CUCEI

ACTIVIDAD 03

INTRODUCCIÓN AL ESCALADO DE APLICACIONES PYTHON DISTRIBUIDAS

MATERIA:
Computación tolerante a fallas (I7036)
NRC:
179961
SECCIÓN:
D06
MAESTRO:
López Franco, Michel Emmanuel
HORARIO:
Lunes y Miércoles, 11:00 – 12:55
FECHA DE ENTREGA:
05/02/2023
ALUMNO:
J. Emmanuel Ortiz Renteria (219747611)

Contenido

Introducción 3

Explicación del programa 3

Código Fuente del programa: 4

Introducción

La **concurrencia** es un término utilizado en informática para describir la capacidad de un sistema para ejecutar múltiples tareas simultáneamente. En un entorno concurrente, las tareas pueden comenzar, ejecutarse y completarse en cualquier orden, y pueden superponerse en el tiempo.

El **threading**, o la creación de hilos, es un concepto fundamental en la programación concurrente que permite a un programa realizar múltiples tareas simultáneamente. En términos simples, un hilo es una secuencia de instrucciones que puede ejecutarse independientemente de otros hilos dentro de un mismo programa.

Explicación del programa

El programa es un ejemplo de uso de threading el cual está sujeta a dos funciones encargadas de manejar dos labels con contadores de tiempo, solo que a uno se le suman 4 horas. Esto esta formateado dentro de una GUI de Tkinter para simular un contador. Así demostrando que no hay retraso en el proceso de obtener el tiempo local de equipo y sumarse más horas.

Código Fuente del programa:

```
import sys
import threading
from PyQt5.QtWidgets import QApplication, QMainWindow, QLabel,
QVBoxLayout, QWidget
from PyQt5.QtCore import QTimer
from datetime import datetime, timedelta

#Funcion para imprimir el tiempo actual en el label_actual y repetirlo
def display_time():
    captura_tiempo = datetime.now()
    label_actual.setText("Current Time: " +
captura_tiempo.strftime("%H:%M:%S"))
    QTimer.singleShot(1000, display_time)  #Repetir la funcion cada
segundo para correseguido

#Funcion para imprimir el tiempo actual mas 4 horas en el label_futuro y
repetirlo
def display_time_futuro():
    captura_tiempo = datetime.now()
    label_futuro.setText("Tiempo actual más 4 horas: " + (captura_tiempo +
timedelta(hours=4)).strftime("%H:%M:%S"))
    QTimer.singleShot(1000, display_time_futuro) #Repetir la funcion cada
segundo para correseguido

#Preparando el GUI
app = QApplication(sys.argv)
window = QMainWindow()
window.setGeometry(100, 100, 300, 100)

central_widget = QWidget()
window.setCentralWidget(central_widget)

#Creando el espaciadpo
layout = QVBoxLayout()

label_actual = QLabel("Tiempo actual: ")
layout.addWidget(label_actual)

label_futuro = QLabel("Tiempo actual más 4 horas: ")
layout.addWidget(label_futuro)

central_widget.setLayout(layout)

#Thread que maneja el label y el contador de tirmpo futuro
thread_futuro = threading.Thread(target=display_time_futuro())
thread_futuro.daemon = True
thread_futuro.start()

#Thread que maneja el label y el contador de tiempo actual
thread_presente = threading.Thread(target=display_time())
thread_presente.daemon = True
thread_presente.start()

window.show()
sys.exit(app.exec_())
```

Captura del programa: