out that if the song is the first item on the list box then clicking on the previous button caused the program to crash. However, I have changed it so that if the music is the first item on the list box, clicking on the next button sends it to the last item of the music box so that it is on a loop.

**Test 15 – Increase the volume of the media player**

When I debugged my program for the test, the music was being played louder as I increased the volume of the media player – this was what I expected and there were no problems. I have also added a label in my media player so that the value of the volume slider is being displayed.

**Test 16 – Decrease the volume of the media player**

Similarly, when I debugged my program for this test, the music was being played quieter as I decreased the volume of the media player – this was what I expected and there were no problems. I have also added a label in my media player so that the value of the volume slider is being displayed.

**Test 17 & 18 – Increasing and decreasing volume slider to highest and lowest possible value**

Both of these test were abnormal data tests and ran as expected increasing the volume slider to the highest possible value made it 100 and increased the volume of the media player loudly. Similarly, decreasing the volume slider to the lowest possible value made it 0 which also worked as expected as it muted the song that was playing. Both of the test also didn’t go above or below what the expected numbers were.

**Normal Data Testing**

Normal data testing is a type of testing where data is accepted and is correct therefore the program shouldn’t have any problems with it. In my program, normal testing is used in different ways such as adding mp3 files or moving the volume slider to appropriate levels.

**Abnormal Data Testing**

Abnormal data testing is a type of testing where data should not and is not accepted by a program or a system. In my program, abnormal data is not used to test the program – this is because the media player mostly run on triggers and doesn’t require any user input (e.g. a search) as I have not implemented the search bar and removed it from the final product of the program. Additionally, another abnormal test that I could have done was use .png or .doc files to see how the media player copes with it – however, in my program I have made it so that it filters out any file type apart from .mp3 files therefore you could not add these kinds of file.

**Boundary Data Testing**

Boundary data testing is a type of testing where the data used to test the program are the extremes. An example of this within my program would be increasing and decreasing the volume slider to their highest and lowest possible value to ensure that using these values still works as expected.

# Evaluation

**Creating the screen designs**

Implementing my screen designs was quite easy and very simple to do, Visual Basic gives you the forms and tools and you can just drag and drop the tools giving me the freedom to create the triggers of media player the way I want. The actual outcome of my media player is quite similar to my screen designs but I have also adapted it by adding triggers to fit situations where I didn’t think about a scenario when creating it. For example, I have added new features such as the add button that is because I have missed to put it in my screen design and it is a vital part of the media player. Additionally, in my screen designs, I did not include a back button for the setting page and the homepage, however in the final product they were added - this is because there was no way for the users to exit the pop up screens when using a mobile phone so a back button is best way get make them disappear so the user can use the media player and play songs.

When I started creating my screen designs – I first started with the home page and initially followed my design. I added all the triggers e.g. the labels, textbox, picture box and combo box. This was quite easy to do as all I had was to drag and drop all these tools.

After the homepage, I then created my play page where I added the Windows Media Player visualiser which acted as the core tool within my media player - this was simple to do as all I had to do was import the media player to my form, get rid of all the controls and only show the visualiser. I then added the main controls such as play, pause and next using picture boxes and then lastly I added the track bar which acted as my volume slider. My volume slider also included a label which tells the user the value of the track bar and a picture box that changes depending on the value of the slider.

However, I did change a part of the play page which was the music box (where the music will be stored). Instead, I have moved the music box to the home page – this is because the play page was getting overcrowded with triggers which wasn’t very user friendly and the home page was quite empty and had enough space for a list box therefore I utilised the space in the home page to put the list box there.

I then created my setting page which consisted of radio buttons and labels which were the themes that the user could choose. I also thought that the setting page was a suitable place to put the ‘Help’ button.

Lastly, I created the ‘Help’ page which was the simplest part of the whole media player as all I had to do was use labels and picture boxes to come up with questions that the user might ask and answer them and I also used picture boxes to show the user the controls and explain to them what it does.

**Coding the Home Screen**

Program the home screen of my media player was quite a daunting task – this was because programming the search box and the genre box proved to be difficult. Therefore, I left these two triggers initially and coded the rest of the triggers within the homepage. However, in the end both of the ideas were left out because of issues concerning time and meeting the deadline, I also thought that a media player without these functions works perfectly fine.

Apart from the two problems that I originally met, coding the rest of the home page was quite easy; for the setting button, all I had to code was to show the setting page on click.

The add button wasn’t in my home screen, however I realised that it was a necessary trigger to allow the users to listen to the song that they want in their media player. Thus, this also caused me to use the ‘open file dialog’ tool which open files from any directory which I used to add the songs that the user wants in their the list box (where the music will be stored).

In the home screen, the last trigger that I coded was the list box which was because this wasn’t intended to be here but it was an improvisation to utilise space and prevent overcrowding on a play page. The list box had quite a fair amount of code consisting of sequence which was to mostly link the items on the list box to the media player so that the media player can play the song.

**Coding the play page**

The play page was the form where there were a lot of triggers as this page enables the user to listen to their song – thus it consisted of the main media player controls such as the previous button, a next button and a play/pause button. The play page is also where the Windows Media Player a main tool in the media player that I am creating to play the songs, and there are also other triggers such as a volume slider, a back button and a scrolling text that tells the user the current song that is playing.

The first trigger in the play page that I programmed was the play/pause button; since I used the Windows Media Player as a core tool for my media player this was quite an easy task. The Windows Media Player has pre-built codes that pauses and plays the song within the media player therefore I used the code and made a working button that plays and pauses the track.

However, I initially had problems when creating the play button, this was because in my mind I was going to have two buttons – one that plays the song and one that pauses the song but when I programmed it, it did not work as I couldn’t show the pause button when a song is playing. Instead, I fixed the problem by only having a single button and used conditional statement, I used the state of the media player to change the image of the button to fit the purpose such as if a song is playing then show the pause image and coded in a way so that it plays the song when it is currently paused and vice versa. In the end, I found that this was the most efficient and best code for my program and made it my final solution for the play and pause button.

The next two triggers that I programmed in my play page were the next and previous button, I had both of these triggers planned out logically in my pseudocode but when it came to programming it I faced some trouble.

When I first programmed the buttons, I initially used the pre-built function that came with the Windows Media Player, however this gave me a logic error – this was because even though the program worked it wasn’t what I wanted because instead of skipping and going back songs it only fast forwarded and rewound the song and so using the Windows Media Player pre-built codes wasn’t the way to go. Therefore, I used my pseudocode and used if statements programmed it in a way that takes one away or adds one to the index using selection.

The code that I used worked fine when it comes to skipping and going back songs, however problem arose when the selected index was the first or the last in the list box – this was because it clicking on the buttons (next or previous) would make the index out of range causing a syntax error and crashing the program. The first solution that I have thought was the hide the buttons if the selected item was the first or last item on the list – I was then able to code the previous button, hiding it when the index was the first on the list and reverse engineered the code to make it work for the next button. However, I did not use the idea for my final solution as it didn’t made the media player look appealing and it limited what the user was able to do.

Instead, I found another solution for the problem where clicking on both buttons looped it round both ways if it was the first or last item on the list. I managed to code this to both of the buttons and they managed to work perfectly looping the songs when they are the first or last item on the list – this solved the syntax error that I was facing and prevent the program from crashing which will allow the user to use it without worries and without limitations of what they can do.

The volume slider was the next trigger that I coded to allow the user to have a more precise way to change the volume instead of using the side buttons of the mobile phone which changes the volume in increments. Programming the volume slider was quite easy, I just linked and made the volume settings of the media player equal to a track bar that has a value of up to 100 which I achieved with a couple of lines of code. Additionally, along with the track bar is a picture box and a label – the label displays the value of the track bar (volume slider) and the picture boxes displays the different levels of volume of the media player (high ,low, mute) changing images depending on the value of the track bar which allows the user to visualise the volume.

Lastly, I coded my label; at first I had a problem with displaying only the file name to the user as when I coded it, it displayed the whole file path which was long and confusing preventing the program from being user friendly. However, after research I found out that you can get the file name without the extension and so I created a new variable and assigned the selected item into that variable to convert it into just a file name – this solution worked and only displayed the file name to the user.

Additionally, I wanted the label to be a scrolling text, if I decided to keep the text stationary then it would’ve been very easy as it would have required only several lines of code. However, I wanted my text to be scrolling to make it more interactive to excite the audience and prevent them from getting bored. Therefore, I had to use a timer and code it to make the label move to left once the timer ticks (the timer ticks once the form loads and music is played) this was quite straightforward and produced a better media player.

After several tests, I have also noticed when the current song ended without clicking on the next button it didn’t automatically play the next song. Therefore, I coded an extra block of code in my timer so that if the media player is stopped then the index is added one automatically playing the next song.

**Coding the setting page**

Coding the setting page was also one of the easy forms that I coded, this was because I only had to code the radio buttons and the lines of code needed wasn’t that much – it was mostly to change the background, font, colours – the style of the media player on click this was because I planned to have two different themes as suggested on my screen designs.

**Coding the back button**

In all of the pages apart from the home screen there are back buttons; this was the simplest and easiest trigger to program and shows an example of sequence as all I had to do was on user click hide the current page and show the home screen.

**Conclusion**

Overall, I am quite happy with the final solution of my program – this was my very first time using Visual Basic and VB.Net as a programming language and I thought that I managed to do quite a lot for a first timer. I have also used different kind of code structures and my code is quite efficient. However, there are also different things that I think I can improve upon; an example of this would be functionalities that I have not implemented that could have improved the overall program vastly – this would include the search bar which would have allowed the user to search the song at their will. Also, a working genre box would have helped better with the media player which would have allowed the user to choose a genre type that they would want to listen to.