# LOVELY PROFESSOINAL UNIVERSITY

## Project Report

# Morse Code Converter

**Submitted By:** **Submitted To:**

**Name :** Akash Singh Panwar Mrs. Pooja Rana Mam

**Registration Number :** 11901641

**Roll Number :** 15

**Section :** K19AP

## Project Description

Our Project was to develop a Morse Code Converter, A morse code is a code written in form of "." (Dot) and "- "(Hyphen). These are used as security purposes, to hide the confidential data.

We can convert a data in two forms:

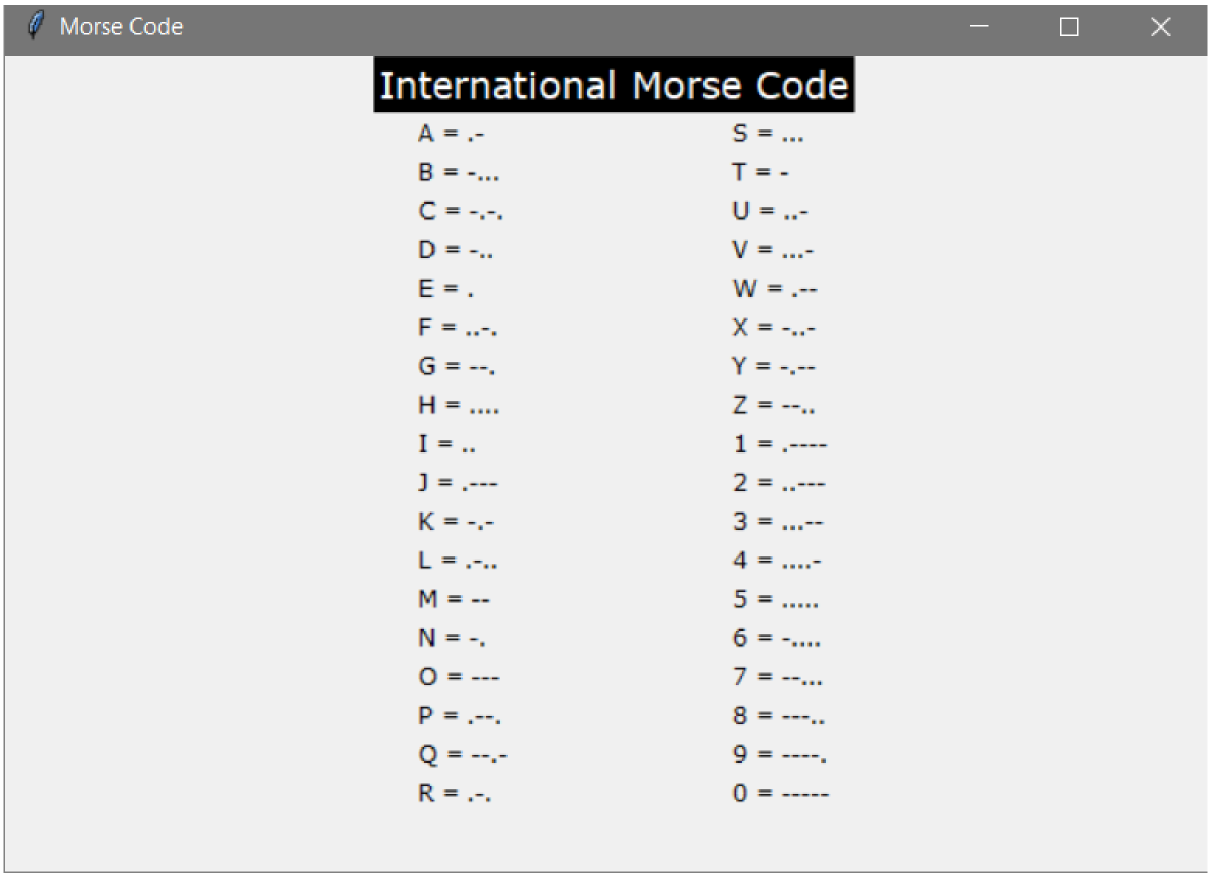
1. Morse code to Text
2. Text to Morse code

## Git Hub Link: https://github.com/RamanSharma100/morse-code-converter-python-project

## My Contribution

1. Morse Code Table page : GUI
2. Guest page : GUI
3. History Page : GUI and Database

## Page 1: Morse Code Table

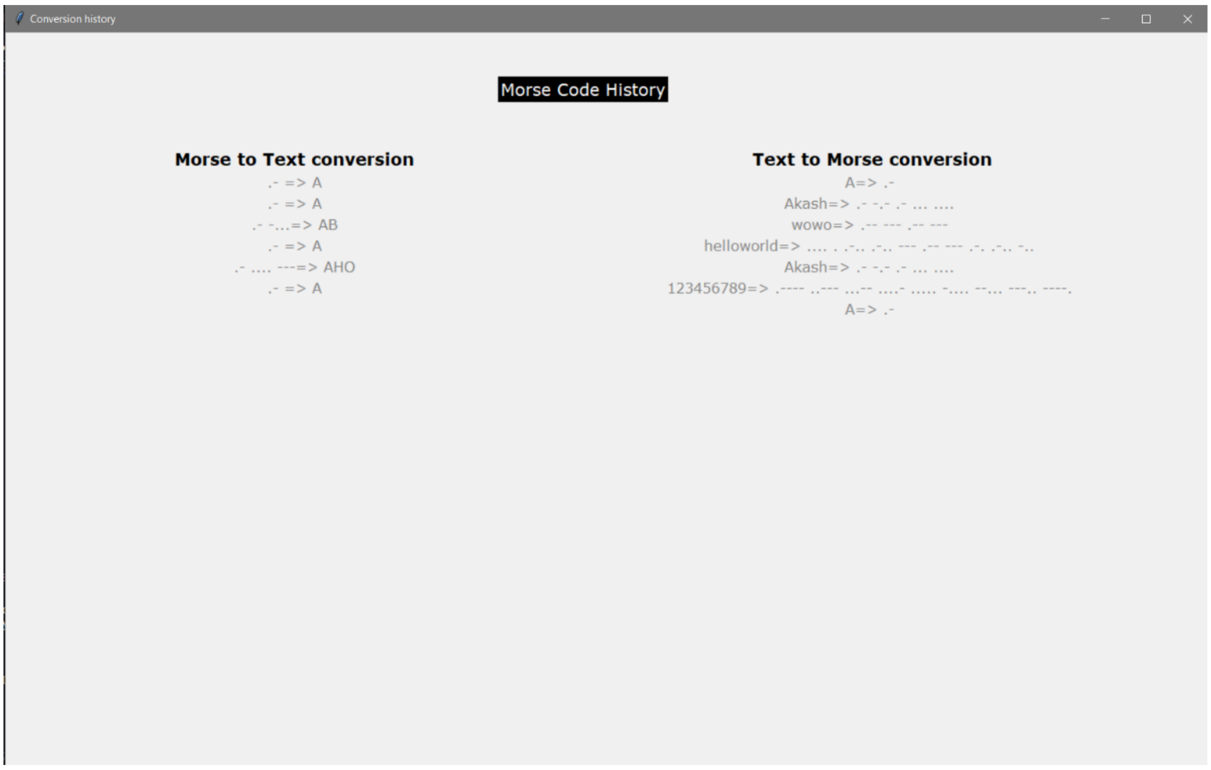


International Morse Code

| | |
|---|---|
| A = .- | S = ... |
| B = -... | T = - |
| C = -.-. | U = ..- |
| D = -.. | V = ...- |
| E = . | W = .-- |
| F = ..-. | X = -..- |
| G = --. | Y = -.-- |
| H = .... | Z = --.. |
| I = .. | 1 = .---- |
| J = .--- | 2 = ..--- |
| K = -.- | 3 = ...-- |
| L = .-.. | 4 = ....- |
| M = -- | 5 = ..... |
| N = -. | 6 = -.... |
| O = --- | 7 = --... |
| P = .--. | 8 = ---.. |
| Q = --.- | 9 = ----. |
| R = .-. | 0 = ----- |

This is Morse Code Table where I have given all the values of alphabets and numbers, this table is provided to both the users, those who have their account and those who have not created their account.

## Page 2: History



Morse Code History

**Morse to Text conversion**

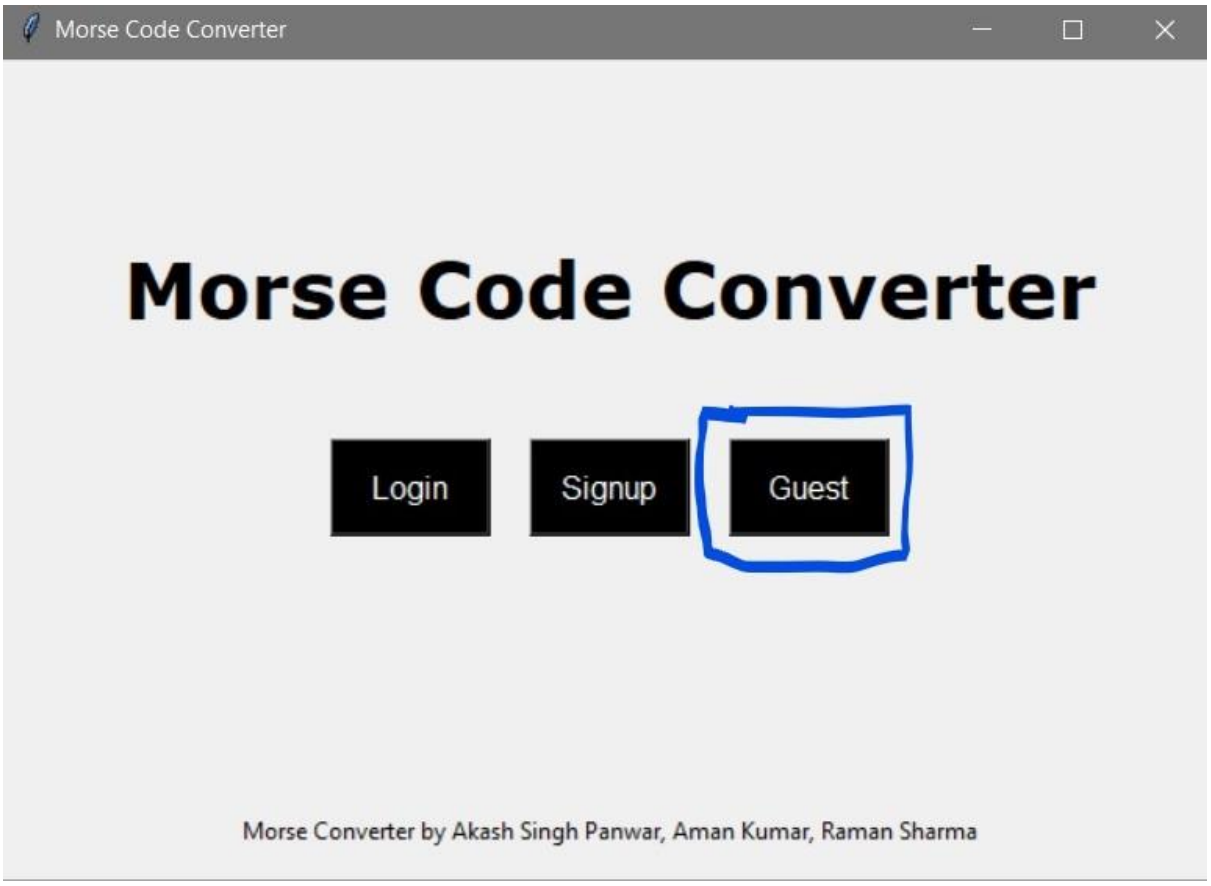
```
.- => A
.- => A
.- -...=> AB
.- => A
.- .... ---=> AHO
.- => A
```

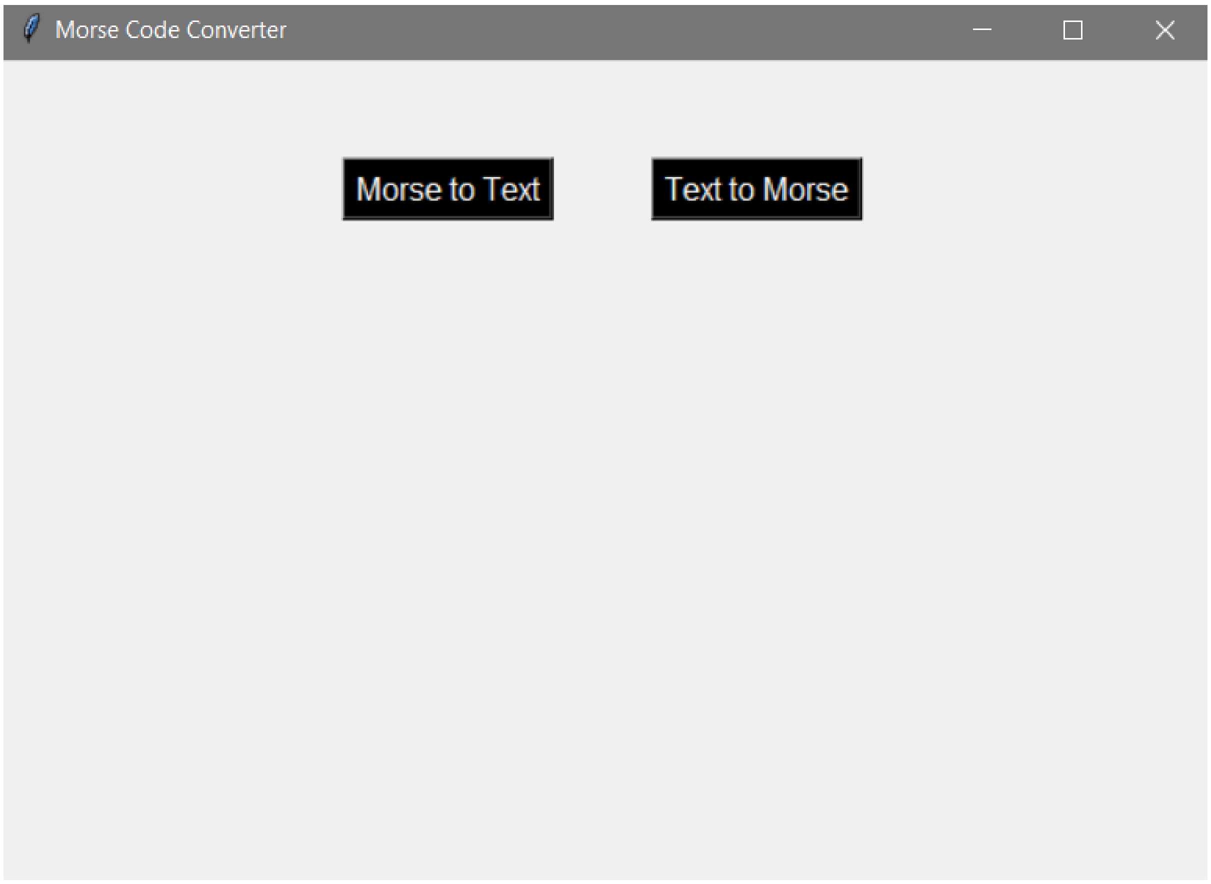
**Text to Morse conversion**

```
A=> .-
Akash=> .- -.- .- ... ....
wowo=> .-- --- .-- ---
helloworld=> .... . .-.. .-.. --- .-- --- .-. .-.. -..
Akash=> .- -.- .- ... ....
123456789=> .---- ..--- ...-- ....- ..... -.... --... ---.. ----.
A=> .-
```

This is History page, where the data of user will be stored i.e. the conversion history of a user is stored, this History feature is not available to guest, in order to access this feature a user should have to create an account, then login. This page has two parts Morse to Text Conversions and Text to Morse Conversions.

## Page 3: Guest


Morse Code Converter
Morse Code Converter
Login
Signup
Guest
Morse Converter by Akash Singh Panwar, Aman Kumar, Raman Sharma



This page has two conversion buttons, Morse to Text and Text to Morse, which will open the conversion page for a user, the guest is not allowed to access history as told earlier, they must have to register themself in signup to access this functionality. The second Image is showing the next page after clicking on Guest button.

## Code Documentation

### 1. Morse Code Table (def morsecodeWin() ):

```
def morsecodeWin():
        windowIndex2 = Tk()
        windowIndex2.geometry("620x420")

        windowIndex2.maxsize(620,420)

        windowIndex2.minsize(620,420)
        windowIndex2.title("Morse Code")
```

```
l1 = Label(windowIndex2, text = "International Morse Code", font = "Verdana 14", bg = "blac
k", fg = "white")
   l1.grid(row = 0, column = 0, padx = 190, pady = 0, columnspan = 2)
```

```
    Label(windowIndex2, text = "A = .-
",font = "verdana 9").grid(row = 1, column = 0,sticky = W, padx = (210,0))
    Label(windowIndex2, text = "B = -
...",font = "verdana 9").grid(row = 2, column = 0,sticky = W, padx = (210,0))
    Label(windowIndex2, text = "C = -.-
.",font = "verdana 9").grid(row = 3, column = 0, sticky = W, padx = (210,0))
    Label(windowIndex2, text = "D = -
..",font = "verdana 9").grid(row = 4, column = 0, sticky = W, padx = (210,0))
    Label(windowIndex2, text = "E = .",font = "verdana 9").grid(row = 5, column = 0, sticky =
W, padx = (210,0))
    Label(windowIndex2, text = "F = ..-
.",font = "verdana 9").grid(row =6 , column = 0, sticky = W, padx = (210,0))
    Label(windowIndex2, text = "G = --
.",font = "verdana 9").grid(row = 7, column = 0, sticky = W, padx = (210,0))
    Label(windowIndex2, text = "H = ....",font = "verdana 9").grid(row = 8, column = 0, sticky =
 W, padx = (210,0))
    Label(windowIndex2, text = "I = ..",font = "verdana 9").grid(row = 9, column = 0, sticky =
W, padx = (210,0))
    Label(windowIndex2, text = "J = .---
",font = "verdana 9").grid(row = 10, column = 0, sticky = W, padx = (210,0))
    Label(windowIndex2, text = "K = -.-
",font = "verdana 9").grid(row = 11, column = 0, sticky = W, padx = (210,0))
    Label(windowIndex2, text = "L = .-
..",font = "verdana 9").grid(row = 12, column = 0, sticky = W, padx = (210,0))
    Label(windowIndex2, text = "M = --
",font = "verdana 9").grid(row = 13, column = 0, sticky = W, padx = (210,0))
    Label(windowIndex2, text = "N = -
.",font = "verdana 9").grid(row = 14, column = 0, sticky = W, padx = (210,0))
    Label(windowIndex2, text = "O = ---
",font = "verdana 9").grid(row = 15, column = 0, sticky = W, padx = (210,0))
    Label(windowIndex2, text = "P = .--
.",font = "verdana 9").grid(row = 16, column = 0, sticky = W, padx = (210,0))
    Label(windowIndex2, text = "Q = --.-
",font = "verdana 9").grid(row = 17, column = 0, sticky = W, padx = (210,0))
    Label(windowIndex2, text = "R = .-
.",font = "verdana 9").grid(row = 18, column = 0, sticky = W, padx = (210,0))

    Label(windowIndex2, text = "S = ...",font = "verdana 9").grid(row = 1, column = 1,sticky =
W, padx = (0,90))
    Label(windowIndex2, text = "T = -
",font = "verdana 9").grid(row = 2, column = 1, sticky = W, padx = (0,90))
    Label(windowIndex2, text = "U = ..-
",font = "verdana 9").grid(row = 3, column = 1,sticky = W, padx = (0,90))
    Label(windowIndex2, text = "V = ...-
",font = "verdana 9").grid(row = 4, column = 1,sticky = W, padx = (0,90))
    Label(windowIndex2, text = "W = .--
",font = "verdana 9").grid(row = 5, column = 1,sticky = W, padx = (0,90))
```

```
   Label(windowIndex2, text = "X = -..-
",font = "verdana 9").grid(row = 6, column = 1,sticky = W, padx = (0,90))
   Label(windowIndex2, text = "Y = -.--
",font = "verdana 9").grid(row = 7, column = 1,sticky = W, padx = (0,90))
   Label(windowIndex2, text = "Z = --
..",font = "verdana 9").grid(row = 8, column = 1,sticky = W, padx = (0,90))
   Label(windowIndex2, text = "1 = .----
",font = "verdana 9").grid(row = 9, column = 1,sticky = W, padx = (0,90))
   Label(windowIndex2, text = "2 = ..---
",font = "verdana 9").grid(row = 10, column = 1,sticky = W, padx = (0,90))
   Label(windowIndex2, text = "3 = ...--
",font = "verdana 9").grid(row = 11, column = 1,sticky = W, padx = (0,90))
   Label(windowIndex2, text = "4 = ....-
",font = "verdana 9").grid(row = 12, column = 1,sticky = W, padx = (0,90))
   Label(windowIndex2, text = "5 = .....",font = "verdana 9").grid(row = 13, column = 1,sticky
= W, padx = (0,90))
   Label(windowIndex2, text = "6 = -
....",font = "verdana 9").grid(row = 14, column = 1,sticky = W, padx = (0,90))
   Label(windowIndex2, text = "7 = --
...",font = "verdana 9").grid(row = 15, column = 1,sticky = W, padx = (0,90))
   Label(windowIndex2, text = "8 = ---
..",font = "verdana 9").grid(row = 16, column = 1,sticky = W, padx = (0,90))
   Label(windowIndex2, text = "9 = ----
.",font = "verdana 9").grid(row = 17, column = 1,sticky = W, padx = (0,90))
   Label(windowIndex2, text = "0 = -----
",font = "verdana 9").grid(row = 18, column = 1,sticky = W, padx = (0,90))

windowIndex2.mainloop()
```

In the above code snippet, the code is written to define the Morse Code Table, the **windowIndex2** is an object of **Tkinter framework**, after this we have defined the geometry of window by **620x420**, after this we have used two functions **maxsize** and **minsize** which have same values in them , this is because we don't want to change the size of the window, we want to keep that size fixed, after this we have given a title to the window i.e. **Morse Code**.

Now here I have made a label as a name **International Morse Code** with font as Verdana and font size 14, with back ground colour as black and foreground colour as white and gave position by using gird function.

After this, I have created 36 labels which is showing the values of text and numbers in Verdana font of size 9, at last I gave them position using **grid** function, in which I have defined row, column, sticky to W, (sticky is a widget in Tkinter), we want the labels to get sticked in west direction, and at last I used **padx** to give them position with respect to x-axis.

## 2. History ( def history() ):

```
def history():
   # connecting to database
   db = mysql.connector.connect(host='localhost',user='root',password='Akku2017@123',da
tabase='pythonproject')
```

in the above code snippet we have defined a **history** function, and made a connection with database by using **mysql.connector.connect** function , inside that function I have added the credentials to make connectivity to database.

```
if(db):
            myCursor = db.cursor()
      #sql statements for texttomorsehistory and morsetotexthistory
      # loginedUserData[0]
      texttomorsehistorySql = r"SELECT * FROM texttomorsehistory WHERE userid = '"+str(lo
ginedUserData[0])+"'"
      morsetotexthistorySql = r"SELECT * FROM morsetotexthistory WHERE userid = '"+str(lo
ginedUserData[0])+"'"
```

in the above code snippet there is a condition that is database is connected then only the next commands will execute ,in this we have created a cursor, so that we can execute all the **sql** commands, now we have created two variables **texttomorsehistorySql** and **morsetotexthistorySql** in which we are storing the user id which is present in 0 index of every row,  we have made it as a raw string to make it pure or in other words we are not considering the function of escape sequences.

```
          myCursor.execute(texttomorsehistorySql)
     result1 = myCursor.fetchall()

      texttomorsehistorylist = list()
      for i in result1:
         texttomorsehistorylist.append(i)
```

in the above code snippet we are executing the texttomorsehistorySql command, that is taking values from all the rows of **texttomorsehistory** table in our database, then we are fetching all the row details of that table, and storing it to **result1** tuple, the fetchall() command returns the list of tuples in our result1 tuple, now we have created a list **texttomorsehistorylist** in which the every data present in result1 tuple is getting stored.

```
          myCursor.execute(morsetotexthistorySql)
      result2 = myCursor.fetchall()

      morsetotexthistorylist = list()
      for i in result2:
         morsetotexthistorylist.append(i)
```

the above code snippet is working same as the previous code snippet, just with a change in fetched values, they are now Morse values which are fetched from database.

Hence, all the conversion history is getting saved in our database.

```
        windowindex3 = Tk()

    windowindex3.geometry("1366x1280")
    windowindex3.maxsize(1920,1280)
    windowindex3.minsize(1366,1280)
    windowindex3.title("Conversion history")
```

in the above code snippet I have made an object of Tk() framework, by providing is size and title as **Conversion history**.

```
l1 = Label(windowindex3, text = "Morse Code History", font = "Verdana 14", bg = "black", fg
= "white")
    l1.grid(row = 0, column = 1,  pady = 50)
l2 = Label(windowindex3, text="Morse to Text conversion", font="Verdana 14 bold",width=5
0)
    l2.grid(row=1, column=0,columnspan=2)
l3 = Label(windowindex3, text="Text to Morse conversion", font="Verdana 14 bold",width=5
0)
    l3.grid(row=1, column=2,columnspan=2)
```

in the above code snippet I have defined 3 labels which are the headings like Morse Code History, Morse to Text conversion and Text to Morse conversion.

```
    global row
    row = 3
    if len(morsetotexthistorylist) != 0: #if the list is not empty then it will print the data in gi
ven format
      for i in morsetotexthistorylist:
        Label(windowindex3, text=str(i[1])+'=> '+str(i[2]), font='Verdana 12', fg="grey").grid
(row=row+1,column=0,columnspan=2)
        row = row+1
    else:
      Label(windowindex3, text="No data Found", font='Verdana 12', fg="grey").grid(row=
4,column=0,columnspan=2)
```

in the above code snippet, I have made a global variable row and assigned it equal to 3, now there is condition, that if the length of morsetotexthistorylist list is not 0 then for that list show all the conversion values, first Morse then the value will be shown, and in every iteration value of row will be incremented, if that list is empty then show NO DATA FOUND.

```
        row = 3
        if len(texttomorsehistorylist) != 0:
          for i in texttomorsehistorylist:
                Label(windowindex3, text=str(i[1])+'=> '+str(i[2]), font='Verdana 12', fg="grey").grid
(row=row+1,column=4,columnspan=3)
                row = row+1

            else:
            Label(windowindex3, text="No data Found", font='Verdana 12', fg="grey").grid(row=
4,column=4,columnspan=3)

        windowindex3.mainloop()
```

the above code snippet is working same as the previous one, but we have assigned row again equal to 3, because we want to show the converted data from the same row as **morsetotexthistorylist,** it will show the values from text to Morse.

```
 else:
        messagebox.showwarning("Warning","Database is not connected!!")
```

the above code snippet is else part of that condition when the database was connected if not then show warning “Database is not connected!”

## 3. Guest ( def guest() ):

```
def guest():
    head.grid_remove()
    loginButton.grid_remove()
    signupButton.grid_remove()
    guestButton.grid_remove()
    bottomFrame.grid_remove()
    logoutButton.grid_remove()
    converterButton.grid_remove()
    historyButton.grid_remove()

    global buttonFrame2

    buttonFrame2 = Frame(windowIndex)
    buttonFrame2.grid(row = 0, column = 0, columnspan = 2, padx = 125, pady = (50,0))

    mtoeButton = Button(buttonFrame2, text = "Morse to Text", font = "10", bg = '#000000', f
g = '#fff', command = Morse_to_English)
    mtoeButton.grid(row = 0, column = 0, padx = 50)

    etomButton = Button(buttonFrame2, text = "Text to Morse", font = "10", bg = '#000000', f
g = '#fff', command = English_to_Morse)
    etomButton.grid(row = 0, column = 1)
```

in the above code snippet I have defined the functionality of guest page , I have used **grid_remove()** function in head, login button, signup button, guest button, bottom frame, logout button, convert and history button, to that all the part of previous pages will be removed.

Then there is a buttonFrame2 global variable which I used for making a frame and that frame is inside windowIndex , then I have provided two buttons with name Morse to Text and Text to Morse, which will direct user to the conversion pages.