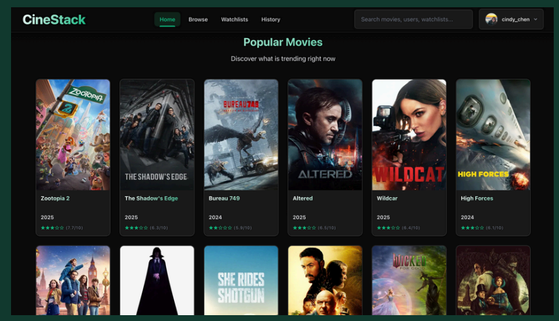
CS 4530: FINAL PROJECT “CINESTACK”

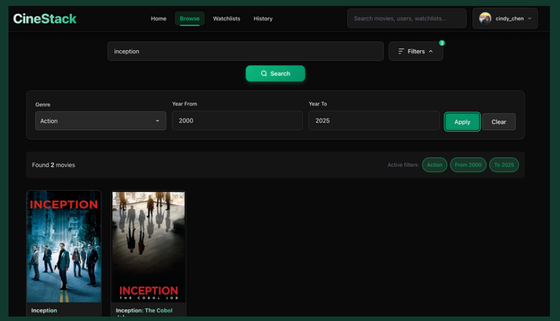
Group 107: Nurayn Guru, Smithi Saladi, Cindy Chen

Demo Site | Repo Site

Our Features:

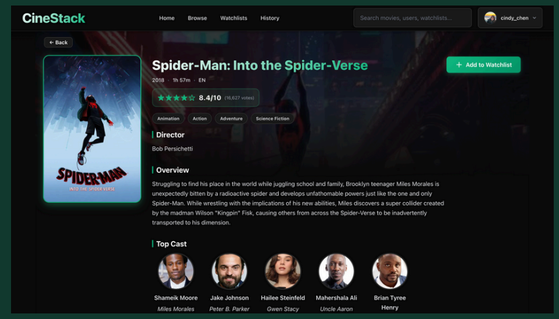
CineStack is a full-stack web app for movie reviews and recommendations. It is developed from FakeStackOverflow by converting the site's Q&A domain into a movie domain. Instead of questions and answers, users browse movies (integrated with TMDB), create/edit reviews with star ratings, like/comment on other users' reviews, and maintain watchlists. The backend REST APIs and WebSocket events support real-time updates (likes, new reviews). The frontend features movie detail pages, review feeds, and personalized recommendation carousels based on user ratings and activity.

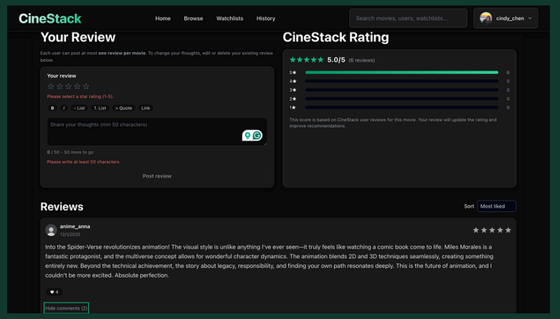




Tech Stack & Design Decision:

CineStack's app implementation uses the core architecture of the original FakeStackOverflow repo: TypeScript/Node/Express for the backend, React + Vite for the frontend, MongoDB via Mongoose ODM, and Socket.IO for real-time chat/notification features. On the server, we retained the existing controller/service/model layering, but replaced Q&A domain objects with movies and reviews, added TMDB integration, and re-used the existing auth pipeline (JWT, middleware, etc.) and shared type definitions. On the client, we use a routed SPA design with a number of reusable components for movie grids, movie details pages, review forms, and recommendation carousels, which are backed by a small set of API and recommendation service modules. We ensure reliability and maintainability with Jest + Supertest unit tests, Cypress E2E tests for a number of high-value flows (posting reviews, liking/recommending) and a GitHub Actions CI workflow that runs linting, unit tests, and E2E tests on each mainline integration.





Future Work:

Future CineStack improvements could include both features and underlying architecture. Feature-wise, we’d like to see richer social interactions (following other users, activity feeds, curated lists), more advanced recommendation signals (collaborative filtering, “because your friends liked…”), improved discovery tools (collections, filters, in-app notifications), and—should we be able to integrate with a licensed streaming provider—an embedded player to let users watch movies without leaving our site. From an engineering perspective, we see opportunity to extract the recommendation logic into its own service module, incorporate stronger caching and background jobs for TMDB sync, tighten type-safety in some legacy areas, and expand automated test coverage (especially E2E) to make future changes safer and easier to ship.