Trabajo Práctico Entrega Final Python Nivel Inicial

ABMC

Integrantes:

- Cairola, Sebastian
- De Bonis, Christian
- Ibarra, Yoel
- Medina, Ivan
- Samudio, Leandro

La aplicación propuesta se basa en un control de reparaciones de dispositivos que lleva a cabo una organización en relación con locales externos. Por ejemplo, el usuario debe poder ingresar un dispositivo que se lleva a reparar, poder modificar y consultar su estado y eliminar su registro cuando el dispositivo vuelve reparado.

- Alta de dispositivo a reparar (Envío a reparación)
- Baja de dispositivo reparado (Recibo del dispositivo)
- Modificación del estado del dispositivo a reparar.
- Consulta del estado de los dispositivos a reparar. Poder visualizar la lista de dispositivos en proceso de reparación.

El ingreso de un nuevo dispositivo en estado de reparación se genera a través de los siguientes datos:

- Tipo de Dispositivo.
- Modelo de Dispositivo.
- Referencia u observación.
- Fecha y hora de salida de Dispositivo detectada por la aplicación.

El Tipo de Dispositivo debe ser una impresora, notebook, proyector o monitor, por lo que el campo de ingreso sólo acepta letras.

El Modelo del dispositivo acepta caracteres alfanuméricos y especiales ya que generalmente son de uso frecuente.

La referencia u observaciones, al igual que el campo de Modelo del dispositivo acepta caracteres alfanuméricos y especiales.

La Fecha y hora es detectada e ingresada automáticamente por la aplicación.

Nota:El ingreso de datos por parte del usuario se encuentra limitado por expresiones regulares tipo PCRE.

```
def alta():
    global mensaje
    if (re.match(re.compile("^[a-zA-Z]+$"), var_tipo.get())):
        id_registro_ingresado = db.insert_producto(
            var_tipo.get(),
            var_modelo.get(),
            var_referencia.get()
```

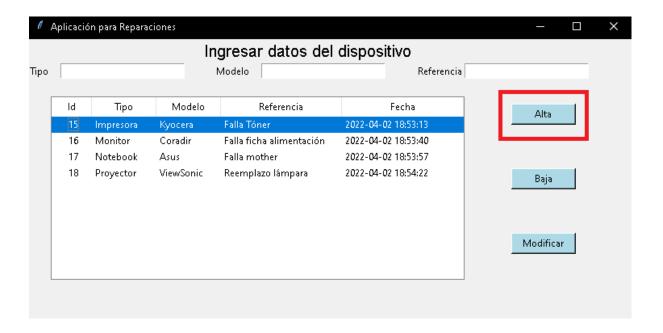
Interfaz de la aplicación.



Nota: La interfaz se encuentra desarrollada con la herramienta Tkinter.

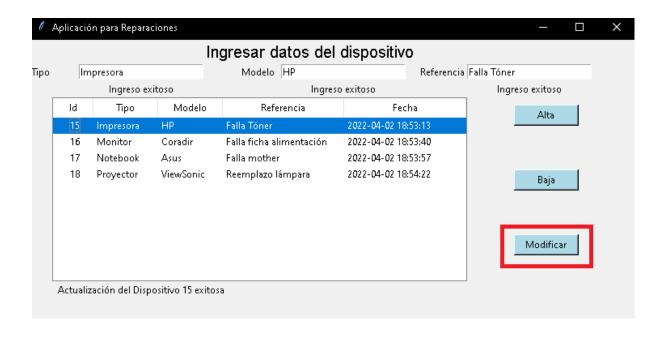
Botón Alta. Luego de ingresar los datos necesarios de un nuevo dispositivo a reparar se debe presionar el botón Alta para que se agregue a nuestra lista de reparaciones en proceso. **Función alta.**

```
def alta():
    global mensaje
    if (re.match(re.compile("^[a-zA-Z]+$"), var_tipo.get())):
        id_registro_ingresado = db.insert_producto(
            var_tipo.get(),
            var_modelo.get(),
            var_referencia.get()
        )
        alta_en_treeview(id_registro_ingresado)
        mensaje['text']="Ingreso de Dispositivo exitoso"
    else:
        mensaje['text']="Tipo no válido, ingrese nuevamente"
```



Botón Modificar. En el caso de que se desee modificar cualquier campo de una determinada reparación ante un error de ingreso o modificación de estado, se debe seleccionar el dispositivo en cuestión e ingresar dicha modificación que se verá reflejada en nuestra lista una vez presionado el botón Modificar. **Función modificar.**

```
def modificar():
    global mensaje
    focused = treeview.focus()
    id_a_modificar = treeview.item(focused)['text']
    if (re.match(re.compile("^[a-zA-Z]+$"), var_tipo.get())):
        db.update_producto(
            var_tipo.get(),
            var_modelo.get(),
            var_referencia.get(),
            id_a_modificar
        actualizar_treeview()
        mensaje['text']="Actualización de Dispositivo"+ str(id_a_modificar) + " exitosa"
    else:
        mensaje['text']="Tipo no válido, ingrese nuevamente"
bu_alta = Button(root, text="Alta", command=alta)
bu_baja = Button(root, text="Baja", command=baja)
bu_modificar = Button(root, text="Modificar", command=modificar)
bu_alta.grid(row=2, column=6, sticky="e")
bu_baja.grid(row=4, column=6, sticky="e")
bu_modificar.grid(row=6, column=6, sticky="e")
actualizar_treeview()
```



Botón Baja. Para eliminar de nuestro listado de reparaciones en proceso un determinado dispositivo, simplemente debemos seleccionarlo y presionar el botón Baja. **Función baja**.

```
def baja():
    global mensaje
    focused = treeview.focus()
    id_a_eliminar = treeview.item(focused)['text']
    db.delete_producto(id_a_eliminar)
    actualizar_treeview()
    mensaje['text'] = "Se dio de baja el registro: "+ str(id_a_eliminar)
```



Consulta. La aplicación se basa y se muestra en una sola ventana por lo que facilita la consulta de los dispositivos que se encuentran en proceso de reparación mostrando una lista ordenada según el ingreso.



Nota: Cada acción que el usuario realice se verá reflejada en un texto informativo en pantalla.

```
def baja():
    global mensaje
    focused = treeview.focus()
    id_a_eliminar = treeview.item(focused)['text']
    db.delete_producto(id_a_eliminar)
    actualizar_treeview()
    mensaje['text'] = "Se dio de baja el registro: "+ str(id_a_eliminar)
```



Nota: La funcionalidad de la aplicación se basa en la relación entre la interfaz con la que el usuario puede interactuar con una base de datos en donde se alojan, modifican o eliminan los registros ingresados. db_config.py

Enlace de Repositorio GIT: https://github.com/Yoellbarra/Tplnicial.git