Group 41, Project2

Group Members:

Mihir Kamat, Mili Jolly, Raj Kalantri

Submission:

The following is a link to our repo:
https://github.com/SoftwareEngg2024/CineScout/tree/ver_i

58 TESTS

75% COVERAGE

CineScout is a dynamic movie recommendation engine that suggests new movies based on your preferences and watch history. CineScout helps users discover new favorites and enhance their viewing experience.

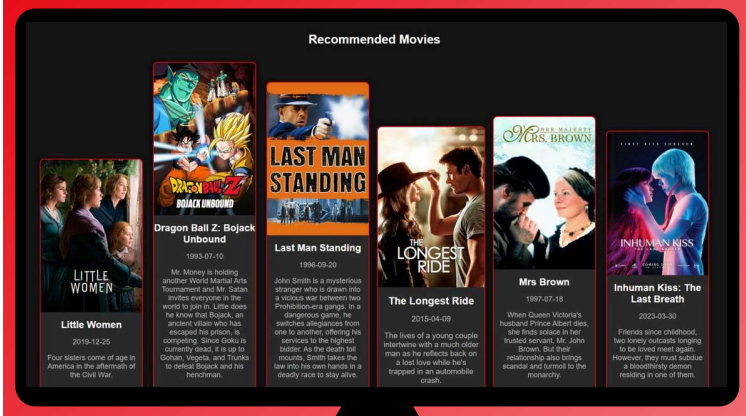
POWERED BY:

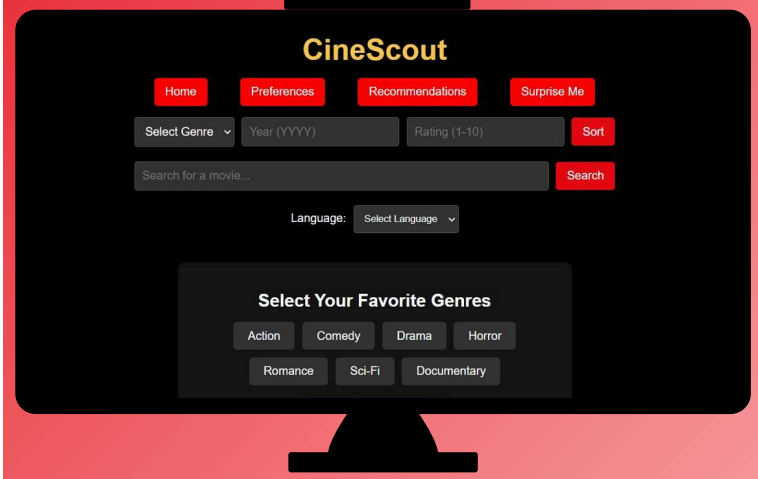
PYTHON FLASK REACT CSS

GROUP 41
Mihir Kamat
Mili Jolly
Raj Kalantri

CINESCOUT

YOUR PERSONALIZED MOVIE COMPANION





New Features Added!

- TMDb based Suggestions: CineScout analyzes watch history to provide personalized suggestions based on data from The Movie Database.
- "Surprise Me" Feature: Want to watch something unexpected? This feature offers recommendations from genres you rarely explore, helping you discover hidden gems.
- Enhanced User Interface with React: Our revamped UI, built with React, lets you choose and refine your movie preferences with ease. A clean, intuitive design to enhance your experience.
- Search & Sort by Year: Now, you can search for movies by title and sort them by release year, giving you full control over your movie search.

Why Work on CineScout?

- Innovative Challenge: The project combines backend, frontend, and data integration, offering a full-stack development challenge.
- Growing Social Element: With features like friend-suggested content, it brings a social media aspect to the movie recommendation space.
- Boost User Engagement: Gamification and interactive quizzes offer cutting-edge techniques to engage users in a fun and rewarding way.
- Creative & Timely Suggestions: Seasonal recommendations add a creative edge, allowing you to craft a truly dynamic engine.

Take this To the Next level!

- Friend-Suggested Content: Connect with friends, share watchlists, and discuss recommendations via MongoDB.
- Gamified Movie-Watching: Earn badges or rewards for trying new genres or watching a set number of films.
- Interactive Quizzes: Discover personalized movie recommendations through fun quizzes, enhancing user engagement.
- Seasonal Suggestions: Get recommendations based on seasonal events like Halloween or summer blockbusters.

Give our repo a visit

Live Demo

Member A: Raj Kalantri

Member B: Mili Jolly

Member C: Mihir Kamat

Notes	A	B	C	evidence
Workload is spread over the whole team (one team member is often Xtimes more productive than the others...	2	2	2	
but nevertheless, here is a track record that everyone is contributing a lot)	3	3	3	https://github.com/SoftwareEngg2024/CineScout/commits/ver_i/
Number of commits	3	3	3	Total commits: 76 https://github.com/SoftwareEngg2024/CineScout/commits/ver_i/
Number of commits: by different people	3	3	3	https://github.com/SoftwareEngg2024/CineScout/commits/ver_i/
Issues reports	2	2	2	https://github.com/SoftwareEngg2024/CineScout/issues
Issues are being closed	2	2	3	https://github.com/SoftwareEngg2024/CineScout/issues?q=is%3Aissue+is%3Aclosed
Docs: doco generated, format not ugly	2	2	2	
Docs: what: point descriptions of each class/function (in isolation)	3	2	3	
Docs: how: for common use cases X,Y,Z mini-tutorials showing worked examples on how to do X,Y,Z	3	2	2	doc page entries

Notes	A	B	C	evidence
Docs: why: docs tell a story, motivate the whole thing, deliver a punchline that makes you want to rush out and use the thing	2	2	2	
Docs: short video, animated, hosted on your repo. That convinces people why they want to work on your code.	2	2	3	
Use of version control tools	3	3	3	
Test cases exist	3	3	3	dozens of tests and those test cases are more than 30% of the code base
Test cases are routinely executed	3	3	3	
Issues are discussed before they are closed	2	2	2	We faced issues when testing the backend manually with postman. When examining the code
Chat channel: exists	1	1	1	Discord: <link> but mostly was unused as we exclusively met physically and discussed the particulars of the project. The bulk of the issues were solved in these discussions.
Test cases: a large proportion of the issues related to handling failing cases.	3	3	3	https://github.com/SoftwareEngg2024/CineScout/issues?q=is%3Aissue+is%3Aclosed
Evidence that the whole team is using the same tools: everyone can get to all tools and files	3	3	3	
Evidence that the whole team is using the same tools (e.g. config files in the repo, updated by lots of different people)	2	2	2	https://github.com/SoftwareEngg2024/CineScout/commits/ver_i/ We have different branches for core_algo, backend and the frontend and we edited each other's files when merging into the final ver_i branch.

Notes	A	B	C	evidence
Evidence that the whole team is using the same tools (e.g. tutor can ask anyone to share screen, they demonstrate the system running on their computer)	2	2	2	https://github.com/SoftwareEngg2024/CineScout/commits/ver_i/ We worked with different tools in different branches but all the tools are readily available to each member after merge.
Evidence that the members of the team are working across multiple places in the code base	2	2	2	Mili worked on the core backend algorithm, Mihir worked on the backend API and Raj worked on the react frontend.
Short release cycles	2	2	2	
The file .gitignore lists what files should not be saved to the repo. See [examples]i(https://github.com/github/gitignore)	3	3	3	https://github.com/SoftwareEngg2024/CineScout/blob/ver_i/.gitignore
The file INSTALL.md lists how to install the code	3	3	3	https://github.com/SoftwareEngg2024/CineScout/blob/ver_i/INSTALL.md
The file LICENSE.md lists rules of usage for this repo	3	3	3	https://github.com/SoftwareEngg2024/CineScout/blob/ver_i/LICENSE.md
The file CODE-OF-CONDUCT.md lists rules of behavior for this repo; e.g. see example	3	3	3	https://github.com/SoftwareEngg2024/CineScout/blob/ver_i/CODE_OF_CONDUCT.md
The file CONTRIBUTING.md lists coding standards and lots of tips on how to extend the system without screwing things up; e.g. see example	3	3	3	https://github.com/SoftwareEngg2024/CineScout/blob/ver_i/CONTRIBUTING.md
The file README.md contains all the following	3	3	3	in GH
Video	3	3	3	2min video of new functionality, showing a significant delta from prior.

Notes	A	B	C	evidence
DOI badge: exists. To get a Digitial Object Indentifier, regiser the project at Zenodo. DOI badges look like this:	3	3	3	https://github.com/SoftwareEngg2024/CineScout/blob/ver_i/README.md
Badges showing your style checkers	3	3	3	https://github.com/SoftwareEngg2024/CineScout/blob/ver_i/README.md
Badges showing your code formatters.	3	3	3	https://github.com/SoftwareEngg2024/CineScout/blob/ver_i/README.md
Badges showing your syntax checkers.	3	3	3	https://github.com/SoftwareEngg2024/CineScout/blob/ver_i/README.md
Badges showing your code coverage tools	3	3	3	https://github.com/SoftwareEngg2024/CineScout/blob/ver_i/README.md We used coveralls for getting code coverage. However, the coverage report is not accurate as we are testing a remote api and not calling the functions directly. We covered what we could by running the backend with coverage, but it might still not be reliable.
Badges showing any other Other automated analysis tools	2	2	2	config files in GH, badges in README

Sustainability Evaluation

Q1 - What your software does

Question 1.1: Does your website and documentation provide a clear, high-level overview of your software?

Yes

No

Question 1.2: Does your website and documentation clearly describe the type of user who should use your software?*

Yes

No

Question 1.3: Do you publish case studies to show how your software has been used by yourself and others?*

Yes

No

Q2 - Your project's and software's identity

Question 2.1: Is the name of your project/software unique?*

Yes

No

Question 2.2: Is your project/software name free from trademark violations?*

Yes

No

Q3 - Availability of your software

Question 3.1: Is your software available as a package that can be deployed without building it?*

Yes

No

Question 3.2: Is your software available for free?*

Yes

No

Question 3.3: Is your source code publicly available to download, either as a downloadable bundle or via access to a source code repository?*

Yes

No

Question 3.4: Is your software hosted in an established, third-party repository likeGitHub (https://github.com), BitBucket (https://bitbucket.org),LaunchPad (https://launchpad.net) orSourceForge (https://sourceforge.net)?*

Yes

No

Q4 - Your software's documentation

Question 4.1: Is your documentation clearly available on your website or within your software?*

Yes

No

Question 4.2: Does your documentation include a "quick start" guide, that provides a short overview of how to use your software with some basic examples of use?*

Yes

No

Question 4.3: If you provide more extensive documentation, does this provide clear, step-by-step instructions on how to deploy and use your software?*

Yes

No

Not applicable

Question 4.4: Do you provide a comprehensive guide to all your software’s commands, functions and options?*

Yes

No

Question 4.5: Do you provide troubleshooting information that describes the symptoms and step-by-step solutions for problems and error messages?*

Yes

No

Question 4.6: If your software can be used as a library, package or service by other software, do you provide comprehensive API documentation?*

Yes

No

Not applicable

Question 4.7: Do you store your documentation under revision control with your source code?*

Yes

No

Not applicable

Question 4.8: Do you publish your release history e.g. release data, version numbers, key features of each release etc. on your web site or in your documentation?*

Yes

No

Q5 - How you support your software

Question 5.1: Does your software describe how a user can get help with using your software?*

Yes

No

Question 5.2: Does your website and documentation describe what support, if any, you provide to users and developers?*

Yes

No

Question 5.3: Does your project have an e-mail address or forum that is solely for supporting users?*

Yes

No

Question 5.4: Are e-mails to your support e-mail address received by more than one person?*

Yes

No

Not applicable

Question 5.5: Does your project have a ticketing system to manage bug reports and feature requests?*

Yes

No

Question 5.6: Is your project's ticketing system publicly visible to your users, so they can view bug reports and feature requests?*

Yes

No

Not applicable

Q6 - Your software's maintainability

Question 6.1: Is your software's architecture and design modular?*

Yes

No

Question 6.2: Does your software use an accepted coding standard or convention?*

Yes

No

Q7 - Open standards and your software

Question 7.1: Does your software allow data to be imported and exported using open data formats?*

e.g. GIF, SVG, HTML, XML, tar, zip, CSV, JSON, NetCDF, or domain specific ones

Yes

No

Question 7.2: Does your software allow communications using open communications protocols?*

e.g. HTTP, FTP, XMPP, SOAP over HTTP, or domain-specific ones

Yes

No

Q8 - Your software's portability

Question 8.1: Is your software cross-platform compatible?*

e.g. does it run under two or more of Windows, Unix/Linux and Mac OS X, or can be used from within two or more of Internet Explorer, Chrome, Firefox and Safari?

Yes

No

Q9 - Your software and accessibility

Question 9.1: Does your software adhere to appropriate accessibility conventions or standards?*

Yes

No

Question 9.2: Does your documentation adhere to appropriate accessibility conventions or standards?*

Yes

No

Q10 - How you manage your source code

Question 10.1: Is your source code stored in a repository under revision control?*

Yes

No

Question 10.2: Is each source code release a snapshot of the repository?*

Yes

No

Not applicable

Question 10.3: Are releases tagged in the repository?*

Yes

No

Not applicable

Question 10.4: Is there a branch of the repository that is always stable? (i.e. tests always pass, code always builds successfully)*

Yes

No

Not applicable

Question 10.5: Do you back-up your repository?*

Yes

No

Not applicable

Q11 - Building and installing your software

Question 11.1: Do you provide publicly-available instructions for building your software from the source code?*

Yes

No

Question 11.2: Can you build, or package, your software using an automated tool?*

e.g. Make (https://www.gnu.org/software/make/), ANT (http://ant.apache.org/), Maven (https://maven.apache.org/), CMake (https://cmake.org/), Python setuptools (https://pypi.python.org/pypi/setuptools), or R package tools (https://cran.r-project.org/doc/manuals/r-devel/R-exts.html)

Yes

No

Question 11.3: Do you provide publicly-available instructions for deploying your software?*

Yes

No

Question 11.4: Does your documentation list all third-party dependencies?*

Yes

No

Question 11.5: Does your documentation list the version number for all third-party dependencies?*

Yes

No

Not applicable

Question 11.6: Does your software list the web address, and licences for all third-party dependencies and say whether the dependencies are mandatory or optional?*

Yes

No

Not applicable

Question 11.7: Can you download dependencies using a dependency management tool or package manager?*

e.g. Ivy (http://ant.apache.org/ivy/), Maven (https://maven.apache.org/), Python pip (https://pypi.python.org/pypi/pip) or setuptools (https://pypi.python.org/pypi/setuptools), PHP Composer (https://getcomposer.org/), Ruby gems (https://rubygems.org), or R PackRat (https://rstudio.github.io/packrat/)

Yes

No

Not applicable

Question 11.8: Do you have tests that can be run after your software has been built or deployed to show whether the build or deployment has been successful?*

Yes

No

Q12 - How you test your software

Question 12.1: Do you have an automated test suite for your software?*

Yes
No

Question 12.2: Do you have a framework to periodically (e.g. nightly) run your tests on the latest version of the source code?*

Yes
No
Not applicable

Question 12.3: Do you use continuous integration, automatically running tests whenever changes are made to your source code?*

Yes
No
Not applicable

Question 12.4: Are your test results publicly visible?*

Yes
No
Not applicable

Question 12.5: Are all manually-run tests documented?*

Yes
No
Not applicable

Q13 - How you engage with your community

Question 13.1: Does your project have resources (e.g. blog, Twitter, RSS feed, Facebook page, wiki, mailing list) that are regularly updated with information about your software?*
e.g. release announcements, publications, workshops, conference presentations

Yes
No

Question 13.2: Does your website state how many projects and users are associated with your project?*

Yes
No

Question 13.3: Do you provide success stories on your website?*

Yes
No

Question 13.4: Do you list your important partners and collaborators on your website?*

Yes
No

Question 13.5: Do you list your project's publications on your website or link to a resource where these are available?*

Yes
No

Question 13.6: Do you list third-party publications that refer to your software on your website or link to a resource where these are available?*

Yes
No

Question 13.7: Can users subscribe to notifications to changes to your source code repository?*

Yes
No

Question 13.8: If your software is developed as an open source project (and, not just a project developing open source software), do you have a governance model?*

Yes
No
Not applicable

Q14 - How you manage contributions

Question 14.1: Do you accept contributions (e.g. bug fixes, enhancements, documentation updates, tutorials) from people who are not part of your project?*

Yes
No

Question 14.2: Do you have a contributions policy?*

Yes
No
Not applicable

Question 14.3: Is your contributions' policy publicly available?*

Yes
No
Not applicable

Question 14.4: Do contributors keep the copyright/IP of their contributions?*

Yes
No
Not applicable

Q15 - Your software's copyright and licensing

Question 15.1: Does your website and documentation clearly state the copyright owners of your software and documentation?*

Yes
No

Question 15.2: Does each of your source code files include a copyright statement?*

Yes
No

Question 15.3: Does your website and documentation clearly state the licence of your software?*

Yes
No

Question 15.4: Is your software released under an open source licence?*

Yes
No
Not applicable

Question 15.5: Is your software released under an OSI-approved open-source licence?*

Yes
No
Not applicable

Question 15.6: Does each of your source code files include a licence header?*

Yes
No
Not applicable

Question 15.7: Do you have a recommended citation for your software?*

Yes
No

Q16 - Your plans for the future

Question 16.1: Does your website or documentation include a project roadmap (a list of project and development milestones for the next 3, 6 and 12 months)?*

Yes
No

Question 16.2: Does your website or documentation describe how your project is funded, and the period over which funding is guaranteed?*

Yes
No

Question 16.3: Do you make timely announcements of the deprecation of components, APIs, etc.?*

Yes
No