**AI~ Movie Recommendation System**



**By**

**Emaan Dar(G1F22UBSCS228)**

**Areen Murtaza(G1F22UBSCS146)**

**Huma Noor(G1F22UBSCS152)**

**Project Report**

**Web Programming**

**BS Computer Science**

**Supervisor**

**Prof. Hamza Afzal**

**Semester – [VI]**

**Fall, 2025**

**Faculty of Computer Science**

**University of Central Punjab**

**Project Report : AI-Movie**

**Recommendation System (MERN + AI)**

1. **Project Title:**

**AI Content Recommendation System:** Suggest movies or shows based on user preferences and trends.

1. **Project Overview:**

This is a **MERN + AI-based Movie Recommendation System** designed to recommend movies to users. The AI engine is developed using **Python** and is responsible for generating movie recommendations. Movie data from CSV files is uploaded and stored in MongoDB using MongoDB Compass. When a user searches for a movie in our web application, the system analyzes the input and recommends similar movies from the database.

The backend is developed using Node.js and Express.js, while the frontend is built using EJS templates to create a smooth and consistent user experience. The interface is kept identical to the reference interface provided—nothing more, nothing less.

1. **System Modules & Features:**

* User Authentication Module
* Movie Recommendation Engine (AI Module)
* Movie Data Import and Management
* Frontend UI (Responsive Pages)
* Routing and API Layer

1. **Current Problem Area:**

Streaming platforms often rely on **generic recommendations,** leading to poor user engagement. A lack of personalized, real-time suggestions based on deep learning results in lower user retention and satisfaction.

1. **Technology Stack:**

|  |  |
| --- | --- |
| **Layer** | **Technologies Used** |
| Frontend | HTML, CSS, EJS (Embedded JavaScript Templates) |
| Backend | Node.js, Express.js |
| Database | MongoDB with Mongoose |
| AI Module | Python (for recommendation logic and data processing) |
| Libraries/Tools | bcrypt, express-session, body-parser, dotenv, MongoDB Compass |

1. **Project Structure Overview:**

movie\_recommendation\_system/

**├──** ai-engine/ # AI recommendation logic (Python)

**├──** data/ # Movie dataset and images

**├──** mongo\_import/ # MongoDB population scripts

**│ ├──** add\_posters.py

**│ └──** upload\_to\_mongodb.py

**├──** web-app/

**│ ├──** controllers/ # Handles business logic

**│ ├──** middleware/ # Authentication guard

**│ ├──** models/ # MongoDB models using Mongoose

**│ ├──** routes/ # API endpoints

**│ ├──** views/ # EJS templates for UI

**│ ├──** public/ # Custom CSS styles

**│ └──** app.js # Main application entry

1. **AI Module Details:**

Was AI Used? ✅ Yes

The AI module, built using Python, employs a collaborative filtering algorithm to recommend movies based on user interactions. It compares user preferences to generate personalized suggestions.  
 -Movie similarity is calculated using user behavior patterns.  
 -The AI module outputs recommendation data based on current user input.  
 -The AI logic is decoupled from the main application for modularity and easy improvements.

**Integration Flow:**  
1. Python scripts process the movie data.

1. AI generates similarity scores and rankings.  
   3. Recommended movies are made available to the backend.  
   4. Node.js retrieves these and renders them in the `recommendations.ejs` view.
2. **System Architecture Diagram:**

[User Browser] → [Frontend (EJS Templates)] → [Backend (Express.js Server)] → [Controllers] → [Models] ↔ [MongoDB] → [Python AI Engine]

1. **Group Member Contributions:**

|  |  |
| --- | --- |
| **Member Name** | **Contribution Area** |
| Areen Murtaza | Developed the AI Recommendation Engine using Python |
| Huma Noor | Built Login and Registration System with session management |
| Emaan Dar | Handled MongoDB imports, backend integration, and UI views |

1. **Conclusion:**

The Movie Recommendation System effectively combines MERN stack technologies with Python-based AI to deliver a smart, responsive, and user-friendly application. By leveraging collaborative filtering techniques, the system intelligently suggests similar movies based on user input, enhancing the overall user experience.

The project showcases practical implementation of multiple core concepts: backend API development with Node.js and Express.js, dynamic frontend rendering using EJS, structured data handling with MongoDB, and intelligent decision-making through a custom AI engine written in Python. The integration of CSV data with MongoDB ensures that the movie dataset remains structured and scalable for future updates.

This system not only fulfills the technical requirements of a modern web application but also demonstrates real-world applicability in domains like streaming services, entertainment platforms, and content recommendation engines. The simplicity and consistency of the interface, as per the design reference, ensures ease of use for end users.