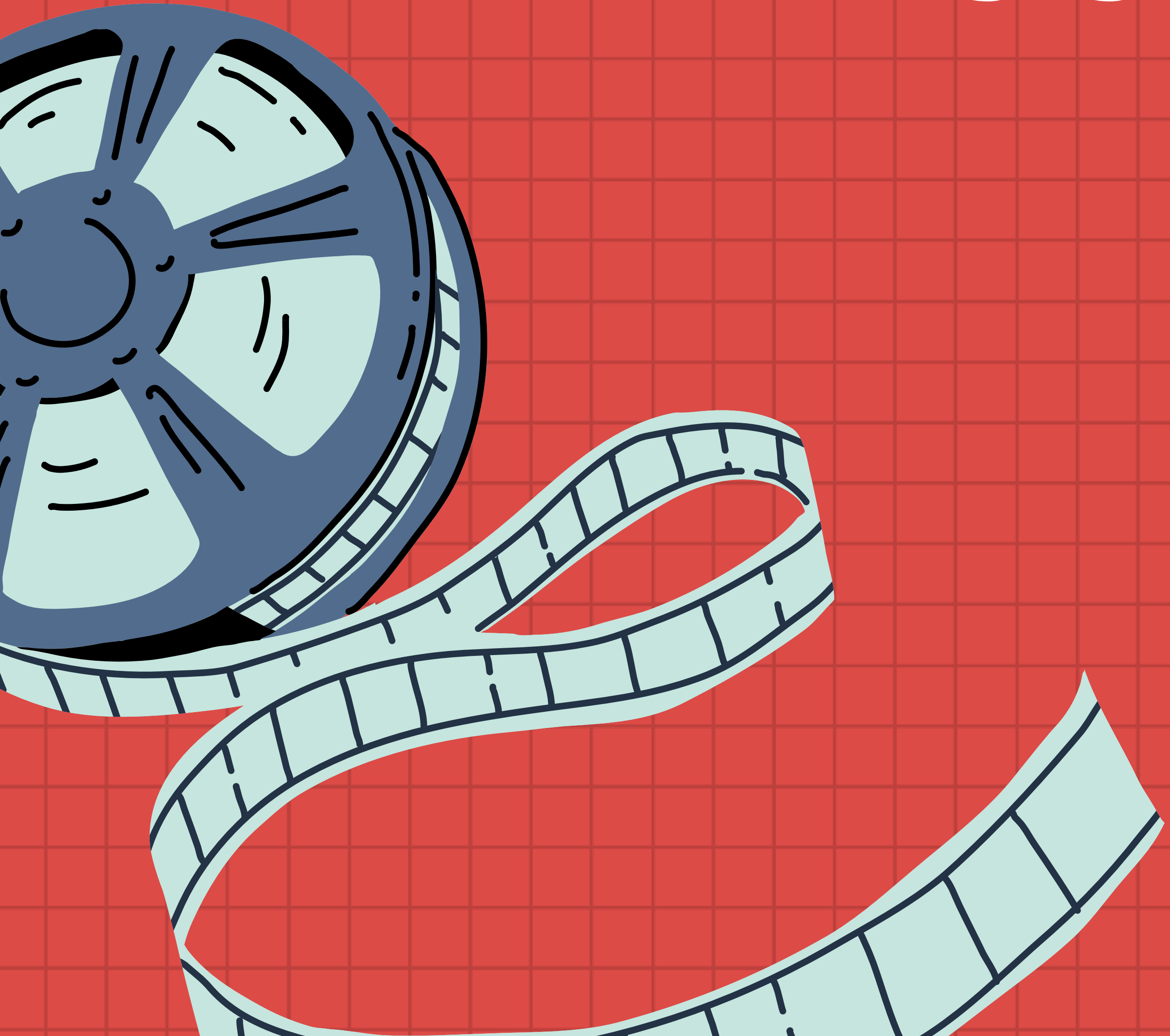
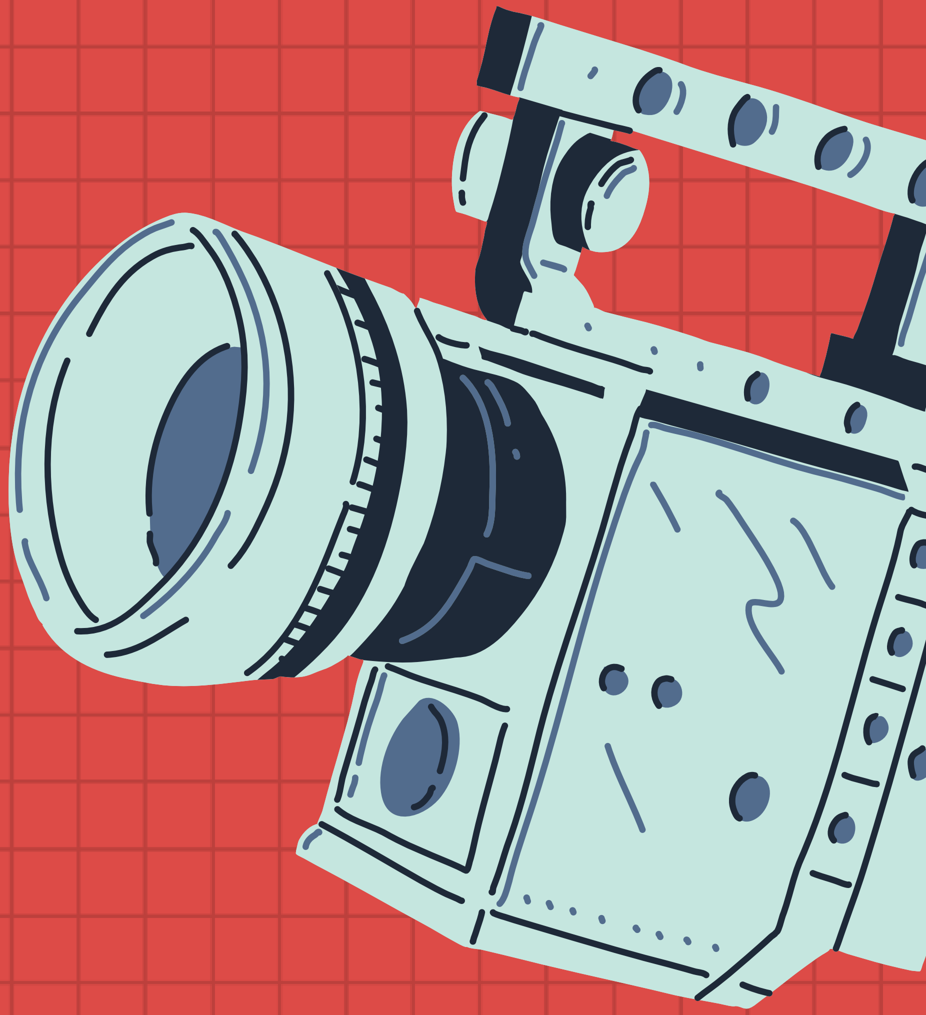
Movie Recommindation System

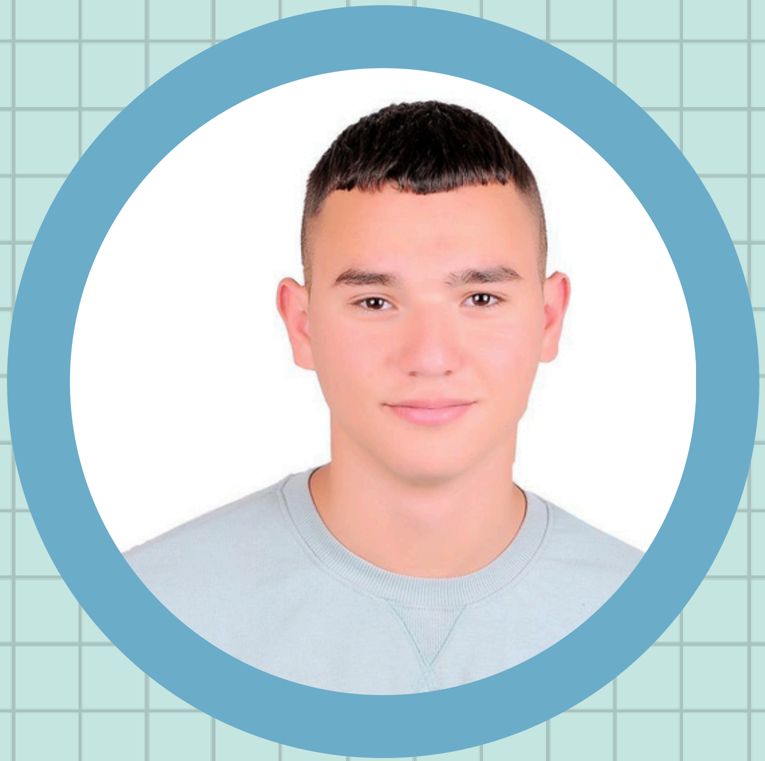
Group 13

Members

Knan Fadi

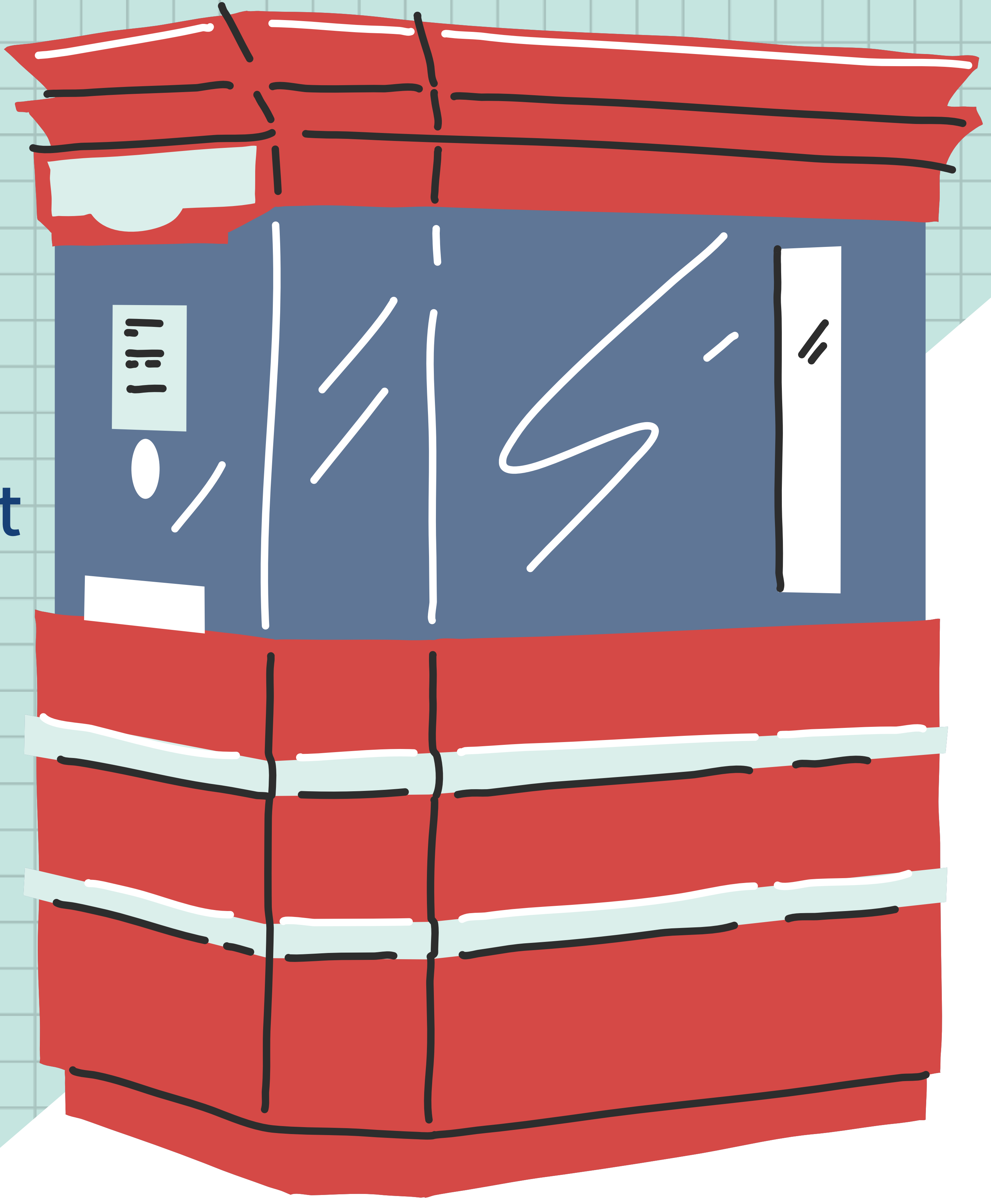
Mohamed Moqbel



Fouad Mahmoud

Ali Reda Ali

Contents

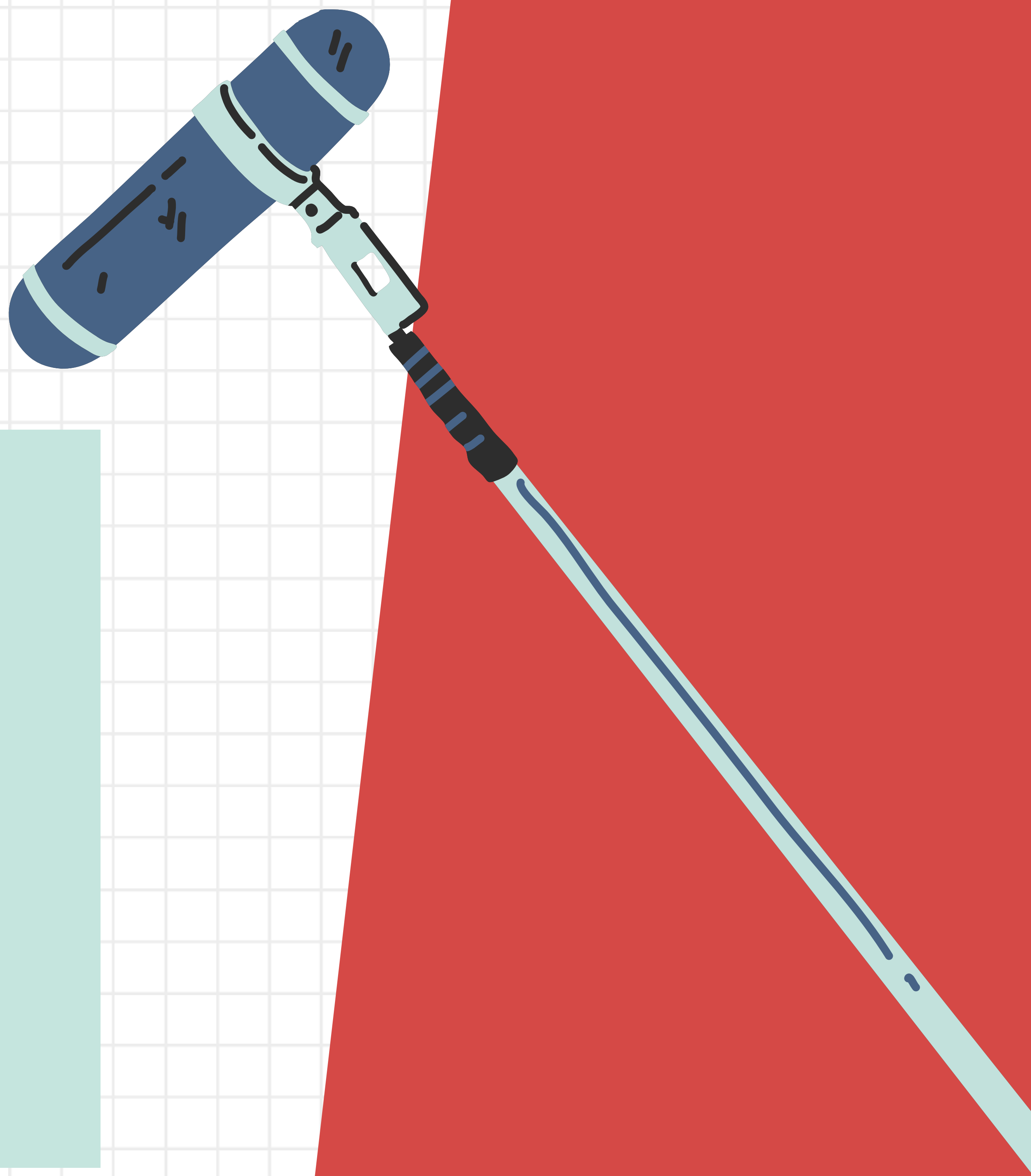


Overview about the project

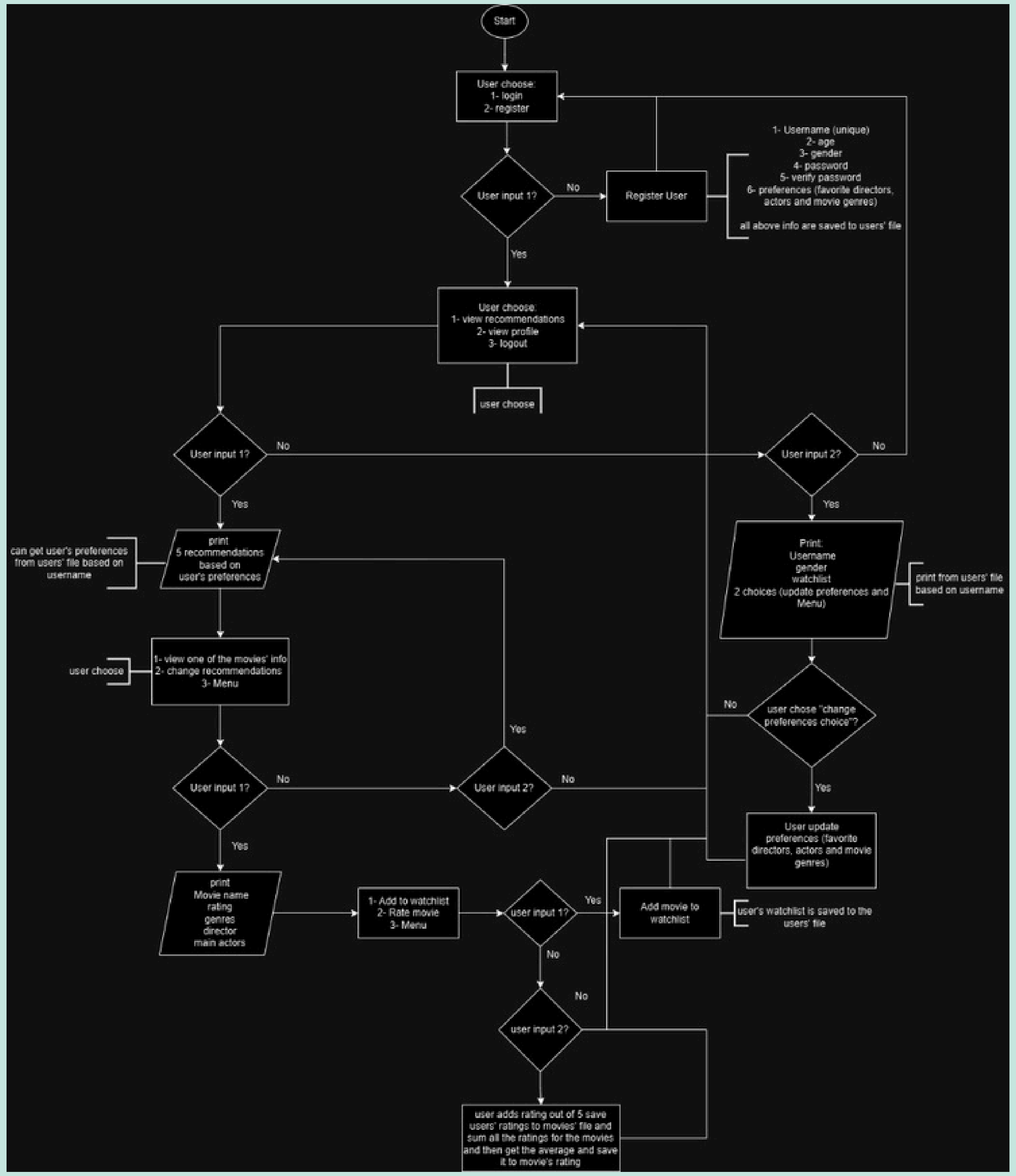
The Movie Recommendation System is designed to assist users in discovering new movies that match their tastes. It aims to make the process of finding movies more enjoyable through the use of algorithm. The system is intended to be user-friendly, ensuring that anyone can navigate and benefit from the recommendations. It will cater to a wide range of preferences, from blockbuster enthusiasts to indie film aficionados, ensuring that everyone finds something they love

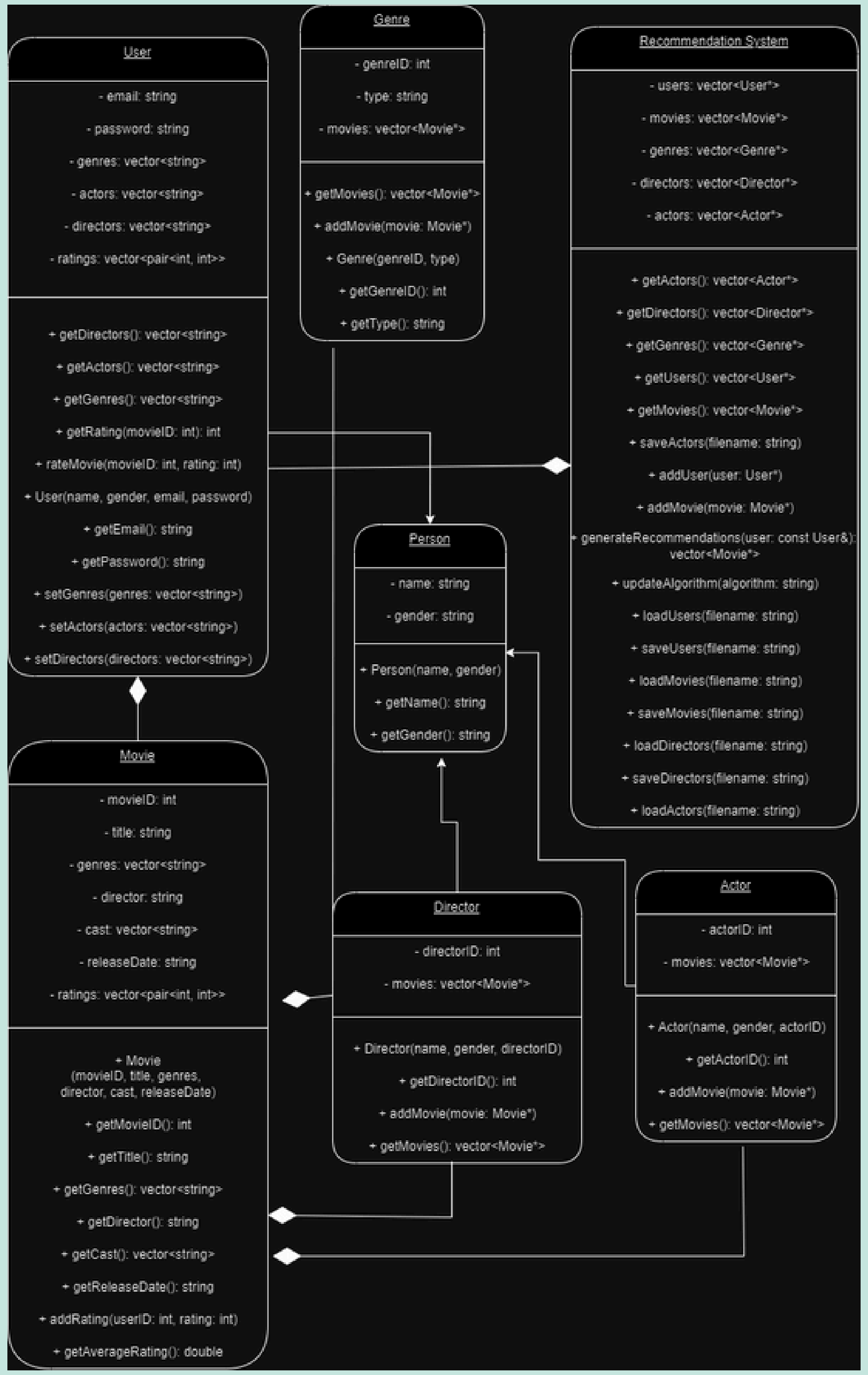
objectives and purpose of the project

1. Help users discover new movies aligned with their tastes
2. Make finding movies easier and more enjoyable
3. Use clever computer techniques to improve the recommendations over time
4. Create a simple and friendly system that anyone can use



Flowchart & UML


Start
User choose:
1- login
2- register
User input 1?
No
Register User
1- Username (unique)
2- age
3- gender
4- password
5- verify password
6- preferences (favorite directors, actors and movie genres)
all above info are saved to users' file
Yes
User choose:
1- view recommendations
2- view profile
3- logout
user choose
User input 1?
No
User input 2?
No
Yes
print
5 recommendations
based on
user's preferences
can get user's preferences from users' file based on username
Print:
Username
gender
watchlist
2 choices (update preferences and Menu)
print from users' file based on username
1- view one of the movies' info
2- change recommendations
3- Menu
user choose
No
user chose "change preferences choice"?
Yes
User input 1?
No
User input 2?
No
Yes
User update preferences (favorite directors, actors and movie genres)
Yes
print
Movie name
rating
genres
director
main actors
1- Add to watchlist
2- Rate movie
3- Menu
user input 1?
Yes
Add movie to watchlist
user's watchlist is saved to the users' file
No
No
user input 2?
user adds rating out of 5 save users' ratings to movies' file and sum all the ratings for the movies and then get the average and save it to movie's rating


User
- email: string
- password: string
- genres: vector<string>
- actors: vector<string>
- directors: vector<string>
- ratings: vector<pair<int, int>>
+ getDirectors(): vector<string>
+ getActors(): vector<string>
+ getGenres(): vector<string>
+ getRating(movieID: int): int
+ rateMovie(movieID: int, rating: int)
+ User(name, gender, email, password)
+ getEmail(): string
+ getPassword(): string
+ setGenres(genres: vector<string>)
+ setActors(actors: vector<string>)
+ setDirectors(directors: vector<string>)
Genre
- genreID: int
- type: string
- movies: vector<Movie*>
+ getMovies(): vector<Movie*>
+ addMovie(movie: Movie*)
+ Genre(genreID, type)
+ getGenreID(): int
+ getType(): string
Recommendation System
- users: vector<User*>
- movies: vector<Movie*>
- genres: vector<Genre*>
- directors: vector<Director*>
- actors: vector<Actor*>
+ getActors(): vector<Actor*>
+ getDirectors(): vector<Director*>
+ getGenres(): vector<Genre*>
+ getUsers(): vector<User*>
+ getMovies(): vector<Movie*>
+ saveActors(filename: string)
+ addUser(user: User*)
+ addMovie(movie: Movie*)
+ generateRecommendations(user: const User&): vector<Movie*>
+ updateAlgorithm(algorithm: string)
+ loadUsers(filename: string)
+ saveUsers(filename: string)
+ loadMovies(filename: string)
+ saveMovies(filename: string)
+ loadDirectors(filename: string)
+ saveDirectors(filename: string)
+ loadActors(filename: string)
Person
- name: string
- gender: string
+ Person(name, gender)
+ getName(): string
+ getGender(): string
Movie
- movieID: int
- title: string
- genres: vector<string>
- director: string
- cast: vector<string>
- releaseDate: string
- ratings: vector<pair<int, int>>
+ Movie (movieID, title, genres, director, cast, releaseDate)
+ getMovieID(): int
+ getTitle(): string
+ getGenres(): vector<string>
+ getDirector(): string
+ getCast(): vector<string>
+ getReleaseDate(): string
+ addRating(userID: int, rating: int)
+ getAverageRating(): double
Director
- directorID: int
- movies: vector<Movie*>
+ Director(name, gender, directorID)
+ getDirectorID(): int
+ addMovie(movie: Movie*)
+ getMovies(): vector<Movie*>
Actor
- actorID: int
- movies: vector<Movie*>
+ Actor(name, gender, actorID)
+ getActorID(): int
+ addMovie(movie: Movie*)
+ getMovies(): vector<Movie*>

OOP concepts used on the code

1. Encapsulation
Encapsulation bundles data and methods within classes, restricting access to some components. For example, attributes like email, password, and ratings in the User class are private and accessed through public methods.

2. Inheritance
Inheritance allows new classes to derive properties and methods from existing classes. The User, Director, and Actor classes inherit from the Person class, gaining attributes name and gender.

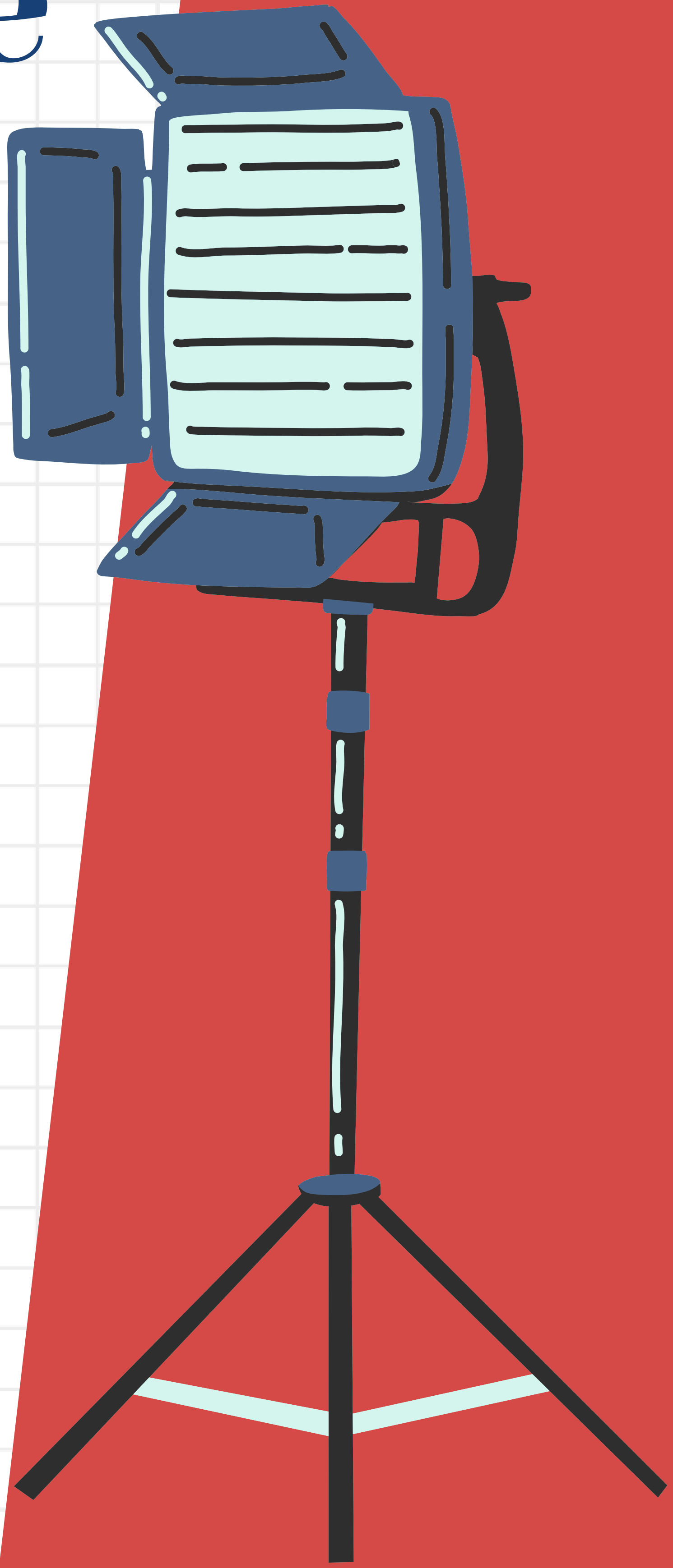
3. Composition
Composition builds complex types from simpler ones, indicating a "has-a" relationship. For instance, the User class contains vectors of genres, actors, and directors, and the RecommendationSystem class manages collections of User*, Movie*, Genre*, Director*, and Actor*.

4. Utility Functions
Functions like isValidEmail and isValidPassword are used for input validation.

5. User Interaction
The main function facilitates user interactions, allowing users to register, log in, view profiles, get recommendations, and rate movies.

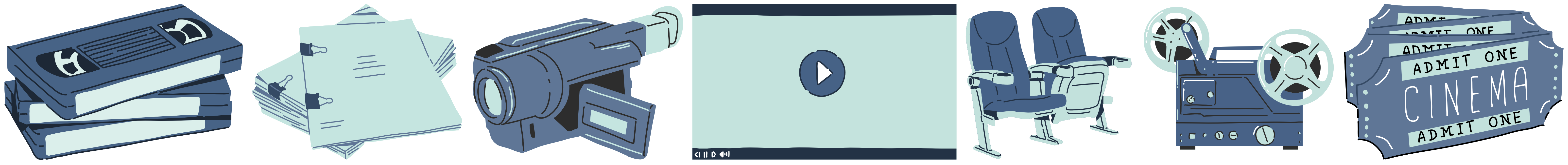
OOP concepts shown in code

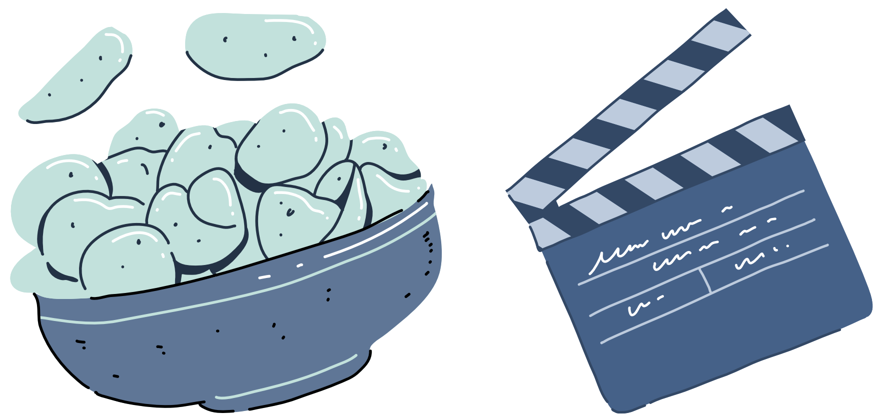
- Classes and Objects: Defines blueprints (Person, User, Movie, etc.) and creates instances of these classes.
- Inheritance: User, Director, and Actor inherit from Person, gaining its properties and methods.
- Encapsulation: Uses private data members (e.g., email, password) and provides public methods (e.g., getEmail(), rateMovie()) to access and modify them.
- Polymorphism: Allows methods to perform different tasks based on the object, inferred through method use.
- Abstraction: Hides complex details, exposing only necessary features via public methods (addUser(), generateRecommendations()).
- Data Management: Manages and manipulates data (users, movies, ratings) and handles file operations (loadUsers(), saveUsers()).
- Object Relationships: Models relationships between entities, like users rating movies and movies associated with genres, actors, and directors.



Demonstration Link

Demonstration Link





Thank you