📈 Week 5 Student Guide – Visual Basketball Report

This guide prepares you for the **Week 5 Assignment**, where you will use the powerful Python visualization libraries **Matplotlib** and **Seaborn** to create a visual report of player performance data.

💻 Part 1: Visualization Library Review

In this course, we use **pandas** for data manipulation, and **Matplotlib** and **Seaborn** for creating charts.

- **Matplotlib (`plt`):** The foundational plotting library. It is used for setting up the figure size, adding titles, labels, and showing the final chart.
  - `plt.figure(figsize=(width, height))`: Sets the size of the canvas.
  - `plt.title("...")`, `plt.xlabel("...")`, `plt.ylabel("...")`: Adds descriptive text to the chart.
  - `plt.show()`: Displays the plot.
- **Seaborn (`sns`):** A high-level library built on top of Matplotlib. It provides functions that make it easy to create beautiful, complex statistical graphics with minimal code, such as bar plots and scatter plots.
  - `sns.set(style="...")`: Applies style themes (e.g., `"whitegrid"`, `"darkgrid"`) to improve the look of the plots.

Key Plot Types for Basketball Data

Plot Type	Seaborn Function	When to Use It	Example Plot
Bar Plot	`sns.barplot()`	**Comparing a single stat** (e.g., Points) across multiple categorical groups (e.g., Players).	
Scatter Plot	`sns.scatterplot()`	**Showing the relationship** between two numerical variables (e.g., Rebounds and Assists).	
Line Plot	`plt.plot()`	**Tracking a single variable over time** or a sequence (e.g., a player's points over a series of games).	

🛠️ Part 2: Week 5 Assignment Tasks

Your assignment is to create three different visual representations of the provided synthetic box score data.

1️⃣ Create Your Data (Code Cell 2)

This step is provided. It creates a DataFrame named `df` with basic box score stats (PTS, REB, AST) for five players.

2️⃣ Bar Chart – Points per Player (Code Cell 3)

Create a **Bar Chart** to visually compare the scoring of each player.

- **Data:** Use the `Player` column for the x-axis and the `PTS` column for the y-axis.
- **Key Code:**

  Python

```
plt.figure(figsize=(8,5))
# Use seaborn.barplot to compare a single stat across categories
sns.barplot(x="Player", y="PTS", data=df, palette="viridis") # Use any palette you like
plt.title("Player Scoring Comparison")
plt.ylabel("Points")
plt.xticks(rotation=15) # Optional: Rotates player names for readability
plt.show()
```

3️⃣ Scatter Plot – Rebounds vs. Assists (Code Cell 4)

Create a **Scatter Plot** to visualize the relationship between two statistics, allowing you to identify players who excel in both (or neither).

- **Data:** Use the `REB` column for the x-axis and the `AST` column for the y-axis.
- **Key Code:**

  Python

```
plt.figure(figsize=(6,5))
# Use seaborn.scatterplot to show the relationship between two numerical variables
sns.scatterplot(x="REB", y="AST", data=df, s=100, color="crimson")
plt.title("Rebounds vs Assists")
plt.xlabel("Rebounds")
plt.ylabel("Assists")
plt.show()
```

4️⃣ Line Chart – Example Player Trend (Code Cell 6)

Although the assignment notebook shows the example data being created in Code Cell 5, you will execute the plot in Code Cell 6. This task uses Matplotlib's native `.plot()` function to show how one player's scoring might change over a series of games.

- **Data:** You are provided with two lists: `games` (x-axis) and `points` (y-axis).
- **Key Code:**

  Python

```
# Data is provided in Code Cell 5: games = ["G1", "G2", "G3", "G4", "G5"], points = [15, 20, 25, 18, 27]
plt.figure(figsize=(7,4))
# Use plt.plot to display sequential data
plt.plot(games, points, marker="o", color="blue")
plt.title("Player A – Points Over 5 Games")
plt.xlabel("Game")
plt.ylabel("Points")
plt.show()
```