Universidad Gerardo Barrios

Facultad de Ciencia y Tecnología

Ingeniería en Sistemas y Redes Informáticas

ASIGNATURA: Programación Computacional III.

TEMA: Contenido a desarrollar (Desarrollo de proyecto Python.)

NOMBRE DE LA ACTIVIDAD: Laboratorio 2 – Semana 18.

PRESENTADO POR:

SMSS063523 – Argueta Portillo, Blanca Leticia.
SMSS084323 – Pérez Bonilla, Xavier Alexander.
SMSS098223 – Salgado Amaya, Jorge Alexis.
SMIS091121 – Zelaya Chávez, Kevin Alejandro.

DOCENTE:
Ing. WILLIAN MONTES.

CICLO VIII - 2024

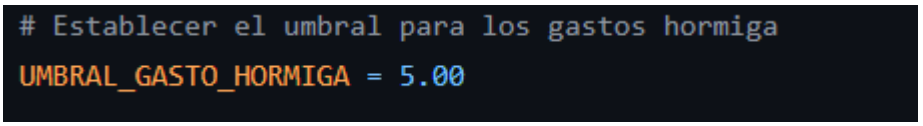
San Miguel, noviembre 24 de 2024

Resolución del plan de trabajo detallado en el tercer avance:

Nuevas Funcionalidades que se han agregado en este tercer avance.

Detección de Gastos Hormiga:
Se verifica si un gasto está por debajo del umbral definido (UMBRAL_GASTO_HORMIGA = $5.00) y, si es así, muestra una alerta para confirmar si el usuario desea agregarlo.

Esta funcionalidad estaba en la lista de alta prioridad, y su implementación sigue lo planeado.



```
# Establecer el umbral para los gastos hormiga
UMBRAL_GASTO_HORMIGA = 5.00
```

```
# Verificar si el gasto es un gasto hormiga
if monto < UMBRAL_GASTO_HORMIGA:
    respuesta = messagebox.askyesno("Gasto Hormiga", f"El monto de {monto} USD parece un gasto hormiga. ¿Estás seguro de que deseas agregarlo?")
    if not respuesta:
        return
```

Simulador de Presupuesto:
Permite al usuario ingresar un presupuesto y calcula el saldo restante considerando los gastos totales. También valida que el presupuesto sea mayor o igual a 0.

Esta funcionalidad cumple parcialmente con el plan: Aunque estaba clasificado como baja prioridad, ya está implementado.

```python
def simular_presupuesto():
    presupuesto_texto = entry_presupuesto.get()

    if not presupuesto_texto:
        messagebox.showerror("Error", "Por favor, ingresa un presupuesto.")
        return

    try:
        presupuesto = float(presupuesto_texto)
    except ValueError:
        messagebox.showerror("Error", "El presupuesto debe ser un número válido.")
        return

    # Validación para presupuesto negativo
    if presupuesto < 0:
        messagebox.showerror("Error", "El presupuesto debe ser mayor o igual a 0.")
        return

    conn = obtener_conexion()
    cursor = conn.cursor()
    cursor.execute("SELECT SUM(monto) FROM gastos")
    gastos_totales = cursor.fetchone()[0] or 0
    saldo_restante = presupuesto - gastos_totales
    messagebox.showinfo("Simulador de Presupuesto", f"Saldo restante: {saldo_restante:.2f} USD")

    conn.close()
```

Reinicio de Datos:

Se agregó la opción de reiniciar todos los datos de ingresos y gastos, lo que no estaba en el plan original pero es útil para gestión de datos.

```python
def reiniciar_presupuesto():
    confirmacion = messagebox.askyesno("Confirmar", "¿Estás seguro de que deseas reiniciar todos los datos?")
    if confirmacion:
        conn = obtener_conexion()
        cursor = conn.cursor()
        cursor.execute("DELETE FROM ingresos")
        cursor.execute("DELETE FROM gastos")
        conn.commit()
        conn.close()
        actualizar_progreso()
        messagebox.showinfo("Reinicio", "Se han reiniciado todos los datos.")
```

Funcionalidades Modificadas o que se han Mejorado.

Análisis de Gastos:

Se utiliza Plotly para crear gráficos circulares que muestran la distribución de los gastos por categoría.

Ahora se ejecuta en un hilo independiente con threading para evitar bloquear la interfaz, consideramos que es una mejora notable para el funcionamiento del programa.

```
def generar_grafico_analisis():
    conn = obtener_conexion()
    cursor = conn.cursor()
    cursor.execute("SELECT categoria, SUM(monto) as total FROM gastos GROUP BY categoria")
    datos = cursor.fetchall()

    if datos:
        df = pd.DataFrame(datos, columns=["Categoria", "Total"])
        fig = go.Figure(data=[go.Pie(labels=df["Categoria"], values=df["Total"], textinfo="label+percent", hole=0.3)])
        fig.update_layout(title="Distribución de Gastos por Categoría", template="plotly_white")
        fig.show()
    else:
        messagebox.showinfo("Sin datos", "No hay gastos registrados para analizar.")
    conn.close()

    label_cargando.config(text="")
    label_cargando.pack_forget()
```

Progreso Mensual:

Se utiliza Plotly para mostrar un gráfico comparativo (barras agrupadas) de ingresos vs. gastos por mes y ahora el gráfico incluye barras más delgadas y ajusta su diseño para mayor claridad.

```
def mostrar_progreso_mensual():
    label_cargando.config(text="Generando gráfico, por favor espere...")
    label_cargando.pack(pady=5)
    thread = threading.Thread(target=generar_grafico_progreso_mensual)
    thread.daemon = True
    thread.start()

def generar_grafico_progreso_mensual():
    conn = obtener_conexion()
    cursor = conn.cursor()
    cursor.execute("""
        SELECT strftime('%Y-%m', fecha) as mes, SUM(monto) as ingresos FROM ingresos GROUP BY mes
    """)
    ingresos_mensuales = cursor.fetchall()
```

Cómo contribuyen al propósito general del proyecto.

Funciones para la Gestión de Ingresos y Gastos.

Las funciones de ingreso y gasto son la base para que los usuarios tengan claridad sobre su situación financiera.

agregar_ingreso()
Permite registrar ingresos en la base de datos con su respectiva fecha.

Proporciona al usuario un seguimiento claro de las entradas financieras, que es fundamental para calcular su capacidad de ahorro y progreso.

agregar_gasto()
Registra los gastos, asociándose con una categoría y una fecha.
Verifica si un gasto es menor al umbral de gastos hormiga y advierte al usuario.

Ayuda a organizar los gastos de manera estructurada, facilitando la identificación de áreas problemáticas (e.g., gastos hormiga).

actualizar_progreso()
Calcula la diferencia entre ingresos y gastos y actualiza el progreso financiero en tiempo real.

Da al usuario una visión clara de su estado financiero, fomentando decisiones más informadas.

Funciones de Análisis Visual

Los gráficos y resúmenes ayudan a interpretar datos, detectar patrones y ajustar comportamientos de gasto.

mostrar_analisis()
Genera un gráfico circular que muestra la distribución de gastos por categoría usando Plotly.

Facilita la comprensión visual de los hábitos de gasto, permitiendo identificar categorías que podrían reducirse.

mostrar_progreso_mensual()
Muestra un gráfico comparativo (ingresos vs. gastos) por mes.
Permite evaluar la consistencia de los ingresos y gastos a lo largo del tiempo, destacando tendencias financieras y posibles desequilibrios.

Funciones de Presupuesto.

Las funciones de simulación y progreso mensual apoyan a los usuarios a establecer límites y metas financieras.

simular_presupuesto()
Calcula el saldo restante tras considerar los gastos acumulados frente al presupuesto ingresado.

Ayuda al usuario a planificar y evaluar si sus gastos están dentro del presupuesto.

ver_presupuesto()
Muestra al usuario el presupuesto total definido.

Refuerza la idea de control financiero al hacer que los objetivos sean tangibles.

reiniciar_presupuesto()
Borra todos los registros de ingresos y gastos.
Proporciona un punto de partida limpio para reestructurar las finanzas cuando el usuario lo necesite.

Funciones de Estado de Cuenta.

Funciones como el estado de cuenta y la categorización de gastos brindan detalles precisos sobre el flujo de dinero.

mostrar_estado_cuenta()
Muestra una vista detallada de ingresos y gastos en una tabla organizada.

Proporciona transparencia y control, permitiendo a los usuarios revisar transacciones pasadas y su impacto en las finanzas.

Optimización de Experiencia del Usuario.

Validaciones y alertas (como en los gastos hormiga) fomentan un uso más consciente de los recursos.

generar_grafico_analisis() y generar_grafico_progreso_mensual() (con threading)

Ejecutan la generación de gráficos en hilos independientes para evitar bloquear la interfaz.

Mejora la experiencia del usuario al mantener la interfaz receptiva y eficiente.

Validaciones de datos (e.g., validar_fecha())
Garantizan que los datos ingresados sean correctos antes de almacenarlos en la base de datos.

Evitan errores y aseguran la integridad de los registros, mejorando la fiabilidad del sistema.

Evaluación.
Alta Prioridad:
Alertas de Gastos Hormiga: Implementada con éxito.
Establecimiento de Objetivos de Ahorro: Aún pendiente.

Media Prioridad:
Notificaciones sobre Límites Presupuestarios: Pendiente.
Sugerencias de Ahorro Personalizado: Pendiente.

Baja Prioridad:
Simulador de Presupuesto: Implementado antes de lo esperado.
Se desarrolló antes de lo esperado ya que notamos que es una funcionalidad más sencilla de desarrollar y ayuda a aumentar la funcionalidad general del sistema.

Cumplimiento del Plan:

El proyecto se ha estado desarrollando en su mayor parte siguiendo el plan original y se han logrado realizar modificaciones en el proyecto los cuales han aportado una mejora notable en el funcionamiento general del programa.

Objetivos Faltantes y Plan de Desarrollo:

Alertas de Gastos Hormiga Avanzadas.

Estado:
Ya se emiten advertencias básicas al detectar un gasto menor a un umbral predefinido.

Planificación:
Ampliar las alertas para incluir notificaciones acumulativas por categoría o tipo de gasto.

Sugerencias Personalizadas de Ahorro.

Estado:
No implementada.

Planificación:
Generar sugerencias dinámicas basadas en el análisis de ingresos y gastos previos. Por ejemplo, sugerencias para reducir gastos en categorías que superen el 50% del presupuesto.

Establecimiento de Objetivos de Ahorro.

Estado:
No implementada.

Planificación:
Permitir al usuario definir metas de ahorro específicas (e.g., $1000 en 6 meses) y mostrar el progreso alcanzado en tiempo real.

Notificarme cuando los gastos comprometan alcanzar la meta.

Análisis Avanzado de Gastos.

Estado:
Gráficos básicos implementados (circular y comparativo mensual).

Planificación:
Añadir gráficos más detallados, como:

- Tendencias de gasto a lo largo del tiempo.
- Comparaciones interanuales (si aplica).
- Incorporar herramientas para filtrar análisis por rango de fechas o categorías.

Simulador de Presupuesto Mejorado.

Estado:
Cálculo básico implementado.

Planificación:
Ampliar el simulador para incluir escenarios personalizados, como:

- Gastos extraordinarios (e.g., vacaciones, emergencias).
- Ajustes en ingresos previstos.

Alcance logrado del desarrollo:

Contribución del Avance al Propósito General del proyecto.

Este avance fortalece el propósito del proyecto al consolidar herramientas prácticas para la gestión financiera personal. La incorporación de alertas de gastos hormiga, análisis avanzado de gastos, y funciones como el simulador de presupuesto y objetivos de ahorro, permite a los usuarios:

Tener mayor conciencia financiera: Detectar gastos innecesarios y controlar el presupuesto.

Facilitar la planificación y el ahorro: Monitorear metas y adaptar su comportamiento financiero.

Visualizar información clave: Gráficos intuitivos y análisis detallados que promueven decisiones informadas.

Estas mejoras alinean directamente el sistema con su objetivo principal: darle un mayor control a los usuarios para poder gestionar su dinero de manera eficiente y alcanzar sus metas financieras.