📊 Sesi 3

Visualisasi Data

HARI 1: Fundamental Analisis Data

Durasi: 1 jam 30 menit (15:30 - 17:00)
Notebooks:

- `01_matplotlib_seaborn.ipynb`
- `02_plotly_interactive.ipynb`

🎯 Tujuan Sesi

Setelah sesi ini, Anda bisa:

- ✅ Membuat visualisasi statistik dengan Matplotlib & Seaborn
- ✅ Membuat grafik interaktif dengan Plotly
- ✅ Memilih tipe chart yang tepat untuk data
- ✅ Menerapkan prinsip data storytelling
- ✅ Customize styling dan theming
- ✅ Export visualisasi dalam berbagai format
- ✅ Membuat dashboard sederhana dengan subplots

📋 Agenda Sesi

Waktu	Topik	Durasi
15:30 - 15:40	Kick-off & Visual Story Goals	10 min
15:40 - 16:05	Matplotlib Fundamentals	25 min
16:05 - 16:30	Seaborn Statistical Plots	25 min
16:30 - 16:50	Plotly Interaktif & Dashboard Mini	20 min
16:50 - 17:00	Data Storytelling & Q&A	10 min

📊 Part 1: Matplotlib # 📊 Part 1: Matplotlib & Seaborn

Penjelasan: Matplotlib untuk buat grafik dasar, Seaborn untuk grafik statistik yang lebih cantik

Static Visualizations for Statistical Analysis

Matplotlib:

- Foundation untuk plotting di Python
- Low-level control
- Customizable

Static Visualizations for Statistical Analysis (lanjutan)

Seaborn:

- Built on top of Matplotlib
- Statistical plots
- Beautiful default themes
- High-level interface

🛠️ Setup & Import

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Load data
df = pd.read_parquet('../../datasets/rup/RUP-PaketPenyedia-Terumumkan-2025.parquet')

# Matplotlib settings
plt.style.use('seaborn-v0_8')
plt.rcParams['figure.figsize'] = (12, 6)
plt.rcParams['font.size'] = 10

# Seaborn settings
sns.set_theme(style="whitegrid")
sns.set_palette("husl")

%matplotlib inline  # Jupyter magic command
```

Matplotlib Basic: Figure & Axes

```
# Simple plot
plt.figure(figsize=(10, 6))
plt.plot([1, 2, 3, 4], [1, 4, 2, 3])
plt.title('Simple Line Plot')
plt.xlabel('X Axis')
plt.ylabel('Y Axis')
plt.grid(True, alpha=0.3)
plt.show()

# Object-oriented approach (recommended)
fig, ax = plt.subplots(figsize=(10, 6))
ax.plot([1, 2, 3, 4], [1, 4, 2, 3], marker='o')
ax.set_title('Line Plot')
ax.set_xlabel('X Axis')
ax.set_ylabel('Y Axis')
ax.grid(True, alpha=0.3)
plt.show()
```

Bar Chart: Categorical Data

```
# Count paket per metode
metode_counts = df['metode_pengadaan'].value_counts()

# Vertical bar chart
fig, ax = plt.subplots(figsize=(10, 6))
metode_counts.plot(kind='bar', ax=ax, color='steelblue', edgecolor='black')
ax.set_title('Jumlah Paket per Metode Pengadaan', fontsize=14, fontweight='bold')
ax.set_xlabel('Metode Pengadaan', fontsize=12)
ax.set_ylabel('Jumlah Paket', fontsize=12)
ax.set_xticklabels(ax.get_xticklabels(), rotation=45, ha='right')

# Add value labels on bars
for i, v in enumerate(metode_counts):
    ax.text(i, v + 50, str(v), ha='center', va='bottom', fontweight='bold')

plt.tight_layout()
plt.show()
```

📊 Horizontal Bar Chart: Top N

```
# Top 10 satker by total pagu
top_satker = df.groupby('nama_satker')['pagu'].sum().sort_values(ascending=True).tail(10)

# Horizontal bar chart
fig, ax = plt.subplots(figsize=(10, 8))
top_satker.plot(kind='barh', ax=ax, color='coral')
ax.set_title('Top 10 Satker by Total Pagu', fontsize=14, fontweight='bold')
ax.set_xlabel('Total Pagu (Rupiah)', fontsize=12)
ax.set_ylabel('Satuan Kerja', fontsize=12)

# Format x-axis as currency
ax.xaxis.set_major_formatter(plt.FuncFormatter(lambda x, p: f'{x/1e9:.1f}B'))

plt.tight_layout()
plt.show()
```

📈 Histogram: Distribution Analysis

```
# Distribution of pagu
fig, ax = plt.subplots(figsize=(12, 6))

# Histogram
ax.hist(df['pagu'], bins=50, edgecolor='black', alpha=0.7, color='skyblue')

# Add mean and median lines
mean_pagu = df['pagu'].mean()
median_pagu = df['pagu'].median()

ax.axvline(mean_pagu, color='red', linestyle='--', linewidth=2, label=f'Mean: {mean_pagu/1e6:.1f}M')
ax.axvline(median_pagu, color='green', linestyle='--', linewidth=2, label=f'Median: {median_pagu/1e6:.1f}M')

ax.set_title('Distribusi Pagu Pengadaan', fontsize=14, fontweight='bold')
ax.set_xlabel('Pagu (Rupiah)', fontsize=12)
ax.set_ylabel('Frekuensi', fontsize=12)
ax.legend()
ax.grid(True, alpha=0.3)

plt.tight_layout()
plt.show()
```

🥧 Pie Chart: Composition

```
# Pie chart untuk metode pengadaan
metode_counts = df['metode_pengadaan'].value_counts()

fig, ax = plt.subplots(figsize=(10, 10))
colors = sns.color_palette('pastel')[0:len(metode_counts)]

ax.pie(metode_counts,
       labels=metode_counts.index,
       autopct='%1.1f%%',
       startangle=90,
       colors=colors,
       explode=[0.05] * len(metode_counts))  # Slight separation

ax.set_title('Distribusi Metode Pengadaan', fontsize=14, fontweight='bold', pad=20)

plt.show()
```

Catatan: Pie charts best untuk max 5-7 categories

Line Chart: Time Series

```
# Time series: Pengumuman per hari
df['tgl_pengumuman_paket'] = pd.to_datetime(df['tgl_pengumuman_paket'])
daily_counts = df.set_index('tgl_pengumuman_paket').resample('D').size()

fig, ax = plt.subplots(figsize=(14, 6))
ax.plot(daily_counts.index, daily_counts.values, linewidth=2, color='steelblue')

# Add moving average
ma_7 = daily_counts.rolling(window=7).mean()
ax.plot(ma_7.index, ma_7.values, linewidth=2, color='red', linestyle='--',
        label='7-Day Moving Average')

ax.set_title('Trend Pengumuman Paket Pengadaan', fontsize=14, fontweight='bold')
ax.set_xlabel('Tanggal', fontsize=12)
ax.set_ylabel('Jumlah Paket', fontsize=12)
ax.legend()
ax.grid(True, alpha=0.3)

plt.tight_layout()
plt.show()
```

Seaborn: Statistical Plots

Why Seaborn?

Kelebihan

- ✅ Beautiful defaults
- ✅ Statistical functions built-in
- ✅ Less code untuk complex plots
- ✅ Better color palettes
- ✅ Automatic legend/labels

Kapan Dipakai

- Distribution analysis
- Relationship plots
- Categorical comparisons
- Statistical estimation
- Multi-plot grids

```
import seaborn as sns
sns.set_theme(style="whitegrid")
sns.set_palette("husl")
```

Seaborn: Histogram & KDE

```
# Distribution plot dengan KDE (Kernel Density Estimate)
fig, axes = plt.subplots(1, 2, figsize=(15, 6))

# Histogram dengan KDE
sns.histplot(data=df, x='pagu', bins=50, kde=True, ax=axes[0])
axes[0].set_title('Distribution dengan KDE', fontsize=12, fontweight='bold')
axes[0].set_xlabel('Pagu (Rupiah)')

# KDE plot only
sns.kdeplot(data=df, x='pagu', fill=True, ax=axes[1])
axes[1].set_title('Kernel Density Estimate', fontsize=12, fontweight='bold')
axes[1].set_xlabel('Pagu (Rupiah)')

plt.tight_layout()
plt.show()
```

📦 Box Plot & Violin Plot

```
fig, axes = plt.subplots(1, 2, figsize=(16, 6))

# Box plot: Pagu by metode
sns.boxplot(data=df, x='metode_pengadaan', y='pagu', ax=axes[0])
axes[0].set_title('Box Plot: Pagu by Metode', fontsize=12, fontweight='bold')
axes[0].set_xticklabels(axes[0].get_xticklabels(), rotation=45, ha='right')
axes[0].set_ylabel('Pagu (Rupiah)')

# Violin plot: Shows distribution + box plot
sns.violinplot(data=df, x='metode_pengadaan', y='pagu', ax=axes[1])
axes[1].set_title('Violin Plot: Pagu by Metode', fontsize=12, fontweight='bold')
axes[1].set_xticklabels(axes[1].get_xticklabels(), rotation=45, ha='right')
axes[1].set_ylabel('Pagu (Rupiah)')

plt.tight_layout()
plt.show()
```

Violin plot = Box plot + KDE 🎻

Count Plot & Bar Plot

```
fig, axes = plt.subplots(1, 2, figsize=(16, 6))

# Count plot (automatic counting)
sns.countplot(data=df, x='jenis_pengadaan', ax=axes[0], palette='Set2')
axes[0].set_title('Count Plot: Jenis Pengadaan', fontsize=12, fontweight='bold')
axes[0].set_xticklabels(axes[0].get_xticklabels(), rotation=45, ha='right')

# Bar plot (dengan penggabungan)
jenis_avg = df.groupby('jenis_pengadaan')['pagu'].mean().reset_index()
sns.barplot(data=jenis_avg, x='jenis_pengadaan', y='pagu', ax=axes[1], palette='Set2')
axes[1].set_title('Average Pagu by Jenis', fontsize=12, fontweight='bold')
axes[1].set_xticklabels(axes[1].get_xticklabels(), rotation=45, ha='right')
axes[1].set_ylabel('Average Pagu (Rupiah)')

plt.tight_layout()
plt.show()
```

🔥 Heatmap: Correlation Matrix

```
# Select numeric columns
numeric_df = df.select_dtypes(include=[np.number])

# Calculate correlation matrix
corr_matrix = numeric_df.corr()

# Heatmap
fig, ax = plt.subplots(figsize=(10, 8))
sns.heatmap(corr_matrix,
            annot=True,           # Show values
            fmt='.2f',            # Format
            cmap='coolwarm',      # Color map
            center=0,             # Center at 0
            square=True,          # Square cells
            linewidths=1,         # Cell borders
            cbar_kws={'shrink': 0.8},
            ax=ax)

ax.set_title('Correlation Matrix', fontsize=14, fontweight='bold', pad=20)
plt.tight_layout()
plt.show()
```

Scatter Plot: Relationship

```
# Scatter plot with regression line
fig, ax = plt.subplots(figsize=(10, 6))

sns.scatterplot(data=df.sample(1000),  # Sample untuk performance
                x='pagu',
                y='nilai_pdn_pekerjaan',
                hue='metode_pengadaan',
                style='jenis_pengadaan',
                s=100,
                alpha=0.6,
                ax=ax)

ax.set_title('Relationship: Pagu vs Nilai PDN', fontsize=14, fontweight='bold')
ax.set_xlabel('Pagu (Rupiah)', fontsize=12)
ax.set_ylabel('Nilai PDN Pekerjaan', fontsize=12)
ax.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

plt.tight_layout()
plt.show()
```

🎨 Customizing Seaborn

```
# Set theme
sns.set_theme(style="darkgrid")  # Opsi: white, dark, whitegrid, darkgrid, ticks

# Set context (scales elements)
sns.set_context("talk")  # Opsi: paper, notebook, talk, poster

# Set color palette
sns.set_palette("husl")  # Opsi: deep, muted, bright, pastel, dark, colorblind

# Custom palette
custom_colors = ["#FF6B6B", "#4ECDC4", "#45B7D1", "#FFA07A"]
sns.set_palette(custom_colors)

# Apply to plot
fig, ax = plt.subplots(figsize=(10, 6))
sns.barplot(data=df, x='metode_pengadaan', y='pagu', ax=ax)
plt.show()
```

📊 Part 2: Plotly - Grafik yang Bisa Diklik

Penjelasan: Plotly buat grafik interaktif: bisa zoom, hover lihat detail, klik legend

Interactive Visualizations

Keunggulan Plotly:

- ✅ **Interaktivity:** Hover, zoom, pan, select
- ✅ **Export:** PNG, SVG, HTML
- ✅ **Responsive:** Auto-resize
- ✅ **Professional:** Publication-quality
- ✅ **3D Support:** 3D scatter, surface plots
- ✅ **Animations:** Animated charts

🛠️ Plotly Setup

```
import plotly.express as px
import plotly.graph_objects as go
from plotly.subplots import make_subplots

# Plotly Express: Tinggi-level, quick plots
# Graph Objects: Rendah-level, full control

# Check version
import plotly
print(f"Plotly version: {plotly.__version__}")
```

Plotly Express vs Graph Objects:

- **Express:** Quick, concise, tapi limited kustomisasi
- **Graph Objects:** Lengkap control, lebih verbose

📊 Bar Chart Interaktif

```
# Top satker dengan plotly
top_satker = df.groupby('nama_satker')['pagu'].sum().sort_values(ascending=False).head(10)
top_satker_df = top_satker.reset_index()
top_satker_df.columns = ['Satker', 'Total Pagu']

fig = px.bar(top_satker_df,
             x='Satker',
             y='Total Pagu',
             title='Top 10 Satker by Total Pagu',
             labels={'Total Pagu': 'Total Pagu (Rupiah)'},
             color='Total Pagu',
             color_continuous_scale='Blues')

fig.update_layout(
    xaxis_tickangle=-45,
    height=600,
    hovermode='x unified'
)

fig.show()
```

Hover untuk lihat detail! 🖱️

📈 Histogram Interaktif

```
# Distribution dengan bins interaktif
fig = px.histogram(df,
                   x='pagu',
                   nbins=50,
                   title='Distribusi Pagu Pengadaan (Interactive)',
                   labels={'pagu': 'Pagu (Rupiah)', 'count': 'Frekuensi'},
                   color_discrete_sequence=['steelblue'])

# Add mean & median as vertical lines
mean_pagu = df['pagu'].mean()
median_pagu = df['pagu'].median()

fig.add_vline(x=mean_pagu, line_dash="dash", line_color="red",
              annotation_text=f"Mean: {mean_pagu/1e9:.2f}B")
fig.add_vline(x=median_pagu, line_dash="dash", line_color="green",
              annotation_text=f"Median: {median_pagu/1e9:.2f}B")

fig.update_layout(height=600)
fig.show()
```

🎯 Scatter Plot dengan Hover Info

```
# Scatter plot dengan custom hover
sample_df = df.sample(1000)

fig = px.scatter(sample_df,
                 x='pagu',
                 y='nilai_pdn_pekerjaan',
                 color='metode_pengadaan',
                 size='pagu',
                 hover_data=['nama_paket', 'nama_satker'],
                 title='Pagu vs Nilai PDN (Interactive)',
                 labels={'pagu': 'Pagu (Rupiah)',
                         'nilai_pdn_pekerjaan': 'Nilai PDN'},
                 opacity=0.6)

fig.update_layout(height=700)
fig.show()
```

Click legend untuk toggle visibility! 👆

📦 Box Plot Interaktif

```
# Box plot dengan comparison
fig = px.box(df,
             x='metode_pengadaan',
             y='pagu',
             color='jenis_pengadaan',
             title='Distribusi Pagu by Metode & Jenis',
             labels={'pagu': 'Pagu (Rupiah)'},
             notched=True,  # Notched box plot
             points='outliers')  # Show outlier points

fig.update_layout(
    height=700,
    xaxis_tickangle=-45
)

fig.show()
```

🌞 Sunburst Chart: Hierarchical Data

```
# Sunburst: Hierarchical composition
metode_jenis = df.groupby(['metode_pengadaan', 'jenis_pengadaan']).size().reset_index(name='count')

fig = px.sunburst(metode_jenis,
                  path=['metode_pengadaan', 'jenis_pengadaan'],
                  values='count',
                  title='Sunburst: Metode → Jenis Pengadaan',
                  color='count',
                  color_continuous_scale='RdYlBu_r')

fig.update_layout(height=700)
fig.show()
```

Click untuk zoom in/out! 🌞

🗺️ Treemap: Proportional Rectangles

```
# Treemap untuk visualisasi proporsi
metode_total = df.groupby('metode_pengadaan').agg({
    'pagu': 'sum',
    'kd_rup': 'count'
}).reset_index()
metode_total.columns = ['Metode', 'Total Pagu', 'Jumlah Paket']

fig = px.treemap(metode_total,
                 path=['Metode'],
                 values='Total Pagu',
                 color='Jumlah Paket',
                 title='Treemap: Metode Pengadaan by Pagu',
                 color_continuous_scale='Viridis',
                 hover_data=['Jumlah Paket'])

fig.update_layout(height=600)
fig.show()
```

Time Series dengan Range Slider

```
# Time series dengan range selector
daily_df = df.groupby(df['tgl_pengumuman_paket'].dt.date).size().reset_index()
daily_df.columns = ['Tanggal', 'Jumlah Paket']

fig = px.line(daily_df,
              x='Tanggal',
              y='Jumlah Paket',
              title='Trend Pengumuman Paket dengan Range Slider')

fig.update_xaxes(
    rangeslider_visible=True,
    rangeselector=dict(
        buttons=list([
            dict(count=7, label="1w", step="day", stepmode="backward"),
            dict(count=1, label="1m", step="month", stepmode="backward"),
            dict(count=3, label="3m", step="month", stepmode="backward"),
            dict(step="all", label="All")
        ])
    )
)

fig.update_layout(height=600)
fig.show()
```

🎨 Graph Objects: Lengkap Control

```
# Using Graph Objects untuk full kustomisasi
fig = go.Figure()

# Add multiple traces
for metode in df['metode_pengadaan'].unique():
    metode_df = df[df['metode_pengadaan'] == metode]
    monthly = metode_df.groupby(metode_df['tgl_pengumuman_paket'].dt.to_period('M')).size()

    fig.add_trace(go.Scatter(
        x=[str(idx) for idx in monthly.index],
        y=monthly.values,
        mode='lines+markers',
        name=metode,
        line=dict(width=2),
        marker=dict(size=6)
    ))
```

🎨 Graph Objects: Lengkap Control (lanjutan)

```
fig.update_layout(
    title='Trend per Metode Pengadaan',
    xaxis_title='Bulan',
    yaxis_title='Jumlah Paket',
    hovermode='x unified',
    height=600
)

fig.show()
```

Subplots dengan Plotly

```
from plotly.subplots import make_subplots

# Create 2x2 subplot
fig = make_subplots(
    rows=2, cols=2,
    subplot_titles=('Bar Chart', 'Pie Chart', 'Histogram', 'Box Plot'),
    specs=[[{'type': 'bar'}, {'type': 'pie'}],
           [{'type': 'histogram'}, {'type': 'box'}]]
)

# Add traces
metode_counts = df['metode_pengadaan'].value_counts()
fig.add_trace(go.Bar(x=metode_counts.index, y=metode_counts.values), row=1, col=1)
fig.add_trace(go.Pie(labels=metode_counts.index, values=metode_counts.values), row=1, col=2)
fig.add_trace(go.Histogram(x=df['pagu'], nbinsx=50), row=2, col=1)
fig.add_trace(go.Box(y=df['pagu'], name='Pagu'), row=2, col=2)

fig.update_layout(height=800, showlegend=False, title_text="Dashboard Multi-Plot")
fig.show()
```

💾 Export Plotly Charts

```
# Export as HTML (interactive)
fig.write_html("interactive_chart.html")

# Export as static image (requires kaleido)
# pip install kaleido
fig.write_image("chart.png", width=1200, height=800)
fig.write_image("chart.pdf")
fig.write_image("chart.svg")

# Save to multiple formats
for fmt in ['html', 'png', 'pdf']:
    filename = f"chart.{fmt}"
    if fmt == 'html':
        fig.write_html(filename)
    else:
        fig.write_image(filename)
    print(f"Saved: {filename}")
```

🎨 Part 3: Ceritakan Data dengan Baik

Penjelasan: Visualisasi bukan cuma buat grafik, tapi ceritakan insight dari data

Pilih Grafik yang Tepat

Tips: Jangan asal bikin grafik! Pilih yang sesuai tujuan: perbandingan, trend, komposisi, dll

Tujuan	Jenis Grafik	Kapan Digunakan
Perbandingan	Bar, Column	Membandingkan kategori
Distribusi	Histogram, Box, Violin	Memahami sebaran
Hubungan	Scatter, Bubble	Menunjukkan korelasi
Komposisi	Pie, Treemap, Sunburst	Bagian dari keseluruhan
Tren	Line, Area	Perubahan dari waktu ke waktu
Peringkat	Horizontal Bar	Menunjukkan urutan

🎨 Data Visualization Principles

Praktik Terbaik

1. **Pilih grafik yang tepat** 📊
    - Sesuaikan jenis grafik dengan jenis data
2. **Buat sesederhana mungkin** ✨
    - Hindari grafik yang terlalu ramai
    - Lebih sedikit lebih baik
3. **Gunakan warna dengan bijak** 🎨
    - Skema warna yang konsisten
    - Palet ramah buta warna

🎨 Data Visualization Principles (lanjutan)

4. **Beri label yang jelas** 🏷️
    - Judul yang deskriptif
    - Label sumbu dengan satuan
    - Sitasi sumber data
5. **Tonjolkan poin-poin penting** 💡
    - Annotations untuk poin penting
    - Hirarki visual
6. **Pertimbangkan audiens Anda** 👥
    - Technical vs non-technical
    - Context matters

🎨 Data Visualization Principles (lanjutan)

7. **Ceritakan sebuah kisah** 📖
    - Clear narrative flow
    - Insight yang actionable
8. **Jujur** ✅
    - Jangan menyesatkan dengan skala
    - Tunjukkan ketidakpastian jika relevan

❌ Common Visualization Mistakes

What to Avoid

1. **3D Pie Charts** 🚫
    - Sulit diinterpretasi
    - Persepsi terdistorsi
2. **Too Many Colors** 🌈
    - Membingungkan dan overwhelming
3. **Truncated Y-Axis** 📉
    - Perbandingan menyesatkan

❌ Common Visualization Mistakes (lanjutan)

4. **Pie Charts with Many Slices** 🥧
    - Use bar chart instead
5. **Dual Y-Axes** ⚖️
    - Can be misleading

🎯 Latihan Praktis

Latihan untuk Anda

1. **Distribution Analysis**
    - Histogram pagu dengan kustomisasi
    - Box plot per kategori
    - KDE plot comparison
2. **Trend Analysis**
    - Time series dengan moving average
    - Multi-line chart per metode
    - Seasonal patterns

🎯 Latihan Praktis (lanjutan)

3. **Comparison Charts**
    - Top N satker horizontal bar
    - Grouped bar chart metode vs jenis
    - Stacked bar chart
4. **Interactive Dashboards**
    - Create subplot dengan 4 charts
    - Sunburst untuk hierarchy
    - Treemap untuk proportions

🎯 Latihan Praktis (lanjutan)

5. **Advanced Customization**
    - Custom color schemes
    - Annotations dan arrows
    - Export untuk presentation
6. **Tell a Story**
    - Pilih 5 charts terbaik
    - Buat narrative flow
    - Export sebagai report

💡 Tips & Tricks

Matplotlib & Seaborn

```
# Save high-resolution
plt.savefig('chart.png', dpi=300, bbox_inches='tight')

# Transparent background
plt.savefig('chart.png', transparent=True)

# Close figure (free memory)
plt.close()

# Style context manager
with plt.style.context('seaborn-v0_8'):
    # Your plot here
    pass
```

💡 Tips & Tricks (lanjutan)

Plotly

```
# Update specific traces
fig.update_traces(marker_size=10, selector=dict(mode='markers'))

# Add shapes (rectangles, lines)
fig.add_shape(type="rect", x0=0, y0=0, x1=1, y1=1)

# Add annotations
fig.add_annotation(x=2, y=3, text="Important Point!")

# Theme templates
fig.update_layout(template="plotly_dark")  # Opsi: plotly, plotly_white, plotly_dark, ggplot2, seaborn

# Responsive sizing
fig.update_layout(autosize=True)
```

📚 Poin Penting

Yang Harus Anda Ingat

- ✅ Matplotlib = Foundation, full control
- ✅ Seaborn = Statistical tampilan, beautiful defaults
- ✅ Plotly = Interactivity, modern web-ready
- ✅ Choose chart type berdasarkan purpose
- ✅ Keep tampilans simple dan clear
- ✅ Color & styling matters
- ✅ Always label axes dan add titles
- ✅ Export dalam format yang sesuai (PNG/PDF/HTML)

Tampilan baik = Clear communication 📊

🔗 Resources

Dokumentasi & Galleries

- **Matplotlib:** https://matplotlib.org/stable/gallery/
- **Seaborn:** https://seaborn.pydata.org/examples/
- **Plotly:** https://plotly.com/python/
- **Python Graph Gallery:** https://python-graph-gallery.com/
- **From Data to Viz:** https://www.data-to-viz.com/

🎉 Penutup Hari 1

Recap Hari 1

- ✅ **Sesi 1:** Python & Pandas untuk Analisis Data
- ✅ **Sesi 2:** DuckDB untuk Query Analitik
- ✅ **Sesi 3:** Visualisasi Data

Besok (Hari 2)

📊 **Sesi 4:** Teknik Analisis Data Lanjutan

- Data cleaning & perubahan
- Time series analysis
- Statistical testing

🎉 Penutup Hari 1 (lanjutan)

🚀 **Sesi 5:** Dashboard Interaktif dengan Streamlit

- Building web apps
- Interactive filters
- Production deployment

🎉 Selesai Hari 1!

Excellent Work Today! 👏

Istirahat yang cukup!

See you tomorrow at 08:00 🚀

Pertanyaan?? 🙋

📞 Tugas Rumah (Opsional)

Latihan untuk Malam Ini

1. **Jelajahi Dataset Lain**
    - Cari dataset di Kaggle
    - Apply teknik hari ini
2. **Buat Sendiri Visualizations**
    - Eksperimen dengan styling
    - Try different chart types

📞 Tugas Rumah (Opsional) (lanjutan)

3. **Baca Dokumentasi**
    - Plotly express gallery
    - Seaborn examples

See you tomorrow! 💪

📞 Tetap Terhubung

Resources & Support

- 👤 **Nama:** [Kurnia Ramadhan,ST.,M.Eng]
- 📧 **Email:** [kurnia@ramadhan.me]

We're here to support your journey! 🤝