**Problem Definition: Movie Recommendation System**

The goal of the **Movie Recommendation System** is to build a software application that suggests movies to users based on their past preferences, interactions, or similarities with other users and items. The system can use various algorithms to predict what movies the user would like to watch, improving their overall experience and engagement on the platform.

There are two main types of recommendation systems:

**Content-Based Filtering**: Recommendations based on the characteristics or metadata of the movies (i.e., similar genres, actors, or directors).

**Milestones for Building a Movie Recommendation System**

1. **Project Planning and Dataset Selection**

**Objective**: Define the scope, goals, and timeline for the project. Select an appropriate dataset.

* **Tasks**:
  + Define the **problem statement**: What kind of recommendations will your system provide? (content-based)
  + Select the dataset: Use popular datasets like the **tmdb** datasets.
  + Set up the development environment (Python, Jupyter notebooks, necessary libraries).

**2. Data Preprocessin**

* **Objective**: Clean and preprocess the dataset to ensure it is ready for analysis and modeling.
* **Tasks**:
  + Handle missing values (e.g., missing ratings, movie metadata).
  + Remove or handle duplicates.
  + Convert categorical data (e.g., genres) into numeric values.
  + Normalize data if necessary (e.g., scaling movie ratings).
  + Split the dataset into training and testing sets.
* **Deliverable**: Cleaned dataset ready for analysis and modeling.
  + Explore **movie metadata**: What are the most common genres? What are the most popular movies?
  + Visualize patterns using **Pandas**, **Matplotlib**, **numpy** , etc.
  + Analyze correlations between movie features (e.g., genre popularity, average ratings).
* **Deliverable**: Report on data insights, trends, and visualizations.

**4. Building a Baseline Recommendation System**

* **Objective**: Implement a simple recommendation system as a baseline for comparison with more advanced models.
* **Tasks**:
  + **Content-Based Filtering**: Build a recommendation system that compares movie features (e.g., genre, actors) to suggest similar movies.
  + **Evaluate performance** using metrics like precision, recall, F1-score, or RMSE (Root Mean Squared Error) for predicted ratings.
* **Deliverable**: A baseline recommendation system with basic functionality.

**5. Evaluation and Optimization**

* **Objective**: Evaluate the system using proper metrics and optimize for better performance.
* **Tasks**:
  + Use evaluation techniques such as **cross-validation** to ensure the model’s generalizability.
  + Compute evaluation metrics (e.g., **RMSE**, **MAE**, precision, recall).
  + Fine-tune hyperparameters to improve model performance (e.g., regularization in matrix factorization, learning rate in neural networks).
* **Deliverable**: Optimized model with a performance report.

**6. Final Presentation and Documentation (Week 16)**

* **Objective**: Prepare and deliver the final project report, code documentation, and project presentation.
* **Tasks**:
  + Write a comprehensive **project report**: Describe the problem, data analysis, modeling approach, evaluation metrics, and final recommendations.
  + **Document code**: Ensure that your code is well-documented with comments and instructions for future use or improvements.
  + Prepare a presentation that includes visuals, key insights, and a demo of the recommendation system.
* **Deliverable**: Final project report, documented code, and presentation.

**Steps for Building a Movie Recommendation Software**

**1. Collect and Understand Data**

* Gather movie datasets (e.g., MovieLens).
* Understand the dataset structure: User IDs, movie IDs, ratings, genres, metadata (e.g., actors, directors).
* Explore data distributions (e.g., how many movies each user rated, average ratings per movie).

**2. Data Preprocessing**

* Clean the data: Handle missing values, duplicates, and irrelevant information.
* Convert movie features (e.g., genres) into usable formats (e.g., binary vectors or embeddings).
* Create train-test splits for model validation.

**3. Build a Simple Recommendation Model**

* **Content Filtering**:
  + Item-based content: Find similar movies based on user ratings.
* Use cosine similarity or Pearson correlation to measure similarity.

**7. Continuous Improvement**

* Tune hyperparameters, add more features (e.g., social network interactions, reviews).
* Incorporate real-time recommendations as user preferences change.