SRI SATHYA SAI VIDYA VIHAR



AISSCE

2020-21

A Project Work in the subject

Computer Science for the

partial fulfillment of the

syllabus as prescribed by

CBSE for the session 2020-21

The Hangman GameC:\sanskar\computer science\hangman project\hangmanicon3.png

Submitted to: Submitted by:

Ms. Jaspal Vishwakarma Sanskar Sharma

CONTENTS

* **Certificate**
* **Acknowledgement**
* **Certificate of Originality**
* **Aim of the Project**
* **Introduction to Project**
* **Introduction to Modules**
* **Data Dictionary**
* **Flowchart**
* **Source code**
* **Sample outputs**
* **Future Enhancements**
* **Bibliography**

CERITIFICATE

This is to certify that **Sanskar Sharma** of class **XII-B** has completed his Project work as per the syllabus and has submitted a satisfactory account of it as a part of fulfillment towards the practical course for All India Senior Secondary Certificate Examination, 2020-21.

Roll no. Date:\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Internal Examiner) (External Examiner)

**Ms. Jaspal Vishwakarma**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(**Principal**)

**Ms. Punita Nehru School Seal**

ACKNOWLEDGEMENT

**I would like to express my special thanks of gratitude to my teacher Ms. Jaspal Vishwakarma as well as our Principal ma’am Ms. Punita Nehru who gave me the golden opportunity to do this wonderful project, which also helped me in doing a lot of research and I came to know about so many new and fascinating things. Therefore I am really thankful to them.** **Secondly i would also like to thank my parents and friends who helped me in finalizing this project within the limited time frame.**

CERTIFICATE OF ORIGINALITY

**This is to certify that is an original work of the student and is being submitted as a part of fulfillment towards the practical course of All India Senior Secondary Certificate Examination 2020-21.**

**This report has not been submitted earlier either to the Institute or to any other institution for the fulfillment of the requirement of a course of study.**

**Jaspal Vishwakarma Sanskar Sharma**

**Project Guide Student**

AIM OF THE PROJECT

Aim of the project is to provide an electronic version that could be used to play the beautiful game without using papers, which would help us to save our resources such as trees used to produce paper and our precious time which we spend to write the names of different topics and words. The game will definitely enhance the thinking skills of players as the game is all about hints and guesses. Overall it’s a complete fun zone which is not addictive and is related to English and words that helps us to upgrade our responses.

INTRODUCTION TO PROJECT

The project is to design the famous game Hangman on computer using Python and SQL. In this project we ask the player to guess a four letter (level-1) or a five letter (level-2) word depending on the level, if he answers incorrectly then after two chances a hint is displayed that gives the name of topic of word and if he answers incorrectly his second last chance then again a hint is displayed that tells the player something special about the given word. After each wrong guess parts of hangman are formed simultaneously and a complete image of it is displayed if the player loses the game. The words and their two hints are stored in SQL which are linked with Python and randomly executed .The game has option to enable user to continue his game even after closing the playing window as the details can be stored in python according to the player’s wish.

INTRODUCTION TO MODULES

* TKINTER MODULE: **Tkinter** is a Python binding to the Tk GUI toolkit. Tk is the original GUI library for the Tcl language. **Tkinter** is implemented as a Python wrapper around a complete Tcl interpreter embedded in the Python interpreter. It helps us to present our code in a user friendly way.
* RANDOM: This module provides functions for generating pseudo-random numbers or variable.
* MYSQL.CONNECTOR: **MySQL Connector**/**Python** allows you to compress the data stream between **Python** and **MySQL** database server using protocol compression. It supports connections using TCP/IP socket and secure TCP/IP connection using SSL. **MySQLConnector**/**Python** is an API implemented using pure **Python**.
* COLLECTIONS: **Collections** in Python are containers that are used to store **collections** of data, for example, list, dict, set, tuple etc. These are built-in **collections**. Several **modules** have been developed that provide additional data structures to store **collections** of data.
* DATETIME: The **datetime module** supplies classes for manipulating dates and times. While date and time arithmetic is supported, the focus of the implementation is on efficient attribute extraction for output formatting and manipulation.
* PYGAME: **Pygame** is a cross-platform set of **Python modules** designed for writing video games. It includes computer graphics and sound libraries designed to be used with the **Python** programming language.

DATA DICTIONARY

---------------------------CLASS playgame ----------------------------

|  |  |
| --- | --- |
| VARIABLE | DESCRIPTION |
| globnum | This number is level of the game. |
| given\_id | The id of the player if new is created. |
| playerid | The id of the player if user resume next round. |
| score | Score of the game. |

|  |  |
| --- | --- |
| MEMBER FUNCTIONS | DESCRIPTION |
| main\_working | It works for the guessed letter and button function. |
| work | It is main function for the working of the game. |
| bestscore\_function | It stores the best score of the user in the MYSQL when the best score broken. |
| button\_function | It helps the entry button. |
| button\_playagain | It works to set the level. |
| play | This gets the words and hints from the database. |
| level | It helps set the level of the game accordingly. |

-------------------------CLASS player\_entry --------------------------

|  |  |
| --- | --- |
| VARIABLE | DESCRIPTION |
| New | This tells us if new player is created(True/False variable). |
| Resume | This tells us if the player has resumed(True/False variable). |
| Save | This tells us if the player has continued a saved game(True/False variable). |

|  |  |
| --- | --- |
| MEMBER FUNCTIONS | DESCRIPTION |
| create\_player | To enter the created player info into MYSQL. |
| resume\_player | To get the resumed player info to check player is there or not. |
| saved\_game | To check if there is a saved game. |
| delete\_savedgame | Deleted the saved game when the user starts the saved game. |
| save\_play | To continue the saved game if any. |

-----------------------CLASS player\_sql\_entry ------------------------

|  |  |
| --- | --- |
| VARIABLE | DESCRIPTION |
| playerid\_checkvariable | Given player id to check it exists in MYSQL records. |
| playername\_checkvariable | Given player name to check it exists in MYSQL records. |
|  |  |

|  |  |
| --- | --- |
| MEMBER FUNCTION | DESCRIPTION |
| create\_new\_player(s) | It is the window function where new players are created(details are given here). |
| resume\_old\_player(s) | It is the window function where played can play the next game(details are given here). |
| saved\_play(s) | It is the window function where user can play the saved game, (details are given here). |

----------------Database-------------------

**player\_table**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data type** | **Brief** |
| **Player ID** | Int | Primary key –used to link with score table |
| **Player Name** | Varchar | Player name user input |
| **Player Age** | Int | Player age |
| **Player Gender** | Varchar | Player gender |

**hint\_table**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data type** | **Brief** |
| **Topic** | Varchar | Topic of the word. |
| **Word** | Varchar | Word |
| **Hint1** | Varchar | Hint1 of the word. |
| **Hint2** | Varchar | Hint2 of the word. |
| **Levelnum** | Int | Level number. |

**record\_table**

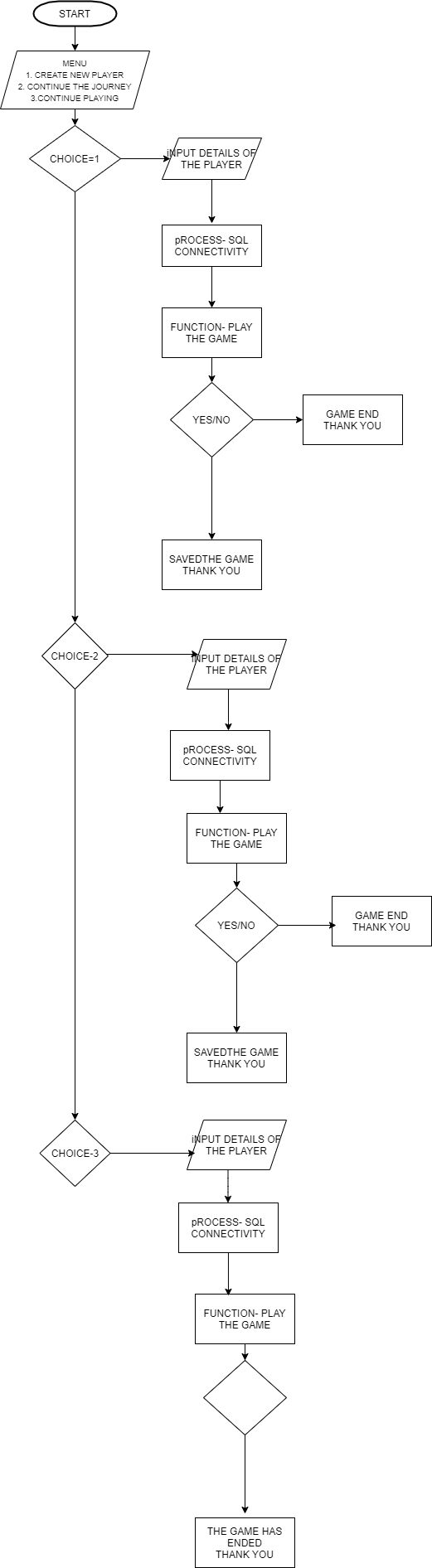
|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data type** | **Brief** |
| **Player ID** | Int | Foreign key –helps to link with the master table. |
| **Best score** | Int | It only updates if the past score for that player is beaten. |
| **Date of the best score** | Date time | Shows the time of their achievement . |

**saved\_game**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data type** | **Brief** |
| **Player ID** | Int | Helps to connect to both the above table. |
| **Playername** | Int | Player name. |
| **Saved\_word** | Int | Word on which user was |
| **Levelplayed** | Int |  |

**score\_game**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data type** | **Brief** |
| **Player ID** | Int | Helps to connect to both the above table. |
| **Number of level passed** | Int | Shows how many levels passed. |
| **Total score** | Int | Total round score. |



FLOWCHART

SOURCE CODE

"""Please save the changes that you have done to Google drive also.thank you ."""

#--------Legal\_Import\_statement\_section-------

import tkinter as tk

from tkinter import \*

from tkinter import PhotoImage

from tkinter import messagebox

from tkinter import ttk

import random

from collections import Counter

import mysql.connector

import datetime

import pygame

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*SQL\_DETAILS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

"""|^\_^|change the details according to your system|^\_^|!"""

global sql\_host,sql\_pass,sql\_database,sql\_user

sql\_host="localhost"

sql\_pass="manmandir"

sql\_database="hangman(player details)"

sql\_user="root"

#-------------Create\_main\_window----------------

window=tk.Tk()

window.title("Hangman|welcome|")

icon=tk.PhotoImage(file ="hangmanicon3.png")

window.iconphoto(False,icon)

window.geometry("800x550")

window.configure(bg="navy")

pygame.mixer.init()

#-----------------Main\_program---------------------

class playgame:

def \_\_init\_\_(s):

s.score=0

s.correct = 0

def main\_working(s):

s.letterGuessed = ''# list for storing the letters guessed by the player

s.chances =8

s.flag = 0

s.wrong\_count=0

playgame. button\_function(s)

def work(s):

s.flag = 0

s.hint\_used=0

s.guess\_string=s.guess.get()

if (s.chances != 0) and s.flag == 0: #flag is updated when the word is correctly guessed

# Validation of the guess

if s.guess\_string.isalpha()==False:

messagebox.showwarning("Message",'Enter only a LETTER')

elif len(s.guess\_string) > 1:

messagebox.showwarning("Message",'Enter only a SINGLE letter')

elif s.guess\_string in s.guessedletter:

messagebox.showinfo("Message",'You have already guessed that letter')

global label\_letterguessed

s.guessedletter=s.guessedletter+s.guess\_string+","

label\_letterguessed.config(text="Letters Used:"+s.guessedletter)

# If letter is guessed correctly

if s.guess\_string in s.word:

s.k = s.word.count(s.guess\_string) #k stores the number of times the guessed letter occurs in the word

for \_ in range(s.k):

s.letterGuessed += s.guess\_string# The guess letter is added as many times as it occurs

# Print the word

s.x=Counter(s.letterGuessed)

s.y=Counter(s.word)

wrong=False#to check if wrong guess

newword=""

for char in s.word:

if char in s.letterGuessed and (s.x != s.y):#going rt on track

if char==s.guess\_string:

global label\_\_

newword=""

for i in s.word:#required for the word to update on the screen

if char ==i:

newword=newword+char# For printing the empty spaces for letters of the word

elif i in s.letterGuessed and char!=i :

newword=newword+i

else:

newword=newword+'\_ '

label\_\_.config(text=newword)

s.correct += 1

# If user has guessed all the letters

elif s.x == s.y : # Correct Guessed

label\_\_.config(text=s.word)

global globnum

globnum+=1

s.buton\_entry.config(state=DISABLED)

label\_word=tk.Label(s.mainwindow,text="Congratulation, you saved the hangman|^o^|"+"The word was: "+s.word,fg="blue",bg="orange",font=("Ink Free",16,"bold")).pack()

s.flag = 1

global sql\_host,sql\_pass,sql\_database,sql\_user

db\_conn=mysql.connector.connect(

host=sql\_host,

user=sql\_user,

passwd=sql\_pass,

database= sql\_database)

s.score=s.correct -s.hint\_used

global playerid,given\_id

global resume,new

c=db\_conn.cursor()

q="INSERT INTO SCORE\_TABLE VALUES(%s,%s,%s)"

if resume== True:

data=(given\_id,globnum,s.score)

elif new==True:

data=(playerid,globnum,s.score)

c.execute(q,data)

db\_conn.commit()

c.close()

db\_conn.close()

player\_sql\_entry.delete\_savedgame(s)

playgame.bestscore\_function(s)

playgame.button\_playagain(s)

break# To break out of the for loop

return

else :

wrong=True

if s.guess\_string not in s.word and wrong== True:

s.chances = s.chances -1

s.wrong\_count+=1

if s.wrong\_count==1:

s.canvas\_main.create\_line(100,300,300,300,fill="orange",width=3)

elif s.wrong\_count==2 :

s.canvas\_main.create\_line(100,300,300,300,fill="orange",width=3)

s.canvas\_main.create\_line(100,100,100,300,fill="orange",width=3)

global hint1,label\_hint

label\_hint=tk.Label(s.mainwindow,text=hint1,fg="blue",bg="orange",font=("Ink Free",15,"bold"))

label\_hint.pack()

s.hint\_used=1

elif s.wrong\_count==3 :

s.canvas\_main.create\_line(100,300,300,300,fill="orange",width=3)

s.canvas\_main.create\_line(100,100,100,300,fill="orange",width=3)

s.canvas\_main.create\_line(100,100,200,100,fill="orange",width=3)

elif s.wrong\_count==4 :

s.canvas\_main.create\_line(100,300,300,300,fill="orange",width=3)

s.canvas\_main.create\_line(100,100,100,300,fill="orange",width=3)

s.canvas\_main.create\_line(100,100,200,100,fill="orange",width=3)

s.canvas\_main.create\_line(200,100,200,175,fill="orange",width=3)

elif s.wrong\_count==5 :

s.canvas\_main.create\_line(100,300,300,300,fill="orange",width=3)

s.canvas\_main.create\_line(100,100,100,300,fill="orange",width=3)

s.canvas\_main.create\_line(100,100,200,100,fill="orange",width=3)

s.canvas\_main.create\_line(200,100,200,175,fill="orange",width=3)

s.canvas\_main.create\_oval(175,175,225,225,width=3,fill="orange")#head

elif s.wrong\_count==6 :

s.canvas\_main.create\_line(100,300,300,300,fill="orange",width=3)

s.canvas\_main.create\_line(100,100,100,300,fill="orange",width=3)

s.canvas\_main.create\_line(100,100,200,100,fill="orange",width=3)

s.canvas\_main.create\_line(200,100,200,175,fill="orange",width=3)

s.canvas\_main.create\_oval(175,175,225,225,width=3,fill="orange")#head

s.canvas\_main.create\_line(200,225,200,275,fill="orange",width=3)#body

s.canvas\_main.create\_line(200,225,150,175,fill="orange",width=3)#hand1

global hint2

label\_hint.config(text=hint2)

s.hint\_used=2

elif s.wrong\_count==7 :

s.canvas\_main.create\_line(100,300,300,300,fill="orange",width=3)

s.canvas\_main.create\_line(100,100,100,300,fill="orange",width=3)

s.canvas\_main.create\_line(100,100,200,100,fill="orange",width=3)

s.canvas\_main.create\_line(200,100,200,175,fill="orange",width=3)

s.canvas\_main.create\_oval(175,175,225,225,width=3,fill="orange")#head

s.canvas\_main.create\_line(200,225,200,275,fill="orange",width=3)#body

s.canvas\_main.create\_line(200,225,150,175,fill="orange",width=3)#hand

s.canvas\_main.create\_line(200,225,250,175,fill="orange",width=3)

elif s.wrong\_count==8 :

s.canvas\_main.create\_line(100,300,300,300,fill="orange",width=3)

s.canvas\_main.create\_line(100,100,100,300,fill="orange",width=3)

s.canvas\_main.create\_line(100,100,200,100,fill="orange",width=3)

s.canvas\_main.create\_line(200,100,200,175,fill="orange",width=3)

s.canvas\_main.create\_oval(175,175,225,225,width=3,fill="orange")

s.canvas\_main.create\_line(200,225,200,275,fill="orange",width=3)

s.canvas\_main.create\_line(200,225,150,175,fill="orange",width=3)#hand

s.canvas\_main.create\_line(200,225,250,175,fill="orange",width=3)#hand

s.canvas\_main.create\_line(200,275,250,290,fill="orange",width=3)#leg

s.canvas\_main.create\_line(200,275,150,290,fill="orange",width=3)#leg

if s.chances <= 0 and (s.x != s.y):

globnum+=1

label\_lost=tk.Label(s.mainwindow,text="You Lost.Better Luck Next Time!"+s.word+" was the word.",fg="blue",bg="orange",font=("Freestyle Script",16,"bold")).pack()

s.buton\_entry.config(state=DISABLED)

playgame.button\_playagain(s)

s.canvas\_main.create\_line(100,300,300,300,fill="orange",width=3)

s.canvas\_main.create\_line(100,100,100,300,fill="orange",width=3)

s.canvas\_main.create\_line(100,100,200,100,fill="orange",width=3)

s.canvas\_main.create\_line(200,100,200,175,fill="orange",width=3)

s.canvas\_main.create\_oval(175,175,225,225,fill="orange",width=2)#head

s.canvas\_main.create\_line(200,225,200,275,fill="orange",width=3)#body

s.canvas\_main.create\_line(200,225,150,175,fill="orange",width=3)#hand

s.canvas\_main.create\_line(200,225,250,175,fill="orange",width=3)#hand

s.canvas\_main.create\_line(200,275,250,290,fill="orange",width=3)#leg

s.canvas\_main.create\_line(200,275,150,290,fill="orange",width=3)#leg

player\_sql\_entry.delete\_savedgame(s)

return

def bestscore\_function(s):#the best score function to store the former.

global sql\_host,sql\_pass,sql\_database,sql\_user

db\_conn=mysql.connector.connect(

host=sql\_host,

user=sql\_user,

passwd=sql\_pass,

database=sql\_database)

global resume,new,playerid,given\_id

c=db\_conn.cursor(buffered=True)

best\_datetime=datetime.datetime.now()

best\_date=datetime.date(best\_datetime.year,best\_datetime.month,best\_datetime.day)

if resume==True:

data=(given\_id,)

q="SELECT BESTSCORE FROM RECORD\_TABLE WHERE PLAYERID=(%s)"

c.execute(q,data)

bestscore=c.fetchone( )

if s.score>bestscore[0]:

q="UPDATE RECORD\_TABLE SET BESTSCORE=%s,SCOREDATE=%s WHERE PLAYERID=%s"

data=(s.score,best\_date,given\_id)

elif new==True:

data=(playerid,)

q="SELECT BESTSCORE FROM RECORD\_TABLE WHERE PLAYERID=(%s)"

c.execute(q,data)

bestscore=c.fetchone( )

if s.score>bestscore[0]:

q="UPDATE RECORD\_TABLE SET BESTSCORE=%s,SCOREDATE=%s WHERE PLAYERID=%s"

data=(s.score,best\_date,playerid)

c.execute(q,data)

db\_conn.commit()

c.close()

db\_conn.close()

def button\_function(s):

s.guess\_variable=tk.StringVar()

s.guess= ttk.Entry(s.mainwindow,textvariable=s.guess\_variable,width=10,justify="center")

s.guess.pack()

s.buton\_entry=tk.Button(s.mainwindow,text="Enter",command= lambda:playgame.work(s),fg="navy",bg="orange",font=("small fonts",16,"bold"),activeforeground="orange",activebackground="navy")

s.buton\_entry.pack()

def button\_playagain(s):

if globnum==1:

buton\_again=tk.Button(s.mainwindow,text="Next Round!",command= lambda:playgame.level(s,1),fg="blue",bg="orange",font=("Freestyle Script",16,"bold"),activeforeground="blue",activebackground="blue").pack()

elif globnum==2 or globnum==3:

buton\_again=tk.Button(s.mainwindow,text="Next Round!",command= lambda:playgame.level(s,2),fg="blue",bg="orange",font=("Freestyle Script",16,"bold"),activeforeground="blue",activebackground="blue").pack()

else:

s.mainwindow.destroy()

def play(s,num\_of\_letters):

global mainwindow

s.mainwindow=tk.Tk()

s.mainwindow.title("Hangman|Play area|")

s.mainwindow.geometry("700x700")

s.mainwindow.configure(bg="orange")

label\_hangman=tk.Label(s.mainwindow,text="HANGMAN",font=("small fonts",50),fg="navy",bg="orange").pack()

global sql\_host,sql\_pass,sql\_database,sql\_user

db\_conn=mysql.connector.connect(

host=sql\_host,

user=sql\_user,

passwd=sql\_pass,

database=sql\_database)

c=db\_conn.cursor()

q="SELECT \* FROM HINT\_TABLE WHERE LEVELNUM=(%s)"

data=(num\_of\_letters,)

c.execute(q,data)

rows=c.fetchmany(60)

n1=random.randint(1,len(rows))#can leave the empty it reads one record (doubt!)

global saved

global globalword

if saved==True:

s.word=globalword#varialbe to have the saved word

else:

try:

s.word=rows[n1][1]

except:

messagebox.showwarning("Message",'|#\_@|Sorry our hangman ran off . Please try again.|@\_#|')

#The main canvas for mr.hangman

s.canvas\_main=tk.Canvas(s.mainwindow,width=350,height=350,background="navy")

s.canvas\_main.pack()#s.canvas\_main.create\_line(x1,y1,x2,y2,fill="colour")

global hint1,hint2

hint1=rows[n1][2]#basic hint to tell about the word type

hint2=rows[n1][3]#a more related hint

global label\_\_,label\_letterguessed

blankword=""

s.guessedletter=""

for i in s.word:

blankword=blankword+'\_ ' # For printing the empty spaces for letters of the word

label\_\_=tk.Label(s.mainwindow,text=blankword,fg="blue",bg="orange",font=("Freestyle Script",25))

label\_\_.pack()

label\_letterguessed=tk.Label(s.mainwindow,text="Guessed Letters: ",fg="blue",bg="orange",font=("Freestyle Script",20))

label\_letterguessed.pack()

button\_save= Button(s.mainwindow, text = "Save!",command=lambda:player\_sql\_entry.save\_play(s,num\_of\_letters,s.word),fg="navy",bg="orange",font=("small fonts",16,"bold"),activeforeground="orange",activebackground="navy")

button\_save.place(relx = 1, x =-2, y = 2, anchor = NE)

playgame.main\_working(s)

s.mainwindow.mainloop()

def level(s,level\_num):

if level\_num==1:

playgame.play(s,1)

elif level\_num==2:

playgame.play(s,2)

#-------------Connecting\_to\_MYSQL---------------"""|^\_^|main sql conections ahead|^\_^|!"""

class player\_sql\_entry :

def \_\_init\_\_(s):

s.rid=100

def create\_player(s):

global playername\_variable, playerage\_variable,playergender\_variable,button\_submit,new

new=True

button\_submit.config(state=DISABLED)

name=playername\_variable.get()

age=playerage\_variable.get()

gender=playergender\_variable.get()

p=playgame()

global sql\_host,sql\_pass,sql\_database,sql\_user

db\_conn=mysql.connector.connect(

host=sql\_host,

user=sql\_user,

passwd=sql\_pass,

database= sql\_database)

s.num= random.randint(10,900)

global playerid

playerid=100+s.num+int(age)

c=db\_conn.cursor()

q="INSERT INTO PLAYER\_TABLE VALUES(%s,%s,%s,%s)"

data=(playerid,name,age,gender)

c.execute(q,data)

db\_conn.commit()

c.close()

db\_conn.close()

db\_conn=mysql.connector.connect(

host=sql\_host,

user=sql\_user,

passwd=sql\_pass,

database= sql\_database)

c=db\_conn.cursor()

best\_datetime=datetime.datetime.now()

best\_date=datetime.date(best\_datetime.year,best\_datetime.month,best\_datetime.day)

q="INSERT INTO RECORD\_TABLE VALUES(%s,%s,%s)"

data=(playerid,0,best\_date)

c.execute(q,data)

db\_conn.commit()

c.close()

db\_conn.close()

s.create\_new\_window2=tk.Tk()

s.create\_new\_window2.title("Hangman|Create Player|")

s.create\_new\_window2.geometry("300x300")

s.create\_new\_window2.configure(bg="orange")

label\_id=tk.Label(s.create\_new\_window2,text="YourID:"+str(playerid),font=("small fonts",17),fg="navy",bg="orange").pack()

label\_rem=tk.Label(s.create\_new\_window2,text="Please remember your ID!",font=("small fonts",12),fg="navy",bg="orange").pack()

button\_play=Button(s.create\_new\_window2,text="Play",command=lambda: p.level(1),bg="navy",fg="orange")

button\_play.pack()

s.create\_new\_window2.mainloop()

def resume\_player(s):

p=playgame()

global playerid\_checkvariable,playername\_checkvariable

global given\_id,resume

resume=True

given\_id=playerid\_checkvariable.get()

given\_name=playername\_checkvariable.get()

global sql\_host,sql\_pass,sql\_database,sql\_user

db\_conn=mysql.connector.connect(

host=sql\_host,

user=sql\_user,

passwd=sql\_pass,

database= sql\_database)

c1=db\_conn.cursor()

q1="SELECT PLAYERNAME FROM PLAYER\_TABLE WHERE PLAYERID = %s ;"

data=(given\_id,)

c1.execute(q1,data)

row=c1.fetchone()

db\_conn.commit()

c1.close()

db\_conn.close()

if row!=None:

if row[0]!= None and row[0] == given\_name:

global button\_continue

button\_continue.config(state=DISABLED)

s.create\_new\_window2=tk.Tk()

s.create\_new\_window2.title("Hangman|Saved Player|")

s.create\_new\_window2.geometry("200x150")

s.create\_new\_window2.configure(bg="orange")

label\_hangman=tk.Label(s.create\_new\_window2,text="HANGMAN",font=("small fonts",29),fg="navy",bg="orange").pack()

label\_welcome=tk.Label(s.create\_new\_window2,text="Welcome "+given\_name+" !",font=("small fonts",17),fg="navy",bg="orange").pack()

button\_play=Button(s.create\_new\_window2,text="Play",command=lambda: p.level(1),fg="navy",bg="orange").pack()

s.create\_new\_window2.mainloop()

else:

s.create\_new\_window2=tk.Tk()

s.create\_new\_window2.title("Hangman|Create Player|")

s.create\_new\_window2.geometry("400x150")

s.create\_new\_window2.configure(bg="orange")

label\_hangman=tk.Label(s.create\_new\_window2,text="HANGMAN",font=("small fonts",50),fg="navy",bg="orange").pack()

label\_welcome=tk.Label(s.create\_new\_window2,text="Sorry wrong username !",font=("small fonts",17),fg="navy",bg="orange").pack()

s.create\_new\_window2.mainloop()

else:

messagebox.showwarning("Message",'|#\_@|You have enter a wrong ID or username|@\_#|')

def saved\_game(s):

p=playgame()

global playerid\_checkvariable,playername\_checkvariable,given\_id,resume

resume=True

given\_id=playerid\_checkvariable.get()

given\_name=playername\_checkvariable.get()

global sql\_host,sql\_pass,sql\_database,sql\_user

db\_conn=mysql.connector.connect(

host=sql\_host,

user=sql\_user,

passwd=sql\_pass,

database= sql\_database)

c1=db\_conn.cursor()

q1="SELECT PLAYERNAME,SAVED\_WORD,LEVELPLAYED FROM SAVED\_GAME WHERE PLAYERID = %s ;"

data=(given\_id,)

c1.execute(q1,data)

row=c1.fetchone()

db\_conn.commit()

c1.close()

db\_conn.close()

if row!=None:

global globalword,saved

globalword=row[1]

saved=True

if row!= None and row[0] == given\_name:

global button\_continue

button\_continue.config(state=DISABLED)

s.create\_new\_window2=tk.Tk()

s.create\_new\_window2.title("Hangman|Create Player|")

s.create\_new\_window2.geometry("200x150")

s.create\_new\_window2.configure(bg="orange")

label\_hangman=tk.Label(s.create\_new\_window2,text="HANGMAN",font=("small fonts",29),fg="navy",bg="orange").pack()

label\_welcome=tk.Label(s.create\_new\_window2,text="Welcome "+given\_name+" !",font=("small fonts",17),fg="navy",bg="orange").pack()

button\_play=Button(s.create\_new\_window2,text="Play",command=lambda: p.level(1),fg="navy",bg="orange").pack()

s.create\_new\_window2.mainloop()

else:

s.create\_new\_window2=tk.Tk()

s.create\_new\_window2.title("Hangman|Create Player|")

s.create\_new\_window2.geometry("400x150")

s.create\_new\_window2.configure(bg="orange")

label\_hangman=tk.Label(s.create\_new\_window2,text="HANGMAN",font=("small fonts",50),fg="navy",bg="orange").pack()

label\_welcome=tk.Label(s.create\_new\_window2,text="Sorry wrong username !",font=("small fonts",17),fg="navy",bg="orange").pack()

s.create\_new\_window2.mainloop()

else:

messagebox.showwarning("Message",'|#\_@|You have enter a wrong ID or username|OR|You donot have a saved game|@\_#|')

def delete\_savedgame(s):

global saved

global given\_id,resume,new,playerid

if saved==True:

db\_conn=mysql.connector.connect(

host=sql\_host,

user=sql\_user,

passwd=sql\_pass,

database= sql\_database)

c=db\_conn.cursor()

q="DELETE SAVED\_GAME WHERE PALYERID=(%s)"

if resume== True:

data=(given\_id,)

elif new==True:

data=(playerid,)

c.execute(q,data)

db\_conn.commit()

c.close()

db\_conn.close()

saved=False

def save\_play(s,level,word):

global sql\_host,sql\_pass,sql\_database,sql\_user

db\_conn=mysql.connector.connect(

host=sql\_host,

user=sql\_user,

passwd=sql\_pass,

database= sql\_database)

c1=db\_conn.cursor()

q1="INSERT INTO SAVED\_GAME VALUES(%s,%s,%s,%s);"

global playerid\_checkvariable,playername\_checkvariable

global resume,new

global globnum,playerid,given\_id

if resume==True:

given\_id=playerid\_checkvariable.get()

given\_name=playername\_checkvariable.get()

data=(given\_id,given\_name,word,level)

elif new==True:

global playerid,playername\_variable

given\_id=playerid

data=(playerid,given\_name,word,level)

c1.execute(q1,data)

db\_conn.commit()

c1.close()

db\_conn.close()

global window,mainwindow,create\_new\_window2

window.destroy()

s.mainwindow.destroy()

exit()

#Creating\_New\_Player|OR|Resume\_Old\_Game

"""|^\_^|Entry code of game.kind of like a login screen for the user|@\_@|!"""

class player\_entry :

def \_\_init\_\_(s):

s.something=0

def create\_new\_player(s):

global b\_createplayer,b\_resume,b\_savedplay

b\_createplayer.config(state=DISABLED)

b\_resume.config(state=DISABLED)

b\_savedplay.config(state=DISABLED)

global new

new=True

s.create\_new\_window=tk.Tk()

s.create\_new\_window.title("Hangman|Create Player|")

s.create\_new\_window.geometry("400x150")

s.create\_new\_window.configure(bg="navy")

global playername\_variable, playerage\_variable,playergender\_variable

playername\_variable=tk.StringVar(s.create\_new\_window)

playerage\_variable=tk.IntVar(s.create\_new\_window)

playergender\_variable=tk.StringVar(s.create\_new\_window)

s.label\_name=tk.Label(s.create\_new\_window,text="Name:",font=("small fonts",17),fg="orange",bg="navy").grid(column=1,row=1)

s.player\_name= tk.Entry(s.create\_new\_window,textvariable=playername\_variable,width=17,justify="center",font=("small fonts",17),bg="orange",fg="navy")

s.player\_name.grid(column=2,row=1)

s.label\_age=tk.Label(s.create\_new\_window,text="Age:",font=("small fonts",17),fg="orange",bg="navy").grid(column=1,row=2)

s.player\_age= tk.Entry(s.create\_new\_window,textvariable=playerage\_variable,width=17,justify="center",font=("small fonts",17),bg="orange",fg="navy")

s.player\_age.grid(column=2,row=2)

s.label\_gender=tk.Label(s.create\_new\_window,text="Gender:",font=("small fonts",17),fg="orange",bg="navy").grid(column=1,row=3)

s.player\_gender= tk.Entry(s.create\_new\_window,textvariable=playergender\_variable,width=17,justify="center",font=("small fonts",17),bg="orange",fg="navy")

s.player\_gender.grid(column=2,row=3)

global button\_submit

button\_submit=Button(s.create\_new\_window,text="Submit",command=lambda: player\_sql\_entry.create\_player(s),bg="navy",fg="orange")

button\_submit.grid(column=2,row=4)

s.create\_new\_window.mainloop()

def resume\_old\_player(s):

global b\_createplayer,b\_resume,b\_savedplay

b\_createplayer.config(state=DISABLED)

b\_resume.config(state=DISABLED)

b\_savedplay.config(state=DISABLED)

global resume

resume=True

p=playgame()

s.create\_new\_window3=tk.Tk()

s.create\_new\_window3.title("Hangman|Resume Player|")

s.create\_new\_window3.geometry("300x150")

s.create\_new\_window3.configure(bg="orange")

global playerid\_checkvariable,playername\_checkvariable

playerid\_checkvariable=tk.IntVar(s.create\_new\_window3)

playername\_checkvariable=tk.StringVar(s.create\_new\_window3)

s.label\_age=tk.Label(s.create\_new\_window3,text="Your ID:",font=("small fonts",17),fg="navy",bg="orange").grid(column=1,row=2)

s.playerid\_check= tk.Entry(s.create\_new\_window3,textvariable=playerid\_checkvariable,font=("small fonts",13),bg="navy",fg="orange",width=17,justify="center")

s.playerid\_check.grid(column=2,row=2)

s.label\_gender=tk.Label(s.create\_new\_window3,text="Your name:",font=("small fonts",17),fg="navy",bg="orange").grid(column=1,row=3)

s.playername\_check= tk.Entry(s.create\_new\_window3,textvariable=playername\_checkvariable,font=("small fonts",15),bg="navy",fg="orange",width=17,justify="center")

s.playername\_check.grid(column=2,row=3)

global button\_continue

button\_continue=Button(s.create\_new\_window3,text="Continue",command=lambda: player\_sql\_entry.resume\_player(s),font=("small fonts",15),bg="orange",fg="navy",activebackground="navy",activeforeground="orange")

button\_continue.grid(column=2,row=4)

s.create\_new\_window3.mainloop()

def saved\_play(s):

global b\_createplayer,b\_resume,b\_savedplay

b\_createplayer.config(state=DISABLED)

b\_resume.config(state=DISABLED)

b\_savedplay.config(state=DISABLED)

global resume

resume=True

p=playgame()

s.create\_new\_window3=tk.Tk()

s.create\_new\_window3.title("Hangman|Continue Saved|")

s.create\_new\_window3.geometry("300x150")

s.create\_new\_window3.configure(bg="orange")

global playerid\_checkvariable,playername\_checkvariable

playerid\_checkvariable=tk.IntVar(s.create\_new\_window3)

playername\_checkvariable=tk.StringVar(s.create\_new\_window3)

s.label\_age=tk.Label(s.create\_new\_window3,text="Your ID:",font=("small fonts",17),fg="navy",bg="orange").grid(column=1,row=2)

s.playerid\_check= tk.Entry(s.create\_new\_window3,textvariable=playerid\_checkvariable,font=("small fonts",13),bg="navy",fg="orange",width=17,justify="center")

s.playerid\_check.grid(column=2,row=2)

s.label\_gender=tk.Label(s.create\_new\_window3,text="Your name:",font=("small fonts",17),fg="navy",bg="orange").grid(column=1,row=3)

s.playername\_check= tk.Entry(s.create\_new\_window3,textvariable=playername\_checkvariable,font=("small fonts",15),bg="navy",fg="orange",width=17,justify="center")

s.playername\_check.grid(column=2,row=3)

global button\_continue

button\_continue=Button(s.create\_new\_window3,text="Continue",command=lambda: player\_sql\_entry.saved\_game(s),font=("small fonts",15),bg="orange",fg="navy",activebackground="navy",activeforeground="orange")

button\_continue.grid(column=2,row=4)

s.create\_new\_window3.mainloop()

#-----------------Front\_Page\_code------------------

logo\_1=tk.PhotoImage(file="hangman1.png")

logo\_label=ttk.Label(window ,image=logo\_1,border=0).pack()

p=player\_entry()

global b\_createplayer,b\_resume,b\_savedplay

b\_createplayer=Button(window,text="Create New Player",command=lambda: p.create\_new\_player(),bg="navy",fg="orange",font=("small fonts",20),activeforeground="navy",activebackground="orange",border=0)

b\_createplayer.pack()

b\_resume=Button(window,text="Countinue The Journey",command=lambda: p.resume\_old\_player(),font=("small fonts",20),bg="navy",fg="orange",activeforeground="navy",activebackground="orange",border=0)

b\_resume.pack()

b\_savedplay=Button(window,text="Countinue Playing",command=lambda: p.saved\_play(),font=("small fonts",20),bg="navy",fg="orange",activeforeground="navy",activebackground="orange",border=0)

b\_savedplay.pack()

global globnum,resume,new,saved#this global var are used to check for old/new player

globnum=0

resume=False

new=False

saved=False

#quotes games

quotes=["Games are the most elevated form of investigation. --Albert Einstein",

"The game itself is bigger than the winning.--Dejan Stojanovic",

"Games lubricate the body and mind.--Benjamin Franklin",

"We do not stop playing because we grow old. \nWe grow old because we stop playing.-- Benjamin Franklin",

"Life is more fun if you play games.--Roald Dahl"]

no\_quote=random.randint(0,4)

pygame.mixer.music.load("Tobu - Hope [NCS Release].mp3")

pygame.mixer.music.play(loops=2)

label\_quote=tk.Label(window,text=quotes[no\_quote],font=("Harrington",20),fg="orange",bg="navy").pack()

label\_thank=tk.Label(window,text="|^\_^|Hope You Have A Great Day|^\_^|.",font=("small fonts",25),fg="orange",bg="navy").pack()

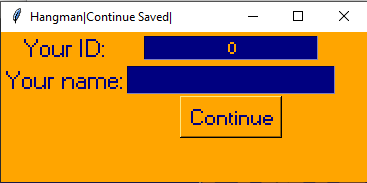
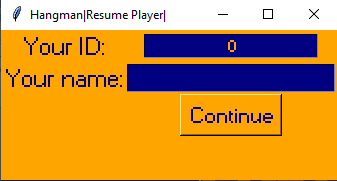
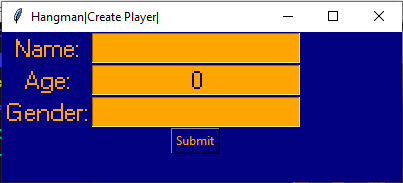
label\_made=tk.Label(window,text="Made by : \n Sanskar Sharma and Devansh Talan",font=("small fonts",25),bg="navy",fg="orange").pack()

#----------------Mainloop\_ahead--------------------

window.mainloop()

#---------------Dead\_End----------------------

SAMPLE OUTPUTS









FUTURE ENHANCEMENTS

Some of the future enhancement are:-

* More number of levels.
* Leaderboard.
* Topic selection by the user.

Bibliography

* Computer science textbook (XII) by Sumita Arora
* [www.flaticon.com](http://www.flaticon.com)
* NCS[No Copyright Sound]
* GeeksforGeeks
* Internet