Project proposal

Title of Project: Movie Recommendation System

Proposed By: Adrian Todd for CPSC 3660 Winter 2025

Project abstract:

I intend to build this project as a combination of final projects for CPSC 3660 and CPSC 3620. This project will implement a movie recommendation system that leverages data from a web-scraper that will scrape data from IMDb and possibly Rotten Tomatoes. A relational database built using MySQL will store movie industry information, (user data, and preferences if time allows). A web crawler will gather data from IMDb, and a recommendation algorithm will suggest movies that match a user's preferences. The system will provide a valuable service for users seeking personalized movie recommendations.

Project scenario and goals: A scenario that highlights how the project will actually be used by an end-user(even if you did not implement it). You might include a sketch of the UI (if there is one). Describe any special constraints (e.g., speed, size, storage, scale, robustness) your design needs to satisfy.

Scenario: A user wants to find new movies to watch. They can either create a new account and specify their favorite genres, actors, directors, and plot keywords, or they can simply start searching for movies and filtering them using predefined filters. The system will then query the database, calculate a match score, and present the user with a ranked list of movie recommendations.

Goals:

- Create a plan and break it down into tasks and add tasks to Kanban board like Trello.
- Design a relational database schema to store movie industry data, user data, and user preferences.
- Create a comprehensive EER diagram. Using Lucid Chart
- Create a relational model. Using Lucid Chart
- Implement database design based on EERD and relational model. Using MySQL Workbench
- Create database queries and test data. Using MySQL
- Build a functional web crawler to extract movie industry data from IMDb and Rotten Tomatoes. Build using Python and appropriate libraries
- Create a simple recommendation algorithm. Implement in Python

- Design and implement a web-based GUI for user interaction. Using Python and Flask or Django
- Connect MySQL database to web interface.
- Thoroughly test application using MySQL Workbench, Postman for API testing, Pytest or similar for basic Unit Testing.

Constraints:

- The IMDb and Rotten Tomatoes movie industry data sets are probably quite large so I will need to make sure queries are optimized to ensure reasonable performance and I that the amount of data I scrape is reasonable. I might want to look into creating indexes for the data if my queries are too slow.
- When scraping data I will need to rate limit my systems interactions with IMDb's web server to ensure that I am not overloading it with requests for data and adhere to their robots.txt file to make sure the api endpoints that I am requesting data from are allowed for this use case.

Design strategy:

The major components for this project will be the EERD, relational data model, DDL SQL script for creating the data base and initial testing, DML SQL queries for retrieving and altering data, integration with web scraper and GUI.

- EERD Will define the main entities and relationships.
- Relational data model will outline the data tables structure.
- The DDL SQL script will create all tables and implement relational, data, and type constraints.
- The DML SQL queries will define how data will be handled, what changes to the database are allowed, and how changes affect the database.
- The integration with the web crawler and GUI will provide an interface for interacting with the database.

Design unknowns/risks:

I don't have experience with web crawlers so it will be challenging and interesting to figure out how to scrape and clean data before inserting it into the database. I am also not overly familiar with indexing, but I will attempt to implement a couple to make my queries more efficient. I don't know if I will have enough time to add user login functionality, I have done this in the past and will incorporate it if time allows.

Implementation plan and schedule:

Week 1 (up to March 7):

- Brainstorm Ideas
- Create EERD
- Create Relational Model
- Write project proposal
- Submit project proposal

Week 2 (up to March 14):

- Assess feedback and make necessary changes.
- Write DDL script to create and populate tables with test data.
- Write DML queries.
- Test SQL queries.

Week 3 (up to March 21):

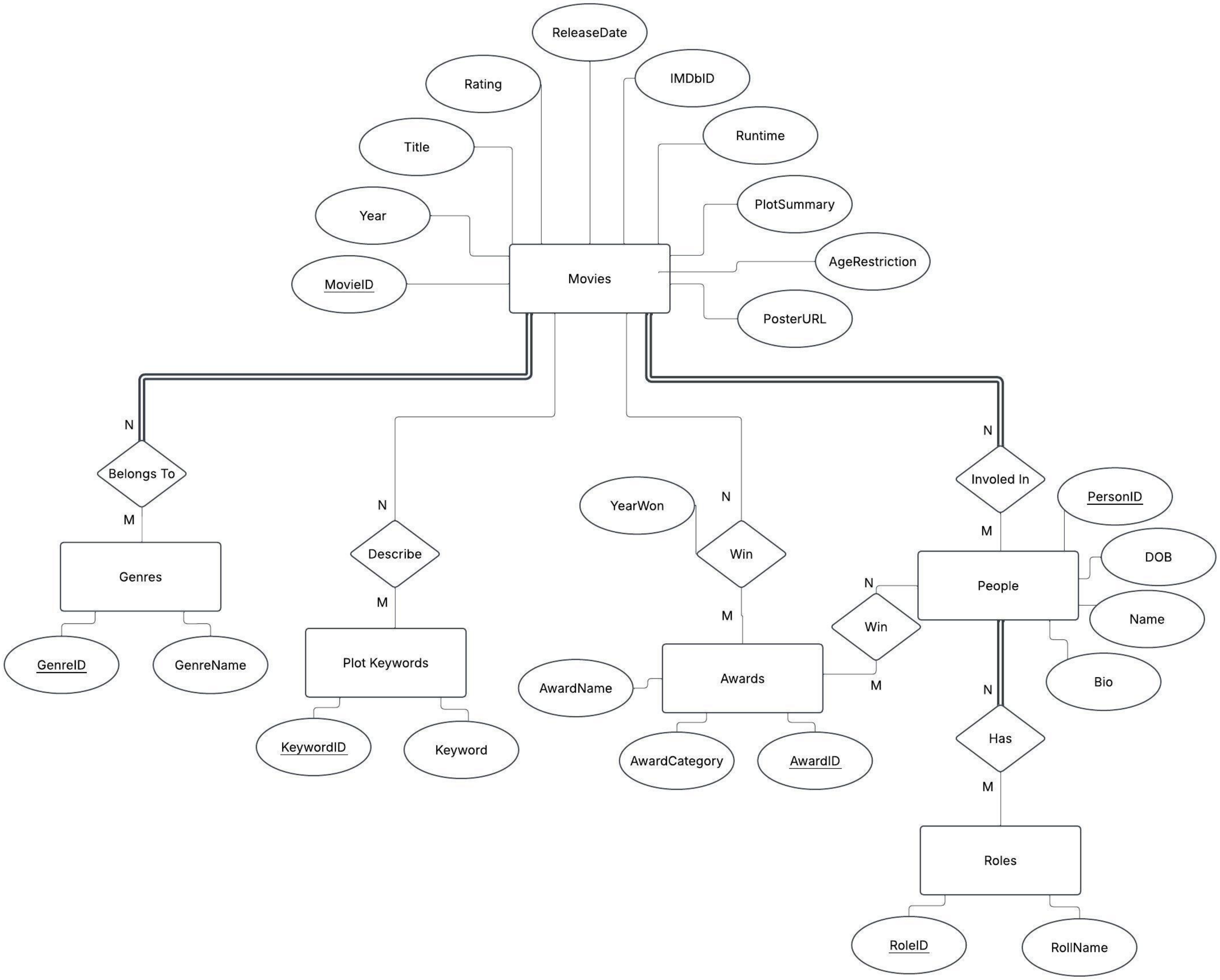
- Integrate web crawler and recommendation system.
- Integrate with frontend.
- Integration testing.
- Add in login for users if time allows
- Prep for class presentation.

Week 4 (up to March 28):

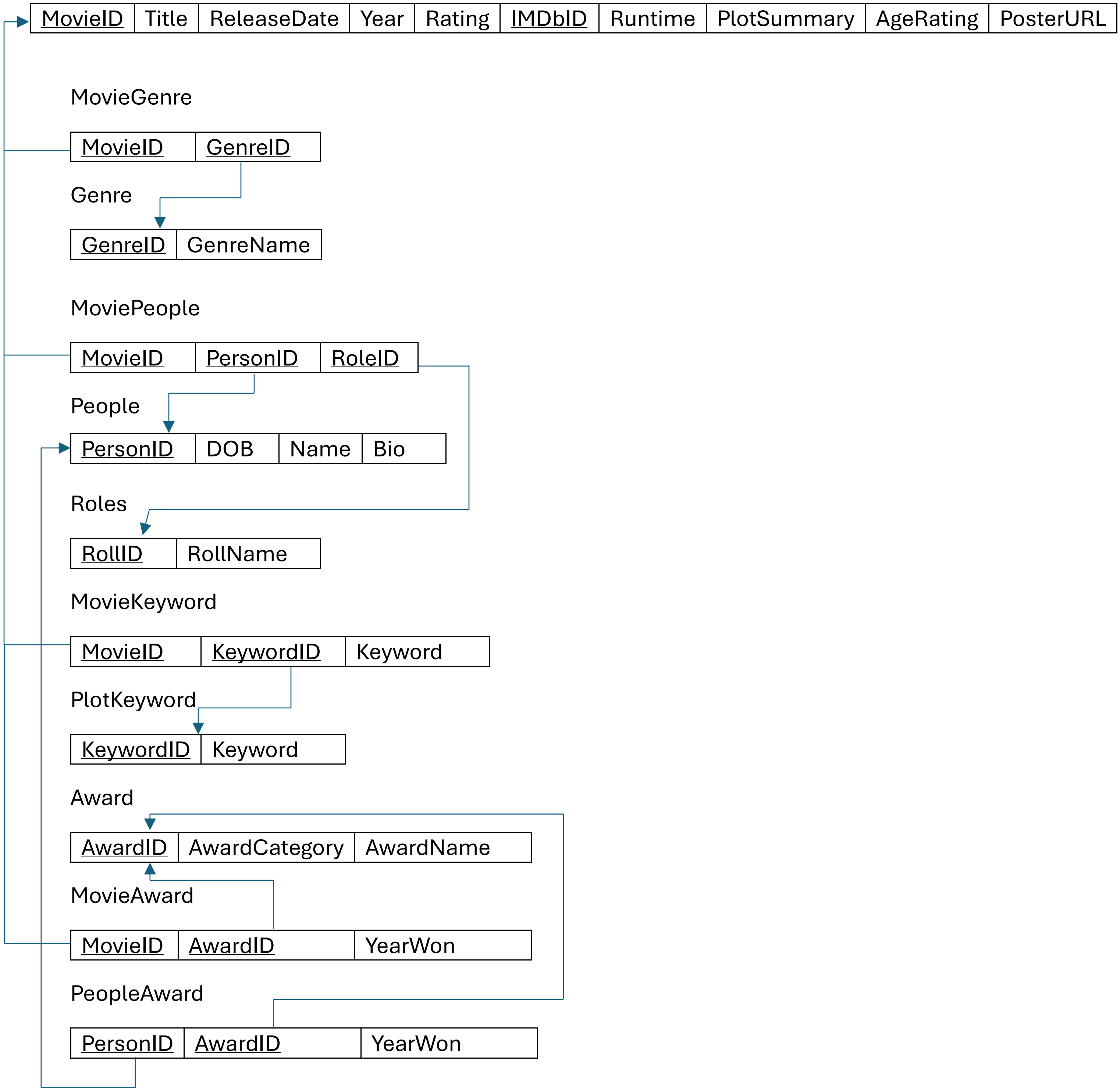
- Assess any feedback and decide what to update.
- Record final video demonstration.

Progress Metrics:

- Submit complete project proposal
- Successful project tracking
- Successful database creation and test population
- Successful data extraction from IMDb
- Accuracy of the recommendation algorithm
- Successful execution of all required SQL queries
- Successful creation of basic frontend
- Successful integration of web crawler and recommendation algorithm
- Successful presentation of final project
- Successful completion of final report


ReleaseDate
Rating
IMDbID
Title
Runtime
Year
PlotSummary
MovieID
Movies
AgeRestriction
PosterURL
N
N
Belongs To
M
Genres
GenreID
GenreName
N
Describe
M
Plot Keywords
KeywordID
Keyword
YearWon
N
Win
M
Awards
AwardName
AwardCategory
AwardID
Involed In
M
PersonID
DOB
N
People
Win
Name
M
Bio
N
Has
M
Roles
RoleID
RollName

Relational Model



Movie

MovieID	Title	ReleaseDate	Year	Rating	IMDbID	Runtime	PlotSummary	AgeRating	PosterURL

MovieGenre

MovieID	GenreID

Genre

GenreID	GenreName

MoviePeople

MovieID	PersonID	RoleID

People

PersonID	DOB	Name	Bio

Roles

RollID	RollName

MovieKeyword

MovieID	KeywordID	Keyword

PlotKeyword

KeywordID	Keyword

Award

AwardID	AwardCategory	AwardName

MovieAward

MovieID	AwardID	YearWon

PeopleAward

PersonID	AwardID	YearWon