

Actividad Colaborativa Módulo 9:

Guía de Diseño Efectivo para Dashboards con Dash

1. Introducción

Los dashboards son herramientas clave para visualizar datos deportivos. Con **Dash** de Plotly, podemos crear interfaces interactivas con Python que combinan datos, gráficos y controles en tiempo real. Esta guía ofrece principios de diseño, organización de layouts, ejemplos prácticos y recomendaciones específicas para aplicaciones deportivas.

2. Principios de diseño visual

Jerarquía visual

- Coloca **métricas clave** (goles, posesión, tiros) en la parte superior.
- Usa **tamaños de letra** y **colores** para destacar lo importante.

Contraste y color

- Usa colores de equipos (por ejemplo, azul vs rojo) para diferenciar visualmente.
- Evita saturar: máximo 2 colores principales + neutros.

Consistencia

- Usa el mismo estilo de fuente, colores y componentes en todo el dashboard.
- Todos los gráficos deben tener el mismo fondo y estilo de leyenda.

Legibilidad

- Números grandes y claros.
- Evita sobrecargar los gráficos con demasiada información.

3. Organización de layouts

Usamos **Dash Bootstrap Components (dbc)** para estructurar el diseño.

Estructura recomendada:

plaintext

CopiarEditar

Encabezado / Título

Filtros (dropdowns)

Métricas clave (cards)

Gráficos principales

Detalles adicionales

Librerías:

python

CopiarEditar

```
import dash
```

```
import dash_bootstrap_components as dbc
```

```
import dash_core_components as dcc
```

```
import dash_html_components as html
```

4. Mejores prácticas UX

- Interactividad intuitiva: usa Dropdown, Checklist, Slider.
- Feedback visual: usa dcc.Loading para mostrar carga.
- Layout responsive: usa dbc.Row y dbc.Col.

5. Patrones de diseño recomendados

- **Cards** para mostrar estadísticas clave:

python

CopiarEditar

```
dbc.Card([
    dbc.CardBody([
        html.H5("Goles", className="card-title"),
        html.P("3", className="card-text")
    ])
])
```

- **Tabs** para cambiar entre temporadas o ligas:

python

CopiarEditar

```
dcc.Tabs([
    dcc.Tab(label="Temporada 2022", children=[...]),
```

```
    dcc.Tab(label="Temporada 2023", children=[...])
])
```

6. Ejemplo práctico (Fútbol)

Supongamos una app con datos de partidos de LaLiga:

Filtros:

```
python
CopiarEditar
dcc.Dropdown(
    id='equipo-dropdown',
    options=[{'label': i, 'value': i} for i in equipos],
    value='Real Madrid'
)
```

Callback:

```
python
CopiarEditar
@app.callback(
    Output('grafico-goles', 'figure'),
    Input('equipo-dropdown', 'value')
)
def actualizar_goles(equipo):
    df_filtrado = df[df['equipo'] == equipo]
    fig = px.bar(df_filtrado, x="partido", y="goles")
    return fig
```

7. Reflexión sobre el diseño

Decidí colocar las tarjetas de estadísticas clave arriba para que el usuario tenga una vista rápida del rendimiento del equipo. Elegí colores que representan a los equipos para hacer la información más familiar. Usé Bootstrap para mantener el diseño ordenado y responsivo. La navegación mediante tabs permite explorar otras temporadas sin recargar la vista.

8. Reflexión final de la actividad

Decidí organizar el dashboard colocando primero un dropdown para seleccionar el equipo, ya que es la variable principal del análisis. Luego, mostré los gráficos en una sección amplia y centrada para que sean fáciles de interpretar. Elegí un diseño limpio y responsivo usando Dash, y traté de usar colores relacionados con los equipos para facilitar la lectura. Aprendí a manejar callbacks para interactividad básica y cómo organizar componentes visuales.

Antonio Llagas Galván.