

G1

grupo 1

# Informe proyecto final

2 de junio del 2022

## INTEGRANTES:

aldair vargas alarcon

José Manuel Tapia Martínez

Benjamín Moisés cruz condori

Ronald Brayan Cusiquispe Paco

registro de pacientes de un hospital



# Quiénes somos

## Misión


Los informes se usan mucho en las empresas para documentar proyectos, códigos y avances, desempeño de las empresas, estrategias de marketing y de redes sociales, y mucho más.

## Visión

Nuestra visión es poder hacer entender a nuestros usuarios de la utilidad de nuestro sistema y la facilidad que nos da el poder utilizar un sistema en nuestro día a día.

## Valores

En nuestros valores podemos destacar el compañerismo a la hora de poder hacer los algoritmos de nuestro sistema. Y la co

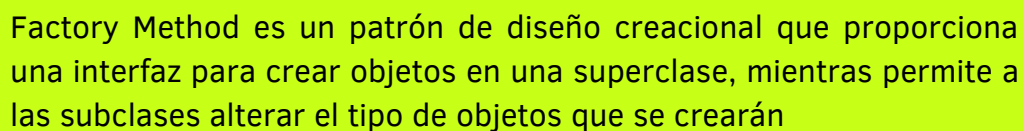


# Patron de diseño

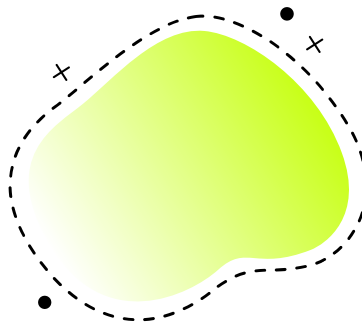


## FACTORY METHOD

También llamado: Método fábrica, Constructor virtual



Factory Method es un patrón de diseño creacional que proporciona una interfaz para crear objetos en una superclase, mientras permite a las subclases alterar el tipo de objetos que se crearán



# Estructura del sistema

## Login

Para nuestro login utilizamos roles en el cual el administrador puede añadir nuevos usuarios a nuestra base de datos empotrada SQLITE,

### ADMIN



A screenshot of a web browser window titled "ADMIN" with a green background. The window has standard OS window controls (minimize, maximize, close) in the top right corner. The main heading is "INICIO DE SESION". Below it, there are two input fields: "E-mail :" with the text "admin" and "Contraseña:" with masked text "\*\*\*\*\*". At the bottom, there are two radio buttons: "CON SEGURO" (selected) and "SIN SEGURO". Below the radio buttons are two buttons: "Iniciar Sesión" and "Registrarse".

Para nuestro rol de usuario tendremos el mismo login con diferentes opciones en el cual al iniciar nuestra sesión podremos ver nuestro registro.

### USUARIO

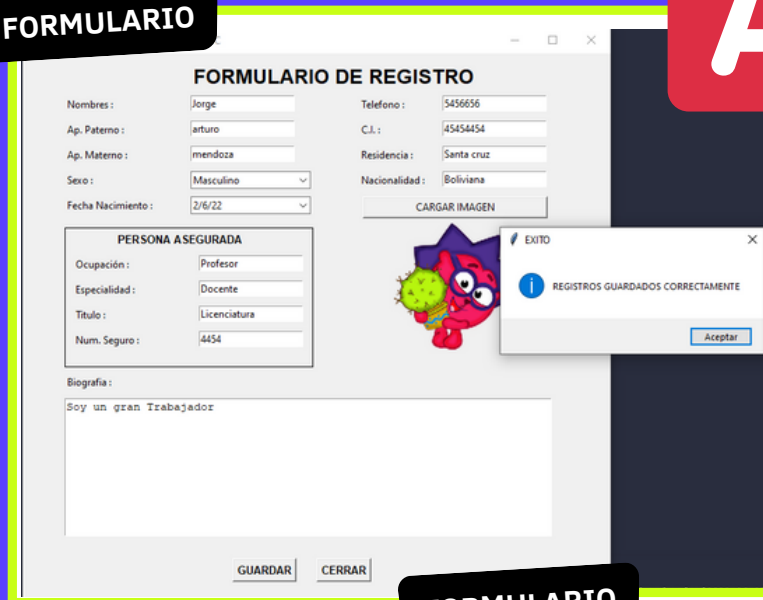


A screenshot of a web browser window titled "USUARIO" with a blue background. The window has standard OS window controls (minimize, maximize, close) in the top right corner. The main heading is "REGISTRO DE USUARIO". Below it, there are two input fields: "E-mail :" with the text "Juan" and "Contraseña:" with masked text "\*\*\*\*". At the bottom, there are two buttons: "Crear Usuario" and "Iniciar Sesión".

# Formulario

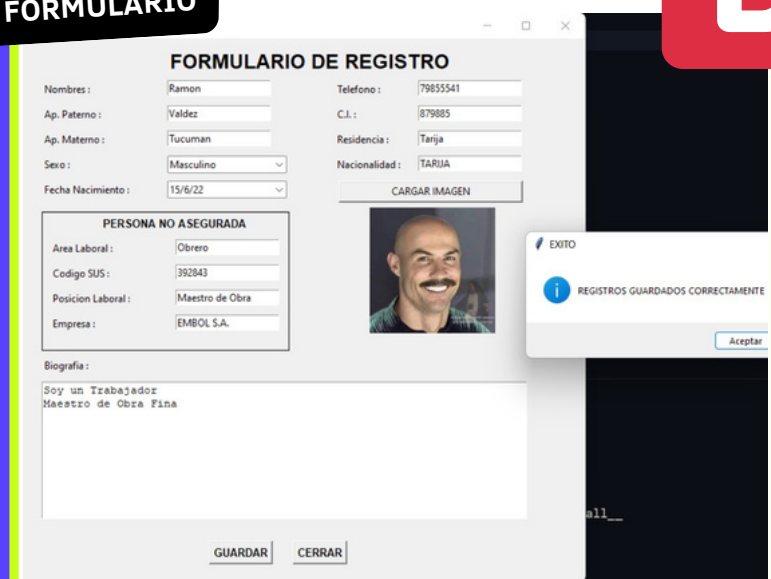
Una vez nos logueamos como usuario tendremos nuestro formulario en el cual tendremos diferentes casillas para rellenar que fuimos añadiendo con PYTHON - TRINTER y a cada label y entry tuvimos que posicionarlo en nuestra ventana y por ultimo tenemos los botones de cerrar y guardar nuestra información, una vez guardada nos aparecera un mensaje para señalar que nuestra información fue guardada con exito.

## FORMULARIO



A screenshot of a web application window titled "FORMULARIO DE REGISTRO". The form contains several input fields for personal information: "Nombres:" (Jorge), "Ap. Paterno:" (arturo), "Ap. Materno:" (mendoza), "Sexo:" (Masculino), "Fecha Nacimiento:" (2/6/22), "Telefono:" (5456656), "C.I.:" (45454454), "Residencia:" (Santa cruz), and "Nacionalidad:" (Boliviana). There is a "CARGAR IMAGEN" button. Below this is a section for "PERSONA ASEGURADA" with fields for "Ocupación:" (Profesor), "Especialidad:" (Docente), "Titulo:" (Licenciatura), and "Num. Seguro:" (4454). A "Biografia:" section contains the text "Soy un gran Trabajador". At the bottom are "GUARDAR" and "CERRAR" buttons. A red square with a white letter 'A' is overlaid on the top right. A small modal window shows a success message: "EXITO REGISTROS GUARDADOS CORRECTAMENTE" with an "Aceptar" button.

## FORMULARIO



A screenshot of the same web application window, but with different data entered. The "Nombres:" field is "Ramon", "Ap. Paterno:" is "Valdez", "Ap. Materno:" is "Tucuman", "Sexo:" is "Masculino", "Fecha Nacimiento:" is "15/6/22", "Telefono:" is "79855541", "C.I.:" is "879885", "Residencia:" is "Tarija", and "Nacionalidad:" is "TARIJA". The "CARGAR IMAGEN" button is now active, showing a photo of a man. The "PERSONA NO ASEGURADA" section has fields for "Area Laboral:" (Obrero), "Codigo SUS:" (392843), "Posicion Laboral:" (Maestro de Obra), and "Empresa:" (EMBOL S.A.). The "Biografia:" section contains the text "Soy un Trabajador Maestro de Obra Fina". At the bottom are "GUARDAR" and "CERRAR" buttons. A red square with a white letter 'B' is overlaid on the top right. A small modal window shows a success message: "EXITO REGISTROS GUARDADOS CORRECTAMENTE" with an "Aceptar" button.

# ADMINISTRADOR DE REGISTRO

Para poder visualizar nuestro administrador de registro debemos de loguearnos como administrador, para enviar todos nuestra información a nuestra tabla debimos de utilizar la API creada en laravel que nos facilito en ingeniero de la materia.

en nuestra tabla tenemos diferentes opciones eliminar, mostrar y imprimir, la opción imprimir es la mas interesante ya que nos permitira descargar toda nuestra información en un pdf

## ADMINISTRADOR

### ADMINISTRADOR DE REGISTROS CEDIMECC

id	ci	nombres	paterno	materno	sexo	fecha_nacimi	biografia	foto	nacionalidad	Genresidencia	ocupacion	especialidad	titulo	numero_s
11	76589945	Dominic	Toreto	None	Masculino	6/1/22	None	https://www.1	Americana	fastnfurius	Taxista	dominic	None	None
12	88889999	RONALD	CUSIQUISPE	PACO	Masculino	24/6/22	ESTUDIANTE	C:/Users/5917	BOLIVIA	SANTA CRUZ	FULLSTACK D	MACHINE LE	ING SISTEMA	655549887
13	82940433	Johny	Depp	None	Masculino	6/23/00	None	https://image	Americana	hol123	Artista	johny	None	None
14	82940433	Arnold	Schawzeneg	None	None	6/2/96	None	https://media	Americana	abc	Culturista	abc	None	None
18	5556	aldair	vargas	alarcon	Masculino	1/6/22	hola esto es u	C:/Users/5917	bolivia	boliana	diseño	diseño	diseño	855
19	123456	Jorge	Barba	Vargas	Masculino	01/08/2000	None	None	Argentino	None	Estudiante	None	Ingeniero	None
21	12345	jij	ccc	ccc	MASCULINO	2022-06-01	aa	None	bolivia	bolivia	estudiante	None	None	12345
22	1234	Josias	CÓndori	Correa	MASCULINO	2022-06-07	A	None	Bolivia	Bolivia	Estudiante	None	None	1234
23	123456789	Oscar	Vargas	Barba	Masculino	10/2/2000	None	None	Argentino	Av/Betania, C	Abogado	None	abogado	None
24	45454454	Jorge	arturo	mendoza	Masculino	2/6/22	Soy un gran T	C:/				Docente	Licenciatura	4454

MOSTRAR

ELIMINAR

IMPR

EXITO

REGISTROS GUARDADOS EN PDF & TXT

Aceptar



# CODIGO FUENTE

## CLASE PADRE PERSONA

La Clase Padre Persona Tiene Atributos Propios que serian los campos en común del formulario, Cuyos parámetros son heredados por las subclases siguientes, que a su vez pueden alterar directamente la estructura del padre

```
1  from registro import register
2  from tkinter import *
3  from tkinter import ttk
4  from tkcalendar import *
5  from PIL import ImageTk, Image
6  from tkinter import messagebox as ms
7  import requests
8  from tkinter import filedialog
9
10
11 class Persona():
12
13     def __init__(self, ventana):
14         #METODO PARA CARGAR LA IMAGEN
15         self.ventana = ventana
16         def cargarImg():
17             global img, ruta
18             #global ruta
19             ventana.filename = filedialog.askopenfilename(filetypes = [
20                 ("image", ".jpeg"),
21                 ("image", ".png"),
22                 ("image", ".jpg")], title="SELECCIONE UNA FOTO", initialdir="/src")
23             ruta = ventana.filename
24             img = ImageTk.PhotoImage(Image.open(ventana.filename).resize((150, 150), Image.ANTIALIAS))
25             Label(image=img).place(x=440, y=200)
26
27
28         lbl_name = Label(ventana, text="Nombres : ").place(x=50, y=50)
29         self.name = Entry(ventana)
30         self.name.place(x=200, y=50)
31
32
33         lbl_apellido = Label(ventana, text="Ap. Paterno : ").place(x=50, y=80)
34         self.apellido1 = Entry(ventana)
35         self.apellido1.place(x=200, y=80)
36
37         lbl_apellido = Label(ventana, text="Ap. Materno : ").place(x=50, y=110)
38         self.apellido2 = Entry(ventana)
39         self.apellido2.place(x=200, y=110)
40
41
42         lbl_sexo = Label(ventana, text="Sexo : ").place(x=50, y=140)
43         self.sexo = ttk.Combobox(ventana,
44             textvariable=StringVar(),
45             state='readonly',
46             values=['Masculino', 'Femenino']
47         )
48         self.sexo.place(x=200, y=140)
49
50
51         lbl_fecha = Label(ventana, text="Fecha Nacimiento : ").place(x=50, y=170)
52         self.fecha = DateEntry(ventana,width=20)
53         self.fecha.place(x=200, y=170)
54
55
56         lbl_history = Label(ventana, text="Biografia : ").place(x=50, y=380)
57         self.biografia = Text(ventana, width=72, height=10)
58         self.biografia.place(x=50, y=410)
59
60
61         lbl_telefono = Label(ventana, text="Telefono : ").place(x=400, y=50)
62         self.telefono = Entry(ventana,textvariable=IntVar())
63         self.telefono.place(x=500, y=50)
64
65
66         lbl_cedula = Label(ventana, text="C.I. : ").place(x=400, y=80)
67         self.cedula = Entry(ventana)
68         self.cedula.place(x=500, y=80)
```

# CLASE PADRE PERSONA

```
lbl_residencia = Label(ventana, text="Residencia : ").place(x=400, y=110)
self.residencia = Entry(ventana)
self.residencia.place(x=500, y=110)

lbl_nacionalidad = Label(ventana, text="Nacionalidad : ").place(x=400, y=140)
self.nacionalidad = Entry(ventana)
self.nacionalidad.place(x=500, y=140)

btn_img = Button(ventana, text="CARGAR IMAGEN", width=30, command=cargarImg).place(x=405, y=170)

#BOTONES DE GUARDAR Y SALIR
btn_save = Button(ventana, text="GUARDAR", font="Helvetica 10 bold", command=self.mandar).place(x=250, y=600)
btn_quit = Button(ventana, text="CERRAR", command=lambda: [quit()], font="Helvetica 10 bold").place(x=350, y=600)

#METODO PARA CONECTAR A LA API CON EL BOTON GUARDAR

def mandar(self):
    requests.post("http://tecnoprofe.com/api/paciente", data = {
        'ci': self.cedula.get(),
        'nombres': self.name.get(),
        'paterno': self.apellido1.get(),
        'materno': self.apellido2.get(),
        'sexo': self.sexo.get(),
        'fecha_nacimiento': self.fecha.get(),
        'biografia': self.biografia.get("1.0", "end-1c"),
        'foto': ruta,
        'nacionalidad': self.nacionalidad.get(),
        'residencia': self.residencia.get(),
        #'ocupacion': txt_ocupacion.get(),
        #'titulo': txt_titulo.get(),
        #'numero_seguro': txt_numseg.get()
    })
    ms.showinfo('EXITO', 'REGISTROS GUARDADOS CORRECTAMENTE')

def cambiar(self):
    self.ventana.destroy()
    register()
```

Hasta Aquí,  
mostramos los  
Atributos de la  
Superclase

# SUBCLASES PRINCIPALES

```
class Asegurado(Persona):
    def __init__(self, ventana):
        super().__init__(ventana)

        global txt_ocupacion, txt_especialidad, txt_titulo, txt_numseg

        dibujar = Canvas(ventana, width= 300, height=170)
        dibujar.place(x=45, y=202)
        dibujar.create_rectangle(5,5,300,170)

        Label(ventana, text="PERSONA ASEGURADA", font="Helvetica 10 bold").place(x=110, y=210)
        Label(ventana, text="Ocupación : ").place(x=60, y=240)
        txt_ocupacion = Entry(ventana)
        txt_ocupacion.place(x=210, y=240)

        Label(ventana, text="Especialidad : ").place(x=60, y=270)
        txt_especialidad = Entry()
        txt_especialidad.place(x=210, y=270)

        Label(ventana, text="Titulo : ").place(x=60, y=300)
        txt_titulo = Entry(ventana)
        txt_titulo.place(x=210, y=300)

        Label(ventana, text="Num. Seguro : ").place(x=60, y=330)
        txt_numseg = Entry(ventana, textvariable=IntVar())
        txt_numseg.place(x=210, y=330)
```

```
class NoAsegurado(Persona):
    def __init__(self, ventana):
        super().__init__(ventana)
        global txt_sus
        dibujar = Canvas(ventana, width= 300, height=170)
        dibujar.place(x=45, y=202)
        dibujar.create_rectangle(5,5,300,170)

        Label(ventana, text="PERSONA NO ASEGURADA", font="Helvetica 10 bold").place(x=120, y=210)
        Label(ventana, text="Area Laboral :").place(x=60, y=240)
        txt_soat = Entry(ventana).place(x=210, y=240)

        Label(ventana, text="Codigo SUS :").place(x=60, y=270)
        txt_sus = Entry(ventana)
        txt_sus.place(x=210, y=270)

        lbl_laburo = Label(ventana, text="Posicion Laboral : ").place(x=60, y=300)
        txt_laburo = Entry(ventana).place(x=210, y=300)

        lbl_emp = Label(ventana, text="Empresa : ").place(x=60, y=330)
        txt_emp = Entry(ventana).place(x=210, y=330)
```

Las Subclases Heredan los Atributos de La Superclase persona, a quien inicialmente invocan, posteriormente, generan sus propios campos en la ventana de Tkinter



# CLASE LOGIN

```
from factory import *
from registro import *
from tkinter import *
from tkinter import messagebox as ms
from hashlib import md5
import sqlite3

class login:
    def __init__(self):
        # Window
        self.root = Tk()
        self.root.title('CEDIMECC')
        self.root.geometry('430x300')
        # Some Usefull variables
        self.username = StringVar()
        self.password = StringVar()
        self.n_username = StringVar()
        self.n_password = StringVar()
        self.option = IntVar()
        #Create Widgets
        self.widgets()
        self.root.mainloop()

#Login Function
def login(self):
    if (self.username.get() == '' or self.password.get() == ''):
        ms.showerror('Error', 'Complete Los Campos !!!')
    #elif(self.password.get() == ''):
    #    ms.showerror('Error', 'Complete Los Campos !!!')
    elif(self.option.get() == 0):
        if (self.username.get() == 'admin' and self.password.get() == 'admin'):
            self.root.destroy()
            register()
        else:
            ms.showerror('Error', 'SELECCIONE TIPO DE FORMULARIO !!!')

    else:
        #Estableciendo Conexion
        with sqlite3.connect('quit.db') as db:
            c = db.cursor()

#Busqueda de Usuario
find_user = ('SELECT * FROM user WHERE username = ? and password = ?')
c.execute(find_user, [(self.username.get()), (md5(self.password.get().encode('utf-8')).hexdigest())])
result = c.fetchall()
if result:

    self.logf.pack_forget()
    self.root.destroy()

    ##### SI EL USUARIO EXISTE ENTRAMOS AL FORMULARIO INDICADO
    form = Tk()
    form.title('FORMULARIO DE REGISTRO CEDIMECC')
    form.geometry('700x650')
    Label(form, text="FORMULARIO DE REGISTRO", font="Helvetica 18 bold").place(x=200, y=10)
    FabricarFormulario().getForm(form, self.option.get()) #SELECCION
    form.mainloop()
    #####

else:
    ms.showerror('Error', 'Usuario no encontrado.')
```

# CLASE LOGIN

```
def new_user(self):
    #Estableciendo Conexion
    if (self.n_username.get() == ''):
        ms.showerror('Error','Complete Los Campos !!!')
    elif(self.n_password.get() == ''):
        ms.showerror('Error','Complete Los Campos !!!')
    else:
        with sqlite3.connect('quit.db') as db:
            c = db.cursor()

        #Verifica si User existe, sino lo registra

        find_user = ('SELECT username FROM user WHERE username = ?')
        c.execute(find_user,[(self.n_username.get())])
        if c.fetchall():
            ms.showerror('Error',' Usuario Existente, Pruebe con uno diferente.')
        else:
            ms.showinfo('EXITO','Cuenta creada Exitosamente')
            self.log()
        #Crear una nueva cuenta
        insert = 'INSERT INTO user(username,password) VALUES(?,?)'
        c.execute(insert,[(self.n_username.get()),(md5(self.n_password.get().encode('utf-8')).hexdigest())])
        db.commit()

#def admin(self):

# Métodos de embalaje del marco
def log(self):
    self.username.set('')
    self.password.set('')
    self.crf.pack_forget()
    self.logf.pack()
def cr(self):
    self.n_username.set('')
    self.n_password.set('')
    if(self.username.get() == 'admin' and self.password.get() == 'admin'):
        self.logf.pack_forget()
        self.crf.pack()
    else:
        ms.showerror('Error','COLOQUE LOS DATOS DEL ADMIN PARA REGISTRAR !!!')
```

```
#Draw Widgets
def widgets(self):

    self.logf = Frame(self.root,padx =10,pady = 20, bg='#0FE465')
    Label(self.logf,text = 'INICIO DE SESION', font = ('',20),pady=20,padx=5, bg='#0FE465').grid(row=0,column=0, columnspan=2)
    Label(self.logf,text = 'E-mail : ',font = ('',20),pady=5,padx=5, bg='#0FE465').grid(sticky = W)
    Entry(self.logf,textvariable = self.username,bd = 5,font = ('',15)).grid(row=1,column=1)
    Label(self.logf,text = 'Contraseña: ',font = ('',20),pady=5,padx=5, bg='#0FE465').grid(sticky = W)
    Entry(self.logf,textvariable = self.password,bd = 5,font = ('',15),show = '*').grid(row=2,column=1)
    #####
    Radiobutton(self.logf, text = 'CON SEGURO', variable = self.option, value = 1, bg='#0FE465',pady=15).grid(row=3,column=0)
    Radiobutton(self.logf, text = 'SIN SEGURO', variable = self.option, value = 2, bg='#0FE465',pady=15).grid(row=3,column=1)
    #####
    Button(self.logf,text = ' Iniciar Sesión ',bd = 3 ,font = ('',10),padx=5,pady=5,command=self.login).grid(row=4,column=0)
    Button(self.logf,text = ' Registrarse ',bd = 3 ,font = ('',10),padx=5,pady=5,command=self.cr).grid(row=4,column=1)
    self.logf.pack()

    self.crf = Frame(self.root,padx =10,pady = 20, bg='#0FC2E4')
    Label(self.crf,text = 'REGISTRO DE USUARIO', font = ('',20),pady=20,padx=5, bg='#0FC2E4').grid(row=0,column=0, columnspan=2)
    Label(self.crf,text = 'E-mail : ',font = ('',20),pady=5,padx=5, bg='#0FC2E4').grid(sticky = W)
    Entry(self.crf,textvariable = self.n_username,bd = 5,font = ('',15)).grid(row=1,column=1)
    Label(self.crf,text = 'Contraseña: ',font = ('',20),pady=5,padx=5, bg='#0FC2E4').grid(sticky = W)
    Entry(self.crf,textvariable = self.n_password,bd = 5,font = ('',15),show = '*').grid(row=2,column=1)
    Label(self.crf,text = '', pady=15, bg='#0FC2E4').grid()
    Button(self.crf,text = 'Crear Usuario',bd = 3 ,font = ('',10),padx=5,pady=5,command=self.new_user).grid(row=4,column=0)
    Button(self.crf,text = 'Iniciar Sesión',bd = 3 ,font = ('',10),padx=5,pady=5,command=self.log).grid(row=4,column=1)
```

# CLASE REGISTROS

```
from tkinter import *
from tkinter import ttk
import requests
from fpdf import FPDF
from tkinter import messagebox as ms
import sys

class register:
    def __init__(self):
        self.reg = Tk()
        self.reg.title('CEDIMECC')

        self.reg.geometry('1250x450')
        self.wid()
        self.reg.mainloop()

    def wid(self):
        Label(self.reg, text="ADMINISTRADOR DE REGISTROS", font="FiraCode 18 bold").place(x=380, y=20)
        Label(self.reg, text=" CEDIMECC ", font="FiraCode 18 bold").place(x=500, y=50)

        self.tabla=ttk.Treeview(self.reg)
        self.tabla.place(x=30,y=100)
        self.desplazate=ttk.Scrollbar(self.reg, orient="vertical", command=self.tabla.yview)
        self.desplazate.place(x=30,y=100,height=230)
        self.tabla.configure(yscrollcommand=self.desplazate.set)
        self.tabla['columns']=(('id', 'ci', 'nombres', 'paterno','materno', 'sexo', 'fecha_nacimiento',
                                'biografia', 'foto', 'nacionalidad', 'residencia', 'ocupacion', 'especialidad',
                                'titulo', 'numero_seguro'))

        self.tabla.column("#0", width=0)
        self.tabla.column("id", width=50)
        self.tabla.column("ci", width=80)
        self.tabla.column("nombres", width=80)
        self.tabla.column("paterno", width=80)
        self.tabla.column("materno", width=80)
        self.tabla.column("sexo", width=80)
        self.tabla.column("fecha_nacimiento", width=80)
        self.tabla.column("biografia", width=80)
        self.tabla.column("foto", width=80)
        self.tabla.column("nacionalidad", width=80)
        self.tabla.column("residencia", width=80)
        self.tabla.column("ocupacion", width=80)
        self.tabla.column("especialidad", width=80)
        self.tabla.column("titulo", width=80)

        self.tabla.heading("#0",text="")
        self.tabla.heading("id",text="id")
        self.tabla.heading("ci",text="ci")
        self.tabla.heading("nombres",text="nombres")
        self.tabla.heading("paterno",text="paterno")
        self.tabla.heading("materno",text="materno")
        self.tabla.heading("sexo",text="sexo")
        self.tabla.heading("fecha_nacimiento",text="fecha_nacimiento")
        self.tabla.heading("biografia",text="biografia")
        self.tabla.heading("foto",text="foto")
        self.tabla.heading("nacionalidad",text="nacionalidad")
        self.tabla.heading("residencia",text="Genresidenciaero")
        self.tabla.heading("ocupacion",text="ocupacion")
        self.tabla.heading("especialidad",text="especialidad")
        self.tabla.heading("titulo",text="titulo")
        self.tabla.heading("numero_seguro",text="numero_seguro")

        btn_mostrar = Button(text="MOSTRAR", font="Helvetica 14 bold", command=lambda:[self.mostrar()]).place(x=400, y=370)
        btn_eliminar = Button(text="ELIMINAR", font="Helvetica 14 bold", command=lambda:[self.eliminar(), self.mostrar()]).place(x=520, y=370)
        btn_imprimir = Button(text="IMPRIMIR", font="Helvetica 14 bold", command=lambda:[self.imprimir()]).place(x=635, y=370)
```

# CLASE REGISTROS

```
def mostrar(self):
    for item in self.tabla.get_children():
        self.tabla.delete(item)

    canciones=requests.get('http://tecnoprofe.com/api/paciente')
    print(canciones.json())
    for dato in canciones.json():
        self.tabla.insert(parent='', index='end',
            values=(dato['id'],dato['ci'], dato['nombres'], dato['paterno'], dato['materno'], dato['sexo'],
                dato['fecha_nacimiento'], dato['biografia'], dato['foto'], dato['nacionalidad'], dato['residencia'],
                dato['ocupacion'], dato['especialidad'], dato['titulo'], dato['numero_seguro']))

def eliminar(self):
    marcado= self.tabla.focus()
    actual=self.tabla.item(marcado)['values'][0]
    dato=actual
    requests.delete('http://tecnoprofe.com/api/paciente/'+ str(dato))

#IMPRIMIR REPORTE
def imprimir(self):
    pdf =FPDF()
    pdf.add_page()
    pdf.set_font("Arial",size=10)
    stoutOrigin = sys.stdout
    sys.stdout=open("reportes/ReportTXT.txt", "w")
    canciones=requests.get('http://tecnoprofe.com/api/paciente')
    for fila in canciones.json():
        print(" [",fila['id'],",- ] [",fila['ci'],"," ] [",fila['nombres'],"," ] [",fila['paterno'],"," ] [ ",
            fila['materno'],"," ] [",fila['sexo'],"," ] [",fila['fecha_nacimiento'],"," ]\n [",fila['biografia'],"," ]\n [",
            fila['foto'],"," ]\n [",fila['nacionalidad'],"," ] [",fila['residencia'],"," ] [",fila['ocupacion'],"," ] [",fila['especialidad'],"," ]\n [",
            fila['titulo'],"," ] [",fila['numero_seguro'],"," ]\n -----+ ")
    sys.stdout=open("reportes/ReportTXT.txt", "r")
    for x in sys.stdout:
        pdf.cell(w=200,h=15,txt = x,ln = 1,align = 'C')

    pdf.output('reportes/ReportPDF.pdf')

    ms.showinfo('EXITO','REGISTROS GUARDADOS EN PDF & TXT')
```

# CLASE FACTORY

```
factory.py > ...
1  from persona import *
2  #from abc import ABC, abstractmethod
3
4  class FabricarFormulario():
5      def getForm(self, ventana, num):
6          if (num == 1):
7              return Asegurado(ventana)
8          elif (num == 2):
9              return NoAsegurado(ventana)
10
```

# Bibliografía

Para lograr el desarrollo del sistema  
tuvimos que recurrir a diferentes tipo de  
información

1. <https://www.youtube.com/c/Codemycom>
2. [https://www.academia.edu/15210965/Manual\\_interfaz\\_grafica\\_en\\_python\\_tkinter](https://www.academia.edu/15210965/Manual_interfaz_grafica_en_python_tkinter)
3. [https://www.youtube.com/watch?v=I\\_URKOojOCO](https://www.youtube.com/watch?v=I_URKOojOCO)
4. [https://www.youtube.com/watch?v=PLvidd1VERI&ab\\_channel=Programaci%C3%B3nDeVerdad](https://www.youtube.com/watch?v=PLvidd1VERI&ab_channel=Programaci%C3%B3nDeVerdad)