Universidad Interamericana de Puerto Rico

Recinto de Bayamón

Escuela de Ingeniería
Departamento de Ingeniería Eléctrica y de Computadora

COEN 2220 – Advanced Programming

Proyecto final

Nombre de estudiante	:	Mireya Vázquez Robles – Y00549598
Nombre de experiencia	:	Aplicación de información de comisiones de arte
Percent of tasks completed (0 - 100 %)	:	100%
Date	:	11/25/2021

1. Especificaciones del programa:

a. Tareas que la aplicación va a desarrollar

Este proyecto consiste en dos archivos de secuencia de comandos de Python.

El primer archivo:

1. Ejecutará el segundo archivo.
2. Medirá el tiempo que toma en ejecutar el segundo archivo.
3. Imprimirá un mensaje.

El segundo archivo:

1. Abrirá una ventana que mostrará una página de inicio con un menú.
2. En el menú habrá botones que conducen a la galería, a información de comisiones de arte, a los términos y condiciones y a estimados.
3. En la galería habrá imágenes y se podrá navegar con una barra de desplazamiento.
4. En la página de información de comisiones habrá información de precios, adiciones e información de contacto y se podrá navegar con una barra de desplazamiento.
5. En la página de términos y condiciones habrá información de los términos y condiciones y se podrá navegar con una barra de desplazamiento.
6. En la página de estimados habrá botones y cajas en donde marcar o ingresar información de pedido.
7. La aplicación podrá calcular el precio total de la comisión y el tiempo de espera que tomará completar la misma en base a la información de pedido ingresada por el usuario.
8. La aplicación mostrará un mensaje en una nueva ventana con los resultados de los estimados.

b. Diagrama de bloques (entradas y salidas)

El programa recibe input del mouse y del teclado y produce un output visible por el monitor. El usuario puede navegar presionando botones y desplazándose para cambiar la página. El usuario también puede obtener un estimado. Para ello, el usuario ingresa la información de su pedido (tamaño, fondo y cantidad de personajes), la aplicación calcula el precio total y el tiempo de espera en base a la información ingresada y por último muestra un mensaje con los resultados del estimado, ya sea para los bocetos o para las pinturas. Ver Fig.1.

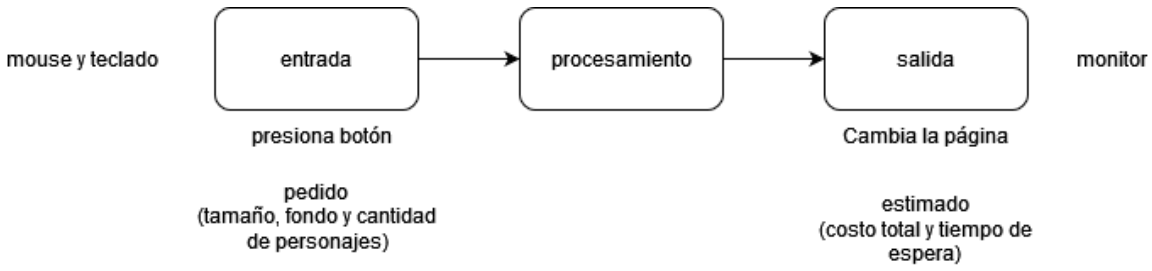

Fig 1. Diagrama de bloques de entradas y salidas

2. Diseño de clases y objetos:

a. Cómo ha identificado las clases, métodos y variables

Las clases fueron identificadas como las distintas partes de la aplicación: base( ), aplicación( ), stats( ), menú( ), inicioinfo( ), galeriainfo( ), cominfo( ), tosinfo( ), estimadosinfo( ), boceto( ) y pintura( ). Las clases que terminan en -info son las que contienen todo lo relacionado a esas páginas.

Los métodos fueron identificados de acuerdo con la acción. Ex ejecuta el programa, operación mide el tiempo que tarda en ejecutar el programa y la otra clase llamada operación imprime un mensaje. Com, estimados, galería y tos son los métodos que esconden y hacen visible los marcos. Selection, selection1 y selection2 determinan cuales fueron las selecciones del usuario. PrecioTotal y tiempoTotal calculan el precio total y el tiempo total que tomará realizar la comisión.

Las variables fueron identificadas como los distintos elementos de la aplicación como botones, etiquetas, textos, marcos, mensajes y otros valores como las selecciones (m, n, c), precio total (p) y tiempo total (t).

b. Diagrama de clases y sus relaciones

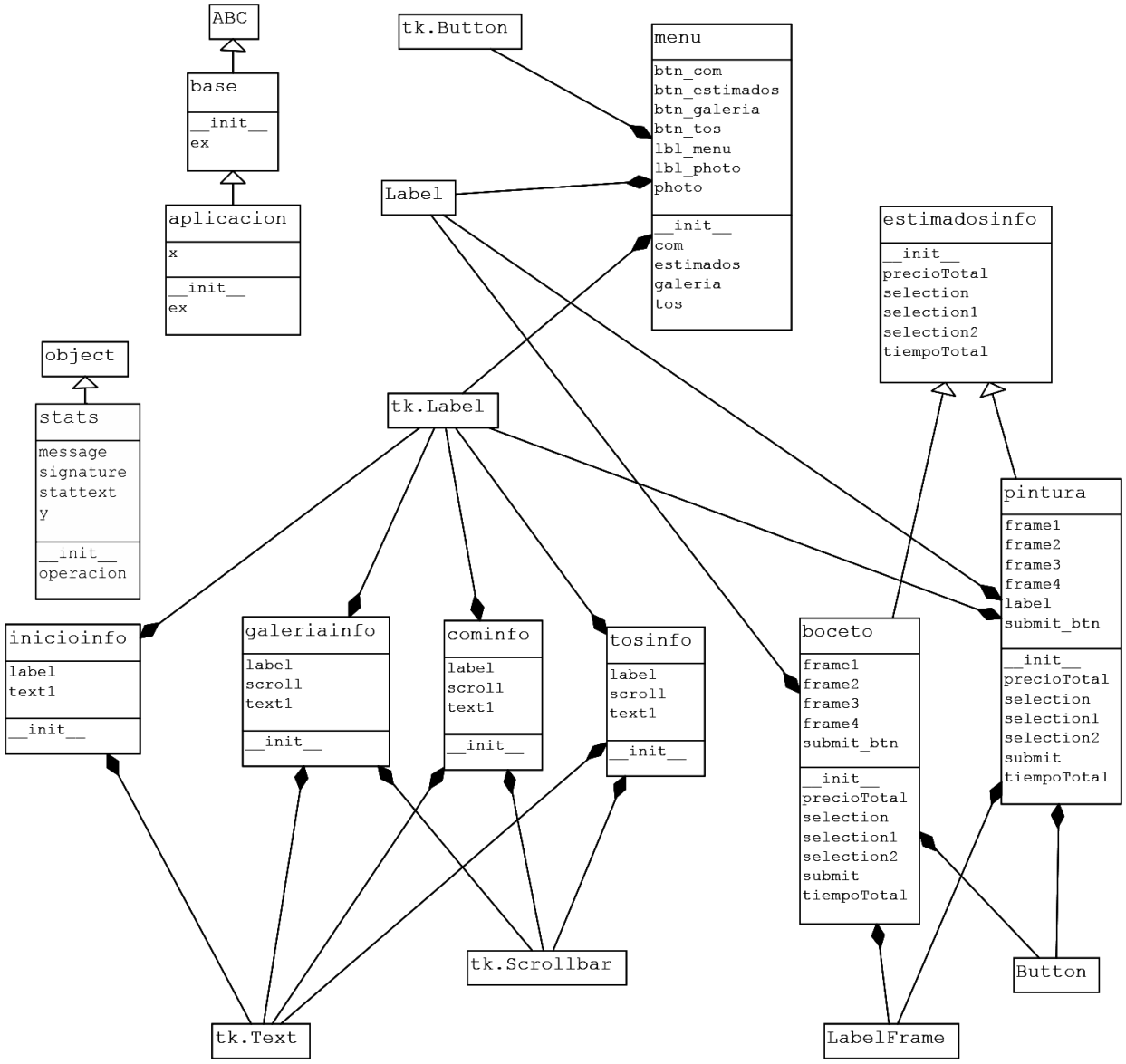

Fig 2. Diagrama de clases y sus relaciones

3. Cumplimiento de los requisitos del proyecto:

a. Variables públicas y privadas

Las variables públicas y privadas se encuentran a través de todo el programa. Algunas de las variables globales lo son aquellas referentes a los marcos y a las fotos como fr_inicio o photo1. La variable privada en esta aplicación es __quote.

b. Variables de Clase y de Instancia

La variable de clase en este caso es stattext dentro de la clase stats( ). Las variables de instancia como self.lbl_menu se encuentran a través de todo el programa dentro de sus respectivas clases y métodos de instancia.

c. Métodos de Clase y de Instancia

Los métodos de clase en este caso lo son las dos clases llamadas operación( ) pertenecientes a la clase stats( ). Los métodos de instancia son los demás métodos como ex( ) o submit( ).

d. Herencia

La herencia ocurre al utilizar una clase padre y clases hijas. En este caso estimados( ) es la clase padre, mientras que boceto( ) y pintura( ) son las clases hijas.

e. Sobre Carga (“Overloading”) de métodos

La sobrecarga ocurre al utilizar dos métodos llamados operación( ) dentro de la clase stats( ). La herramienta “dispatch” hace esto posible.

f. Sobre Escritura(“Overriding”) de métodos

La sobreescritura ocurre al utilizar métodos dentro de la clase boceto( ) y la clase pintura( ) que fueron nombrados igual a los métodos en la clase estimadosinfo( ).

g. Clases Abstractas

La clase abstracta base( ) sirve como plano para la clase aplicación( ).

h. Uso de gráficos

La interfaz gráfica del usuario fue creada utilizando el paquete tkinter y pillow. La aplicación contiene ventanas, marcos, botones, etiquetas, textos, cajas e imágenes.

4. Resultados

a. Explique las pruebas realizadas y resultados obtenidos. Mostrar "screen pictures" de la salida de su programa.

1. Instalar los paquetes: tkinter, pillow y multipledispatch.

2. Abrir el proyecto en Spyder IDE.

3. Correr el script " base.py". La aplicación deberá de abrir en la página de inicio.

4. Presionar el botón de galería. La página deberá cambiar y mostrar la galería.

5. Utilizar la barra de desplazamiento para ver las demás imágenes en la galería.

6. Presionar el botón de Información. La página deberá cambiar y mostrar la información de comisiones como precios y adiciones.

7. Utilizar la barra de desplazamiento para leer toda la información.

8. Presionar el botón de Términos y Condiciones. La página deberá cambiar y mostrar los términos y condiciones.

9. Utilizar la barra de desplazamiento para leer todos los términos y condiciones.

10. Presionar el botón de Estimados. La página deberá de cambiar y mostrar una serie de cajas para seleccionar tamaño y tipo de fondo y escribir la cantidad de personajes y botones de "submit" tanto para bocetos como para pinturas.

11. Seleccionar el tamaño, tipo de fondo y escribir la cantidad de personajes para bocetos y presionar el botón de "submit". Una nueva ventana deberá aparecer con un mensaje que dice la información de pedido, el precio total y el tiempo total de espera.

12. Presionar "ok" para cerrar la ventana con el mensaje.

13. Repetir el proceso de selección con todas las opciones para bocetos. De esta manera se podrá asegurar que el programa funciona como debido.

14. Repetir el proceso de selección con todas las opciones, pero para las pinturas.

15. Cerrar la aplicación precionando la "x".

16. Ver la consola para obtener el tiempo que de ejecución del programa y un mensaje.

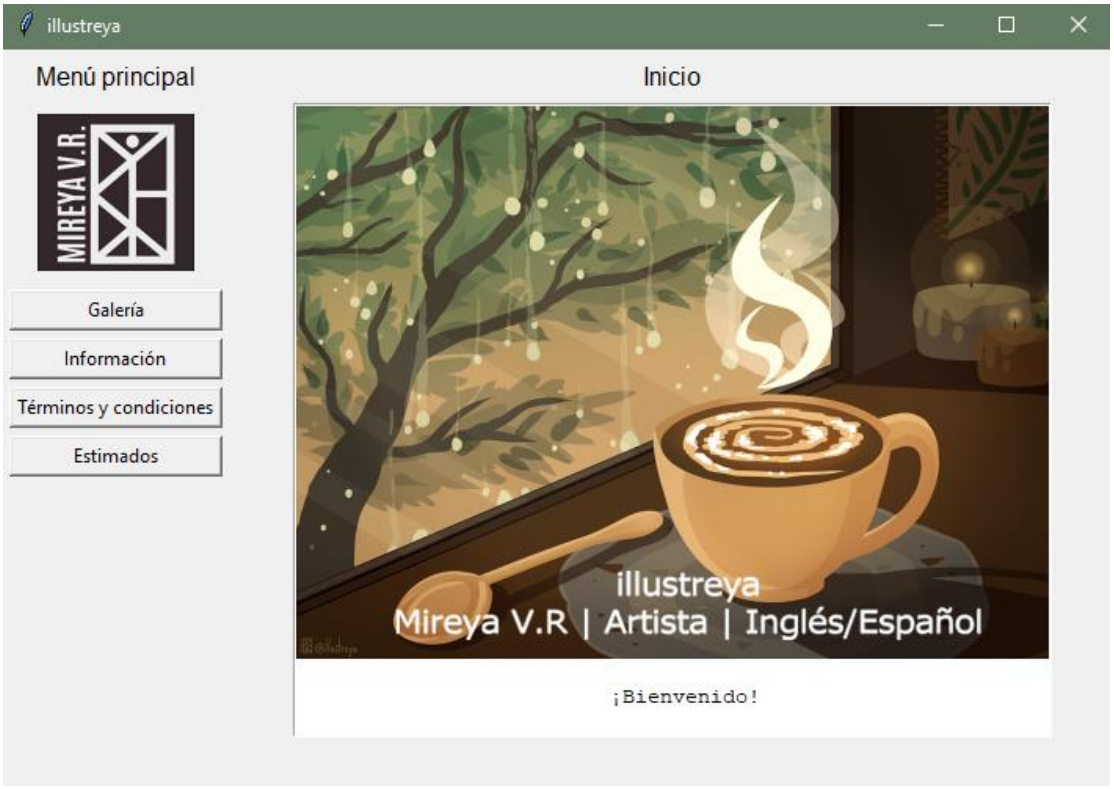

Fig 3. Página de inicio

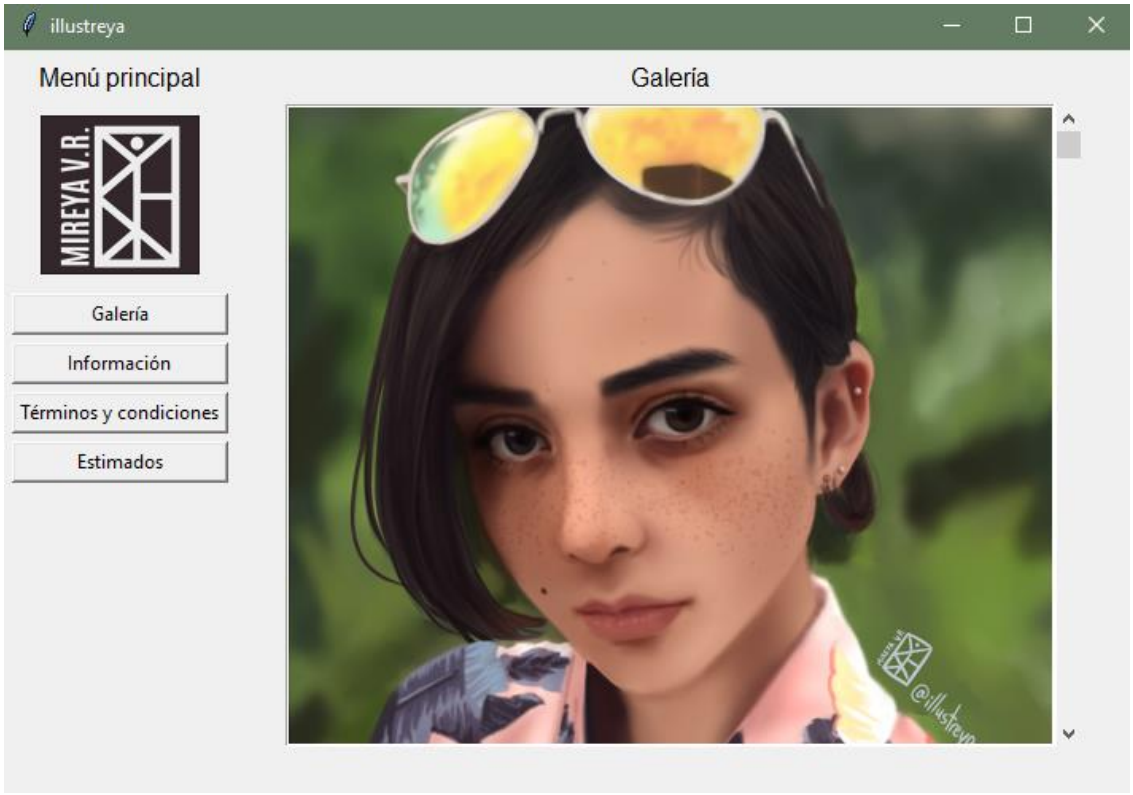

Fig 4. Página de galería

Fig 5. Página de información de comisiones

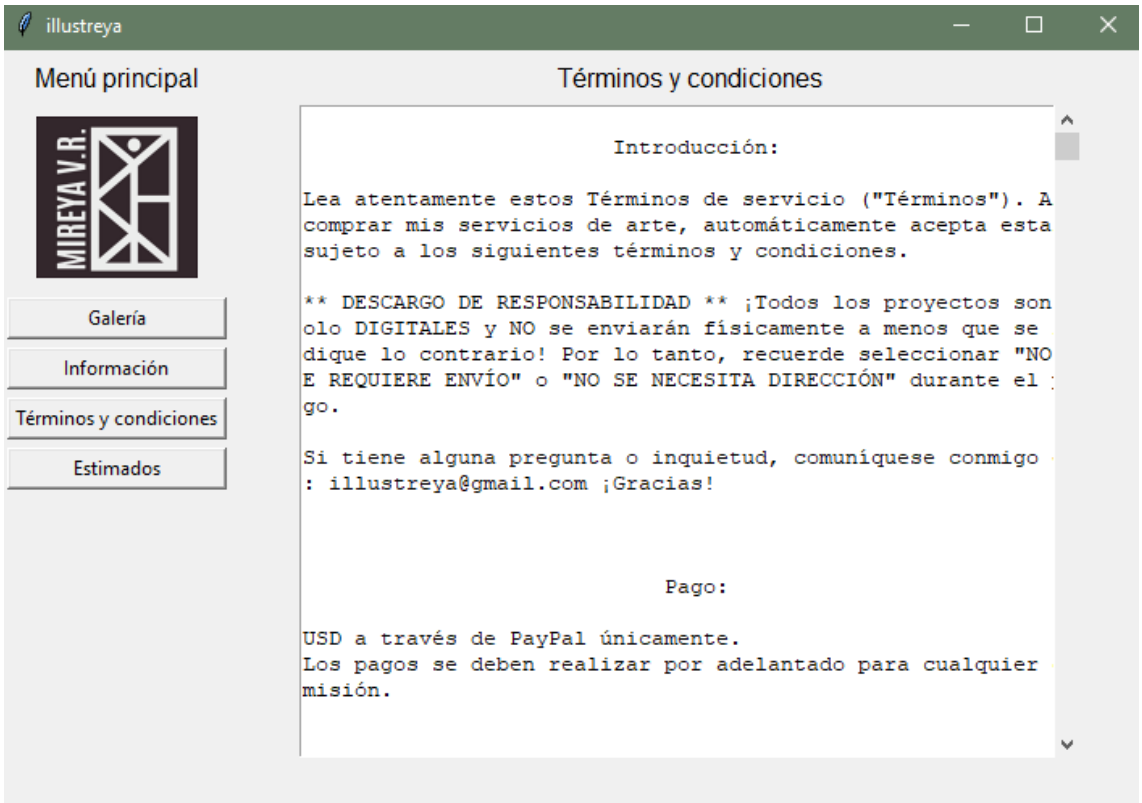

Fig 6. Página de términos y condiciones

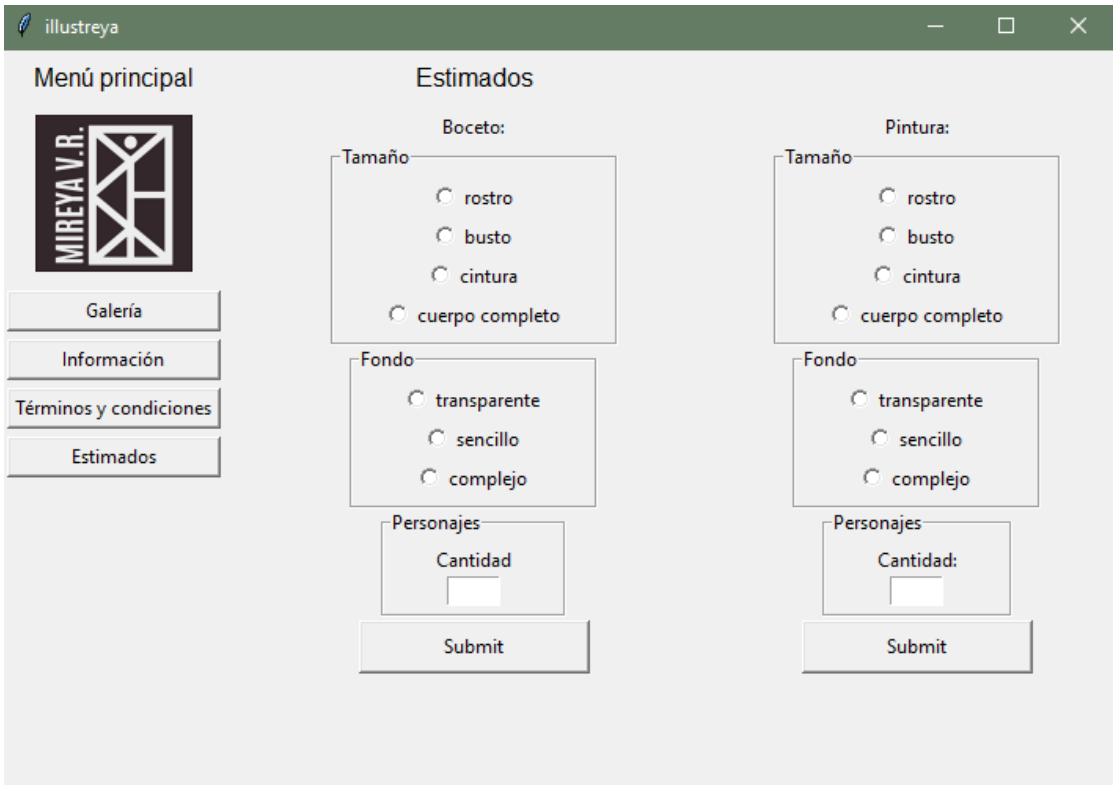

Fig 7. Página de estimados

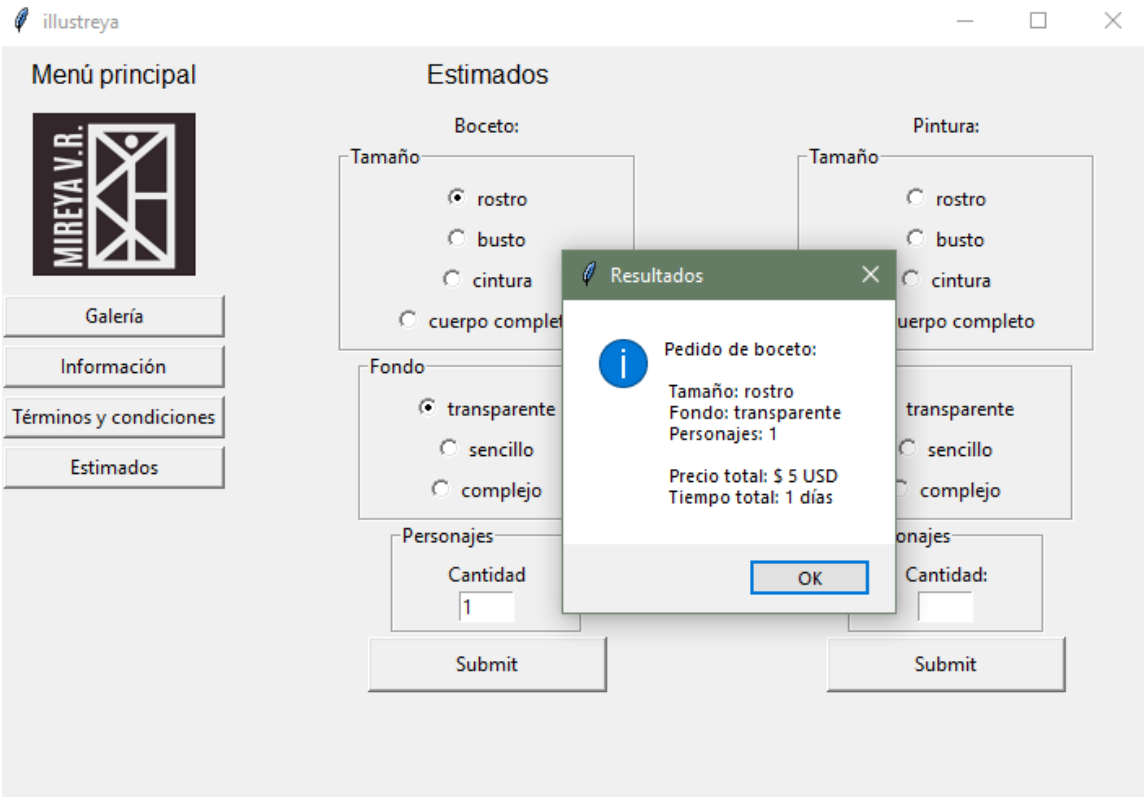

Fig 8. Estimado de boceto

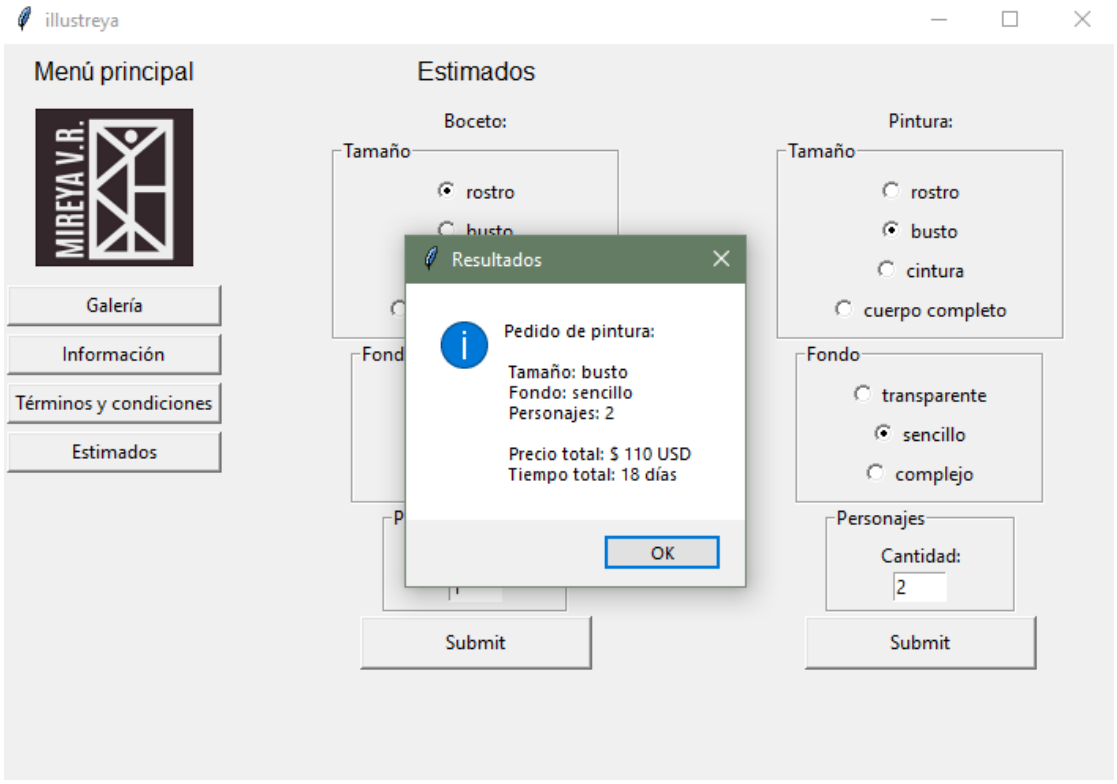

Fig 9. Estimado de pintura

```
Runtime:

The time of execution of this program is : 202.41070818901062

Thank you for using this program  -Mireya Vázquez
```

Fig 10. Tiempo de ejecución del programa

b. Tabla con la lista de cada archivo y su descripción

Tabla 1. Lista de archivos

Nombre de archivo	Descripción
aplicacion.py	Segundo archivo de secuencia de comandos de Python.
base.py	Primer archivo de secuencia de comandos de Python.
instrucciones.txt	Archivo que contiene las instrucciones para correr el programa.
photo.png	Imagen del logo ubicado en el menu.
photo1.png	Imagen ubicada en la página de inicio.
photo2.png	Imagen ubicada en la galería de la primera pintura.
photo3.png	Imagen ubicada en la galería de la segunda pintura.
photo4.png	Imagen ubicada en la galería de la tercera pintura.
photo5.png	Imagen ubicada en la galería de la cuarta pintura.
photo6.png	Imagen ubicada en la galería de la quinta pintura.
photo7.png	Imagen ubicada en la galería de la sexta pintura.
photo8.png	Imagen ubicada en la galería del primer boceto.
photo9.png	Imagen ubicada en la galería del segundo boceto.
photo10.png	Imagen ubicada en la galería del tercer boceto.
photo11.png	Imagen ubicada en la galería del cuarto boceto.

5. Conclusiones

a. Describir todos los inconvenientes que se tuvieron en el transcurso del desarrollo del programa y las soluciones empleadas.

Los inconvenientes enfrentados en el transcurso del desarrollo del programa fueron el lograr cumplir con todos los requisitos, ubicar los elementos e imágenes, lograr que los botones funcionaran y lograr que las páginas cambiaran. Para obtener las soluciones se realizaron búsquedas en internet. Para lograr cumplir con todos los requisitos comencé por el más dificil, el uso de gráficos. Una vez tenía la mayoría del GUI entonces me enfoqué en organizarlo todo dentro de clases y métodos. Luego, los reestructuré para que hubiera clases, métodos y variables de todo tipo y añadí nuevas funciones para cumplir con los requisitos que faltaban. Para ubicar los elementos e imágenes instalé tkinter y pillow. Para que los botones funcionaran y lograr que las páginas cambiaran cree métodos que son llamados al presionar los botones. En estos métodos se hace visible el marco con la información que se quiere que se vea y se esconden los que no, así es como se cambian las páginas.

b. Indicar lo alcanzado en el proyecto versus lo esperado inicialmente (objetivos).

Todos los objetivos fueron alcanzados y considero que el resultado final del proyecto superó mis expectativas. Inicialmente la idea era crear un programa que contuviera la galería, información de las comisiones, los términos y condiciones y que pudiera calcular y mostrar los estimados. Sin embargo, hubo ideas que surgieron según lo trabajaba como el medir el tiempo de ejecución del programa y la adición de la barra de desplazamiento que resultaron ser buenos aditamentos. Logré aplicar todo lo aprendido en clase y aprender por encima de ello, particularmente el tema de la interfaz gráfica del usuario. Con este proyecto he obtenido la experiencia de desarrollar una aplicación desde cero.

6. Referencias y Apéndices

a. Referencias

Kumar, B., & KumarEntrepreneur, B. (2021, June 10). *Python tkinter text box widget + examples*. Python Guides. Retrieved November 25, 2021, from https://pythonguides.com/python-tkinter-text-box/.

Kumar, B., & KumarEntrepreneur, B. (2021, October 6). *Python tkinter frame*. Python Guides. Retrieved November 25, 2021, from https://pythonguides.com/python-tkinter-frame/.

M, R. (2021, September 21). *How to add images in Tkinter - using the python pillow package*. ActiveState. Retrieved November 25, 2021, from https://www.activestate.com/resources/quick-reads/how-to-add-images-in-tkinter/.

Python tkinter - entry widget. GeeksforGeeks. (2021, February 1). Retrieved November 25, 2021, from https://www.geeksforgeeks.org/python-tkinter-entry-widget/.

Python tkinter course. GUI Programming with Python: Checkboxes in Tkinter. (n.d.). Retrieved November 25, 2021, from https://python-course.eu/tkinter_checkboxes.php.

Python tkinter course. GUI Programming with Python: Text Widget. (n.d.). Retrieved November 25, 2021, from https://python-course.eu/tkinter_text_widget.php.

Reading images with Python - Tkinter. GeeksforGeeks. (2021, October 20). Retrieved November 25, 2021, from https://www.geeksforgeeks.org/reading-images-with-python-tkinter/?ref=lbp.

Real Python. (2021, April 3). *Python GUI programming with Tkinter*. Real Python. Retrieved November 25, 2021, from https://realpython.com/python-gui-tkinter/#assigning-widgets-to-frames-with-frame-widgets.

Real Python. (2021, April 3). *Python GUI programming with Tkinter*. Real Python. Retrieved November 25, 2021, from https://realpython.com/python-gui-tkinter/#displaying-text-and-images-with-label-widgets.

Real Python. (2021, April 3). *Python GUI programming with Tkinter*. Real Python. Retrieved November 25, 2021, from https://realpython.com/python-gui-tkinter/#getting-user-input-with-entry-widgets.

YouTube. (2020). *Menu Bars With tKinter - Python Tkinter Gui Tutorial #46. YouTube*. Retrieved November 25, 2021, from https://www.youtube.com/watch?v=ZS2_v_zsPTg.

YouTube. (2020). *Using Frames With Menus - Python Tkinter Gui Tutorial #47. YouTube*. Retrieved November 25, 2021, from https://www.youtube.com/watch?v=1cWWiXU02-g.

b. Source code

#base.py

```
# -*- coding: utf-8 -*-
"""
@author: yeya
"""

import tkinter as tk
from tkinter import *
from tkinter import messagebox
from PIL import ImageTk,Image

import time
from abc import ABC, abstractmethod
from multipledispatch import dispatch

class base(ABC):
    @abstractmethod
    def __init__(self):
        pass
    @abstractmethod
    def ex(self):
        pass



class aplicacion(base):
```

```
    def __init__(self,x):
        self.x = x

    def ex(self):

        #execute the program
        exec(compile(open( self.x , "rb").read(), self.x, 'exec'))



class stats(object):
    stattext = "Runtime:"
    def __init__(self,y,message,signature):
        self.y = y
        self.message = message
        self.signature = signature

    @classmethod
    @dispatch(type,str)
    def operacion(cls,y):

    #compute the time of execution
        start = time.time()
        Obj1 = aplicacion('aplicacion.py' )
        Obj1.ex()

        end = time.time()
        print("\n",y, end-start)

    @classmethod
```

```
    @dispatch(type,str,str)
    def operacion(cls,message,signature):

        #print final message
        print("\n",message,signature)

print("\n",stats.stattext)
stat = stats.operacion("The time of execution of this program is :")
stat = stats.operacion("Thank you for using this program ","-Mireya Vázquez")
```

#Aplicacion.py

```
# -*- coding: utf-8 -*-
"""
Created on Sat Nov  6 11:53:01 2021

@author: yeya
"""


import tkinter as tk
from tkinter import *
from tkinter import messagebox
from PIL import ImageTk,Image

window = tk.Tk()
window.title("illustreya")
window.rowconfigure(1, minsize=1, weight=1)
window.columnconfigure(1, minsize=1, weight=1)
```

```
class menu():

    def __init__(self,master):

        #frame
        fr_menu = tk.Frame(master)
        fr_menu.grid(row=0,rowspan=3, column=0, sticky="ns")

        #label
        self.lbl_menu = tk.Label(fr_menu,text="Menú principal",font='bold')

        self.lbl_menu.grid(row=0, column=0, sticky = "ew",padx=5, pady=5)

        #logo
        self.photo = ImageTk.PhotoImage(Image.open("photo.png"))
        self.lbl_photo = Label(fr_menu,image=self.photo)
        self.lbl_photo.grid(row=1, column=0, sticky = "ew",padx=5, pady=5)

       # buttons
        self.btn_galeria = tk.Button(fr_menu, text="Galería", command=self.galeria)
        self.btn_com = tk.Button(fr_menu, text="Información", command=self.com)
        self.btn_tos = tk.Button(fr_menu, text="Términos y condiciones", command=self.tos)
        self.btn_estimados = tk.Button(fr_menu, text="Estimados", command=self.estimados)

        self.btn_galeria.grid(row=2, column=0, sticky="ew", padx=5, pady=5)
        self.btn_com.grid(row=3, column=0, sticky="ew", padx=5)
        self.btn_tos.grid(row=4, column=0, sticky="ew", padx=5, pady=5)
        self.btn_estimados.grid(row=5, column=0, sticky="ew", padx=5)
```

```
def galeria(self):
    # #Hide all frames except the one for galeria
    fr_inicio.grid_forget()
    fr_cominfo.grid_forget()
    fr_tosinfo.grid_forget()
    fr_estimadosinfo.grid_forget()
    fr_bocetoinfo.grid_forget()
    fr_pinturainfo.grid_forget()
    fr_galeria.grid(row=0,column=1,sticky="ns")

def com(self):
    #Hide all frames except the one for comisiones
    fr_inicio.grid_forget()
    fr_galeria.grid_forget()
    fr_tosinfo.grid_forget()
    fr_estimadosinfo.grid_forget()
    fr_bocetoinfo.grid_forget()
    fr_pinturainfo.grid_forget()
    fr_cominfo.grid(row=0,rowspan=5,column=1,sticky="ns")

def tos(self):
    #Hide all frames except the one for terminos y condiciones
    fr_inicio.grid_forget()
    fr_galeria.grid_forget()
    fr_cominfo.grid_forget()
    fr_estimadosinfo.grid_forget()
    fr_bocetoinfo.grid_forget()
```

```
        fr_pinturainfo.grid_forget()
        fr_tosinfo.grid(row=0,column=1,sticky="ns")

    def estimados(self):
        #hide all frames except the ones for estimados
        fr_inicio.grid_forget()
        fr_galeria.grid_forget()
        fr_cominfo.grid_forget()
        fr_tosinfo.grid_forget()
        fr_estimadosinfo.grid(row=0,column=1,sticky="ns")
        fr_bocetoinfo.grid(row=1,column=1,sticky="ns")
        fr_pinturainfo.grid(row=1,column=2,sticky="ns")

class inicioinfo():

    def __init__(self,master):

        #frame
        global fr_inicio, photo1
        fr_inicio =tk.Frame(master)
        fr_inicio.grid(row=0,column=1,sticky="ns")

        #label
        self.label = tk.Label(fr_inicio,text="Inicio",font='bold')
        self.label.grid(row=0, column=1, sticky="n",padx=5, pady=5)

        #contents
        self.text1 = tk.Text(fr_inicio,height=25,width=60)
        self.text1.grid(row=1,column=1,sticky="n")
```

```
        self.text1.tag_configure("center", justify='center')
        photo1 = tk.PhotoImage(file='photo1.png')
        self.text1.image_create(tk.END, image=photo1)
        self.text1.insert(tk.END, '\n\n                    ¡Bienvenido!')

        self.text1.config(state='disabled')

class galeriainfo():

    def __init__(self,master):

        #frame
        global fr_galeria, photo2, photo3, photo4, photo5, photo6, photo7, photo8, photo9, photo10,
photo11
        fr_galeria =tk.Frame(master)

        #label
        self.label = tk.Label(fr_galeria,text="Galería",font='bold')
        self.label.grid(row=0, column=1, sticky="n",padx=5, pady=5)

        #contents
        self.text1 = tk.Text(fr_galeria,height=25,width=60)
        self.text1.tag_configure("center", justify='center')

        photo2 = tk.PhotoImage(file='photo2.png')
        self.text1.image_create(tk.END, image=photo2)

        photo3 = tk.PhotoImage(file='photo3.png')
        self.text1.insert(tk.END, '\n\n')
```

```
self.text1.image_create(tk.END, image=photo3)

photo4 = tk.PhotoImage(file='photo4.png')
self.text1.insert(tk.END, '\n\n')
self.text1.image_create(tk.END, image=photo4)

photo5 = tk.PhotoImage(file='photo5.png')
self.text1.insert(tk.END, '\n\n')
self.text1.image_create(tk.END, image=photo5)

photo6 = tk.PhotoImage(file='photo6.png')
self.text1.insert(tk.END, '\n\n')
self.text1.image_create(tk.END, image=photo6)

photo7 = tk.PhotoImage(file='photo7.png')
self.text1.insert(tk.END, '\n\n')
self.text1.image_create(tk.END, image=photo7)

photo8 = tk.PhotoImage(file='photo8.png')
self.text1.insert(tk.END, '\n\n')
self.text1.image_create(tk.END, image=photo8)

photo9 = tk.PhotoImage(file='photo9.png')
self.text1.insert(tk.END, '\n\n')
self.text1.image_create(tk.END, image=photo9)

photo10 = tk.PhotoImage(file='photo10.png')
self.text1.insert(tk.END, '\n\n')
self.text1.image_create(tk.END, image=photo10)
```

```
        photo11 = tk.PhotoImage(file='photo11.png')
        self.text1.insert(tk.END, '\n\n')
        self.text1.image_create(tk.END, image=photo11)

        self.text1.config(state='disabled')
        self.text1.grid(row=1,column=1,sticky="n")

        self.scroll = tk.Scrollbar(fr_galeria, command=self.text1.yview)
        self.scroll.grid(row=1,column=2,sticky="nse")

class cominfo():

    def __init__(self,master):

        #frame
        global fr_cominfo
        fr_cominfo =tk.Frame(master)

        #label
        self.label = tk.Label(fr_cominfo,text="Información de comisiones",font='bold')
        self.label.grid(row=0, column=1, sticky="n",padx=5, pady=5)

        #contents
        __quote = """
    Precios:

    Boceto:
    rostro..... $ 5 USD
```

busto ..... $ 10 USD

cintura ..... $ 15 USD

cuerpo entero ..... $ 20 USD

Pintura digital:

busto..... $ 30 USD

cintura..... $ 50 USD

cuerpo entero..... $ 70 USD

Adiciones:

fondo:

transparente.....$ 0 USD

sencillo.....$ 10 USD

complejo.....$ 20 USD

personajes adicionales..... x cantidad

¡Para realizar un encargo, envíeme su pedido por correo electrónico! illustreya@gmail.com

*Cuando me contrata, acepta haber leído y aceptado los Términos y condiciones.

"""

self.text1 = tk.Text(fr_cominfo, height=25, width=60)

```
        self.text1.insert(tk.END, __quote)
        self.text1.config(state='disabled')
        self.text1.grid(row=1,column=1,sticky="n")

        self.scroll = tk.Scrollbar(fr_cominfo, command=self.text1.yview)
        self.scroll.grid(row=1,column=1,sticky="nse")


class tosinfo():

    def __init__(self,master):


        #frame
        global fr_tosinfo
        fr_tosinfo =tk.Frame(master)

        #label
        self.label = tk.Label(fr_tosinfo,text="Términos y condiciones",font='bold')
        self.label.grid(row=0, column=1, sticky="n",padx=5, pady=5)

        __quote = """
                Introducción:
```

Lea atentamente estos Términos de servicio ("Términos"). Al comprar mis servicios de arte, automáticamente acepta estar sujeto a los siguientes términos y condiciones.

** DESCARGO DE RESPONSABILIDAD ** ¡Todos los proyectos son solo DIGITALES y NO se enviarán físicamente a menos que se indique lo contrario! Por lo tanto, recuerde seleccionar "NO SE REQUIERE ENVÍO" o "NO SE NECESITA DIRECCIÓN" durante el pago.

Si tiene alguna pregunta o inquietud, comuníquese conmigo en: illustreya@gmail.com ¡Gracias!

Pago:

USD a través de PayPal únicamente.

Los pagos se deben realizar por adelantado para cualquier comisión.

Obras en curso y cambios:

¡Es su responsabilidad proporcionarme los detalles que necesitaré para hacer un trabajo exitoso antes del pago! Sea lo más completo posible y proporcione referencias si están disponibles. No habrá cambios para la obra de arte terminada.

En el caso de piezas completamente coloreadas, puede solicitar un trabajo en progreso ("wip") justo antes de que empiece a colorear. ¡Los cambios significativos solo se pueden aplicar en este momento!

Artista:

Al contratarme, solo compra mi trabajo artístico. Conservo todos los derechos de autor sobre mi trabajo, que incluyen, entre otros: el derecho a distribuir, reproducir o usar la imagen como muestra para ventas / autopromoción, a menos que se acuerde lo contrario.

La propiedad intelectual de los personajes retratados permanece en sus respectivos dueños.

La obra de arte puede tardar entre 1 y 21 días en completarse. Si tiene una fecha límite / límite de tiempo, infórmeme antes del pago. Si tarda más, me pondré en contacto contigo de inmediato.

Tengo derecho a rechazar pedidos.

Enviaré su copia de la obra de arte por correo electrónico o mensajería directa.

Cliente:

Las piezas encargadas son solo para uso personal / privado, lo que significa que no se pueden utilizar con fines comerciales o de lucro (no deben aparecer en ningún producto que se venda).

Dé el crédito adecuado cuando vuelva a publicarlo / usarlo para otro propósito.

No puede reclamar que la obra de arte es suya, editarla o configurarla para su redistribución. Tampoco debe utilizarlo para proyectos externos a menos que se acuerde.

Política de reembolsos:

Solo se pueden solicitar reembolsos si aún no he comenzado a trabajar en la comisión. Una vez iniciado no habrá reembolsos.

Al contratarme, significa que ha leído y aceptado todos los términos descritos anteriormente.

Te adherirás a la política de usar el arte para fines personales, sin fines de lucro y me proporcionarás todos los créditos de imagen, illustreya.

Este documento puede y será editado en cualquier momento sin necesidad de previo aviso.

Última actualización: 8/2/2021

```
"""
    self.text1 = tk.Text(fr_tosinfo,height=25, width=60)
```

```
        self.text1.grid(row=1,column=1,sticky="n")
        self.text1.insert(tk.END, __quote)
        self.text1.config(state='disabled')


        self.scroll = tk.Scrollbar(fr_tosinfo, command=self.text1.yview)
        self.scroll.grid(row=1,column=1,sticky="nse")




class estimadosinfo():

    def __init__(self,master):

        #frame
        global fr_estimadosinfo
        fr_estimadosinfo =tk.Frame(master)


    def selection(self):
        pass
    def selection1(self):
        pass
    def selection2(self):
        pass
    def precioTotal(self):
        pass
    def tiempoTotal(self):
```

```
        pass

class boceto(estimadosinfo):

    def __init__(self,master):

        #frame
        super().__init__(fr_estimadosinfo)

        global fr_bocetoinfo
        fr_bocetoinfo =Label(master)

        #contents
        self.frame1 = Label(fr_bocetoinfo)
        self.frame1.pack()

        Label(self.frame1, text="Boceto:").grid(row=0,column=4,sticky="ew")
        self.frame2 = LabelFrame(self.frame1, text='Tamaño', padx=30, pady=5)
        self.frame3 = LabelFrame(self.frame1, text='Fondo', padx=30, pady=5)
        self.frame4 = LabelFrame(self.frame1, text='Personajes',padx=30,pady=5)

        global var, var1, var2
        var = IntVar()
        var1 = IntVar()

        Radiobutton(self.frame2, text='rostro', variable=var, value=1,command=self.selection).pack()
        Radiobutton(self.frame2, text='busto', variable=var, value=2,command=self.selection).pack()
        Radiobutton(self.frame2, text='cintura', variable=var, value=3,command=self.selection).pack()
```

```
        Radiobutton(self.frame2, text='cuerpo completo', variable=var,
value=4,command=self.selection).pack()


        Radiobutton(self.frame3, text='transparente', variable=var1,
value=1,command=self.selection1).pack()
        Radiobutton(self.frame3, text='sencillo', variable=var1, value=2,command=self.selection1).pack()
        Radiobutton(self.frame3, text='complejo', variable=var1, value=3,command=self.selection1).pack()


        Label(self.frame4, text="Cantidad").grid(row=0,column=1)



        var2 = Entry(self.frame4, width = 5)
        var2.grid(row=1, column=1, columnspan=30)


        self.frame2.grid(row=4, column=4,padx=30)
        self.frame3.grid(row=5, column=4,padx=30)
        self.frame4.grid(row=6, column=4,padx=30)


        self.submit_btn = Button(self.frame1, text="Submit", command=self.submit, padx=50, pady=5)
        self.submit_btn.grid(row=7, column=4, pady=2)



    def selection(self):
        choice = var.get()


        if choice == 1:
            m = 'rostro'
        elif choice == 2:
            m = 'busto'
```

```
    elif choice == 3:
        m = 'cintura'
    elif choice == 4:
        m = 'cuerpo completo'
    else:
        pass
    return m

def selection1(self):
    choice1 = var1.get()

    if choice1 == 1:
        n = 'transparente'
    elif choice1 == 2:
        n = 'sencillo'
    elif choice1 == 3:
        n = 'complejo'
    else:
        pass
    return n

def selection2(self):
    choice2 = var2.get()
    c = choice2
    return c

def precioTotal(self):
    choice = var.get()
    choice1 = var1.get()
```

```
choice2 = var2.get()

if ((choice == 1) and (choice1 == 1)):
    p = 5
elif ((choice == 1) and (choice1 == 2)):
    p = 15
elif ((choice == 1) and (choice1 == 3)):
    p = 25

elif ((choice == 2) and (choice1 == 1)):
    p = 10
elif ((choice == 2) and (choice1 == 2)):
    p = 20
elif ((choice == 2) and (choice1 == 3)):
    p = 30

elif ((choice == 3) and (choice1 == 1)):
    p = 15
elif ((choice == 3) and (choice1 == 2)):
    p = 25
elif ((choice == 3) and (choice1 == 3)):
    p = 35

elif ((choice == 4) and (choice1 == 1)):
    p = 20
elif ((choice == 4) and (choice1 == 2)):
    p = 30
elif ((choice == 4) and (choice1 == 3)):
    p = 40
```

```
    else:
        pass
    p = (p * (int(choice2)))
    return p


def tiempoTotal(self):
    choice = var.get()
    choice1 = var1.get()
    choice2 = var2.get()

    if ((choice == 1) and (choice1 == 1)):
        t = 1
    elif ((choice == 1) and (choice1 == 2)):
        t = 2
    elif ((choice == 1) and (choice1 == 3)):
        t = 3

    elif ((choice == 2) and (choice1 == 1)):
        t = 4
    elif ((choice == 2) and (choice1 == 2)):
        t = 5
    elif ((choice == 2) and (choice1 == 3)):
        t = 6

    elif ((choice == 3) and (choice1 == 1)):
        t = 7
    elif ((choice == 3) and (choice1 == 2)):
        t = 8
```

```
        elif ((choice == 3) and (choice1 == 3)):
            t = 9

        elif ((choice == 4) and (choice1 == 1)):
            t = 11
        elif ((choice == 4) and (choice1 == 2)):
            t = 12
        elif ((choice == 4) and (choice1 == 3)):
            t = 13
        else:
            pass

        t = (t * (int(choice2)))

        return t


    def submit(self):
        m = self.selection()
        n = self.selection1()
        c = self.selection2()
        p = self.precioTotal()
        t = self.tiempoTotal()

        return messagebox.showinfo('Resultados', f'Pedido de boceto: \n \n Tamaño: {m} \n Fondo: {n} \n
Personajes: {c} \n\n Precio total: $ {p} USD \n Tiempo total: {t} días')
```

```
class pintura(estimadosinfo):

    def __init__(self,master):

        #frame
        global fr_estimadosinfo  , fr_pinturainfo
        super().__init__(fr_estimadosinfo)
        fr_estimadosinfo =tk.Frame(master)
        fr_pinturainfo =Label(master)

        #label
        self.label = tk.Label(fr_estimadosinfo,text="Estimados",font='bold')
        self.label.grid(row=0, column=1, sticky="n",padx=5, pady=5)

        #contents
        self.frame1 = Label(fr_pinturainfo)
        self.frame1.pack()
        Label(self.frame1, text="Pintura:").grid(row=0,column=4,sticky="ew")
        self.frame2 = LabelFrame(self.frame1, text='Tamaño', padx=30, pady=5)
        self.frame3 = LabelFrame(self.frame1, text='Fondo', padx=30, pady=5)
        self.frame4 = LabelFrame(self.frame1, text='Personajes',padx=30,pady=5)
        global var3, var4, var5
        var3 = IntVar()
        var4 = IntVar()

        Radiobutton(self.frame2, text='rostro', variable=var3, value=1,command=self.selection).pack()
        Radiobutton(self.frame2, text='busto', variable=var3, value=2,command=self.selection).pack()
        Radiobutton(self.frame2, text='cintura', variable=var3, value=3,command=self.selection).pack()
```

```
        Radiobutton(self.frame2, text='cuerpo completo', variable=var3,
value=4,command=self.selection).pack()


        Radiobutton(self.frame3, text='transparente', variable=var4,
value=1,command=self.selection1).pack()
        Radiobutton(self.frame3, text='sencillo', variable=var4, value=2,command=self.selection1).pack()
        Radiobutton(self.frame3, text='complejo', variable=var4, value=3,command=self.selection1).pack()


        Label(self.frame4, text="Cantidad:").grid(row=0,column=1)



        var5 = Entry(self.frame4, width = 5)
        var5.grid(row=1, column=1, columnspan=30)


        self.frame2.grid(row=4, column=4,padx=30)
        self.frame3.grid(row=5, column=4,padx=30)
        self.frame4.grid(row=6, column=4,padx=30)


        self.submit_btn = Button(self.frame1, text="Submit", command=self.submit, padx=50, pady=5)
        self.submit_btn.grid(row=7, column=4, pady=2)




    def selection(self):
        choice3 = var3.get()


        if choice3 == 1:
            m1 = 'rostro'
        elif choice3 == 2:
```

```
        m1 = 'busto'
    elif choice3 == 3:
        m1 = 'cintura'
    elif choice3 == 4:
        m1 = 'cuerpo completo'
    else:
        pass
    return m1

def selection1(self):
    choice4 = var4.get()

    if choice4 == 1:
        n1 = 'transparente'
    elif choice4 == 2:
        n1 = 'sencillo'
    elif choice4 == 3:
        n1 = 'complejo'
    else:
        pass
    return n1

def selection2(self):
    choice5 = var5.get()
    c1 = choice5
    return c1

def precioTotal(self):
    choice3 = var3.get()
```

```
choice4 = var4.get()
choice5 = var5.get()

if ((choice3 == 1) and (choice4 == 1)):
    p1 = 30
elif ((choice3 == 1) and (choice4 == 2)):
    p1 = 40
elif ((choice3 == 1) and (choice4 == 3)):
    p1 = 50

elif ((choice3 == 2) and (choice4 == 1)):
    p1 = 45
elif ((choice3 == 2) and (choice4 == 2)):
    p1 = 55
elif ((choice3 == 2) and (choice4 == 3)):
    p1 = 65

elif ((choice3 == 3) and (choice4 == 1)):
    p1 = 50
elif ((choice3 == 3) and (choice4 == 2)):
    p1 = 60
elif ((choice3 == 3) and (choice4 == 3)):
    p1 = 70

elif ((choice3 == 4) and (choice4 == 1)):
    p1 = 70
elif ((choice3 == 4) and (choice4 == 2)):
    p1 = 80
elif ((choice3 == 4) and (choice4 == 3)):
```

```
        p1 = 90
    else:
        pass
    p1 = (p1 * (int(choice5)))
    return p1


def tiempoTotal(self):
    choice3 = var3.get()
    choice4 = var4.get()
    choice5 = var5.get()

    if ((choice3 == 1) and (choice4 == 1)):
        t1 = 5
    elif ((choice3 == 1) and (choice4 == 2)):
        t1 = 6
    elif ((choice3 == 1) and (choice4 == 3)):
        t1 = 7

    elif ((choice3 == 2) and (choice4 == 1)):
        t1 = 8
    elif ((choice3 == 2) and (choice4 == 2)):
        t1 = 9
    elif ((choice3 == 2) and (choice4 == 3)):
        t1 = 10

    elif ((choice3 == 3) and (choice4 == 1)):
        t1 = 11
    elif ((choice3 == 3) and (choice4 == 2)):
```

```
            t1 = 12
        elif ((choice3 == 3) and (choice4 == 3)):
            t1 = 13

        elif ((choice3 == 4) and (choice4 == 1)):
            t1 = 14
        elif ((choice3 == 4) and (choice4 == 2)):
            t1 = 15
        elif ((choice3 == 4) and (choice4 == 3)):
            t1 = 16

        else:
            pass

        t1 = (t1 * (int(choice5)))

        return t1


    def submit(self):
        m1 = self.selection()
        n1 = self.selection1()
        c1 = self.selection2()
        p1 = self.precioTotal()
        t1 = self.tiempoTotal()

        return messagebox.showinfo('Resultados', f'Pedido de pintura: \n \n Tamaño: {m1} \n Fondo: {n1}
\n Personajes: {c1} \n\n Precio total: $ {p1} USD \n Tiempo total: {t1} días')
```

```
obj = menu(window)
obj1 = inicioinfo(window)
obj2 = galeriainfo(window)
obj3 = cominfo(window)
obj4 = tosinfo(window)
obj5 = estimadosinfo(window)
obj6 = boceto(window)
obj7 = pintura(window)

window.geometry("710x470")
window.mainloop()
```