**PROJECT ON**

**ACCOUNT MANAGEMENT SYSTEM**

|  |
| --- |
| **SAILI TOTARE - 56-SE/IT/B**  **DHANASHREE SHINDE -50-SE/IT/B**  **MAYUR PAYMODE- 42-SE/IT/B** |

**What is SQlite?**

SQLite is an in-process library that implements a [self-contained](https://www.sqlite.org/selfcontained.html),[serverless](https://www.sqlite.org/serverless.html), [zero-configuration](https://www.sqlite.org/zeroconf.html), [transactional](https://www.sqlite.org/transactional.html) SQL database engine. The code for SQLite is in the [public domain](https://www.sqlite.org/copyright.html) and is thus free for use for any purpose, commercial or private. SQLite is the [most widely deployed](https://www.sqlite.org/mostdeployed.html)database in the world with more applications than we can count, including several [high-profile projects.](https://www.sqlite.org/famous.html)

**WHAT IS TKINTER?**

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter outputs the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.  
**To create a tkinter:**

1. Importing the module – tkinter
2. Create the main window (container)
3. Add any number of widgets to the main

**Code-**

from tkinter import \*

import ledger\_bk

window = Tk()

window.title("Account Management")

def \_\_init\_\_(self, master):

self.master = master

self.left = Frame(master, width=1000, height=500, bg='olivedrab1')

self.left.pack(side=LEFT)

def view\_command():

lb.delete(0,END)

for row in ledger\_bk.viewall():

lb.insert(END,row)

def search\_command():

lb.delete(0,END)

for row in ledger\_bk.search(name=name.get(),user=user.get(),password=password.get(),category=category.get()):

lb.insert(END,row)

def add\_command():

ledger\_bk.add(name.get(),user.get(),password.get(),category.get(),cdate.get())

lb.delete(0,END)

lb.insert(END,name.get(),user.get(),password.get(),category.get(),cdate.get())

def get\_selected\_row(event):

try:

global selected\_tuple

index=lb.curselection()[0]

selected\_tuple = lb.get(index)

e1.delete(0,END)

e1.insert(END,selected\_tuple[1])

e2.delete(0,END)

e2.insert(END,selected\_tuple[2])

e3.delete(0,END)

e3.insert(END,selected\_tuple[3])

e4.delete(0,END)

e4.insert(END,selected\_tuple[4])

e5.delete(0,END)

e5.insert(END,selected\_tuple[5])

except IndexError:

pass

def update\_command(): ledger\_bk.update(selected\_tuple[0],name.get(),user.get(),password.get(),category.get(),cdate.get())

view\_command()

def delete\_command():

ledger\_bk.delete(selected\_tuple[0])

view\_command()

#lb.delete(END,get\_selected\_row.selected\_tuple)

def clear\_command():

lb.delete(0,END)

e1.delete(0,END)

e2.delete(0,END)

e3.delete(0,END)

e4.delete(0,END)

e5.delete(0,END)

l1 = Label(window,text="Name",width=10,height=2,font=(' algerian',15),bg='mistyrose3')

l1.grid(row=0,column=0,columnspan=1)

l2 = Label(window,text="Email",width=10,height=2,font=(' algerian',15),bg='grey')

l2.grid(row=1,column=0,columnspan=1)

l3 = Label(window,text="Password",width=10,height=2,font=(' algerian',15),bg='mistyrose3')

l3.grid(row=2,column=0,columnspan=1)

l4 = Label(window,text="Category",width=10,height=2,font=(' algerian',15),bg='grey')

l4.grid(row=3,column=0,columnspan=1)

l5 = Label(window,text="Date",width=10,height=2,font=(' algerian',15),bg='mistyrose3')

l5.grid(row=4,column=0,columnspan=1)

name=StringVar()

e1 = Entry(window,textvariable=name,width=50)

e1.grid(row=0,column=0,columnspan=15)

user=StringVar()

e2 = Entry(window,textvariable=user,width=50)

e2.grid(row=1,column=0,columnspan=15)

password=StringVar()

e3 = Entry(window,textvariable=password,show='\*',width=50)

e3.grid(row=2,column=0,columnspan=15)

category=StringVar()

e4 = Entry(window,textvariable=category,width=50)

e4.grid(row=3,column=0,columnspan=15)

cdate=StringVar()

e5 = Entry(window,textvariable=cdate,width=50)

e5.grid(row=4,column=0,columnspan=15)

b1 = Button(window,text="Add",width=23,height=2,bg='Lemonchiffon4',command=add\_command)

b1.grid(row=5,column=0)

b2 = Button(window,text="Update",width=23,height=2,bg='Lemonchiffon3',command=update\_command)

b2.grid(row=5,column=1)

b3 = Button(window,text="Search",width=23,height=2,bg='Lemonchiffon4',command=search\_command)

b3.grid(row=5,column=2)

b4 = Button(window,text="View All",width=23,height=2,bg='Lemonchiffon3',command=view\_command)

b4.grid(row=5,column=3)

b5 = Button(window,text="Delete",width=23,height=2,bg='Lemonchiffon4',command=delete\_command)

b5.grid(row=5,column=4)

b6 = Button(window,text="Cancel",width=23,height=2,bg='Lemonchiffon3',command=window.destroy)

b6.grid(row=5,column=5)

b7 = Button(window,text="Clear All",width=20,height=1,bg='Lemonchiffon4',command=clear\_command)

b7.grid(row=0,column=5)

lb=Listbox(window,height=30,width=150)

lb.grid(row=6,column=0,columnspan=6)

sb=Scrollbar(window)

sb.grid(row=6,column=6,rowspan=6)

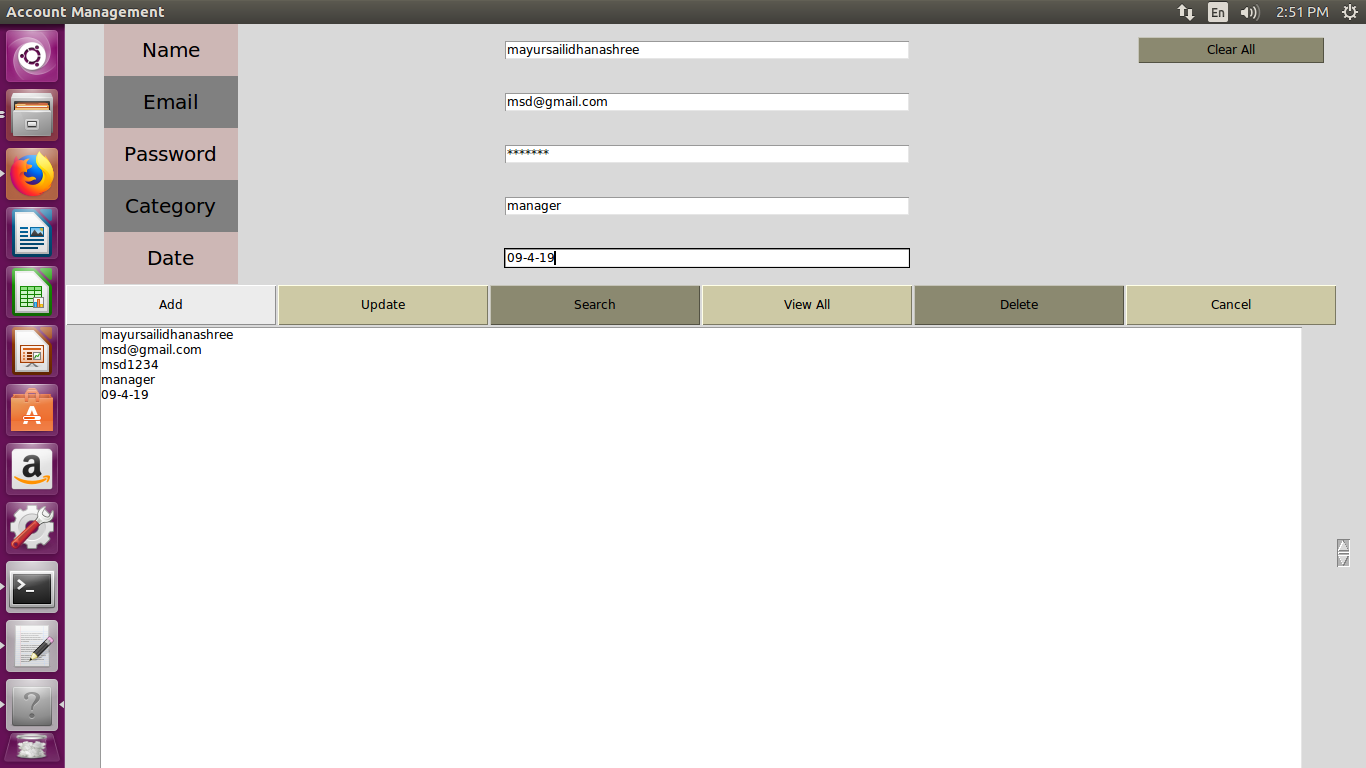
lb.configure(yscrollcommand=sb.set)

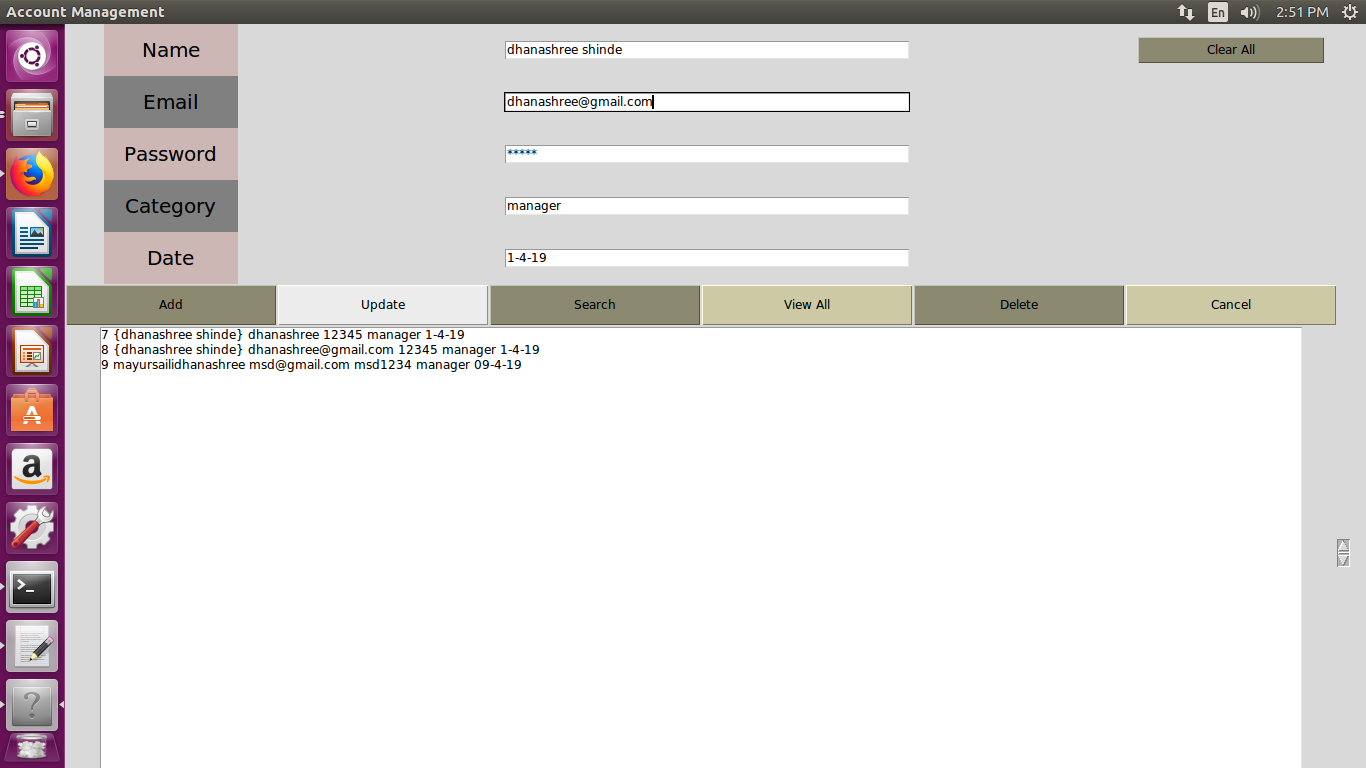
sb.configure(command=lb.yview)

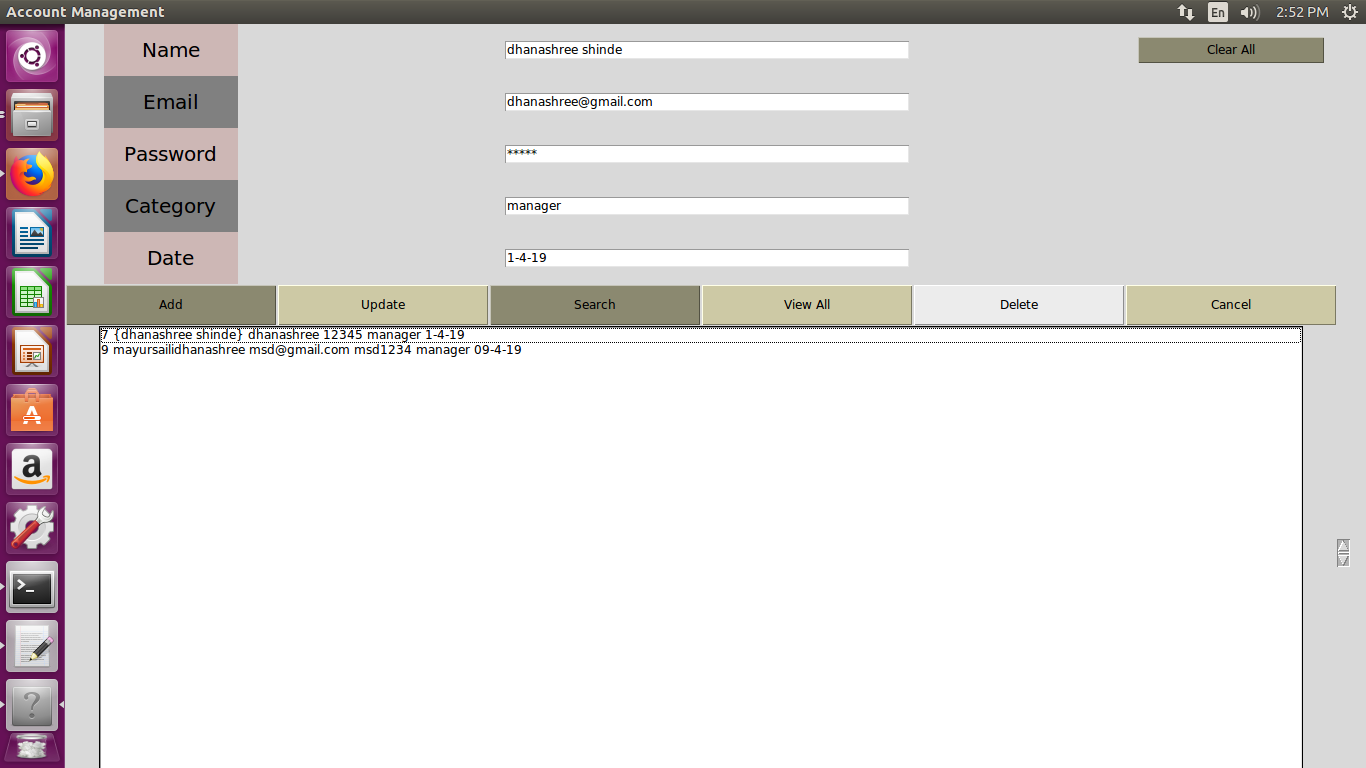
lb.bind('<<ListboxSelect>>',get\_selected\_row)

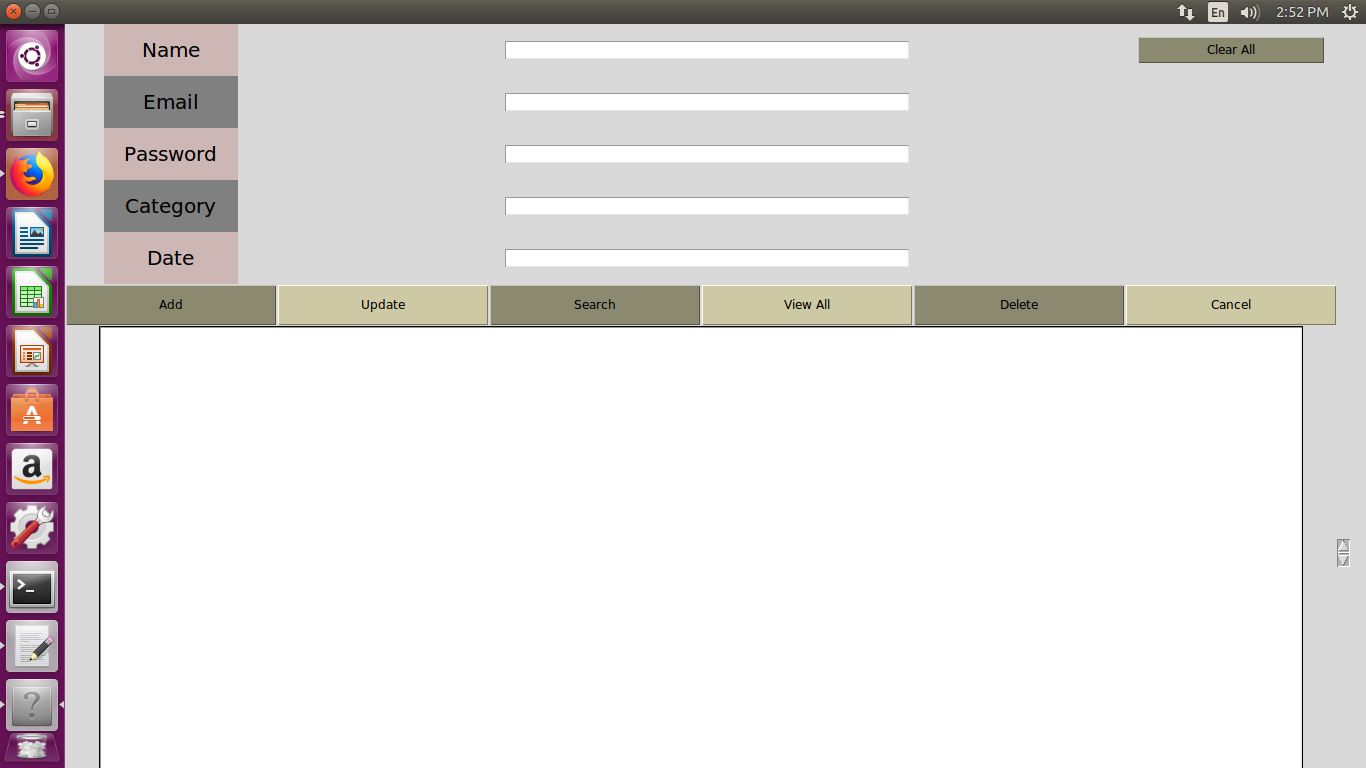
window.geometry('1000x1000')

window.mainloop()

**OUTPUT:**







**Conclusion:**

* **Basically the techniques of managerial accounting are applicable in an organization to** **help in devising planning, supporting decisions of management, and performance management** **system and provide management assistant to formulate and interpret the organizational strategies** **in order to generate profitability.**
* **It helps ensure organizational success.**
* **The managerial** **accounting techniques used in different areas support management decisions and enhance the** **chances for a high rate of return in the future.**