---

# 🎬 Capstone Project – *Movie Ratings Analytics with PySpark*

### Business Scenario

You are working for a **movie streaming platform**. You have data about users, movies, and ratings. Your task is to analyze viewing patterns, customer preferences, and movie popularity using **PySpark**.

---

## Step 1: Setup in Google Colab

```
!pip install pyspark
from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("Movie-Capstone").getOrCreate()
sc = spark.sparkContext
```

---

## Step 2: Prepare Data

### Users Data

```
users_data = [
    (1, "Rahul", 25, "Bangalore"),
    (2, "Priya", 30, "Delhi"),
    (3, "Aman", 22, "Hyderabad"),
    (4, "Sneha", 28, "Chennai"),
    (5, "Arjun", 35, "Mumbai")
]
users_cols = ["user_id", "name", "age", "city"]
users_df = spark.createDataFrame(users_data, users_cols)
```

### Movies Data

```
movies_data = [
    (101, "Inception", "Sci-Fi", 2010),
    (102, "Avengers", "Action", 2012),
    (103, "3 Idiots", "Comedy", 2009),
    (104, "Dangal", "Drama", 2016),
    (105, "Interstellar", "Sci-Fi", 2014)
]
movies_cols = ["movie_id", "title", "genre", "year"]
movies_df = spark.createDataFrame(movies_data, movies_cols)
```

### Ratings Data

```
ratings_data = [
    (1, 101, 5),
```

```
    (2, 101, 4),
    (1, 102, 3),
    (3, 103, 4),
    (4, 104, 5),
    (2, 103, 5),
    (5, 105, 4),
    (6, 101, 5)  # Rating from non-existent user
]
ratings_cols = ["user_id", "movie_id", "rating"]
ratings_df = spark.createDataFrame(ratings_data, ratings_cols)
```

---

## Step 3: Capstone Tasks

### 🔹 Part A – DataFrame Basics

1. Show all users who are older than 28.
2. List all distinct movie genres.
3. Find all movies released after 2010.

---

### 🔹 Part B – Aggregations

4. Find the **average age of users per city**.
5. Find the **average rating for each movie**.
6. Find the **highest-rated movie in each genre**.

---

### 🔹 Part C – Joins

7. Join `ratings` with `users` to see who rated what.
8. Join `ratings` with `movies` to see ratings with movie names.
9. Find all users who have **not rated any movie**.
10. Find all movies that have **never been rated**.

---

### 🔹 Part D – SQL Queries

11. Register `users`, `movies`, and `ratings` as temp views.
12. Write a SQL query to find the **top 2 cities by number of ratings given**.
13. Write a SQL query to find users who gave **at least one 5-star rating**.
14. Write a SQL query to find the **most popular genre** by number of ratings.

---

### 🔹 Part E – File I/O

15. Save `ratings_df` as CSV and load it back.
16. Save `movies_df` as JSON and reload it.

---

### 🔹 Part F – Visualization

17. Convert PySpark DataFrame → Pandas ( `toPandas()` ).
18. Plot **average rating per genre** (bar chart).
19. Plot **number of ratings per year of movie release** (line chart).
20. Plot **age vs average rating given** (scatter plot).

---