Manual de uso Técnico

Tabla de Contenido

Contenido

Introducción

El siguiente documento es para uso técnico el cual describe los aspectos técnicos del sistema elaborado para control de pensum de estudios. El documento pretende facilitar al personal técnico especializado encargado de las actividades de mantenimiento, revisión, solución de problemas, instalación y configuración del sistema.

Información destacada

El manual técnico hace referencia a la información necesaria con el fin de orientar al personal en la concepción, planteamiento análisis programación e instalación del sistema. Es de notar que la redacción propia del manual técnico está orientada a personal con conocimientos en sistemas y tecnologías de información, conocimientos de programación avanzada y el uso de gramáticas y autómatas en el entorno de Python.

Objetivos

Instruir al técnico designado en los procesos de modificación, mantenimiento y reestructuración de la aplicación.

1.Requerimientos

El sistema pude ser instalado en cualquier sistema operativo que cumpla con los siguientes requerimientos.

2.Instalación y Configuración

La aplicación es ejecutable no requiere ningún proceso de instalación, simplemente tener instalado
Python en el equipo, se ejecuta en cualquier sistema operativo, gracias a ser una ampliación Python , para mantenimiento y modificación del mismo se requiere un entorno de desarrollo para Python en el cual se podrá realizar dichas actividades

3. Inicial

```
import tkinter as tk
from menup import menuprincipal

root = tk.Tk()
root.geometry('700x300')
root.title('Proyecto2.')
b=tk.Label(root, text="Bienvenidos", font=('Times New Roman',15), fg='#000000', bg='#DCD
c=tk.Label(root, text="Lenguajes Formales y de Programación N", font=('Times New Roman',
d=tk.Label(root, text="Henry Alexander García Montúfar", font=('Times New Roman',15), fg
e=tk.Label(root, text="Carnet:201407049", font=('Times New Roman',15), fg='#000000', bg=


def MostrarVentana():
    a=menuprincipal()
```

Configuración del sistema

En esta parte se muestra el inicio de los formularios realizados utilizando Tkinter y un timer para poder desaparecer la pantalla inicial.

Función Update

Este método integra los procesos para actualización del sistema

```
def update(rows):
    global mydata
    mydata = rows
    trv.delete(*trv.get_children())
    for i in rows:
        trv.insert('', 'end', values=i)
```

Función Search

Contiene los procesos para poder buscar dentro de la base de datos

```
def search():
    q2=q.get()
    query = "SELECT id, nombre, prerquisito, opcionalidad, semestre, creditos, estado FROM datos WHERE nombre LIKE '%"+q2+"%'"
    cursor.execute(query)
    rows = cursor.fetchall()
    update(rows)
```

Función Clear

Esta función permite limpiar la pantalla.

Función getrow

Esta función integra la selección de filas para poder recorrer los archivos de la lista.

```
def getrow(event):
    rowid = trv.identify_row(event.y)
    item = trv.item(trv.focus())
    t1.set(item['values'][0])
    t2.set(item['values'][1])
    t3.set(item['values'][2])
    t4.set(item['values'][3])
    t5.set(item['values'][4])
    t6.set(item['values'][5])
    t7.set(item['values'][6])
    t8.set(item['values'][8])
```

Función UP

Función usada para poder actualizar los cursos por separado.

```
def up():
    fname = t2.get()
    lpre = t3.get()
    opc = t4.get()
    sem = t5.get()
    cre = t6.get()
    estado = t7.get()
    id = t1.get()

    if messagebox.askyesno("Confirmar", "Deseas actualizar?"):
     query= "UPDATE datos SET nombre=%s, prerequisito=%s, opcionalidad=%s, semestre=%s, creditos=%s, estado=%s WHERE id=%s"
     cursor.execute(query, (fname, lpre, opc, sem, cre, estado, id))
     mydb.commit()
     clear()
    else:
     return True
```

Función nuevo

Esta función permite poder actualizar un curso seleccionado.

```
def nuevo():
    id=t1.get()
    fname = t2.get()
    lpre = t3.get()
    opc =t4.get()
    sem =t4.get()
    cre =t4.get()
    estado =t4.get()
    query = "INSERT INTO datos VALUES(%s, %s, %s, %s, %s, %s, %s)"
    mydb.commit()
    cursor.execute(query, (id, fname, lpre, opc, sem, cre, estado))
    clear()
```

Función Borrar

Función que contiene el algoritmo para poder borrar un curso seleccionado.

```
def borrar():
    id = t1.get()
    if messagebox.messagebox.askyesno("Quieres eliminar"):

        query="DELETE FROM datos WHERE id= "+id
        cursor.execute(query)
        mydb.commit()
        clear()
    else:
        return True
```

Función Importar y exportar

Contiene la lógica para poder exportar e importar archivos en formato CSV

```
def export():
    if len(mydata) < 1:
        messagebox.showerror("No hay datos")
        return False
    fln = filedialog.asksaveasfilename(initialdir=os.getcwd(), title="Guardar", filetypes=(("CSV File", "*.csv"), ("Todos los Archios", "*.*")))
    with open(fln, mode='w') as myfile:
        exp_writer = csv.writer(myfile, delimiter=',')
        for i in mydata:
            exp_writer.writerow(i)
    messagebox.showinfo("Guardado correctamente", "Su archivo fue exportado "+os.path.basename(fln)+"")


def importcsv():
    mydata.clear()
    fln = filedialog.askopenfilename(initialdir=os.getcwd(), title="Abrir", filetypes=(("Archivo CSV", "*.csv"), ("Todos los Archivos", "*.*")))
    with open(fln) as myfile:
        csvread = csv.reader(myfile, delimiter=",")
        for i in csvread:
            mydata.append(i)
    update(mydata)
```

Función Savedb

Contiene el algoritmo para poder guardar la información cargada en una base de datos.

```
def export():
    if len(mydata) < 1:
        messagebox.showerror("No hay datos")
        return False
    fln = filedialog.asksaveasfilename(initialdir=os.getcwd(), title="Guardar", filetypes=(("CSV File", "*.csv"), ("Todos los Archios", "*.*")))
    with open(fln, mode='w') as myfile:
        exp_writer = csv.writer(myfile, delimiter=',')
        for i in mydata:
            exp_writer.writerow(i)
    messagebox.showinfo("Guardado correctamente", "Su archivo fue exportado "+os.path.basename(fln)+"")


def importcsv():
    mydata.clear()
    fln = filedialog.askopenfilename(initialdir=os.getcwd(), title="Abrir", filetypes=(("Archivo CSV", "*.csv"), ("Todos los Archivos", "*.*")))
    with open(fln) as myfile:
        csvread = csv.reader(myfile, delimiter=",")
        for i in csvread:
            mydata.append(i)
    update(mydata)
```

Los datos antes mencionados son configuraciones dadas por el administrador de dominio, en lo anterior se muestra un ejemplo de los datos que deben ir en cada campo y en el archivo existe una explicación completa de cada apartado.

Las vistas son la interfaz de usuario esto quiere decir que aquí es donde se guarda todo lo que ve el usuario en su entorno gráfico y lo que envía el controlador a la vista

Todos los archivos y directorios no mencionados son parte importante para el funcionamiento del sistema, no se hacen referencia en este documento debido a que solo se enfatizan los archivos que el técnico administrador puede en un dado caso modificar, con conocimiento previo de lo que se hace.

4.Desarrollo

Se desarrollo en lenguaje Python y para su modificación se requiere conocimientos previos del lenguaje.