```
import org.apache.mahout.cf.taste.eval.DataModelBuilder;
import org.apache.mahout.cf.taste.impl.model.file.FileDataModel;
import org.apache.mahout.cf.taste.model.DataModel;
import org.apache.mahout.cf.taste.similarity.UserSimilarity;
import org.apache.mahout.cf.taste.impl.similarity.PearsonCorrelationSimilarity;
import org.apache.mahout.cf.taste.neighborhood.UserNeighborhood;
import org.apache.mahout.cf.taste.impl.neighborhood.NearestNUserNeighborhood;
import org.apache.mahout.cf.taste.recommender.RecommendedItem;
import org.apache.mahout.cf.taste.recommender.Recommender;
import org.apache.mahout.cf.taste.impl.recommender.GenericUserBasedRecommender;

import java.io.File;
import java.util.List;

public class RecommendationSystem {
    public static void main(String[] args) {
        try {
            // Load the data from a CSV file
            File dataFile = new File("data.csv"); // Make sure this file exists
            DataModel model = new FileDataModel(dataFile);

            // Create similarity measure (Pearson)
            UserSimilarity similarity = new PearsonCorrelationSimilarity(model);

            // Define user neighborhood
            UserNeighborhood neighborhood = new NearestNUserNeighborhood(2, similarity, model);

            // Create recommender
            Recommender recommender = new GenericUserBasedRecommender(model, neighborhood,
similarity);

            // Recommend items for a specific user (example: userID = 3)
            List<RecommendedItem> recommendations = recommender.recommend(3, 2);

            System.out.println("Recommendations for user 3:");
            for (RecommendedItem recommendation : recommendations) {
                System.out.println("Item ID: " + recommendation.getItemID() + " | Score: " +
recommendation.getValue());
            }

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```