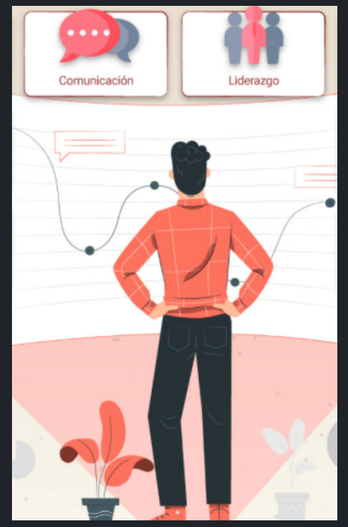
<!--Pes2025 Proramacion 1-->

GastroController 5000 {

<Por=["Mario Castro","Diego Chan]>

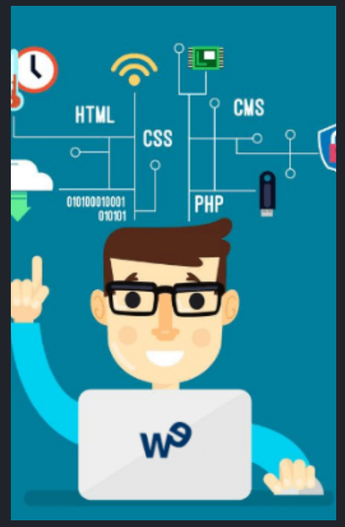
}

Equipo {



MARIO CASTRO

DIRECCIÓN DE PROYECTO



DIEGO CHAN

DESARROLLADOR

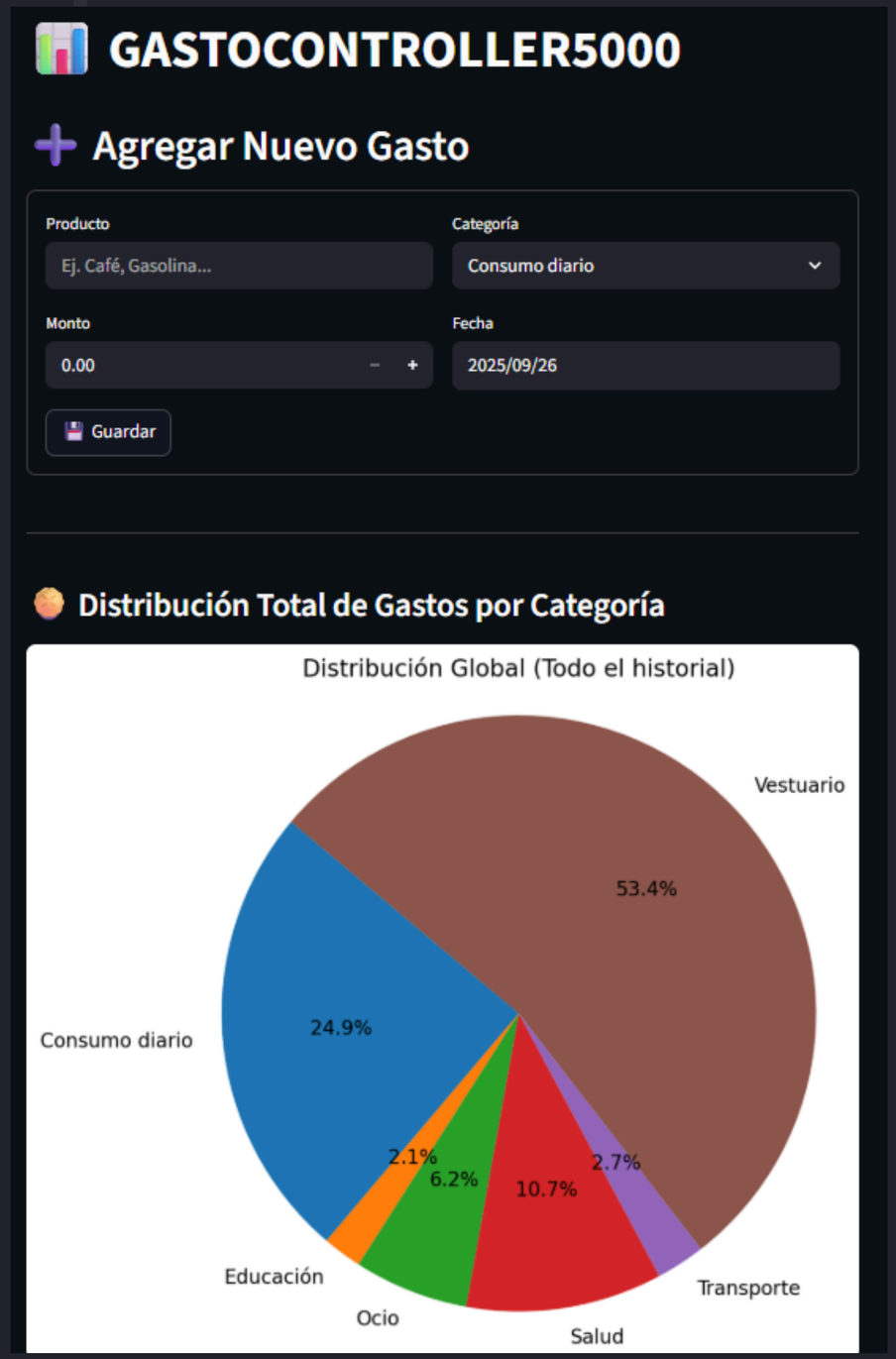
}

Contenidos

Introducción {

GastoController5000 es una aplicacion que nacio para suplir las necesidades del ejecutivo de hoy, su inovadora interfaz y herramientas de ultima generacion hace que el control de gastos sea tan facil como un par de click.

Sus funciones principal es proporcionar una proyectada en el tiempo de a donde se esta dirigiendo sus gastos y donde deberia tener cuidado. De manera grafica podra saber siempre donde esta su dinero o a donde fue.



}

Puntos clave {

01

Ingeso de datos por Formulario Web

02

Se visualiza gastos graficamente

03

Se hace un listado de gastosdescargable

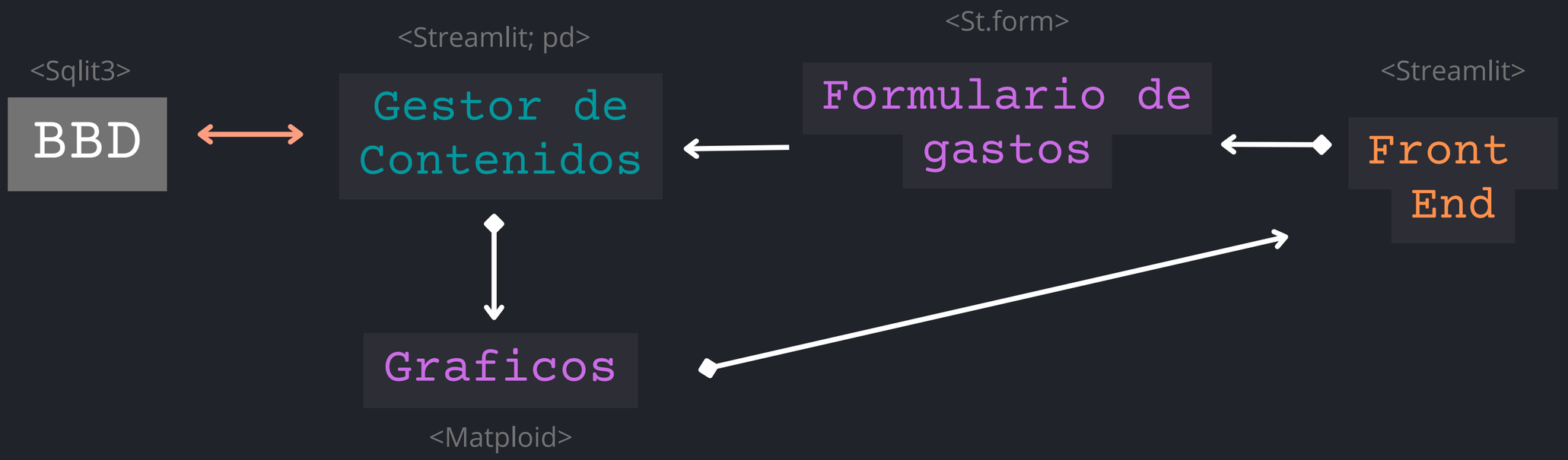
04

Optimizacion de busquedas

05

Accesibilidad en multiplataforma

Estructura {


<Sqlit3>
BBD
<Streamlit; pd>
Gestor de Contenidos
<St.form>
Formulario de gastos
<Streamlit>
Front End
Graficos
<Matploid>

}

Funciones Auxilires {

BASE DE DATOS SQL

```
# Inicializar la base de datos
def inicializar_base_de_datos():
    conexion = sqlite3.connect(ARCHIVO_BD)
    cursor = conexion.cursor()
    cursor.execute('''
        CREATE TABLE IF NOT EXISTS gastos (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
```

BASE DE DATOS CSV

```
def load_expenses():
    """Loads expense data from the CSV file."""
    global expenses_df
    if os.path.exists(CSV_FILE):
        expenses_df = pd.read_csv(CSV_FILE)
        # Ensure the 'Amount' column is numeric after loading
        expenses_df['Amount'] = pd.to_numeric(expenses_df['Amount'], errors='coe
    else:
```

Con librerias pandas o SQLit3 se realizo implementacion por pandas para tratamientos de base de datos CSV, y manipulacion SQL para la version con BD

}

Funciones Itermedias {

BUSQUEDAS Y FILTROS

```python
# Agrupar gastos por periodo (semanal, mensual, anual)
def agrupar_por_periodo(df, periodo='Mensual'):
    df['fecha'] = pd.to_datetime(df['fecha'])
    if periodo == 'Semanal':
        df['periodo'] = df['fecha'].dt.to_period('W').apply(lambda r: r.start_time)
    elif periodo == 'Mensual':
        df['periodo'] = df['fecha'].dt.to_period('M').apply(lambda r: r.start_time)
    elif periodo == 'Anual':
        df['periodo'] = df['fecha'].dt.to_period('Y').apply(lambda r: r.start_time)
    else:
        df['periodo'] = df['fecha']  # Diario
```

METRICAS Y ANALISIS TEMPORAL

```python
# Tabla y métricas del análisis temporal
st.header("📑 Gastos Filtrados (Análisis Temporal)")
if not gastos_filtrados_df.empty:
    st.dataframe(gastos_filtrados_df[['producto', 'categoria', 'monto', 'fecha']].reset_index(
    total = gastos_filtrados_df['monto'].sum()
    promedio = gastos_filtrados_df['monto'].mean()
    st.metric("💰 Total Gastos (Filtrado)", f"Q{total:,.2f}")
    st.metric("📈 Promedio por Gasto", f"Q{promedio:,.2f}")
else:
```

Con el apoyo de pandas, streamlit y operaciones de toma de deciciones se realiza las metricas de gastos.

}

Funciones FRONTEND {

STREMLIT FORMS

```python
# Formulario para agregar gasto
st.header("➕ Agregar Nuevo Gasto")
with st.form("Formulario de Gasto"):
    col1, col2 = st.columns(2)
    with col1:
        producto = st.text_input("Producto", placeholder="Ej. Café, Gasolina...")
    with col2:
        categoria = st.selectbox("Categoría", options=CATEGORIAS)
```

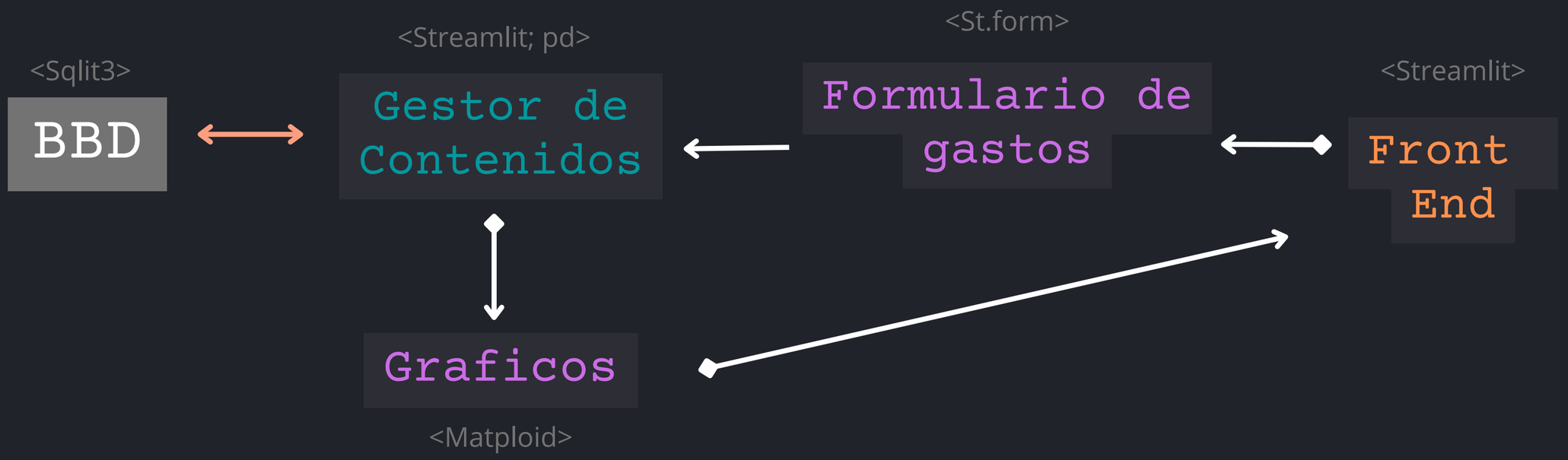
MATPLOTLIB FIGURE

```python
# 🥚 Gráfica de pastel: TOTAL DE TODOS LOS TIEMPOS (sin filtros)
st.subheader("🥚 Distribución Total de Gastos por Categoría")
if not todos_los_gastos_df.empty:
    totales_por_categoria = todos_los_gastos_df.groupby('categoria')['monto'].sum()
    fig_pastel, ax_pastel = plt.subplots(figsize=(6, 6))
    ax_pastel.pie(totales_por_categoria, labels=totales_por_categoria.index, autopct='%1.1f
    ax_pastel.set_title("Distribución Global (Todo el historial)")
    ax_pastel.axis('equal')
    st.pyplot(fig_pastel)
else:
    st.info("No hay datos históricos para mostrar.")
```

Con librerias streamlit para mostrar el FrontEnd y MATPLOTLIB para graficas de las proyecciones de gastos en el tiempo

}

Estructura {


<Sqlit3>
BBD
<Streamlit; pd>
Gestor de Contenidos
<St.form>
Formulario de gastos
<Streamlit>
Front End
Graficos
<Matploid>

}

El Código {

Vamos a visual code a revisar el codigo

}

}

Planificación {

FASE 1

Propuesta de proyecto , lluvia de ideas y eleccion del proyecto, un gestor dee gastos con proyecciones.

FASE 2

Verificacion de que libreriass nos ayudan a resolver el proyecto escogido

FASE 3

Levantamiento de los entornos de programacion, librerias y planeaccion de la base de datos, estructuras de cotejo y graficacion

FASE 4

Programacion en version de base de datos por CSV, se define stremlit para muestra e iteraccion con el usuario

ACTUALIDAD

FASE 5

Subirlo a un entorno real (Web) y entrega del primer prototipo.

}

PES 2025 Programacion I

Gracias {

<Por="Mario castro", “Diego Cham”>

}