PROJECT ABSTRACT

# Movie Recommendation System using Collaborative Filtering

Every streaming platform lives or dies by one question: what should you watch next? Netflix attributes most watched hours to its recommender.

## 01 Title

Movie Recommendation System using Collaborative Filtering

## 02 Introduction / Background

Every streaming platform lives or dies by one question: what should you watch next? Netflix attributes most watched hours to its recommender. Behind both sits collaborative filtering — the insight that people who agreed in the past will agree in the future. This project builds that algorithmic core from scratch and evaluates it like a real system.

## 03 Problem Statement

Choice overwhelms users: thousands of titles, no way to find the right one. Platforms need a system that learns taste from ratings, predicts unseen-movie ratings, and explains its recommendations. This project delivers that: user-based and item-based collaborative filtering plus SVD matrix factorization, compared with RMSE and top-N metrics, served through the CineMatch web app.

## 04 Objectives

- Implement user-based CF with cosine and Pearson similarity.
- Implement item-based CF for "because you watched" recommendations.
- Implement SVD matrix factorization (50–100 latent factors).
- Evaluate all three: RMSE on held-out split, Precision@K / Recall@K (K=10).
- Document cold-start handling with a popularity fallback for new users.
- Deliver CineMatch: Flask app with user profiles, predicted ratings, genre rows.

## 05 Proposed System / Working

The MovieLens 100K matrix (100,000 ratings, 943 users, 1,682 movies) is loaded and split into train/test sets. User-based CF predicts from nearest neighbours' tastes; item-based CF computes movie–movie similarity. SVD (surprise library) factorizes the sparse matrix into latent factors, reconstructing missing ratings with the lowest error. Built in Python 3.10 with scikit-learn, served

via CineMatch (Flask); a 1M-rating upgrade path is documented.

SVD achieves the lowest error — RMSE ~0.87–0.93 on the held-out test — beating both CF variants, with per-method numbers documented. Precision@K and Recall@K (K=10) measure top-10 ranking quality. In the live CineMatch demo, an examiner picks a user profile and instantly sees personalized picks, predicted ratings, "because you watched" rows and explanations.

## 06 Technologies / Components Used

**Python 3.10** · **pandas** · **NumPy** · **scikit-learn** · **surprise 1M-rating upgrade path included** · **Flask** · **Jinja templates** · **Bootstrap-style Netflix UI** · **MovieLens dataset included**

## 07 Key Features

- User-based CF (cosine/Pearson similarity).
- Item-based CF ("because you watched" rows).
- SVD matrix factorization (surprise library).
- RMSE + Precision@K / Recall@K across all methods.
- Netflix-style Flask UI: posters, genres, predicted ratings.
- Per-recommendation explanations ("because you rated X highly").
- Cold-start popularity fallback for new users.
- Search, genre browsing, user profiles.

## 08 Applications / Use Cases

- Streaming and video platforms
- e-commerce product recommendations
- music
- news personalization

## 10 Conclusion

This project delivers the algorithmic core of Netflix and Amazon as a working, benchmarked system: three collaborative-filtering methods implemented from scratch, honestly compared, and served through an app examiners can explore. RMSE plus ranking metrics, documented cold-start handling and a MovieLens 1M upgrade path make it a defensible recommender-systems build.