

Netflix roulette

User experience assessment report

Quick links

[GitHub repo](#)

[XD wireframes](#)

[Demo](#)

[Project assessment](#)

This document is intended to be viewed along the [project assessment report](#).

Purpose

This application will be designed to serve users with a selection of a random Netflix title. Its sole purpose is to help users choose a title from Netflix, so that they can stop playing hot potato with their Apple TV remotes.

Research methodologies

The following are the three primary methods used to research and gather feedback from the target audience. Information gathered from these methods are then used to further develop the application to ensure that it meets the needs of the intended audience.

User surveys and interviews

Participants are met one-on-one to discuss in depth about their experience with the potential designs and the application.

Surveys and interviews allows me to better understand the users and their attitude towards the application or in general. It also reveals their usual approach to the problem and how they think about it: what do they usually do when they can't think of something to watch? What could be better in the application that could help them solve the issue?

These interviews enables me to draw out the type of details needed to develop useful or better features from users, which will help me to design features in the application that addresses the problem.

This is the primary method of gathering information from potential users.

Due to technical restrictions and security concerns, it was not possible to engage users with email surveys or intercept surveys.

Participatory designs

Participants are asked to provide feedback and construct a potential model for the application that they find ideal.

The primary focus of this method is to ensure that the application maintains user engagement by asking them to demonstrate the values that matters to them and meeting those set expectations.

Participatory designs helps highlights potential flaws in the design. Participants can help provide inputs on whether the implementation will work, or how it could be improved, or if other users would use it. Having users involved in the design process also can also assist in understanding how a user would interact with the design, as opposed to developing designs based on pure speculation. Knowing how actual users interacts with the designs would yield a higher chance of ensuring an effective and successful design.

User interviews are also often conducted during test phases in order to tweak the application to the feedback provided.

Not all participants are engaged in participatory designs. These participants are primarily chosen from my computer science class.

Usability testing

Participants are observed as they are using the application to accomplish the task.

Given that this application use case is mainly based on a single process, users are simply asked to generate a Netflix title. This is followed by a survey, in which users are asked for their feedback on how the application could be improved to better suit their needs.

Observations of the user can help reveal the things that worked well in the application, and things that didn't. Usability testing is carried out throughout the development cycle of the application, right from the wireframes sketches. Small-scale tests are carried out in order to identify potential issues and improvements early, as to not waste time developing the application the wrong way, and to avoid repeating pitfalls that may arise during the cycle.

Note on feedback collection

Wireframes made on Adobe XD can be easily shared and viewed on browsers. However, the application itself could not be easily shared via something like a URL for users to test. In order for users to run the compiled program, their systems must have Java runtime environments

installed, which is less common among users to have, and is therefore harder to get them to test on their systems.

Due to unresolvable issues with the compiler, a runnable JAR file could not be created for distribution, even for systems that meets the requirements. During testing, it appeared that the compiler had failed to include the dependencies required to run the program.

In addition, further security limitations prevents the application to be distributed for testing, as the end users will not have the credentials required for the application to function properly. Instead, feedback is gathered by asking participants to directly interact with the program from my computer.

Target demographic

Anyone with a Netflix subscription.

Of the users that use a streaming service, 84% of them uses Netflix.¹ Demographics that are likely to use this application are younger audience. About 77% of 19-29 year olds have an active Netflix subscription.²

From personal experience, there are times where other people and myself struggle to find a title to watch on Netflix, passing the remote endlessly. This issue appears to be common among other young people, even becoming an internet meme that some may refer to.

Given the abundance of users that fall into the target demographic at school, the majority of respondents have reported that they also run into a similar situation with their family and other people that they watch Netflix with. One respondent said that having a tool to select a title for them would be “pretty handy” and could potentially “improve the experience [watching Netflix].”

Design goals

Given its simple purpose, the application must be easy and straightforward to use. If a user is looking for a title to watch, it is likely that they would want to spend as little time as possible looking a title.

In order to improve the experience of users watching Netflix, the application must serve a random Netflix title effortlessly, with as little interaction as possible.

¹

<https://www.forbes.com/sites/jeffewing/2019/02/12/new-research-highlights-streaming-demographic-trends/>

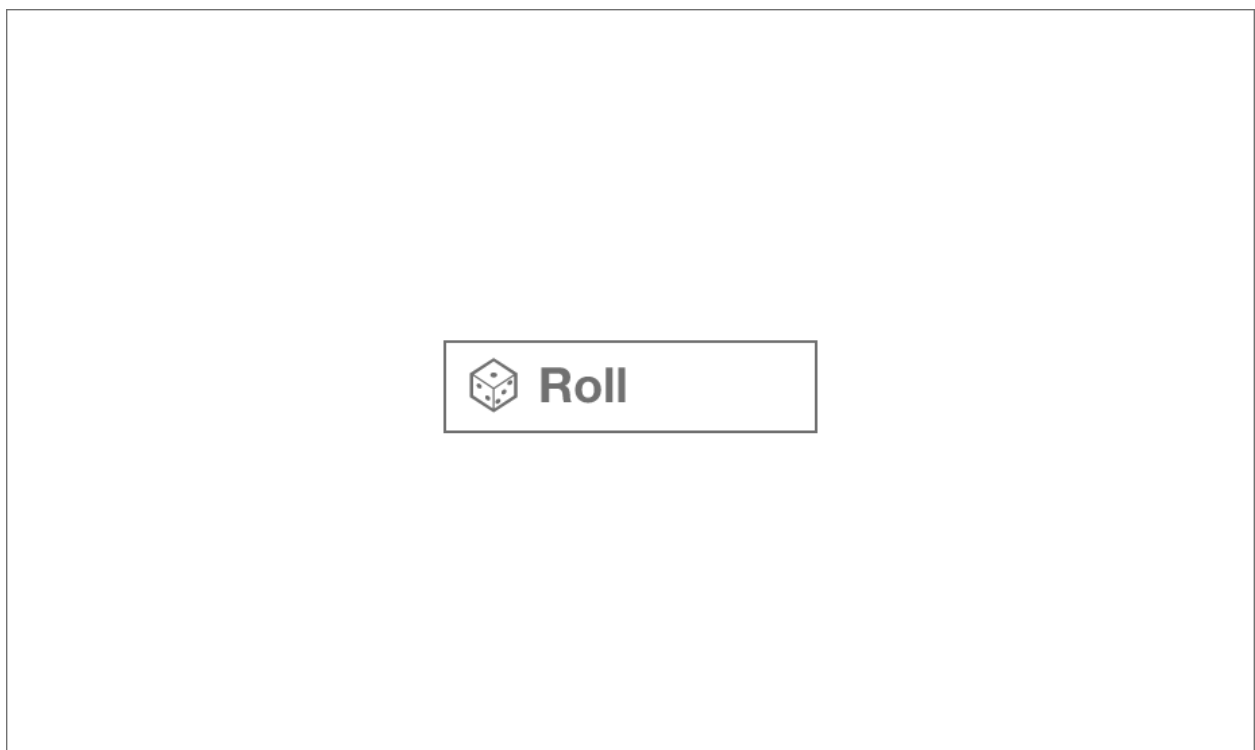
² <https://www.statista.com/statistics/720723/netflix-members-usa-by-age-group/>

Initial wireframes

Before developing the application, wireframes were made in order to gather feedback from users.

MVP

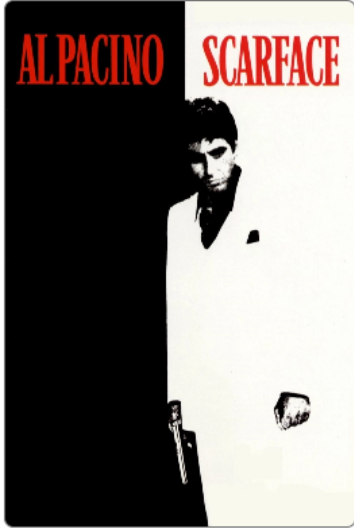
The [first design](#) showcases what the application may look like. This design displays the most basic features that the application will have as a minimum viable prototype. It features a single-button main menu that simply says “Roll.”



Upon clicking the button, a selected title is displayed, alongside its name, poster image, synopsis, and other information.

- The **name of the title** written in bold and in the largest font size and placed at the top, with its **runtime and duration** below it.
- The **poster image** is on the left hand side, and spans the majority of the height, so that the user can easily identify the title.

This initial design is made so that the user will be served with a title by just clicking one button, streamlining the interaction as much as possible. And if the user is not satisfied with the selected title, they would only need to click the “Roll again” button.



AL PACINO SCARFACE

Scarface

1983 · 2h 50m

After getting a green card in exchange for assassinating a Cuban government official, Tony Montana (Al Pacino) stakes a claim on the drug trade in Miami. Viciously murdering anyone who stands in his way, Tony eventually becomes the biggest drug lord in the state, controlling nearly all the cocaine that comes through Miami. But increased pressure from the police, wars with Colombian drug cartels and his own drug-fueled paranoia serve to fuel the flames of his eventual downfall.



Feedback summary

Overall, participants appeared to be satisfied with the design. The majority of the participants have noted that the design is “simple to use and effective” and is generally “good 😊.”

Several participants have pointed out that there is no button to return to the main page. While the button will need to be included later on as the application develops, there is currently no need for the button to be added for the application to complete the minimal task.

Participants have also requested that the application should include a button where the user can access the displayed Netflix title from the application, instead of having to search for it again on the Netflix platform.

Genre selection

The [second design](#) includes an interface for the selection of genres. This design was intended to be developed after the MVP has been developed. This is to ensure that the application was able to accomplish the minimal task, before developing it into a more complex program.


This wireframe was made alongside the MVP design. Note that these were developed prior to the technical development of the application.

Genre

<input checked="" type="checkbox"/> Action	<input checked="" type="checkbox"/> Cult	<input checked="" type="checkbox"/> Horror	<input checked="" type="checkbox"/> Romance
<input checked="" type="checkbox"/> Anime	<input checked="" type="checkbox"/> Documentaries	<input checked="" type="checkbox"/> Independent	<input checked="" type="checkbox"/> Sci-Fi
<input checked="" type="checkbox"/> Childrens	<input checked="" type="checkbox"/> Dramas	<input checked="" type="checkbox"/> International	<input checked="" type="checkbox"/> Sports
<input checked="" type="checkbox"/> Classics	<input checked="" type="checkbox"/> Spirituality	<input checked="" type="checkbox"/> Music	<input checked="" type="checkbox"/> Thrillers
<input checked="" type="checkbox"/> Comedies	<input checked="" type="checkbox"/> LGBT	<input checked="" type="checkbox"/> Musicals	

Type

<input checked="" type="checkbox"/> Movie	<input checked="" type="checkbox"/> Series
---	--

 **Roll**

The purpose of this design is relatively straightforward. This interface is designed to allow users to filter the results to only include specific genres.

You may notice that all of the genres are selected in the screenshot above. This is intended to be the default state of the checkboxes when the application is launched. So that, by default, all of the genres are included in the result, and the user will be able to click “Roll” straightaway, without taking any additional actions. In turn, they may unselect the genres that they would like to exclude from their results, and then click the “Roll” button.

The interface that displays the selected title remains the same, aside from the addition of a button for users to return to the main menu. This is to allow users to modify the selected filters after rolling. Alternatively, they may simply choose to reroll with the same filters.



John Rambo

2008 · 1h 39m

Having long-since abandoned his life as a lethal soldier, John Rambo (Sylvester Stallone) lives a solitary life near the Thai border. Two weeks after guiding a missionary (Julie Benz) and her comrades into Burma, he gets an urgent call for help. The missionaries have not returned and although he is reluctant to embrace violence again, Rambo sets out to rescue the captives from the Burmese army.

Main menu



Roll again

Feedback summary

Participants were generally satisfied with the design.

When participants were asked about the addition of the checkboxes, they appeared to be satisfied with the design. A participant noted that “it is good that [the checkboxes] are alphabetically-ordered.” The same participant also suggested the addition of a “select all” and “clear all” buttons for the genres, citing that it could be more convenient for users that are on;y looking for a particular set of genres.

A participant noted that the buttons on the window where the selected title is displayed should be “more consistent.” Where the reroll button has an icon beside it, but the main menu does not.

Application versions

After collecting feedback from potential users with the wireframes, the application is then built, largely based on those designs, alongside the improvements and feedback.

Note that the MVP version of the application is built first. As noted earlier, this is done to ensure that the minimum required components of the application is able to function properly, before adding additional features, such as genre selection. In addition to genre selection, more features are also added later on during the development of the application.

0.0: MVP

git checkout 6b6e2ed

See PR [#17](#). Last commit on branch at [6b6e2ed](#).

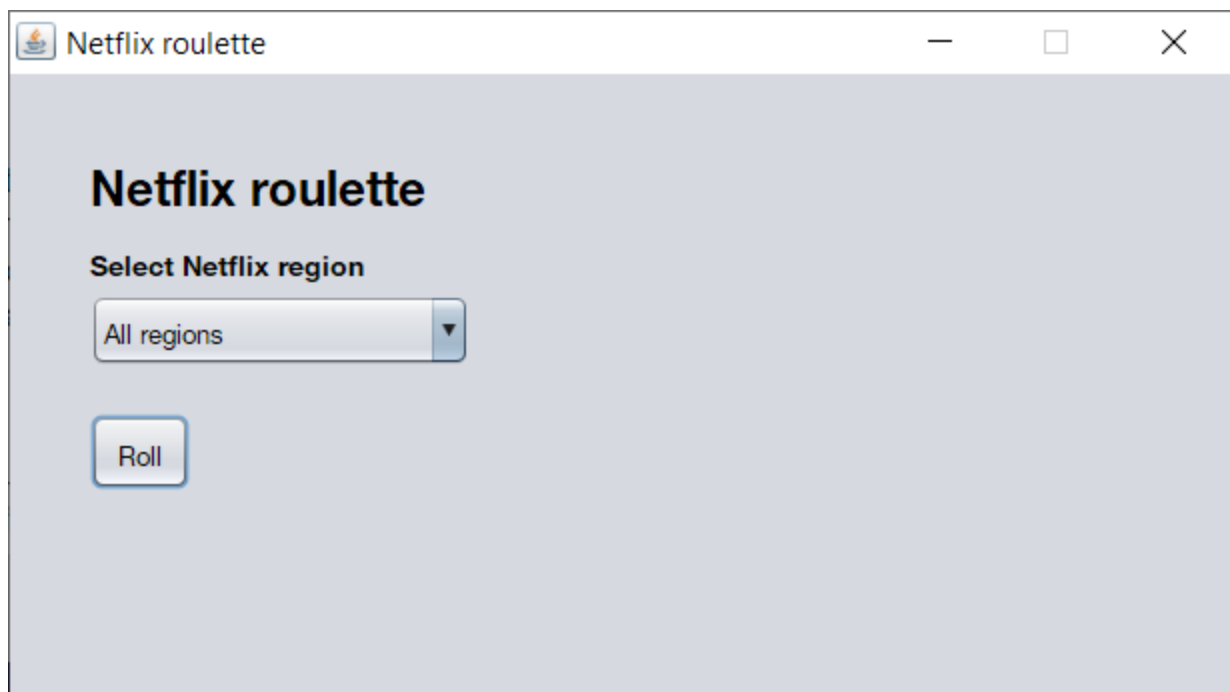
As you may have noticed straight away, the interface looks a bit rough compared to the wireframe. Keep in mind that the earlier versions are made to ensure that the application functions properly. The frontend of the applications are polished later on.

Aside from the less-appealing interface, you may have also noticed that the window is a little bit more populated from the wireframe of the MVP.

A drop-down list was added to the window for users to select their Netflix regions. During the development of the application, it was realized that users needed to select their Netflix regions in order to ensure that the titles that are being displayed to them can actually be viewed from their regions.

Keeping ease of use in mind, “All regions” is selected by default, in case the user ignores the field. This is so that the application will still be able to display a random title regardless.

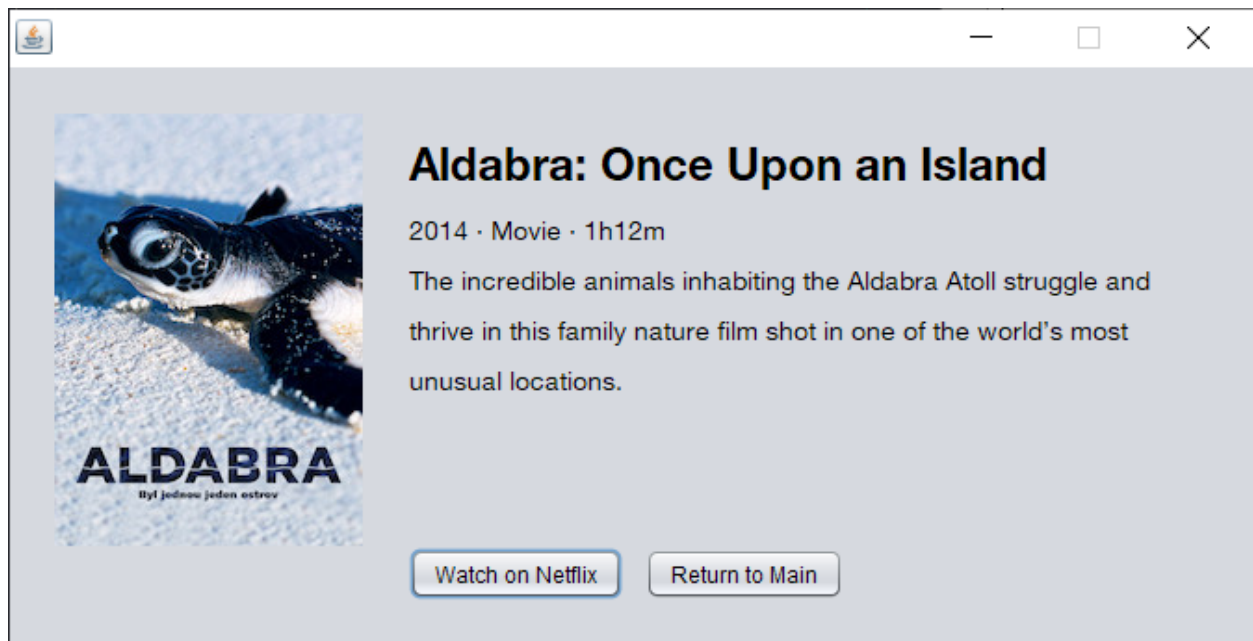
The “Roll” button has also been moved to the top-left corner instead, respective to the other components. The dice emoji has also been removed, due to rendering issues.



Upon clicking “Roll,” the user is then presented with a window that displays the selected title. This window is similar to the wireframe made for it, apart from the ugly [Nimbus](#) theme.

A “Watch on Netflix” button has been added, so that users can access the selected title directly from the application, as suggested by one of the participants.

You may notice that the “Reroll” button is removed from this window. This is because its implementation on the technical side is a little more complicated, so it’s been removed for now. It will make an appearance later on.



Feedback summary

Aside from the absence of the “Reroll” button, participants are generally satisfied with the current interface.

Most participants noted that the current interface is different from the wireframes, which is expected, as this is to simply serve as an MVP to model the application on.

A proportion of the participants that did not see the initial wireframe requested a feature to filter the type of titles to be returned. This is a planned feature for the following iteration of the application, [despite initial decision not to implement it](#).

With the addition of the Netflix regions selection, participants reported having trouble finding their region (New Zealand), despite the list being ordered in alphabetical order. This is because New Zealand does not appear in the list of available Netflix regions. The list is derived from the API that is used for requesting titles, and appears to be non-exhaustive. For more information

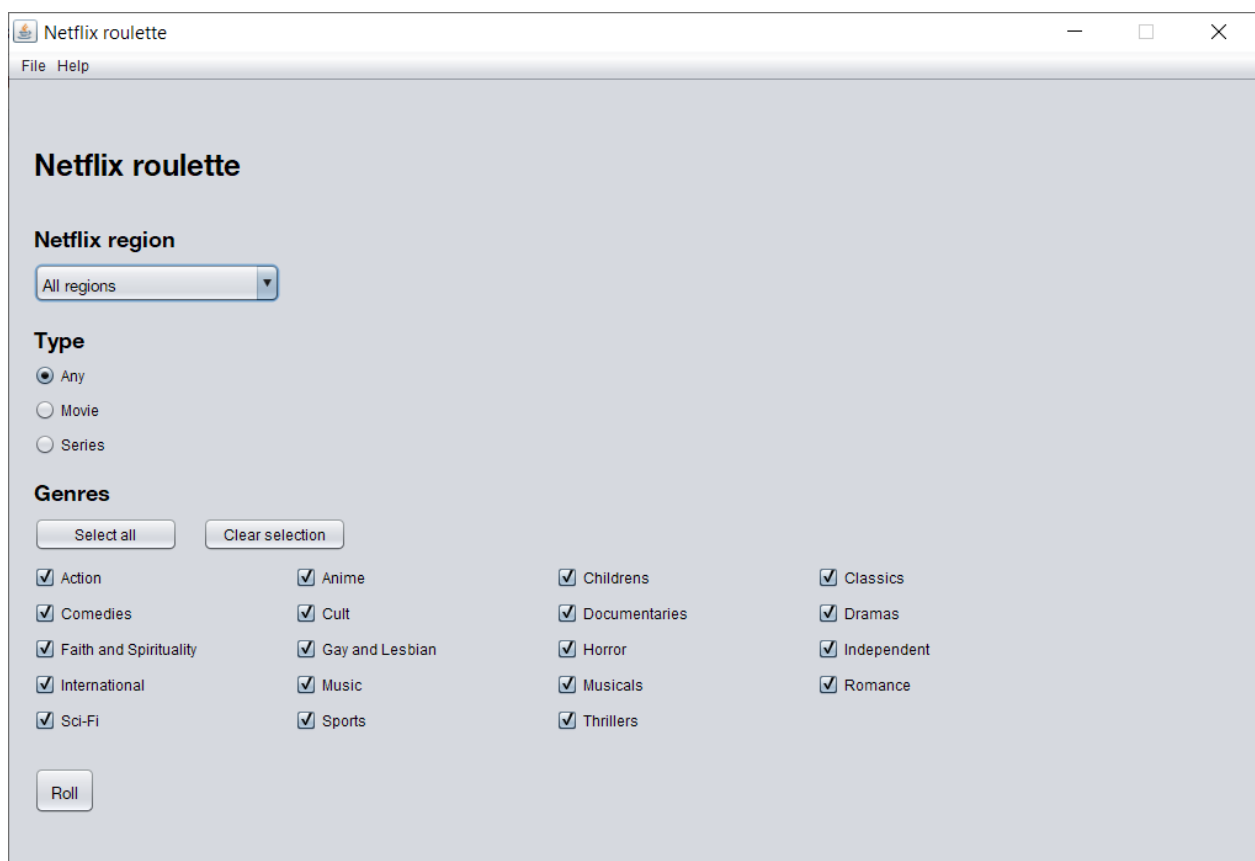
regarding this issue, please read on how [response-dependent GUI components](#) function in this application. Participants have suggested that clarifications should be made, for example, in the form of a help menu.

Participants have also mentioned that error messages that may appear during the use of the application, for example the “No matching titles” dialog, should also be addressed in a help menu.

0.1: Genre selection

git checkout 75ae2d2

See PR [#21](#) and [#23](#). Last commit on branch(es) at [75ae2d2](#).



The checkboxes for genre selection have been added to the interface, similar to those in the wireframe. This iteration also includes the option for users to select the title type (either movie or series).

A “Select all” and “Clear selection” button has been added alongside the checkboxes for genre selection, following user suggestions from initial wireframes.

A navigation bar is added to the top of the application in preparation for additional features to be added to the application, one of which is the help menu that a user requested. For now, it remains non-functional.

The selected title window remains the same.

Feedback summary

Following repeated user requests to add the ability to filter out specific genres, the feature was finally implemented. Both first-time users and participants that have requested this feature are generally satisfied with the implementation.

A participant that requested this feature earlier noted that this was a “crucial feature” and that the current implementation, alongside with the selection of title type, should “pretty much meet most people’s needs.”

Several participants that have requested this feature and one first-time user requested a feature that would allow them to filter the responses by the rating of the title. This feature was planned to be implemented in the next iteration of the application.

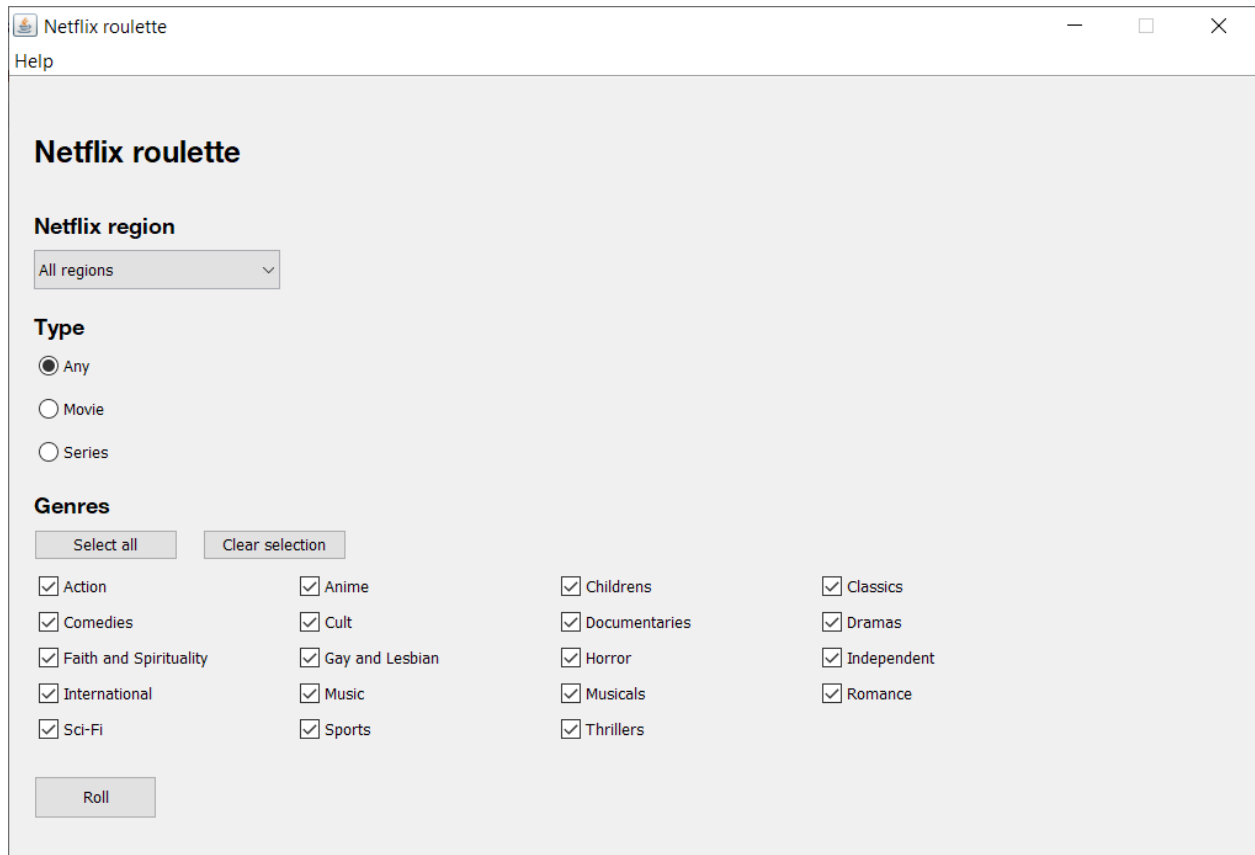
One participant suggested that the radio buttons for title type selection should be horizontal and moved up beside the region selection combobox, so that the window does not have to be “unnecessarily tall.” This suggestion came from a participant that took part in participatory designs.

A first-time user noted that the buttons on the window that displays the selected titles are “ugly.” When questioned, the respondent said “I don’t know, it’s just ugly.” Note that the iteration is not made to be aesthetically-pleasing.

0.2: GUI enhancements

`git checkout a9bee3c`

See PR [#24](#). Last commit on branch at [a9bee3c](#).



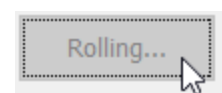
Before more features were added, the aesthetics-side of the application is addressed to make it look a bit more tidy.

This update addresses most of the feedback from users, including the functional and aesthetic components of the application.

The Nimbus theme is ditched for OS-dependent ones. In this case, the application uses the Windows theme. While it isn't exactly a fancy interface with colors and graphics, the theme gives it a much more cleaner and brighter look. Drop-down lists, buttons, and other interactive elements looks a million times better than it did in the Nimbus theme.

The fonts have been made to look more consistent. Non-interactive components will use Helvetica. Other interactive components will use Tahoma, consistent with Windows' UI fonts. Even though this is a small change, this can greatly help the users to distinguish the components at a glance.

Visual indication has been added for the "Roll" button when it is fetching titles, and also to prevent users from spamming the button, which may cause unintended results.

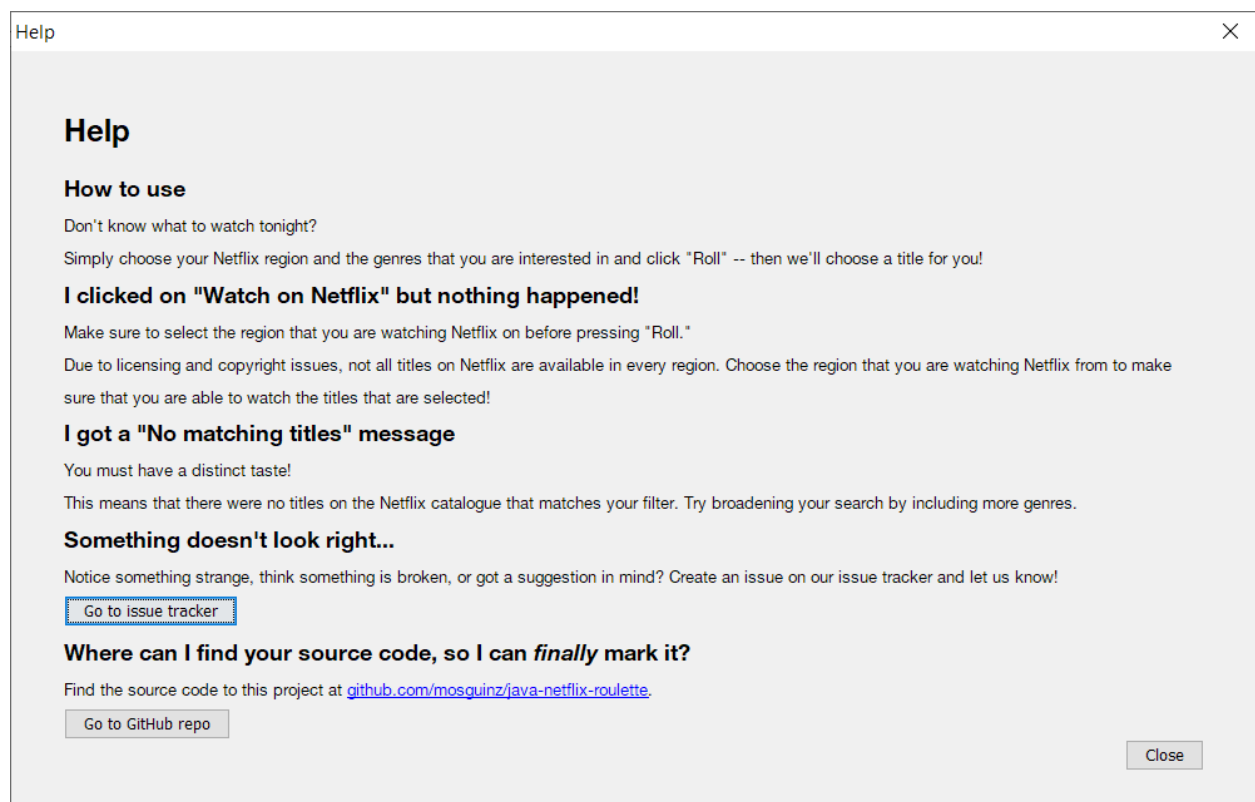
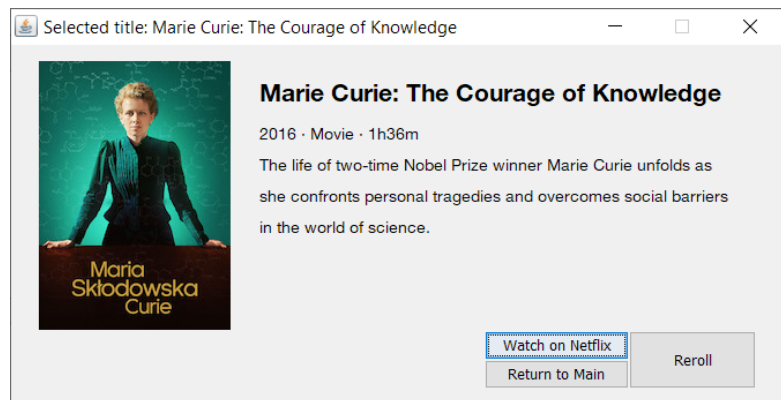


The buttons on the selected title dialog has been rearranged with the return of the “Reroll” button.

With the “Watch on Netflix” button being the first interactive component in the window, it is automatically highlighted to hint users that they can click on the button to watch the selected title.

The reroll button is made to be twice the height of the other two buttons, to maintain the consistency of the large, wide “Roll” button in the main menu. This also provides a subtle visual cue to the user that these buttons are used for requesting a title.

The help dialog has been added, following a user suggestion. The dialog addresses some of the common issues that the users may run into as they use the application. It includes some of the FAQs and links to the GitHub issues page for users to report bugs.



Other minor QoL features, such as the addition of tooltip texts and using modal dialogs, are also addressed. For an exhaustive list of changes made, please refer to the referenced pull request above.

Feedback summary

Participants have generally responded that the interface is much more cleaner and “looks a lot more polished” compared to previous iterations.

Participants have also noted that the addition of tooltip text can be particularly helpful for users that may use accessibility tools, and generally for those that may require guidance to use the application.

One participant noted that the contrast between the font color and the background is “much better” than the other one, and that the application feels similar to other Windows executables.

Aside from additional feature requests, participants were satisfied with the current layout of the application. No additional comments or suggestions were made regarding the interface.

0.3: Rating filters

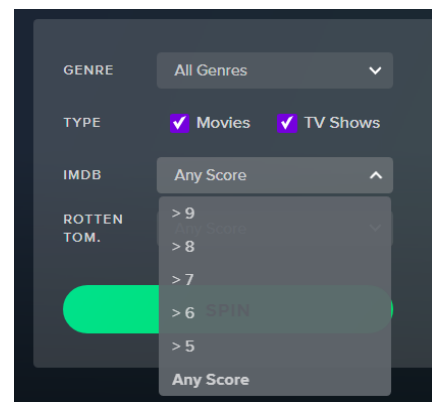
git checkout b1027f5

See PR [#25](#). Last commit on branch at [b1027f5](#).

This update is made in response to the feature request made by several participants, over different iterations of the application.

It was initially planned to be implemented as a combo box, where users may choose ratings in increments of two, where each would return titles that have ratings greater than the selected values. For example, > 6.0, > 8.0, > 9.0, etc.

This idea initially came from the implementation on [reelgood.com](#), where a list was used. However, during a co-designing session with a group of target audience, one user said that there should be a way to specify both the minimum and maximum rating values, rather than just a “greater than x” selection, reasoning that some users may want to look for “bad movies.”



Netflix roulette

Help

Netflix region

All regions

Type

☒ Any

☐ Movie

☐ Series

Viewer ratings

Minimum **3.0** Maximum **9.0**

Genres

Select all Clear selection

☒ Action ☒ Anime ☒ Childrens ☒ Classics

☒ Comedies ☒ Cult ☒ Documentaries ☒ Dramas

☒ Faith and Spirituality ☒ Gay and Lesbian ☒ Horror ☒ Independent

☒ International ☒ Music ☒ Musicals ☒ Romance

☒ Sci-Fi ☒ Sports ☒ Thrillers

Roll

Feedback summary

Because the addition of the rating filters was a little less straightforward and obvious, a rough plan was made to discuss its implementation, prior to developing the feature itself.

A co-designing session was held with my CS classmates to discuss how the rating filters could be implemented in the application. A participant suggested the usage of sliders, instead of a drop-down box, as a rating filter.

Participants that tested previous iterations of the application were also surveyed on how this feature should be implemented, where a consensus couldn't be reached. Participants suggested various methods, including the use of sliders, text fields, spinners, and radio buttons... where the last method was definitely not even up for consideration in the slightest.

Apart from users that were involved in participatory designs, other participants are generally satisfied with the addition of the sliders.

Users that do not wish to filter titles by their ratings do not have to use the sliders, as its default value will allow it to include titles with all ratings.

Viewer ratings

Minimum **4.2** Maximum **6.9**

Several participants requested that the sliders support more precise increments. Since IMDb ratings goes up to the nearest tenth, a slight modification have been made to support this change. Participants have also cited familiarity regarding the rating values being in accurate up to one decimal point. This is likely the result of the widespread use of IMDb ratings.

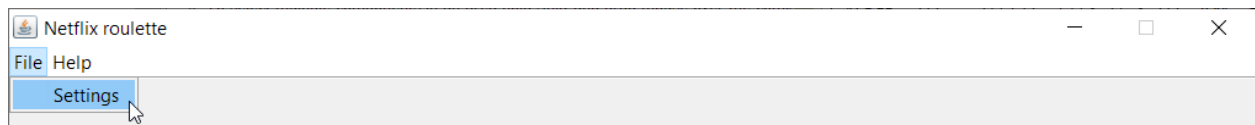
1.0: Settings dialog

git checkout f2309aa

See PR [#26](#). Last commit on branch at [f2309aa](#).

This was the last addition of features to the application. The layout and features of this version reflects the final version of the application. Changes made following this pull request are all bug fixes, minor adjustments, and QoL features.

Aside from the menu bar looking strange with just one menu button at the top, it was decided that a dialog should be added in order for users to easily access the folder where the response from the API are cached. Read more about how and why responses are cached [here](#).



From here, users can open the folder location of the cache. A button to clear all local cache has also been implemented, in case the user is getting unexpected (possibly outdated) results.



Information on the size of the cache is also available, should the user feel the need to delete the folder. Even though it takes up little to no space.

Not too many users would actually benefit from this feature. It's just a little gimmick that was developed to make the application seem a little cooler. However, it may be useful for those that are interested in how the application works. Say, for example, those that may be assessing it.

This is a purely supplemental feature that would be the least likely to be utilized. It is deliberately kept away in the menu bar, in order to not steer users away from the main function of the application. And to also ensure that the user's interaction with the application remains streamlined as much as possible for users that may *only* be interested in generating a random title without utilizing other features, such as additional filters.

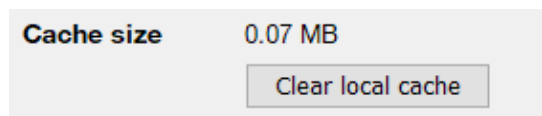
Feedback summary

Unlike other iterations of the applications, where participants would simply be asked to interact with the application whichever way they'd like, users are instead asked to complete a set of tasks.

Since this feature does not interact with the main part of the application, users are explicitly asked to access the cache folder from the application and clear it during usability testing. The observation would demonstrate whether the designed interface is obvious enough for participants to complete the task efficiently.

All of the participants were able to locate the dialog within about 15 seconds. The majority of users noted that the location of the menu was "quite obvious" as all other components in the main menu does not appear to "have anything to do with the settings stuff."

All participants from my CS class suggested that the file size units displayed in the dialog should instead be in megabytes, rather than bytes, citing familiarity with the units in real life, opposed to the inflated figures in bytes. A modification was made to reflect this feedback.



Note

The above list only highlights the most notable changes and improvements made as a result of using the aforementioned UX-building methodologies. There are many more minor changes that were made from these UX methods and otherwise that are not listed. For a complete list of changes made due to user feedback, please refer to the [commit messages](#).

Relevant implications

Legal

A disclaimer has been placed in the “About” window to notify end users that the usage of the trademark “Netflix” is only for demonstration purposes only.

The data obtained about titles and catalogue are provided by a third-party API provider, Unofficial Netflix Online Global Search (“uNoGS”) via RapidAPI.

As an API provider on RapidAPI, uNoGS agrees that it adheres to and are responsible for all the support and all claims relating thereto (e.g., product liability, legal compliance or intellectual property infringement). As mentioned in RapidAPI’s Terms of Service (<https://rapidapi.com/terms>).

This application is not endorsed by, directly affiliated with, maintained, authorized, or sponsored by Netflix, Inc., uNoGS, or RapidAPI. All product names, trademarks, and registered trademarks are property of their respective owners. All company, product and service names used in this application are for identification purposes only. Use of these names, trademarks, and brands does not imply endorsement.

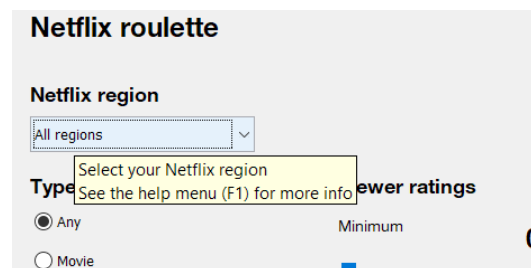
Poster images are owned by their respective publisher or creator of the work depicted. The images used are for demonstration purposes only and not solely for illustrations. These images are of low resolution and may qualify as fair use or fair dealing for educational purposes.

Accessibility

While the application is not built with accessibility in mind, some measures have been put in place to ensure that most users could use the application.

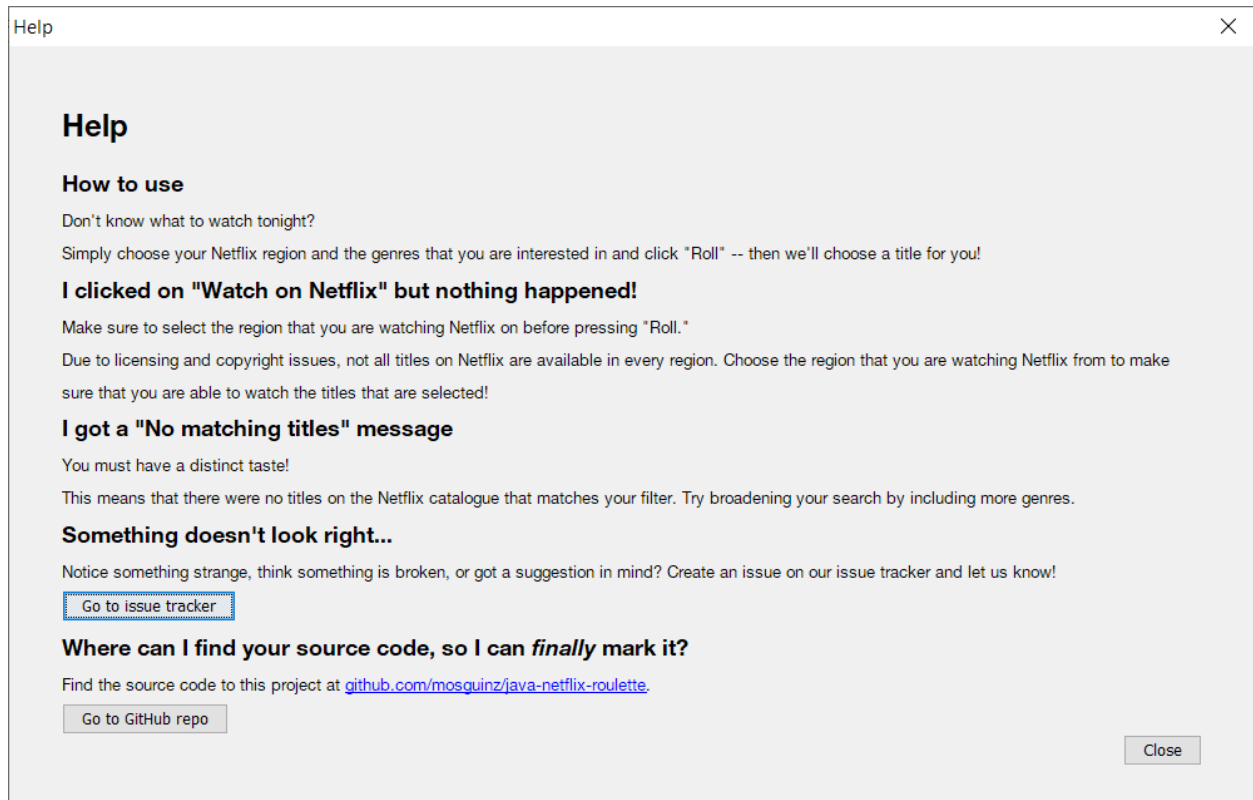
The sequence of the GUI components have been ensured that they are ordered in a sensible sequence, so that screen readers would read out the text labels and tooltips in the correct order.

Tooltip texts have also been added to buttons and components to describe their purpose. Explanatory texts are added to less self-explanatory labels and interactive components for screen readers, and also simply to guide users.



End-user considerations

A help dialog is included in the application to guide the user on how to use the application and how to troubleshoot common issues that they may run into. A button to open the issue tracker on GitHub has also been added for users to report bugs.



Aesthetics

Fonts

The application is designed to mainly use Helvetica as its default font. However, not all systems -- in fact, most non-macOS -- will not have the font installed. Instead, sans-serif alternatives, such as Arial, will be used in place of the font. While this isn't ideal, as Arial is obviously inferior and uglier, this issue is unavoidable due to font licensing issues.

Even with Helvetica installed on my Windows machine and access to countless Adobe Typekit fonts, they cannot be embedded along with the application for distribution, as the licences for these fonts explicitly prohibits this.

Color palette & designs

As you've noticed from the screenshots of the final version of the application, it's not ugly, but it's not exactly a pretty one either. To my surprise, most users seemed to have agreed, noting that the interface "looks fine."

This could be something that may be developed further, given that most of the functionality is limited by the API. And that most filters have been developed for this application.

Window and image sizes

The dialog that displays the selected title is about 700 by 300 pixels. This is bounded by the size of the poster image that is returned from the API.

The poster image of titles that are returned from the API are all 166 px in width, most of which are also 233 px in height. Because of this, this dialog, and the rest of the application has to be designed around the size of this image.



It was decided that other components should be sized with respect to the size of the poster. This is why the dialog is designed to be of a fixed size, so that the proportion between the image and other components are appropriate. The poster image could have also been resized, if the dialog were made to be resizable. However, this would result in the poster image appearing to be blurry as its native resolution is low.

Another reason why the low resolution image is used is because its usage in this application are believed to be qualified as fair use or fair dealing for educational purposes.

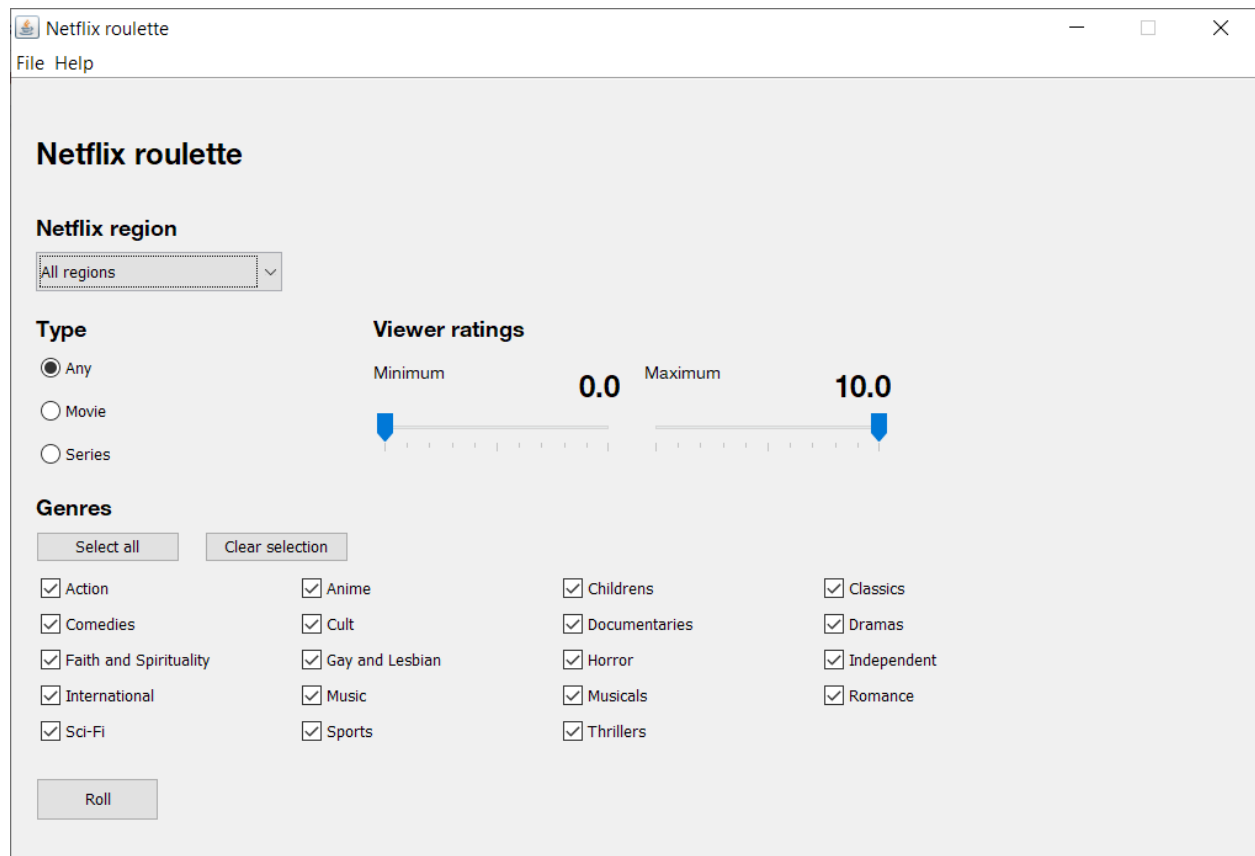
Final design assessment

The application is designed with ease of use in mind. It should be able to serve a random Netflix title to the user with as little interaction as possible.

Even with additional features added, the application is designed in such a way that would still allow any user to request a title with one click.

All of the filters that are available in the main menu will all have a default value upon launch. This eliminates the need for the user to interact with any of those components, and can just dive straight into finding a title.

Other features, such as the help menu and the settings dialog are tucked away in the menu bar, allowing the user to focus on the main part of the application, but can be easily accessed when needed.



Users can also easily customize filters to suit their taste easily. Interactive components on the GUI are designed to minimize the interaction from the user. Checkboxes and radio buttons have been used, so that users can simply click on the type of titles that wish to view. Sliders allows title ratings to be easily selected, without having to type in or spam click to get the desired values.

Other minor supplemental features can be accessed from the menu bar, where they are kept away from the main component of the main menu. Users can open the help dialog, settings, and read the legal mumbo jumbo from the menu bar.



The selected title dialog allows the user to access the displayed title straight from the application, completely eliminating the need for the user to search for the title again on the platform, allowing them to just dive straight in.

A large “Reroll” button on the side allows the user to look for another title with the existing filter that they’ve chosen -- or, they could “Return to Main” to modify the filters.

Overall, this application is designed so that users could simply fire it up, and with a couple of clicks, take the decision-making part away from them, presenting them with a title -- ready to go.

It can be as simple as clicking a button, or as complex as finding titles that fits a defined criteria of genres and ratings. Either way, it is designed so the user can accomplish both without impeding the efficiency or complexity of the other.

User surveys and interviews of the initial wireframe sketches has helped in identifying users’ needs early, which has allowed more effort to be focused on developing solutions to address those needs.

Later on in the development cycle when functional prototypes are created, surveys and interviews are conducted to gather feedback on the user’s experience with the application. In addition to surveys and interviews, select participants are also involved in co-designing sessions.

These are conducted often and throughout the development period as more features are implemented. This is done so that more detailed feedback can be gathered on each

implementation of a given feature, and to avoid wasting time developing features that users may not be satisfied with. Additionally, with most of the users' needs identified along with a developed plan, the feedback gathered from surveys and participatory designs are used to design a solution that delivers a better experience.

Consistently conducting usability tests has ensured that the application is usable and meets the core design goals of the application. This was a crucial part of the development cycle as the feedback loop has continually involved the addition of filters and other features.

As the application grows more complex, it was important to ensure that it was still simple and straightforward to use. The application must be able to generate and display a random Netflix title with minimal interaction for users that do not wish to use the filters. Other features of the application must also be simple to use and meet the users' expectations in order to maintain the overall ease of use of the application.

Further developments

Two of the main areas where this application could be developed is its aesthetics and the accessibility of the application itself.

The platform problem

While the application is designed to be as easy to use (and complex) as possible for the target user, the application itself isn't easily accessible. It is a Java application that requires users to have: A: a Java runtime environment installed on their computers; and B: the appropriate credentials to access the API used in the development of this application.

The concept and the execution of the application itself allows the user to generate a Netflix title easily, choose genres, et cetera, et cetera... but getting the application itself to run on the machine is not as easy.

For the application itself to be accessible to the target audience, it could be adapted into a web application, where users could access the application via a URL; or a mobile application; or even as a chatbot on a popular messaging service.

The aesthetic problem

Well, I mean, this isn't really a problem... as participants in surveys and tests have said. It can, however, definitely be better.

At its current state, the layout of the application appears to be appropriate for its purpose according to the feedback gathered from test users.

To attract more of the target users and retain their engagement, the application could probably use some colors. For example, the main menu could be designed to use Netflix's signature theme color, red, and gradients of those variations -- since gradients seems to be making a comeback nowadays. Fonts like, **Proxima Nova** could be used in the application to make it look more like Netflix's custom font. Using these design combinations would further help remind the user that this application is a Netflix roulette, though a non-affiliation disclaimer will probably have to be a little more prominent than just being in the "About" section, hidden away in the menu.

In the selected title dialogs, the background could utilize an algorithm to analyze the colors in the poster image, and make them blend in with the title to make the colors pop out a little more. Take Spotify's mobile application as an example:



Notice how the background are gradients based on the colors from the artwork.

The lyrics button at the bottom of the screen also uses a color that goes with the background. Notice how this color is also derived from the artwork.

This addition could make the application a little more engaging to use, rather than a project that is waiting to be completed.

The implementation would likely work well on a web application with the use of the HTML5 stack.

