Simple Register and Login Application Using Python 3 and MySQL Database Server

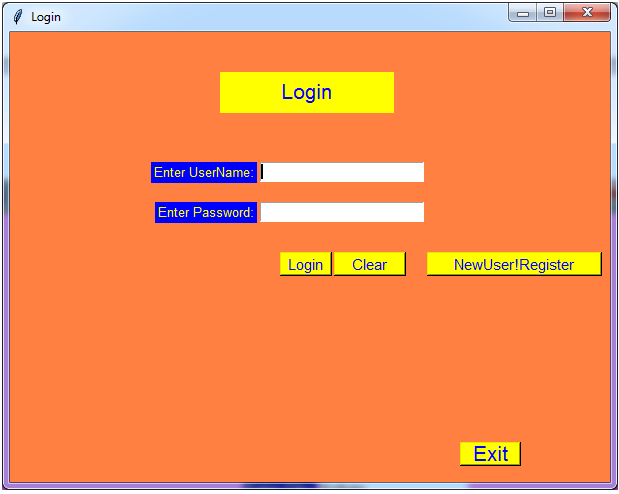
Introduction

This is a simple login and register desktop application using Python 3.7 Tkinter and MySQL Database server. For MySQL, the database server XAMPP version 3.2.4 is used.

The latest version of XAMPP or MySQL Database server can be used.

Mysql.connector is used after downloading and installing it for connecting to the MySQL Database Server.

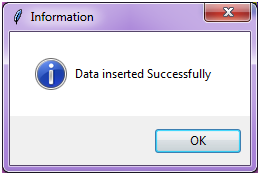
After running the application, the login window shown below opens up. This is where registered users can log in by entering username and password. A new user has to register to log in by clicking the NewUser!Register Button.



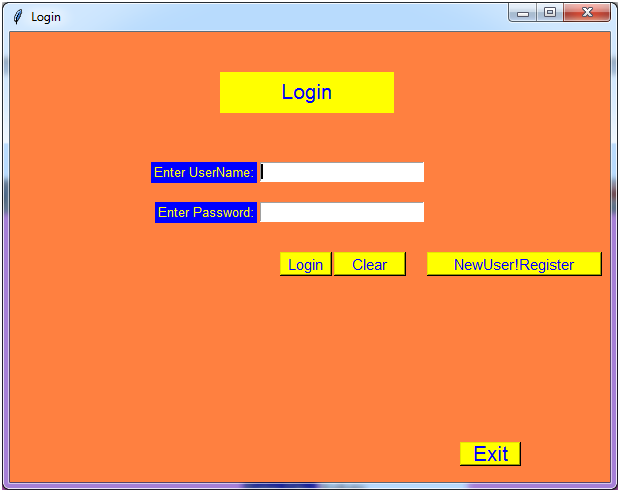
After clicking NewUser!Register Button Register window shown below opens where a new user can register himself. Then a Database named ‘User’ is created and a Table Named ‘User’ is created. Then a new User record is inserted in table ‘User’.



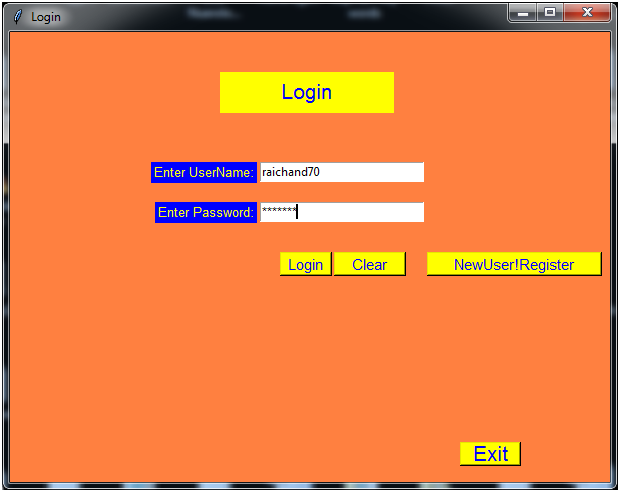
After clicking the ‘Register’ button below the message window shows up if registration is successful.



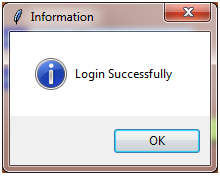
The registered user would come back to the ‘Login’ Window after clicking the ‘Back’ button. The login window opens, as shown below.



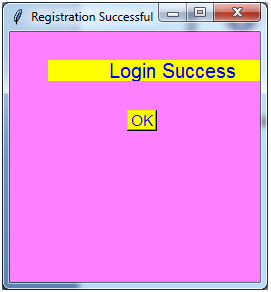
Then the registered user enters his registered username and password, as shown below.

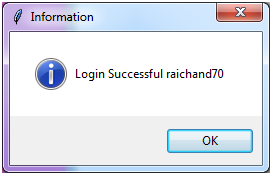


If the entered username and password match the username and password stored in the database, then the successful login message pops up, as shown below.

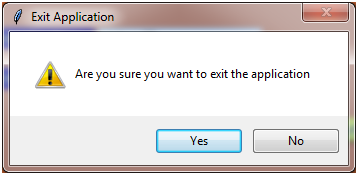


Also, the successful login window as shown below opens up. If the ok button is clicked, then another login successful message box pops up.





If Exit Button is clicked to close the application then a confirmation Message box pops up and on clicking ‘yes’ button the application closes.



There is one Python file named **Login.py** created. The code written in it is shown below.

1. ##
2. connecting to the database using 'connect()'
3. method## it takes 3 required parameters 'host', 'user', 'password'
4. **db\_connection = mysql.connector.connect(host = "localhost", user = "raichand70", password = "1America")**
5. In the above code snippet reader should change user **and** password to his username **and** password to connect to MySQL Database Server.
6. - -- -- -- -- -- -- -- -- -- -- -- -Login.py-- -- -- -- -- -- -- -- --- --- --- --- --
7. **import** tkinter as tk
8. **import** tkinter.messagebox as mb
9. **import** random
10. **import** tkinter.ttk## Connecting to the database## importing 'mysql.connector'
11. **for** connection to mysql database
12. **import** mysql.connector## connecting to the database using 'connect()'
13. method## it takes 3 required parameters 'host', 'user', 'password'
14. db\_connection = mysql.connector.connect(host = "localhost", user = "raichand70", password = "1America")# creating database\_cursor to perform SQL operation
15. db\_cursor = db\_connection.cursor(buffered = True)# buffered = True would
16. **return** actual number of records selected otherwise would
17. **return** -1# Using Toplevel widget to create a new window named Register Successful Window
18. **class** Login\_Success\_Window(tk.Toplevel): **def** \_\_init\_\_(self, parent): super().\_\_init\_\_(parent)
19. self.original\_frame = parent
20. self.geometry("250x250")
21. self.title("Registration Successful")
22. self.configure(background = "#ff80ff")
23. self.lbl\_Login\_success = tk.Label(self, text = "Login Success", font = ("Helvetica", 15), bg = "yellow", fg = "blue")
24. self.lbl\_Login\_success.place(relx = 0.150, rely = 0.111, height = 21, width = 250)# create OK button
25. self.btn\_register = tk.Button(self, text = "OK", font = ("Helvetica", 11), bg = "yellow", fg = "blue", command = self.delete\_login\_success)# self.btn\_register.pack(side = tk.BOTTOM)
26. self.btn\_register.place(relx = 0.467, rely = 0.311, height = 21, width = 30)
27. **def** delete\_login\_success(self): mb.showinfo('Information', "Login Successful " + str(username))
28. self.destroy()
29. self.original\_frame.show()# Using Toplevel widget to create a new window named RegisterWindow to register a new user
30. **class** RegisterWindow(tk.Toplevel): **def** \_\_init\_\_(self, parent): super().\_\_init\_\_(parent)
31. self.original\_frame = parent
32. self.geometry("600x450+485+162")
33. self.title("Register")
34. self.configure(background = "#ff80ff")
35. self.lblRegister = tk.Label(self, text = "Register", font = ("Helvetica", 16), bg = "yellow", fg = "blue")
36. self.lblFName = tk.Label(self, text = "Enter FirstName:", font = ("Helvetica", 10), bg = "blue", fg = "yellow")
37. self.lblLName = tk.Label(self, text = "Enter LastName:", font = ("Helvetica", 10), bg = "blue", fg = "yellow")
38. self.lblLName = tk.Label(self, text = "Enter LastName:", font = ("Helvetica", 10), bg = "blue", fg = "yellow")
39. self.lblUId = tk.Label(self, text = "Enter UserId:", font = ("Helvetica", 10), bg = "blue", fg = "yellow")
40. self.lblPwd = tk.Label(self, text = "Enter Password:", font = ("Helvetica", 10), bg = "blue", fg = "yellow")# self.lblPin = tk.Label(self, text = "Enter Pin:", font = ("Helvetica", 10), bg = "blue", fg = "yellow")
41. self.lblContactNo = tk.Label(self, text = "Enter Contact No:", font = ("Helvetica", 10), bg = "blue", fg = "yellow")
42. self.lblCity = tk.Label(self, text = "Enter City:", font = ("Helvetica", 10), bg = "blue", fg = "yellow")
43. self.lblState = tk.Label(self, text = "Enter State:", font = ("Helvetica", 10), bg = "blue", fg = "yellow")
44. self.txtFName = tk.Entry(self)
45. self.txtLName = tk.Entry(self)
46. self.txtUId = tk.Entry(self)
47. self.txtPwd = tk.Entry(self)
48. self.txtContact = tk.Entry(self)
49. self.txtCity = tk.Entry(self)
50. self.txtState = tk.Entry(self)
51. self.btn\_register = tk.Button(self, text = "Register", font = ("Helvetica", 11), bg = "yellow", fg = "blue", command = self.register)
52. self.btn\_cancel = tk.Button(self, text = "<-Back", font = ("Helvetica", 11), bg = "yellow", fg = "blue", command = self.onClose)
53. self.lblRegister.place(relx = 0.467, rely = 0.111, height = 21, width = 100)
54. self.lblFName.place(relx = 0.318, rely = 0.2, height = 21, width = 100)
55. self.lblLName.place(relx = 0.319, rely = 0.267, height = 21, width = 100)
56. self.lblUId.place(relx = 0.355, rely = 0.333, height = 21, width = 78)
57. self.lblPwd.place(relx = 0.319, rely = 0.4, height = 21, width = 100)
58. self.lblContactNo.place(relx = 0.310, rely = 0.467, height = 21, width = 105)
59. self.lblCity.place(relx = 0.375, rely = 0.533, height = 21, width = 66)
60. self.lblState.place(relx = 0.369, rely = 0.6, height = 21, width = 70)
61. self.txtFName.place(relx = 0.490, rely = 0.2, height = 20, relwidth = 0.223)
62. self.txtLName.place(relx = 0.490, rely = 0.267, height = 20, relwidth = 0.223)
63. self.txtUId.place(relx = 0.490, rely = 0.333, height = 20, relwidth = 0.223)
64. self.txtPwd.place(relx = 0.490, rely = 0.4, height = 20, relwidth = 0.223)
65. self.txtContact.place(relx = 0.490, rely = 0.467, height = 20, relwidth = 0.223)
66. self.txtCity.place(relx = 0.490, rely = 0.533, height = 20, relwidth = 0.223)
67. self.txtState.place(relx = 0.490, rely = 0.6, height = 20, relwidth = 0.223)
68. self.btn\_register.place(relx = 0.500, rely = 0.660, height = 24, width = 63)
69. self.btn\_cancel.place(relx = 0.605, rely = 0.660, height = 24, width = 54)
70. **def** register(self): **if** db\_connection.is\_connected() == False: db\_connection.connect()# executing cursor with execute method and pass SQL query
71. db\_cursor.execute("CREATE DATABASE IF NOT EXISTS User")# Create a Database Named AradhanaBank
72. db\_cursor.execute("use User")# Interact with Bank Database# creating required tables
73. db\_cursor.execute("Create table if not exists USER(uid VARCHAR(30) NOT NULL PRIMARY KEY,password VARCHAR(30),fname VARCHAR(30),lname VARCHAR(30),city VARCHAR(20),state VARCHAR(30),mobileno VARCHAR(10))")
74. db\_connection.commit()
75. fname = self.txtFName.get()# Retrieving entered first name
76. lname = self.txtLName.get()# Retrieving entered last name
77. uid = self.txtUId.get()# Retrieving entered user id
78. pwd = self.txtPwd.get()# Retrieving entered password# pin = self.txtPin.get()# Retrieving entered ATM pin number
79. contact\_no = self.txtContact.get()# Retrieving entered contact number
80. city = self.txtCity.get()# Retrieving entered city name
81. state = self.txtState.get()# Retrieving entered state name# validating Entry Widgets
82. **if** fname == "": mb.showinfo('Information', "Please Enter Firstname")
83. self.txtFName.focus\_set()
84. **return**
85. **if** lname == "": mb.showinfo('Information', "Please Enter Lastname")
86. self.txtLName.focus\_set()
87. **return**
88. **if** uid == "": mb.showinfo('Information', "Please Enter User Id")
89. self.txtUId.focus\_set()
90. **return**
91. **if** pwd == "": mb.showinfo('Information', "Please Enter Password")
92. self.txtPwd.focus\_set()
93. return
94. **if** contact\_no == "": mb.showinfo('Information', "Please Enter Contact Number")
95. self.txtContact.focus\_set()
96. return
97. **if** city == "": mb.showinfo('Information', "Please Enter City Name")
98. self.txtCity.focus\_set()
99. **return**
100. **if** state == "": mb.showinfo('Information', "Please Enter State Name")
101. self.txtState.focus\_set()
102. **return**# Inserting record into bank table of bank database
103. db\_cursor.execute("use User")# Interact with Bank Database
104. query = "INSERT INTO User(uid,password,fname,lname,city,state,mobileno) VALUES ('%s','%s','%s','%s','%s','%s','%s')" % (uid, pwd, fname, lname, city, state, contact\_no)
105. **try**: #implement sql Sentence
106. db\_cursor.execute(query)
107. mb.showinfo('Information', "Data inserted Successfully")# Submit to database
108. **for** execution
109. db\_connection.commit()
110. **except**: mb.showinfo('Information', "Data insertion failed!!!")# Rollback in
111. case there **is** any error
112. db\_connection.rollback()# Close database connection
113. db\_connection.close()
114. **def** onClose(self): ""
115. ""
116. ""
117. self.destroy()
118. self.original\_frame.show()
119. **class** LoginApp(tk.Tk): **def** \_\_init\_\_(self): super().\_\_init\_\_()
120. self.title("Login")
121. self.geometry("600x450+351+174")
122. self.configure(bg = "#ff8040")
123. self.lblHeading = tk.Label(self, text = "Login", font = ("Helvetica", 16), bg = "yellow", fg = "blue")
124. self.lbluname = tk.Label(self, text = "Enter UserName:", font = ("Helvetica", 10), bg = "blue", fg = "yellow")
125. self.lblpsswd = tk.Label(self, text = "Enter Password:", font = ("Helvetica", 10), bg = "blue", fg = "yellow")
126. self.txtuname = tk.Entry(self, width = 60)
127. self.txtpasswd = tk.Entry(self, width = 60, show = "\*")
128. self.btn\_login = tk.Button(self, text = "Login", font = ("Helvetica", 11), bg = "yellow", fg = "blue", command = self.login)
129. self.btn\_clear = tk.Button(self, text = "Clear", font = ("Helvetica", 11), bg = "yellow", fg = "blue", command = self.clear\_form)
130. self.btn\_register = tk.Button(self, text = "NewUser!Register", font = ("Helvetica", 11), bg = "yellow", fg = "blue", command = self.open\_registration\_window)
131. self.btn\_exit = tk.Button(self, text = "Exit", font = ("Helvetica", 16), bg = "yellow", fg = "blue", command = self.exit)
132. self.lblHeading.place(relx = 0.35, rely = 0.089, height = 41, width = 174)
133. self.lbluname.place(relx = 0.235, rely = 0.289, height = 21, width = 106)
134. self.lblpsswd.place(relx = 0.242, rely = 0.378, height = 21, width = 102)
135. self.txtuname.place(relx = 0.417, rely = 0.289, height = 20, relwidth = 0.273)
136. self.txtpasswd.place(relx = 0.417, rely = 0.378, height = 20, relwidth = 0.273)
137. self.btn\_login.place(relx = 0.45, rely = 0.489, height = 24, width = 52)
138. self.btn\_clear.place(relx = 0.54, rely = 0.489, height = 24, width = 72)
139. self.btn\_register.place(relx = 0.695, rely = 0.489, height = 24, width = 175)
140. self.btn\_exit.place(relx = 0.75, rely = 0.911, height = 24, width = 61)
141. **def** open\_registration\_window(self): self.withdraw()
142. window = RegisterWindow(self)
143. window.grab\_set()
144. **def** open\_login\_success\_window(self): self.withdraw()
145. window = Login\_Success\_Window(self)
146. window.grab\_set()
147. **def** show(self): ""
148. ""
149. ""
150. self.update()
151. self.deiconify()
152. **def** login(self): **if** db\_connection.is\_connected() == False: db\_connection.connect()# executing cursor with execute method and pass SQL query
153. db\_cursor.execute("CREATE DATABASE IF NOT EXISTS User")# Create a Database Named Bank
154. db\_cursor.execute("use User")# Interact with Bank Database# creating required tables
155. db\_cursor.execute("create table if not exists USER(uid VARCHAR(30) NOT NULL PRIMARY KEY,password VARCHAR(30),fname VARCHAR(30),lname VARCHAR(30),city VARCHAR(20),state VARCHAR(30),mobileno VARCHAR(10))")
156. db\_connection.commit()
157. **try**: **global** username
158. username = str(self.txtuname.get())# Retrieving entered username
159. passwd = str(self.txtpasswd.get())# Retrieving entered password
160. **if** username == "": mb.showinfo('Information', "Please Enter Username")
161. self.txtuname.focus\_set()
162. **return**
163. **if** passwd == "": mb.showinfo('Information', "Please Enter Password")
164. self.txtpasswd.focus\_set()
165. **return**
166. **print**(username)
167. **print**(passwd)
168. query = "SELECT \* FROM User WHERE uid = '" + username + "' AND password = '" + passwd + "'"
169. **print**(query)# implement sql Sentence
170. db\_cursor.execute(query)
171. rowcount = db\_cursor.rowcount
172. **print**(rowcount)
173. **if** db\_cursor.rowcount == 1: mb.showinfo('Information', "Login Successfully")
174. self.open\_login\_success\_window()
175. **else** :mb.showinfo('Information', "Login failed,Invalid Username or Password.Try again!!!")
176. **except**: #Closing Connection
177. db\_connection.disconnect()
178. **def** clear\_form(self): self.txtuname.delete(0, tk.END)
179. self.txtpasswd.delete(0, tk.END)
180. self.txtuname.focus\_set()
181. **def** exit(self): MsgBox = mb.askquestion('Exit Application', 'Are you sure you want to exit the application', icon = 'warning')
182. **if** MsgBox == 'yes': self.destroy()
183. **if** \_\_name\_\_ == "\_\_main\_\_": app = LoginApp()
184. app.mainloop()   
     ------------------------------------------------------------------------------------------