Java Code: AI-Based Recommendation System

```
import java.util.*;

public class RecommendationSystem {

        private static final Map<String, Map<String, Integer>> userRatings = new
HashMap<>();

    public static void main(String[] args) {
        addRating("Alice", "Matrix", 5);
        addRating("Alice", "Titanic", 3);
        addRating("Bob", "Matrix", 4);
        addRating("Bob", "Titanic", 2);
        addRating("Bob", "Avatar", 5);
        addRating("Charlie", "Matrix", 2);
        addRating("Charlie", "Avatar", 5);

        String targetUser = "Alice";
        List<String> recommendations = getRecommendations(targetUser);

                 System.out.println("Recommendations for " + targetUser + ": " +
recommendations);
    }

    private static void addRating(String user, String item, int rating) {
        userRatings.computeIfAbsent(user, k -> new HashMap<>()).put(item, rating);
    }

    private static List<String> getRecommendations(String user) {
        Map<String, Integer> targetRatings = userRatings.get(user);
        Map<String, Double> scores = new HashMap<>();

        for (String otherUser : userRatings.keySet()) {
            if (otherUser.equals(user)) continue;

                                double similarity = cosineSimilarity(targetRatings,
userRatings.get(otherUser));

                                        for (Map.Entry<String, Integer> entry :
userRatings.get(otherUser).entrySet()) {
                String item = entry.getKey();
                int rating = entry.getValue();

                if (!targetRatings.containsKey(item)) {
                         scores.put(item, scores.getOrDefault(item, 0.0) + similarity *
rating);
                }
            }
        }

        List<Map.Entry<String, Double>> sorted = new ArrayList<>(scores.entrySet());
        sorted.sort(Map.Entry.<String, Double>comparingByValue().reversed());
```

```java
        List<String> recommendedItems = new ArrayList<>();
        for (Map.Entry<String, Double> entry : sorted) {
            recommendedItems.add(entry.getKey());
        }

        return recommendedItems;
    }

     private static double cosineSimilarity(Map<String, Integer> ratings1, Map<String, 
Integer> ratings2) {
        Set<String> commonItems = new HashSet<>(ratings1.keySet());
        commonItems.retainAll(ratings2.keySet());

        if (commonItems.isEmpty()) return 0.0;

        double dotProduct = 0.0;
        double normA = 0.0;
        double normB = 0.0;

        for (String item : commonItems) {
            int rating1 = ratings1.get(item);
            int rating2 = ratings2.get(item);
            dotProduct += rating1 * rating2;
        }

        for (int rating : ratings1.values()) {
            normA += rating * rating;
        }
        for (int rating : ratings2.values()) {
            normB += rating * rating;
        }

        return dotProduct / (Math.sqrt(normA) * Math.sqrt(normB));
    }
}
```