Six Weeks Industrial Training Project Report

On

Music Player Application

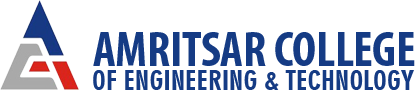
Submitted in the partial fulfillment of the requirement for the Award of Degree of

**Bachelor of Technology**

**In**

**INFORMATION TECHNOLOGY**

**Batch (2016-2020)**

****

|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
| **Submitted To:**  Er. Amarpreet Singh  Department of CSE/IT |  |

|  |  |
| --- | --- |
| **Submitted By:**  Devi Parsad Samal (1600736)  Kamalpreet Singh (1600755)  Gurpreet Singh (1600612) |  |

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING / INFORMATION TECHNOLOGY**

**Amritsar College Of Engineering & Technology, Amritsar**

**(Autonomous college under UGC Act – 1956|a (f) and 12 (B)|)**

**AKNOWLEDGEMENT**

It is a matter of immense pleasure that we take this opportunity to express our heart felt gratitude to **“Mr. Amarpreet Singh”**, Head, Department of Computer Science and Information Technology, **ACET, Amritsar** for the invaluable ideas, assistance and encouragement that he provided during our study. We wish to express thanks to him for his helpful reviews, comments and continual support on our studies. We shall always visualize his encouraging gesture, competent guidance and constant inspiration to learn more and more.

The completion of this project would not have been possible without the help and encouragement of our teacher. We would like to thank all the people who have helped us in one way or another to complete our project. Finally, we would like to thank all our family members for their cooperation and good wishes in completing our work. We feel short of words to express our deep sense of gratitude towards the **Almighty God** with whose grace present study has been completed.

Devi Parsad Samal (1600736)

Kamalpreet Singh (1600755)

Gurpreet Singh (1600612)

**Table Of Contents**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Topic** | **Page No.** |
| 1. | About Training Organization | 4-6 |
| 2. | Introduction To PYTHON | 7-10 |
| 3. | Introduction To Project | 11-12 |
| 4. | Modules Used In Project | 13-15 |
| 5. | Tools To Be Used For Development | 16-17 |
| 6. | Programming Code | 18-34 |
| 7. | Screen Shots Of The Project | 35-39 |
| 8. | Hardware And Software Requirements | 40 |
| 9. | Bibliography | 41 |
|  |  |  |

**About Organization: Piford Technologies**



**1. Details Of Piford Technologies:**

Piford Technologies is a USA based Software Development Company. It has vast experience in working with Independent Software Vendors (ISVs) for their Custom Development needs. Client segments that we cater to range from early stage startups to large and established market leaders. Our technical expertise encompasses various technologies such as [Android](http://www.piford.com/android-app-development-training-course-in-chandigarh/) | [iPhone](http://www.piford.com/iphone-app-development-training-course-in-chandigarh/) | [JAVA](http://www.piford.com/java-training-course-in-chandigarh/) | [PHP](http://www.piford.com/php-training-course-in-chandigarh/) | [Python](http://www.piford.com/python-training-course-in-chandigarh/) | [Hadoop](http://www.piford.com/hadoop-training-course-in-chandigarh/) | [Big Data](http://www.piford.com/big-data-training-course-in-chandigarh/) | [.Net](http://www.piford.com/dot-net-training-course-in-chandigarh/) | [Web Design](http://www.piford.com/web-design-training-course-in-chandigarh/) | [Software Testing](http://www.piford.com/software-testing-training-course-in-chandigarh/) | [C](http://www.piford.com/c-training-course-in-chandigarh/) | [C++](http://www.piford.com/c-plus-plus-training-course-in-chandigarh/) and many more…!!! We have expertise in architecting complex, state-of-the-art software products on various advanced technology platforms.

Our teams are highly skilled in building Smartphone Applications, component-based, service-oriented architectures (SOA), Software as a Service (SAAS) and Web Services. Our expertise ranges across various leading-edge development platforms, Operating Systems, Application Servers, and Databases (Object and Relational).

# 2. Training Provide By Piford Technologies:

Piford Technologies (www.piford.com) is a "Software Development Company" with its development and training divison at IT Park, Mohali. We have business development offices at USA, Israel, UK and Austria. Piford Technologies specializes in application development for Android, iPhone and Windows 10 UWP platforms. Our company provides contract services in Java, C#, PHP, Embedded System, Multicore Processor Programming and Cloud Computing.

**Employability Enhancement Program (EEP) Training Program in Chandigarh, Mohali & Panchkula:**

In Training Programs are broadly divided into following categories:

## Smartphone APP Development

* ANDROID App Development (With Core JAVA, C++).
* Windows Phone 7 (WP7) App Development (with C#.Net, C++).
* iPhone App Development.

### Software Programming

* JAVA (With C++).
* .NET (With C++).

### Website Development

* PHP
* ASP.NET
* JSP

### Advanced Technologies

* Multicore Programming
* Digital Signal Processing

### Software Testing

* Software Testing

### Embedded Systems

* Embedded Systems

**About Organization: SEBIZ Square Infotech Pvt. Ltd.**



Sebiz Square Infotech Pvt. Ltd. Company is a USA based Software Development Company.

Sebiz can help tachieve your IT and training goals. We are the IT partner you can depend on to increase business efficiencies and help your business be more visible online.  To date we have more than 3000 projects under our belt and many more underway. We provide [Software Development](http://www.sebiz.net/index.php/services/servise/)services including Web & Mobile application development and [Search Engine Marketing](http://www.sebiz.net/index.php/services/search-engine-marketing/) services to businesses big and small. Under the aegis of the [Sebiz Finishing School](http://www.sebiz.net/index.php/sebiz-finishing-school/sebiz-finishing-school/) we have also flagged off a number of important training and educational inititatives.

Sebiz has also been involved in training and education for over 10 years and for the past 4 years has been actively partnering in [Government Projects](http://www.sebiz.net/index.php/govt-project/govt-project/) for IT and vocational training for lower income and BPL categories with both – Central and State Government organizations. We are also a NCVT and SSC affiliated company for assessment and certification.

[Sebiz Finishing School](http://www.sebiz.net/index.php/sebiz-finishing-school/sebiz-finishing-school/sebiz_finishing_school.html) or SFS as it is popularly known is a brainchild of Sebiz. It is fast gaining reputation for high quality training in the Northern Region. It was set up with the goal of making fresh graduates industry-ready with highly focused programs and emphasis on lab work rather than practical. Today SFS has widened its repertoire to include Industrial Training, Online Courses and Weekend Workshops for corporates and working professionals. We are proud of the fact that in recent years SFS has helped train over 2000 IT graduates and placed over 1500 graduates in jobs.

**Introduction To PYTHON**

**What is Python?**



Python is an [interpreted](https://en.wikipedia.org/wiki/Interpreted_language) [high-level programming language](https://en.wikipedia.org/wiki/High-level_programming_language) for [general-purpose programming](https://en.wikipedia.org/wiki/General-purpose_programming_language). Created by [Guido van Rossum](https://en.wikipedia.org/wiki/Guido_van_Rossum) and first released in 1991, Python has a design philosophy that emphasizes [code readability](https://en.wikipedia.org/wiki/Code_readability), notably using [significant whitespace](https://en.wikipedia.org/wiki/Significant_whitespace). It provides constructs that enable clear programming on both small and large scales.

Python features a [dynamic type](https://en.wikipedia.org/wiki/Dynamic_type) system and automatic [memory management](https://en.wikipedia.org/wiki/Memory_management). It supports multiple [programming paradigms](https://en.wikipedia.org/wiki/Programming_paradigm), including [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming), [imperative](https://en.wikipedia.org/wiki/Imperative_programming), [fuctional](https://en.wikipedia.org/wiki/Functional_programming) and [procedural](https://en.wikipedia.org/wiki/Procedural_programming), and has a large and comprehensive [standard library](https://en.wikipedia.org/wiki/Standard_library).

Python interpreters are available for many [operating systems](https://en.wikipedia.org/wiki/Operating_system). [CPython](https://en.wikipedia.org/wiki/CPython), the [reference implementation](https://en.wikipedia.org/wiki/Reference_implementation) of Python, is [open source](https://en.wikipedia.org/wiki/Open_source) software[[29]](https://en.wikipedia.org/wiki/Python_(programming_language)#cite_note-29) and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit [Python Software Foundation](https://en.wikipedia.org/wiki/Python_Software_Foundation).

**History:**

**

Guido Van Rossum, the creator of PYTHON.

Python was conceived in the late 1980s, and its implementation began in December 1989 by [Guido van Rossum](https://en.wikipedia.org/wiki/Guido_van_Rossum) at [Centrum Wiskunde & Informatica](https://en.wikipedia.org/wiki/Centrum_Wiskunde_%26_Informatica) (CWI) in the [Netherlands](https://en.wikipedia.org/wiki/Netherlands) as a successor to the [ABC language](https://en.wikipedia.org/wiki/ABC_(programming_language)) (itself inspired by [SETL](https://en.wikipedia.org/wiki/SETL)) capable of [exception handling](https://en.wikipedia.org/wiki/Exception_handling) and interfacing with the [Amoeba](https://en.wikipedia.org/wiki/Amoeba_(operating_system)) operating system. Van Rossum remains Python's principal author. His continuing central role in Python's development is reflected in the title given to him by the Python community: [Benevolent Dictator For Life](https://en.wikipedia.org/wiki/Benevolent_Dictator_For_Life) (BDFL).

Python 2.0 was released on 16 October 2000 and had many major new features, including a [cycle-detecting](https://en.wikipedia.org/wiki/Cycle_detection) [garbage collector](https://en.wikipedia.org/wiki/Garbage_collection_(computer_science)) and support for [Unicode](https://en.wikipedia.org/wiki/Unicode). With this release, the development process became more transparent and community-backed.

**Libraries of Python:**

Python's large [standard library](https://en.wikipedia.org/wiki/Standard_library), commonly cited as one of its greatest strengths, provides tools suited to many tasks. For Internet-facing applications, many standard formats and protocols such as [MIME](https://en.wikipedia.org/wiki/MIME) and [HTTP](https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol) are supported. It includes modules for creating [graphical user interfaces](https://en.wikipedia.org/wiki/Graphical_user_interface), connecting to [relational databases](https://en.wikipedia.org/wiki/Relational_database), [generating pseudorandom numbers](https://en.wikipedia.org/wiki/Pseudorandom_number_generator), arithmetic with arbitrary precision decimals, manipulating [regular expressions](https://en.wikipedia.org/wiki/Regular_expression), and [unit testing](https://en.wikipedia.org/wiki/Unit_testing).

Some parts of the standard library are covered by specifications (for example, the [Web Server Gateway Interface](https://en.wikipedia.org/wiki/Web_Server_Gateway_Interface) (WSGI) implementation “wsgiref” follows PEP 333), but most modules are not. They are specified by their code, internal documentation, and test suites (if supplied). However, because most of the standard library is cross-platform Python code, only a few modules need altering or rewriting for variant implementations.

As of March 2018, the [Python Package Index](https://en.wikipedia.org/wiki/Python_Package_Index) (PyPI), the official repository for third-party Python software, contains over 130,000, packages with a wide range of functionality, including:

* Graphical user interfaces
* Web frameworks
* Multimedia
* Databases
* Networking
* Test frameworks
* Automation
* Web scraping
* Documentation
* System administration
* Scientific computing
* Text processing
* Image processing

**Flavours of Python:**

We always want different flavors in everything, don’t we? As they say varieties add spice to life. So python need not be exception to this. How many times have you wondered why there is python,Cpython, Jython, IronPython and many more .\*ython, that is, these different flavors of python:

1 Cpython

2 Jython

3 IronPython

4 Pypy  
5 Stackless Python

6 ActiveState ActivePython

7 Pythonxy

8 Portable Python

9 Anaconda Python

# Python Features:

#### 1. Easy to Learn and Use

Python is easy to learn and use. It is developer-friendly and high level programming language.

2. Expressive Language

Python language is more expressive means that it is more understandable and readable.

3. Interpreted Language

Python is an interpreted language i.e. interpreter executes the code line by line at a time. This makes debugging easy and thus suitable for beginners.

4. Cross-platform Language

Python can run equally on different platforms such as Windows, Linux, Unix and Macintosh etc. So, we can say that Python is a portable language.

5. Free and Open Source

Python language is freely available at [offical web address](https://www.python.org/).The source-code is also available. Therefore it is open source.

6. Object-Oriented Language

Python supports object oriented language and concepts of classes and objects come into existence.

7. Extensible

It implies that other languages such as C/C++ can be used to compile the code and thus it can be used further in our python code.

8. Large Standard Library

Python has a large and broad library and prvides rich set of module and functions for rapid application development.

9. GUI Programming Support

Graphical user interfaces can be developed using Python.

10. Integrated

It can be easily integrated with languages like C, C++, JAVA etc.

**Introduction To Project: Avatar Music Player**

**Introduction:-**

The project is about Music Player Application. In Today’s modern scenario, we all love to listen awesome and popular songs all around the world with great sound qualities.

So, “Avatar Music Player” allows us to do these amazing stuffs. It is a open source cross platform audio player that plays most media files like mp3, m4a, wav, ogg, etc. It is purely developed in Python to available for all devices.

Avatar Music Player is a full audio player with a complete database, filters and amazing visualizations playing all weird audio formats.

This application as a media library for audio files and allows to browse folders directly.

It also includes a widget for audio control, support audio headsets control, cover art and a complete audio media library.

**Purpose:-**

The purpose of developing Avatar Music Player application is to give best and amazing performance in audio streams and also makes listening experience at best level for music lovers. It is more encoded with open source, so that any user will be able to listen music at different devicees. This application allows user to listen all music with best experience at low cost.

It is also a light weight application that makes any OS to run smoothly without any bug or crashes. Avatar Music Player is also able to manipulate and manage system time, easy use with basic console, all operations at a single touch which makes this application easy to use.

Another purpose for developing this software is to make media files at speed up for all Windows OS at low cost and give smart experience over multiple media libraries.

It also has the feature of fast and efficient working with low memory and CPU usage.

**Scope:-**

In towards generations, people wants to listen music at very best and highly experience audio formats for better audio performance, So this software provides more efficient work at low cost and portable to all devices in future to make music at different platform.

The scope of the project is to give people a better experience over different audio formats i.e. the project is developed as a desktop application, and it will work for all Windows OS easily. But later on the project can be modified to operate it different OS with some more upgrades like multimedia files as well as discs, DVD’s, network streaming protocols, more equalizers and much more.

**Modules Used In Project**

**1. Tkinter: (Front End)**



Tkinter provides us with a variety of common GUI elements which we can use to build our interface – such as buttons, menus and various kinds of entry fields and display areas. We call these elements widgets. We are going to construct a tree of widgets for our GUI – each widget will have a parent widget, all the way up to the root window of our application. For example, a button or a text field needs to be inside some kind of containing window.

The widgets are:

1. The Button Widget

2. The Canvas Widget

3. The Checkbutton Widget

4. The Entry Widget

5. The Frame Widget

6. The Label Widget

7. The Progressbar Widget

8. The Listbox Widget

9. The Menu Widget

10. The Combo Box Widget

11. The Message Widget

12. The OptionMenu Widget

13. The PanedWindow Widget

14. The Radiobutton Widget

15. The Scale Widget

16. The Scrollbar Widget

17. The Spinbox Widget

18. The Text Widget

19. The Toplevel Widget

20. The Image Widget

21. The Notebook Widget

22. The Treeview Widget

23. The Status bar Widget

**2. Pygame: (Back End)**

****

Pygame is a [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) set of [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) modules designed for writing [video games](https://en.wikipedia.org/wiki/Video_game). It includes [computer graphics](https://en.wikipedia.org/wiki/Computer_graphics) and sound [libraries](https://en.wikipedia.org/wiki/Library_(computing)) designed to be used with the Python [programming language](https://en.wikipedia.org/wiki/Programming_language).

Pygame was originally written by Pete Shinners to replace PySDL after its development stalled. It has been a [community](https://en.wikipedia.org/wiki/Free_software_community)project since 2000 and is released under the [open source](https://en.wikipedia.org/wiki/Open_source) [free software](https://en.wikipedia.org/wiki/Free_software) [GNU Lesser General Public License](https://en.wikipedia.org/wiki/GNU_Lesser_General_Public_License).

Built over the [Simple DirectMedia Layer](https://en.wikipedia.org/wiki/Simple_DirectMedia_Layer) (SDL) library, with the intention of allowing [real-time](https://en.wikipedia.org/wiki/Real-time_computer_graphics) [computer game](https://en.wikipedia.org/wiki/Computer_game) development without the [low-level](https://en.wikipedia.org/wiki/Low-level_programming_language) mechanics of the [C programming language](https://en.wikipedia.org/wiki/C_(programming_language)) and its derivatives. This is based on the assumption that the most [expensive](https://en.wikipedia.org/wiki/Computationally_expensive) functions inside games (mainly the [graphics](https://en.wikipedia.org/wiki/Computer_graphics) part) can be abstracted from the [game logic](https://en.wikipedia.org/wiki/Model-view-controller), making it possible to use a [high-level programming language](https://en.wikipedia.org/wiki/High-level_programming_language), such as Python, to structure the game.

Other features that SDL doesn't have include vector math, collision detection, 2d sprite scene graph management, [MIDI](https://en.wikipedia.org/wiki/MIDI) support, camera, pixel array manipulation, transformations, filtering, advanced freetype font support, and drawing.

Applications using pygame can run on Android phones and tablets with the use of Pygame Subset for Android (pgs4a). Sound, vibration, keyboard, and accelerometer are supported on Android.

**3. Pillow: (Back End)**

Python Imaging Library (abbreviated as PIL) (in newer versions known as Pillow) is a [free](https://en.wikipedia.org/wiki/Free_and_open_source_software) library for the [Python programming language](https://en.wikipedia.org/wiki/Python_(programming_language)) that adds support for opening, manipulating, and saving many different image file formats. It is available for Windows, Mac OS X and Linux. The latest version of PIL is 1.1.7, was released in September 2009 and supports Python 1.5.2–2.7, with [Python 3](https://en.wikipedia.org/wiki/Python_3)support to be released "later".

Development appears to be discontinued with the last commit to the PIL repository coming in 2011. Consequently, a successor project called Pillow has forked the PIL repository and added Python 3.x support. This fork has been adopted as a replacement for the original PIL in [Linux distributions](https://en.wikipedia.org/wiki/Linux_distribution) including [Debian](https://en.wikipedia.org/wiki/Debian_GNU/Linux) and [Ubuntu](https://en.wikipedia.org/wiki/Ubuntu_(operating_system)) (since [13.04](https://en.wikipedia.org/wiki/Ubuntu_13.04_Raring_Ringtail)).

**4. Pyinstaller:**

PyInstaller is a program that freezes (packages) Python programs into stand-alone executables, under Windows, Linux, Mac OS X, FreeBSD, Solaris and AIX. Its main advantages over similar tools are that PyInstaller works with Python 2.7 and 3.3—3.6, it builds smaller executables thanks to transparent compression, it is fully multi-platform, and use the OS support to load the dynamic libraries, thus ensuring full compatibility.

**Tools To Be Used For Development**

**Pycharm IDE :-**

With PyCharm you can develop applications in Python. In addition, in the Professional edition, one can develop Django, Flask and Pyramid applications. Also, it fully supports HTML (including HTML5), CSS, JavaScript, and XML: these languages are bundled in the IDE via plugins and are switched on for you by default. Support for the other languages and frameworks can also be added via plugins (go to Settings | Plugins or PyCharm | Preferences | Plugins for macOS users, to find out more or set them up during the first IDE launch).

PyCharm is a cross-platform IDE that works on Windows, macOS, and Linux.

If you need assistance installing PyCharm, see the installation instructions: [Requirements, Installation and Launching](https://www.jetbrains.com/help/pycharm/install-and-set-up-pycharm.html).

****

**Features Of PyCharm :-**

* Coding assistance and [analysis](https://en.wikipedia.org/wiki/Code_analysis), with [code completion](https://en.wikipedia.org/wiki/Autocomplete), syntax and error highlighting, linter integration, and quick fixes
* Project and code navigation: specialized project views, file structure views and quick jumping between files, classes, methods and usages
* Python [refactoring](https://en.wikipedia.org/wiki/Refactoring): including rename, extract method, introduce variable, introduce constant, pull up, push down and others
* Support for web frameworks: [Django](https://en.wikipedia.org/wiki/Django_(web_framework)), [web2py](https://en.wikipedia.org/wiki/Web2py) and [Flask](https://en.wikipedia.org/wiki/Flask_(web_framework))
* Integrated Python [debugger](https://en.wikipedia.org/wiki/Debugger)
* Integrated [unit testing](https://en.wikipedia.org/wiki/Unit_testing), with line-by-line [code coverage](https://en.wikipedia.org/wiki/Code_coverage)
* [Google App Engine](https://en.wikipedia.org/wiki/Google_App_Engine) Python development

It competes mainly with a number of other Python-oriented IDEs, including [Eclipse](https://en.wikipedia.org/wiki/Eclipse_(software))'s [PyDev](https://en.wikipedia.org/wiki/PyDev), and the more broadly focused [Komodo IDE](https://en.wikipedia.org/wiki/ActiveState_Komodo).

**Programming Code: Avatar Music Player**

from tkinter.filedialog import askdirectory,askopenfilename

from tkinter import \*

from tkinter import messagebox as msgb

from tkinter import ttk

import pygame

import random

import os

class Avatar\_Music\_Player:

def \_\_init\_\_(self,my\_music\_wm):

self.first\_click=0

self.list\_of\_songs=[]

self.single\_song\_name=''

self.song\_index=0

self.single\_song\_name=''

self.volume=0.5

self.my\_loc=os.getcwd()

self.check\_me=-1

self.check\_pause=0

self.check\_mute=0

self.check\_stop=0

self.gif=[]

################################################## Main Window ###################################################################

self.my\_music\_wm=my\_music\_wm

self.my\_music\_wm.title('Avatar Music Player')

self.my\_music\_wm.resizable(0,0)

self.my\_music\_wm.geometry('1200x600')

self.my\_music\_wm.configure(background='black')

self.my\_music\_wm.wm\_iconbitmap('Avatar Icon.ico')

################################################# Menu Bar ###################################################################

self.avatar\_menubar=Menu(self.my\_music\_wm)

self.my\_music\_wm.configure(menu=self.avatar\_menubar)

self.media\_menu=Menu(self.my\_music\_wm,tearoff=0)

self.media\_menu.add\_command(label='Open file',command=self.browse\_file)

self.media\_menu.add\_command(label='Open folder',command=self.browse\_folder)

self.media\_menu.add\_separator()

self.media\_menu.add\_command(label='Exit',command=self.exit\_me)

self.avatar\_menubar.add\_cascade(label='Media',menu=self.media\_menu)

self.playback\_menu=Menu(self.my\_music\_wm,tearoff=0)

self.playback\_menu.add\_command(label='Play Music',command=self.play\_song)

self.playback\_menu.add\_command(label='Stop Music',command=self.stop\_song)

self.playback\_menu.add\_separator()

self.playback\_menu.add\_command(label='Next Song',command=self.next\_song)

self.playback\_menu.add\_command(label='Previous Song',command=self.prev\_song)

self.avatar\_menubar.add\_cascade(label='Playback',menu=self.playback\_menu)

self.audio\_menu=Menu(self.my\_music\_wm,tearoff=0)

self.audio\_menu.add\_command(label='Increase Volume',command=self.music\_up)

self.audio\_menu.add\_command(label='Decrease Volume',command=self.music\_down)

self.audio\_menu.add\_command(label='Mute',command=self.mute\_song)

self.audio\_menu.add\_separator()

self.audio\_menu.add\_command(label='Visualizaions',command=self.change\_vis)

self.avatar\_menubar.add\_cascade(label='Audio',menu=self.audio\_menu)

self.help\_menu=Menu(self.my\_music\_wm,tearoff=0)

self.help\_menu.add\_command(label='About Avatar',command=self.about\_dev)

self.help\_menu.add\_command(label='Help Docs',command=self.help\_me)

self.avatar\_menubar.add\_cascade(label='Help',menu=self.help\_menu)

########################################################### Middle Logo ##########################################################

self.ava=PhotoImage(file='Avatar Logo.png')

self.avatar\_logo=Label(self.my\_music\_wm,image=self.ava,height=400,width=800,bg='black')

self.avatar\_logo.pack(padx=100,pady=20)

########################################################## Song Label Bar ##########################################################

self.song\_label\_value=StringVar()

self.song\_label=Label(self.my\_music\_wm,textvariable=self.song\_label\_value,bg='black',fg='white',font=('Arial',10,'bold'))

self.song\_label.pack(fill=X,pady=5)

self.song\_label\_value.set("Hiii... I'm Avatar. Here You Rock N Roll With Amazing Music Channels.")

########################################################## Music Console #########################################################

self.play\_button=Button(self.my\_music\_wm,text='\u25B6',width=3,font=('Arial',26,'bold'),bd=20,activebackground='grey',command=self.play\_pause\_song)

self.play\_button.pack(side=LEFT)

self.prev\_button=Button(self.my\_music\_wm,text='I\u25c0\u25c0',width=3,font=('Arial',26,'bold'),bd=20,activebackground='grey',command=self.prev\_song)

self.prev\_button.pack(side=LEFT)

self.stop\_button=Button(self.my\_music\_wm,text='\u25A0',width=3,font=('Arial',26,'bold'),bd=20,activebackground='grey',command=self.stop\_song)

self.stop\_button.pack(side=LEFT)

self.next\_button=Button(self.my\_music\_wm,text='\u25B6\u25B6I',width=3,font=('Arial',26,'bold'),bd=20,activebackground='grey',command=self.next\_song)

self.next\_button.pack(side=LEFT)

self.voldown\_button=Button(self.my\_music\_wm,text='\u2B07',width=3,font=('Arial',26,'bold'),bd=20,activebackground='grey',command=self.music\_down)

self.voldown\_button.pack(side=RIGHT)

self.volup\_button=Button(self.my\_music\_wm,text='\u2B06',width=3,font=('Arial',26,'bold'),bd=20,activebackground='grey',command=self.music\_up)

self.volup\_button.pack(side=RIGHT)

self.empty\_label=Button(self.my\_music\_wm,height=8,width=80,text='AVATAR',font=('Colonna MT',70,'bold'),relief=FLAT,bd=0,fg='white',command=self.flash\_avatar,activeforeground='black')

self.empty\_label.pack(side=RIGHT)

def change\_vis(self):

os.chdir(self.my\_loc)

self.vis=os.chdir('Visualizations\\')

for files in os.listdir(self.vis):

if files.endswith(".gif"):

self.gif.append(files)

self.chgif=random.choice(self.gif)

self.temp\_gif=PhotoImage(file=self.chgif)

self.avatar\_logo.configure(image=self.temp\_gif)

if self.check\_me==0:

os.chdir(self.single\_song\_loc)

else:

os.chdir(self.folder\_name)

def flash\_avatar(self):

self.mycolor=['black','white','red','blue','green','yellow','pink','skyblue','purple','grey','gold','orange','darkgreen','lightgreen','darkblue','cyan','magenta','#00FF00','#FF00FF','#C0C0C0','brown','#800000','#808000','#302217','#2554C7','#25587E','#2B547E','#8D38C9','#F6358A','#800517','#7E354D','#437C17','#805817','#EDE275','#B5EAAA','#437C17']

self.chcolor=random.choice(self.mycolor)

self.empty\_label.configure(fg=self.chcolor,activeforeground=self.chcolor)

def music\_up(self):

if self.check\_mute==1:

pygame.mixer.music.set\_volume(1.0)

pygame.mixer.music.get\_volume()

self.check\_mute=0

else:

self.volume+=0.1

pygame.mixer.music.set\_volume(self.volume)

pygame.mixer.music.get\_volume()

self.check\_mute=0

def music\_down(self):

if self.check\_mute==1:

pygame.mixer.music.set\_volume(1.0)

pygame.mixer.music.get\_volume()

self.check\_mute=0

else:

self.volume-=0.1

pygame.mixer.music.set\_volume(self.volume)

pygame.mixer.music.get\_volume()

self.check\_mute=0

def next\_song(self):

if self.check\_me==0:

pass

else:

if self.check\_stop==1 or self.check\_pause==1:

pass

else:

self.song\_index+=1

pygame.mixer.music.load(self.list\_of\_songs[self.song\_index])

pygame.mixer.music.set\_volume(self.volume)

pygame.mixer.music.play()

self.update\_song\_label()

self.play\_button.configure(text='II')

def prev\_song(self):

if self.check\_me==0:

pass

else:

if self.check\_stop==1 or self.check\_pause==1:

pass

else:

self.song\_index-=1

pygame.mixer.music.load(self.list\_of\_songs[self.song\_index])

pygame.mixer.music.set\_volume(self.volume)

pygame.mixer.music.play()

self.update\_song\_label()

self.play\_button.configure(text='II')

def play\_pause\_song(self):

if self.check\_stop==1:

if self.check\_me==1:

pygame.mixer.music.load(self.list\_of\_songs[self.song\_index])

pygame.mixer.music.set\_volume(self.volume)

pygame.mixer.music.play()

self.play\_button.configure(text='II')

self.check\_stop=0

self.update\_song\_label()

if self.check\_me==0:

pygame.mixer.music.play()

self.play\_button.configure(text='II')

self.check\_stop=0

self.update\_song\_label()

else:

if self.check\_pause==0:

if self.check\_me==0 and self.first\_click==0:

pass

if self.check\_me==0 and self.first\_click==1:

pygame.mixer.music.pause()

self.play\_button.configure(text='\u25B6')

self.check\_pause=1

if self.check\_me==1 and self.first\_click==0:

pass

if self.check\_me==1 and self.first\_click==1:

pygame.mixer.music.pause()

self.play\_button.configure(text='\u25B6')

self.check\_pause=1

else:

if self.check\_me==0 and self.first\_click==0:

pass

if self.check\_me==0 and self.first\_click==1:

pygame.mixer.music.unpause()

self.play\_button.configure(text='II')

self.check\_pause=0

if self.check\_me==1 and self.first\_click==0:

pass

if self.check\_me==1 and self.first\_click==1:

pygame.mixer.music.unpause()

self.play\_button.configure(text='II')

self.check\_pause=0

def mute\_song(self):

self.check\_mute=1

pygame.mixer.music.set\_volume(0.0)

pygame.mixer.music.get\_volume()

def about\_dev(self):

msgb.showinfo("Adout Avatar","Version :\t\t1.0\nDeveloped By :\tKamal Preet Singh\nPowered By :\tIron Boy\n\nCopyright \u00A9 2018 Avatar Pvt. Ltd.")

def help\_me(self):

os.chdir(self.my\_loc)

self.sub\_window=Toplevel(self.my\_music\_wm)

self.sub\_window.title('Help Docs')

self.sub\_window.wm\_iconbitmap('Avatar Icon.ico')

self.sub\_window.geometry('600x400')

self.sub\_window.resizable(0,0)

self.sub\_window.lift(self.my\_music\_wm)

self.sub\_window.state('normal')

self.temp\_logo\_pic=PhotoImage(file='Avatar Logo.png')

self.logo\_label=Label(self.sub\_window,image=self.temp\_logo\_pic).pack()

self.avatar\_name=Label(self.sub\_window,text='Avatar',font=('Colonna MT',70,'bold'),relief=FLAT,bd=0,fg='blue').pack(pady=20)

self.step\_note=Label(self.sub\_window,text="1.\tSelect music file or folder from 'Media' Menu.\n2.\tThen, Click on 'Play' button to play the song.\n3.\tUse different consoles to manipulate the audio streams.\n4.\tClick on 'Exit' from 'Media' menu to quit the player.").pack(fill=X,side=LEFT)

if self.check\_me==0:

os.chdir(self.single\_song\_loc)

else:

os.chdir(self.folder\_name)

def browse\_file(self):

self.first\_click=1

self.check\_me=0

single\_song\_name=''

self.file\_name=askopenfilename(filetypes=(('MP3 Files',"\*.mp3"),("OGG Files","\*.ogg"),("M4A Files","\*.m4a"),('All files','\*.\*')))

self.single\_song\_loc=os.getcwd()

self.temp\_file=self.file\_name[::-1]

for i in self.temp\_file:

if i=="/":

break

else:

self.single\_song\_name+=i

self.single\_song\_name=self.single\_song\_name[::-1]

pygame.mixer.music.load(self.file\_name)

pygame.mixer.music.set\_volume(self.volume)

pygame.mixer.music.play()

self.update\_song\_label()

self.play\_button.configure(text='II')

def update\_song\_label(self):

if self.check\_me==0:

self.song\_label\_value.set(self.single\_song\_name)

return

if self.check\_me==1:

self.song\_label\_value.set(self.list\_of\_songs[self.song\_index])

return

def browse\_folder(self):

self.check\_me=1

self.first\_click=1

self.list\_of\_songs=[]

self.folder\_name=askdirectory()

os.chdir(self.folder\_name)

for files in os.listdir(self.folder\_name):

if files.endswith(".mp3"):

self.list\_of\_songs.append(files)

pygame.mixer.music.load(self.list\_of\_songs[0])

pygame.mixer.music.set\_volume(self.volume)

pygame.mixer.music.play()

self.update\_song\_label()

self.play\_button.configure(text='II')

def exit\_me(self):

self.exit\_confirm=msgb.askyesno("Avatar Music Player","Are you sure to exit.")

if self.exit\_confirm==True:

self.my\_music\_wm.destroy()

else:

pass

def play\_song(self):

if (self.check\_me==0 and self.check\_stop==0) or self.check\_stop==1:

pygame.mixer.music.play()

self.update\_song\_label()

self.play\_button.configure(text='II')

self.check\_stop=0

if (self.check\_me==1 and self.check\_stop==0) or self.check\_stop==1:

pygame.mixer.music.load(self.list\_of\_songs[self.song\_index])

pygame.mixer.music.set\_volume(self.volume)

pygame.mixer.music.play()

self.update\_song\_label()

self.play\_button.configure(text='II')

self.check\_stop=0

def stop\_song(self):

if self.check\_me==0:

pygame.mixer.music.stop()

self.song\_label\_value.set("Click on play button to listen the music.")

self.play\_button.configure(text='\u25B6')

self.check\_stop=1

if self.check\_me==1:

pygame.mixer.music.stop()

self.song\_label\_value.set("Click on play button to listen the music.")

self.play\_button.configure(text='\u25B6')

self.check\_stop=1

def main():

my\_music=Tk()

pygame.mixer.init()

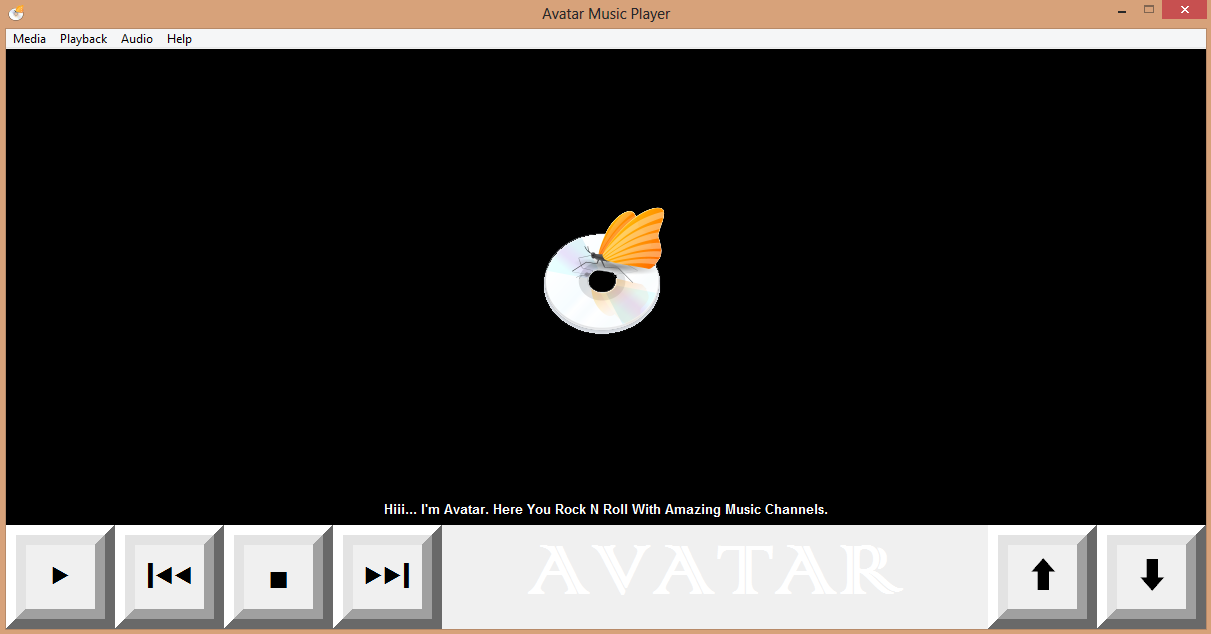
Music\_Player=Avatar\_Music\_Player(my\_music)

my\_music.mainloop()

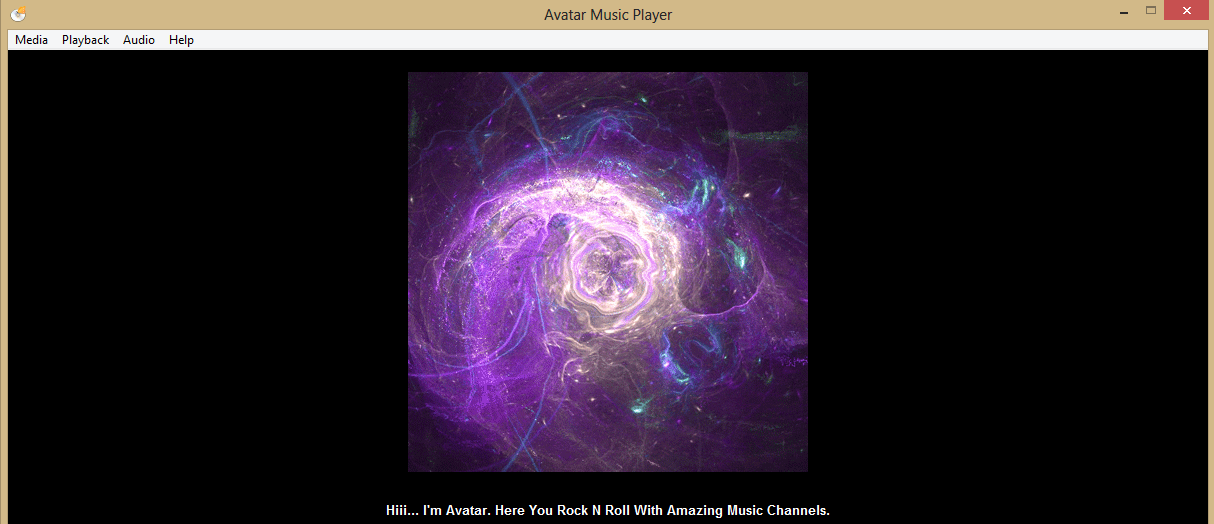
if \_\_name\_\_=="\_\_main\_\_":main()

**ScreenShots Of Avatar Music Player**

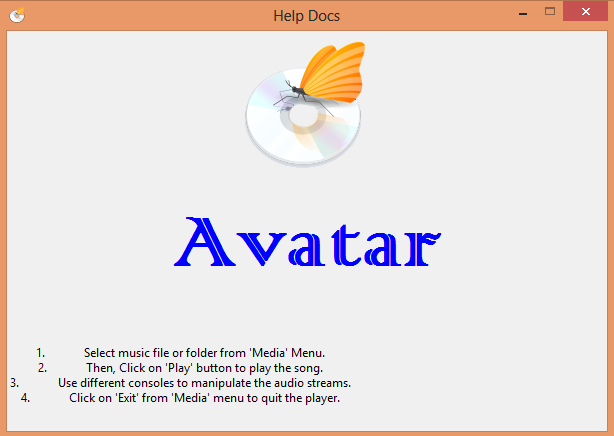
**1. Main Window:**



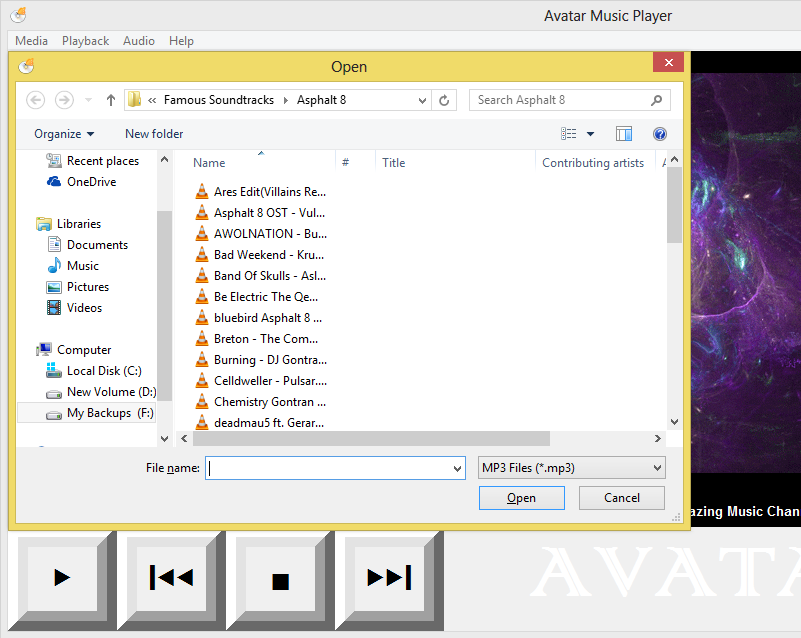
**2. Visualizations:**



**3. Help Docs:**



**4. Open File Dialog Box:**



**5. Console Panel:**



**Hardware And Software Requirements**

**Hardware Requirements:**

1. Intel i3/i5/i7 processor

2. 2 GB RAM

3. 80 GB+ Hard Disk Space

4. Optical Drive

5. Keyboard And Mouse

**Software Requirements:**

1. Windows 7/8.1/10

2. Microsoft Framework

3. Pycharm IDE

**Bibliography**

**1. Books:**

(i) Programming With Python By **R. Nageswara Rao**

(ii) Python For Everybody By **Charles Severance**

**2. Websites:**

(i) <https://www.tutorialspoint.com/python/index.htm>

(ii) <http://effbot.org/tkinterbook/>

(iii) <https://www.youtube.com/user/sentdex/featured>