## horizontal line



BILL GENERATOR

MODERN SOLUTION FOR HANDWRITTEN BILLS

ANIKET SHINDE

SHRAVAN KEGADE

PRATHMESH GAWANDE

# 

# RESEARCH CONTEXT

# E-commerce (electronic commerce) is the combination between the traditional business and the improvement of information processing brought by the internet. This combination allows companies to exchange sales information, do financial transactions, deliver goods & services and process invoices in an automatic way. E-commerce is leveraged by a safe communication medium which potential world-wide buyers can easily access. Some of the biggest advantages of e-commerce (B2C and B2B) to a company are:

* Brings world-wide visibility: the business is available anytime, anywhere.
* Reduces the values associated with the consumer/ supplier or partner/partner transactions.
* Reduces product delivery time.
* Allows access to new markets with small financial effort.
* Reduces competitive advantages of big companies over the small ones: a portal does not depend on the company’s financial strength; the client buys from the most trustworthy supplier and from the one who serves the client better.
* Reduces the bureaucracy associated with a purchase.
* Obtains information directly from a CRM (Customer Relationship Management): e.g. the customer’s personal tastes and the market evolution.
* Keeps the company permanently in touch with its business partners, thus increasing both ends’ satisfaction and diminishing communication costs.

A particular kind of traditional business where e-commerce may be beneficial is store commerce. Typically store commerce involves human interaction, waiting in line, the transaction of money and dealing with paper receipts and invoices. With these factors economic, time and ecologic issues emerge. E-commerce is a simple solution for these problems that can potentially bring a huge profit margin.Studying how intelligent environments can help customers make the best purchasing decisions has become an important research area. One research issue in particular is to try and understand how mobile devices can help users to shop intelligently and intuitively.

# PROJECT SUMMARY

# An important aspect of shopping is the receipt that is given to customers. The customer may find the paper receipt less than ideal for a number of reasons: it can be easily lost, damaged or destroyed. This could be a problem when the customer wishes to use the paper receipt to organize his personal finances or to exchange the item of purchase. If the receipt is damaged to a point where it’s longer valid, the customer may not be able to properly document or manage his accounting. Therefore, an improved system that keeps digital receipts of commercial transactions would be beneficial.

# The main goal of this project is to develop and implement a system of digital receipts for local transactions in commercial spaces, thus suppressing the need for paper receipts and reducing costs for business entities. This system will likely include the establishment of a connection between the mobile device of the customer and the fixed terminal in the store, and the transmission and storage of the receipt.

# We are considering different ways of implementing this system. For example, we could use a smart-card (a pocket-sized card with embedded integrated circuits) or a device with a display such as the Open Moko phone or the Gphone.

# The development of this project assumes that the creation of the digital receipt is independent of the means of payment and that the required data will be available to fill the receipt properly. In the final phase of the work, it should be possible to study the impact and acceptance of such a system by the merchants, institutions and customers.

# 

# 

# 

# 

# 

# 

# WORKPLAN

# In the workplan for this project, 4 major assignments were identified:

* Specification of the protocol
* Implementation – development of the protocol
* Evaluation – considering the simplest case of all: one client and one store
* Documentation – write Thesis and Scientific Paper

Implementation could be divided in 3 different stages:

* Development of the communication protocol in a virtual environment,
* Implementation of the protocol over a Bluetooth connection and a Smart Card,
* Understand how this system behaves with multiple clients or/and stores.

# INTRODUCTION

Study of the state of the art in digital invoices, certificates and authentication, and security. Definition of the requirements for a local digital invoice system, in particular in terms of the mobile device used by the customer, the fixed device in the commercial establishment, and the distributed invoice system. Development of the on-site exchange protocol of fiscal information between client and seller (i.e. in the commercial establishment and at the time of purchase). Study invoice upload options for the mobile device and the Internet. Implementation of the protocol in the client and vendor devices, focusing on the digital signatures and the own communication channels taking into account the possibility of the proximity of several client devices and devices of establishment. Evaluation of the proposed solution and the prototype, in particular in terms of security (robustness to various attacks, eg man-in-the middle) and performance (transactions per minute, for example)

# SOFTWARE REQUIREMENTs

* **PYTHON**

Python is an interpreted, high-level, general purpose programming language. Created by Guido Van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readibility, notably using significant whitespaces. It provides constructs that enable clear programming on both small and large scales. Van Rossum led the language community until stepping down as leader in July 2018.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigm, including object oriented, imperative, functional and procedural. It also has a comprehensive standard library.

Python interpreters are available for many operating system. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of Python's other implementations. Python and CPython are managed by the non-profit Python Software Foundation.

* **Tkinter**

Tkinter is a Python Binding to the Tk GUI toolkit. It is the standard Python interface to the Tk GUI toolkit, and is Python's *de facto* standard GUI.Tkinter is included with standard Linux, Microsoft Windows and Mac OSx installs of Python.

The name *Tkinter* comes from *Tk interface*. Tkinter was written by Fredrik Lundh.

Tkinter is free software released under a Python license.

As with most other modern Tk bindings, Tkinter is implemented as a Python wrapper around a complete Tcl interpreter embedded in the Python interpreter. Tkinter calls are translated into Tcl commands which are fed to this embedded interpreter, thus making it possible to mix Python and Tcl in a single application.

Python 2.7 and Python 3.1 incorporate the "themed Tk" ("ttk") functionality of Tk 8.5. This allows Tk widgets to be easily themed to look like the native desktop environment in which the application is running, thereby addressing a long-standing criticism of Tk (and hence of Tkinter).

There are several popular GUI library alternatives available, such aswxPython,PyQT, Pygame, Pyglet, and PyGTK.

# CODE

from tkinter import\*

import random

import time

import datetime

root=Tk()

root.geometry("1600x8000")

root.title("Restaurant Management System")

Tops=Frame(root, width=1200,relief=SUNKEN)

Tops.pack(side=TOP)

f1=Frame(root,width=800,height=700,relief=SUNKEN)

f1.pack(side=LEFT)

localtime=time.asctime(time.localtime(time.time()))

lblInfo=Label(Tops,font=('helvetica',50,'bold'),text="SHRAVAN RESTAURANT ",fg="Black",bd=10,anchor='w')

lblInfo.grid(row=0,column=0)

lblInfo=Label(Tops,font=('arial',20,'bold'),text=localtime,fg="Steel Blue",bd=10,anchor='w')

lblInfo.grid(row=1,column=0)

def Ref():

x=random.randint(10908,500876)

randomRef=str(x)

rand.set(randomRef)

if (FRIES.get()==""):

CoFRIES=0

else:

CoFRIES=float(FRIES.get())

if (NOODLES.get()==""):

CoNOODLES=0

else:

CoNOODLES=float(NOODLES.get())

if (SOUP.get()==""):

CoSOUP=0

else:

CoSOUP=float(SOUP.get())

if (MISAL.get()==""):

CoMISAL=0

else:

CoMISAL=float(MISAL.get())

if (SANDWICH.get()==""):

CoSANDWICH=0

else:

CoSANDWICH=float(SANDWICH.get())

if (DRINKS.get()==""):

CoD=0

else:

CoD=float(DRINKS.get())

CostofFRIES =CoFRIES \* 140

CostofDRINKS=CoD \* 65

CostofNOODLES = CoNOODLES\* 90

CostofSOUP = CoSOUP \* 140

CostMISAL = CoMISAL\* 260

CostSANDWICH=CoSANDWICH \* 300

CostofMeal= "Rs", str('%.2f' % (CostofFRIES+CostofDRINKS+CostofNOODLES+CostofSOUP+CostMISAL+CostSANDWICH))

PayTax=((CostofFRIES+CostofDRINKS+CostofNOODLES+CostofSOUP+CostMISAL+CostSANDWICH) \* 0.2)

TotalCost=(CostofFRIES+CostofDRINKS+CostofNOODLES+CostofSOUP+CostMISAL+CostSANDWICH)

Ser\_Charge= ((CostofFRIES+CostofDRINKS+CostofNOODLES+CostofSOUP+CostMISAL+CostSANDWICH)/99)

Service = "Rs", str ('%.2f' % Ser\_Charge)

OverAllCost ="Rs", str ('%.2f' % (PayTax+TotalCost+Ser\_Charge))

PaidTax= "Rs", str ('%.2f' % PayTax)

Service\_Charge.set(Service)

Cost.set(CostofMeal)

Tax.set(PaidTax)

SubTotal.set(CostofMeal)

Total.set(OverAllCost)

def qExit():

root.destroy()

def Reset():

rand.set("")

FRIES.set("")

NOODLES.set("")

SOUP.set("")

SubTotal.set("")

Total.set("")

Service\_Charge.set("")

DRINKS.set("")

Tax.set("")

Cost.set("")

MISAL.set("")

SANDWICH.set("")

rand = StringVar()

FRIES=StringVar()

NOODLES=StringVar()

SOUP=StringVar()

SubTotal=StringVar()

Total=StringVar()

Service\_Charge=StringVar()

DRINKS=StringVar()

Tax=StringVar()

Cost=StringVar()

MISAL=StringVar()

SANDWICH=StringVar()

lblReference= Label(f1, font=('arial', 16, 'bold'),text="Reference",bd=16,anchor="w")

lblReference.grid(row=0, column=1)

txtReference=Entry(f1, font=('arial',16,'bold'),textvariable=rand,bd=10,insertwidth=4,bg="powder blue",justify='right')

txtReference.grid(row=0,column=2)

lblFRIES= Label(f1, font=('arial', 16, 'bold'),text="FRIES",bd=16,anchor="w")

lblFRIES.grid(row=1, column=1)

txtFRIES=Entry(f1, font=('arial',16,'bold'),textvariable=FRIES,bd=10,insertwidth=4,bg="powder blue",justify='right')

txtFRIES.grid(row=1,column=2)

lblNOODLES= Label(f1, font=('arial', 16, 'bold'),text="NOODLES",bd=16,anchor="w")

lblNOODLES.grid(row=2, column=1)

txtNOODLES=Entry(f1, font=('arial',16,'bold'),textvariable=NOODLES,bd=10,insertwidth=4,bg="powder blue",justify='right')

txtNOODLES.grid(row=2,column=2)

lblSOUP= Label(f1, font=('arial', 16, 'bold'),text="SOUP",bd=16,anchor="w")

lblSOUP.grid(row=3, column=1)

txtSOUP=Entry(f1, font=('arial',16,'bold'),textvariable=SOUP,bd=10,insertwidth=4,bg="powder blue",justify='right')

txtSOUP.grid(row=3,column=2)

lblMISAL= Label(f1, font=('arial', 16, 'bold'),text="MISAL",bd=16,anchor="w")

lblMISAL.grid(row=4, column=1)

txtMISAL=Entry(f1, font=('arial',16,'bold'),textvariable=MISAL,bd=10,insertwidth=4,bg="powder blue",justify='right')

txtMISAL.grid(row=4,column=2)

lblSANDWICH= Label(f1, font=('arial', 16, 'bold'),text="SANDWICH",bd=16,anchor="w")

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

txtSANDWICH=Entry(f1, font=('arial',16,'bold'),textvariable=SANDWICH,bd=10,insertwidth=4,bg="powder blue",justify='right')

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

lblDRINKS= Label(f1, font=('arial', 16, 'bold'),text="DRINKS",bd=16,anchor="w")

lblDRINKS.grid(row=0, column=3)

txtDRINKS=Entry(f1, font=('arial',16,'bold'),textvariable=DRINKS,bd=10,insertwidth=4,bg="powder blue",justify='right')

txtDRINKS.grid(row=0,column=4)

lblCost= Label(f1, font=('arial', 16, 'bold'),text="Cost of Meal",bd=16,anchor="w")

lblCost.grid(row=1, column=3)

txtCost=Entry(f1, font=('arial',16,'bold'),textvariable=Cost,bd=10,insertwidth=4,bg="powder blue",justify='right')

txtCost.grid(row=1,column=4)

lblService= Label(f1, font=('arial', 16, 'bold'),text="Service Charge",bd=16,anchor="w")

lblService.grid(row=2, column=3)

txtService=Entry(f1, font=('arial',16,'bold'),textvariable=Service\_Charge,bd=10,insertwidth=4,bg="powder blue",justify='right')

txtService.grid(row=2,column=4)

lblStateTax= Label(f1, font=('arial', 16, 'bold'),text="State Tax",bd=16,anchor="w")

lblStateTax.grid(row=3, column=3)

txtStateTax=Entry(f1, font=('arial',16,'bold'),textvariable=Tax,bd=10,insertwidth=4,bg="powder blue",justify='right')

txtStateTax.grid(row=3,column=4)

lblSubTotal= Label(f1, font=('arial', 16, 'bold'),text="Sub Total",bd=16,anchor="w")

lblSubTotal.grid(row=4, column=3)

txtSubTotal=Entry(f1, font=('arial',16,'bold'),textvariable=SubTotal,bd=10,insertwidth=4,bg="powder blue",justify='right')

txtSubTotal.grid(row=4,column=4)

lblTotalCost= Label(f1, font=('arial', 16, 'bold'),text="Total Cost",bd=16,anchor="w")

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

txtTotalCost=Entry(f1, font=('arial',16,'bold'),textvariable=Total,bd=10,insertwidth=4,bg="powder blue",justify='right')

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

btnTotal=Button(f1,padx=16,pady=8,bd=16,fg="black",font=('arial',16,'bold'),width=10,text="Total",bg="powder blue",command=Ref).grid(row=7,column=2)

btnReset=Button(f1,padx=16,pady=8,bd=16,fg="black",font=('arial',16,'bold'),width=10,text="Reset",bg="powder blue",command=Reset).grid(row=7,column=3)

btnExit=Button(f1,padx=16,pady=8,bd=16,fg="black",font=('arial',16,'bold'),width=10,text="Exit",bg="powder blue",command=qExit).grid(row=7,column=4)

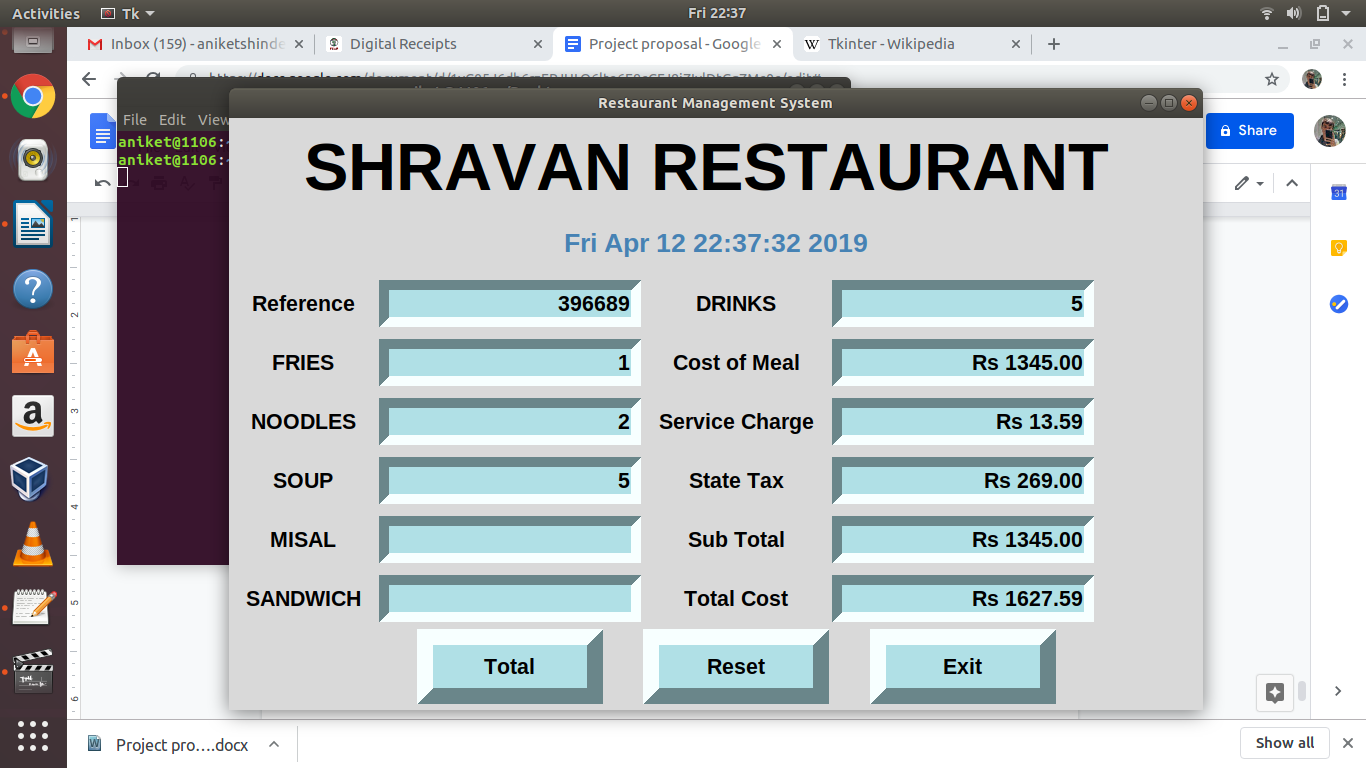
root.mainloop()

# OUTPUT

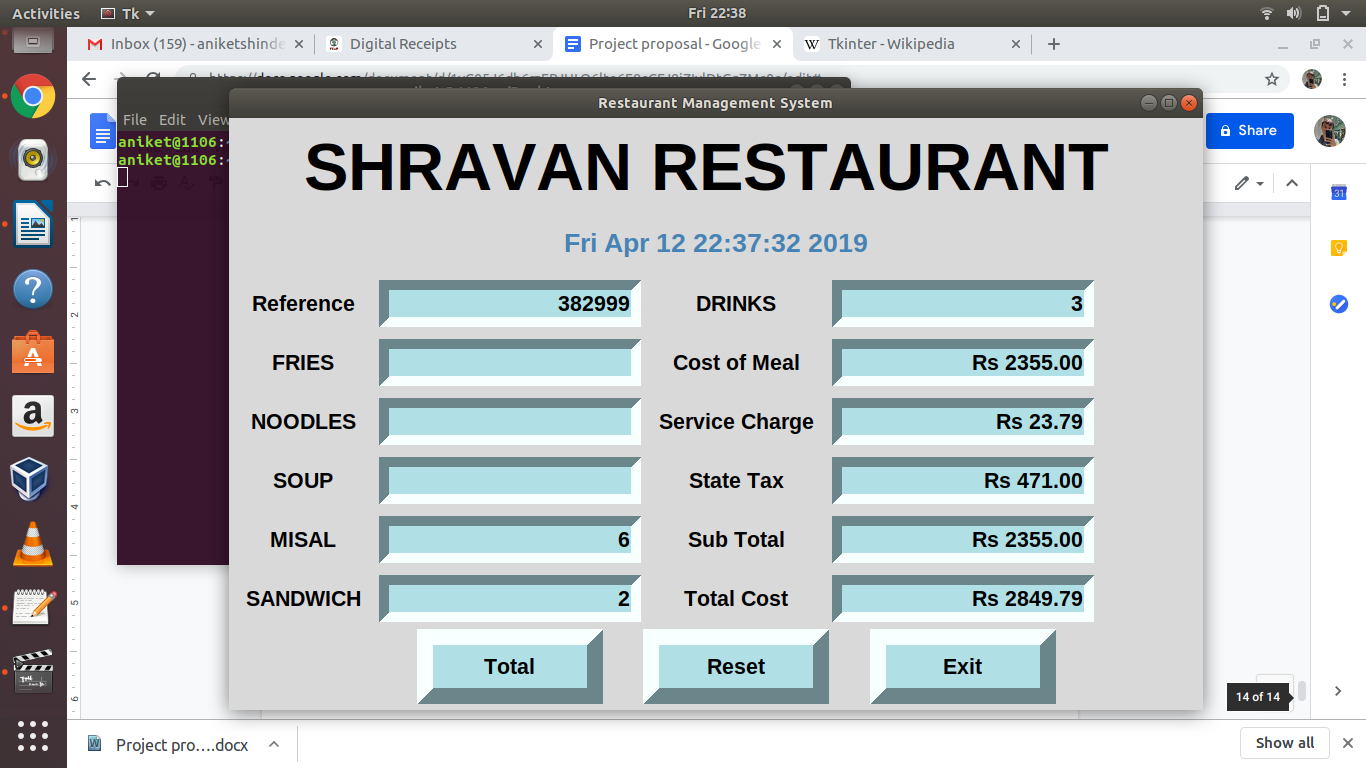
**NORMAL WINDOW**

# 

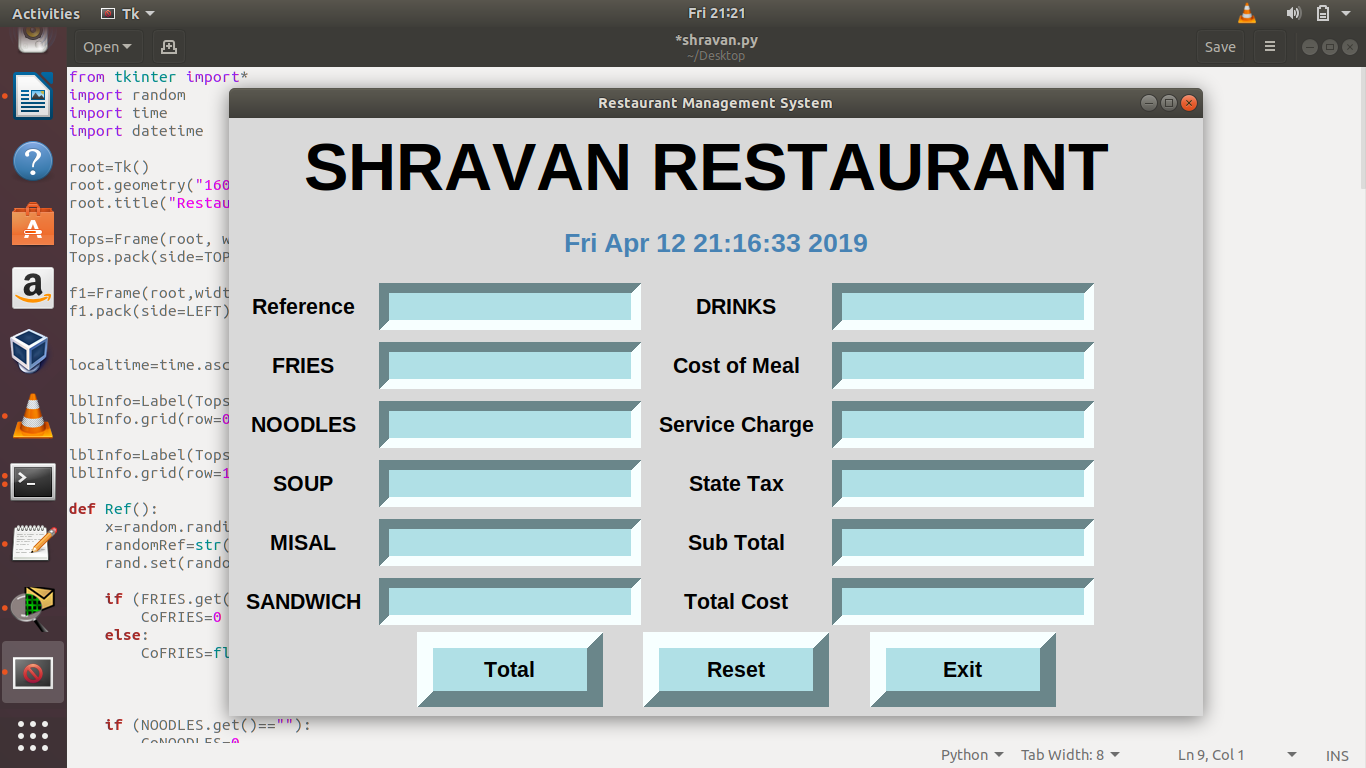
# ORDER 1



**ORDER 2**



**AFTER RESETING**



# 