PSDL Mini project writeup

C.no – UCE2021531, UCE2021532, UCE2021533, UCE2021535

Member names: Shweta Katkar, Gunjan Khatari, Bhakti Khokad, Prerna Khotari

Topic name: Restaurant billing system

1) Problem Statement:

To create a restaurant management system which helps the restaurant manager to manage the restaurant more effectively and efficiently by computerising meal ordering, billing and inventory control.

2) Abstract:

Traditionally the method in which customers specify their desired menu to the waiter who takes the order on a paper. Personally, he then takes the order to the kitchen department and then he supply the food item to the customer. So, it was a time-consuming process. It leads to wastage of paper and also it requires reprinting of all menu cards. Also, in many cases for small change to be making in menu card it is not convenient to print all menu cards again and again. Simply saying that the menu card once printed can't be changed. After some days, the menu card lost its worthy look and attractiveness.

Many of the hotel are managed their workflow and services by paper work and manually which take time and high budget for management. busy day of people schedule and value of time become important day by day so Smart Restaurant handle the system as booking and ordering by digitally which become save manpower and time of staff of hotel. Generating KOT by application is easy to save paper, staff and time. The hotel are services and facialized in a traditional way,

from waiters to kitchen also on account. More recently, this system is one of the major a problem for hotel to manage their manpower and guests.

3) Module wise description:

- import sys: It lets us access system-specific parameters and functions, import sys. First, we have to import the sys module in our program before running any functions. sys.modules.
- import os: Python os system function allows us to run a command in the Python script, just like if I was running it in my shell.
- from tkinter import *: if we want to define every widget by importing all the functions and modules in it, then we can use "from tkinter import *" method.
- import math ,random: Use a random. randint() function to get a random integer number from the inclusive range. For example, random. randint(0, 10) will return a random number from [0, 1, 2, 3, 4, 5, 6, 7, 8 ,9, 10].
- from tkinter import messagebox: The tkMessageBox module is used to display message boxes in your applications. This module provides a number of functions that you can use to display an appropriate message. Some of these functions are showinfo, showwarning, showerror, askquestion, askokcancel, askyesno, and askretryignore.
- from tkinter import ttk: ttk is a module that is used to style the tkinter widgets. Just like CSS is used to style an HTML element, we use tkinter. ttk to style tkinter widgets.
- import datetime: imports all the content from the datetime module, but requires you to precede names that are imported from that module with the datetime. qualifier in your code.
- import time: The Python time module provides many ways of representing time in code, such as objects, numbers, and strings. It also provides functionality other than representing time, like waiting during code execution and measuring the efficiency of your code.

4) Technology selected and technology features:

➢ **Tkinter Programming:**

- Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.
- Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps:

  – Example:Import the Tkinter module.

  – Create the GUI application main window.

  – Add one or more of the above mentioned widgets to the GUI application.

  – Enter the main event loop to take action against each event triggered by the user.

➢ **Features:**

- Layered approach:

  The layered approach used in designing Tkinter gives Tkinter all of the advantages of the TK library. Therefore, at the time of creation, Tkinter inherited from the benefits of a GUI toolkit that had been given time to mature. This makes early versions of Tkinter a lot more stable and reliable than if it had been rewritten from scratch. Moreover, the conversion from Tcl/Tk to Tkinter is really trivial, so that Tk programmers can learn to use Tkinter very easily.

- Accessibility

  Learning Tkinter is very intuitive, and therefore quick and painless. The Tkinter implementation hides the detailed and complicated calls in simple, intuitive methods. This is a continuation of the Python way of thinking, since the language excels at quickly building prototypes. It is therefore expected that its preferred GUI library be implemented using the same approach. For example, here is the code for a typical “Hello world”-like application:

```
 from Tkinter import *

root = Tk( )

root.title("A simple application")

root.mainloop( )
```

The first 2 lines allow me to create a complete window. Compared to MFC programming, it makes no doubt that Tkinter is simple to use. The third line sets the caption of the window, and the fourth one makes it enter its event loop.

- Portability

  Python scripts that use Tkinter do not require modifications to be ported from one platform to the other. Tkinter is available for any platform that Python is implemented for, namely Microsoft Windows, X Windows, and Macintosh. This gives it a great advantage over most competing libraries, which are often restricted to one or two platforms. Moreover, Tkinter will provide the native look-and-feel of the specific platform it runs on.

- Availability

  Tkinter is now included in any Python distribution. Therefore, no supplementary modules are required in order to run scripts using Tkinter.

➢ Major widgets

1. **Button**:To add a button in your application, this widget is used.
2. Canvas: It is used to draw pictures and other complex layout like graphics, text and widgets.

3. **CheckButton:** To select any number of options by displaying a number of options to a user as toggle buttons.
4. **Entry:**It is used to input the single line text entry from the user.. For multi-line text input, Text widget is used.
5. **Frame:** It acts as a container to hold the widgets. It is used for grouping and organising the widgets.
6. **Label**: It refers to the display box where you can put any text or image which can be updated any time as per the code.
7. **Listbox**: It offers a list to the user from which the user can accept any number of options.
8. **MenuButton**: It is a part of top-down menu which stays on the window all the time. Every menubutton has its own functionality.
9. **Menu**: It is used to create all kinds of menus used by the application.
10. **Message**: It refers to the multi-line and non-editable text. It works same as that of Label.
11. **RadioButton:** It is used to offer multi-choice option to the user. It offers several options to the user and the user has to choose one option.
12. **Scale:** It is used to provide a graphical slider that allows to select any value from that scale.
13. **Scrollbar**: It refers to the slide controller which will be used to implement listed widgets
14. **Text:** To edit a multi-line text and format the way it has to be displayed.
15. **TopLevel:** This widget is directly controlled by the window manager. It don’t need any parent window to work on.

5) Reference: Let Us Python - 5th Edition Yashavant Kanetkar, Aditya Kanetkar