**Testing and Evidence Log**

|  |  |
| --- | --- |
| Before getting into the programming, I had to do some planning and set up some project management tools. Shown to the right is a screenshot of my trello board, which will help me keep track of this program and my progress on it. The planning I have done is in the Planning document. |  |
| Here is the testing plan I set up for the first section of the program that I want to get working. For now, I will just code it so that the program prints to the shell so I know that it is rating. This will not be in the final program, but I need a way to tell the rating of the movie before I set up the search option. | Test plan for setting up the rating frame:   * Press no rating   + The program should print no rating into the shell * Press rating 1   + The program should print rating 1 into the shell * Press rating 2   + The program should print rating 2 into the shell * Press rating 3   + The program should print rating 3 into the shell * Press rating 4   + The program should print rating 4 into the shell * Press rating 5   + The program should print rating 5 into the shell |
| In version one of my program, I set up the variables and the tkinter for just the rating frame. For testing purposes, I coded it so that the rating that the user chooses out of the radio buttons will print to the shell. However, a few problems presented themselves. First, regardless of what button is pressed the program will only output “No Rating”. Secondly, the layout of the radio buttons is not in line. The No Rating button is not aligned with the others, and it is confusing to look at.  I will fix these problems in version two. |  |
| In version two, I got the radio buttons to work as expected, and fixed some issues with the alignment in the layout.  The first problem was that pressing different radio buttons did not change the variable they were assigned to. This was because they were assigned to a regular variable, and not a StringVar. This was simple fix, I changed self.rating to a StringVar and when I printed it in the printrate command, I put .get() at the end of it so it would actually print out the value I wanted.  The second problem was that the layout of the program was confusing. I fixed this by adding padding to the radio button frame and the “Your Rating:” label, and aligning that label to the north and the radio buttons to the west. |  |
| To check that everything was working before I moved onto the next component of the program, I went through the small testing plan I created (two log entries ago). The video for this is in the videos folder, entitled “TESTING 1 – RADIO BUTTONS”.  With the testing done, I had finished the first programming component of the program. I could now get to work on the next one. |  |
| Here is the loose testing plan I set up for the second section of the program. Essentially, it’s the same as with one movie, but here I have listed some other things that I need to be aware of. This is to do with the next and back buttons, and when they should be disabled, and how movie ratings should be able to be kept as the default if they haven’t been rated, and how ratings for each movie should not be forgotten. | Test Plan for Multiple Movies:   * Press next button repeatedly   + Should switch movies until it reaches the last one in the list, where the button will then disable. * Press back button repeatedly   + Should switch movies until it reaches the first one in the list, where the button will then disable * Rate all but one movie   + Should keep the rating for the unrated movie as No Rating, but the others should have ratings * [Note that the ratings for each movie should save – if I rate one movie as three and then rate another movie and return to that first one, it should still be rated as three, with the three radio button selected] |
| In version three, I added some more movies so there were four movies in total to cycle through. I added back and next buttons, though at this point the back button does not do anything. The next button, however, calls a method called “go\_next” when it is pressed. This adds one to the “position” variable, and changes the relevant label and variable so that the user can rate the next movie. It also calls another method called “check\_pos”. This method checks the position variable and, if it is on the last movie in the list, will disable the next button to prevent the user pressing it again and prompting an error message. | Testing video in videos folder as “TESTING 2A – NEXT BUTTON” |
| In version 4, I got the back button to work. The back button calls the “go\_back” method. This functions almost identically the same as “go\_next”, but it puts the position back by one. I added some code to “check\_pos” so that it also checks if it is on the first movie in the list, and then disables the back button to prevent errors. Because the program starts on the first movie in the list, the back button is disabled by default at creation. |  |
| Now the multiple movies component was done, I tested to make sure it was working (using the test plan detailed a few log entries ago). The video for this is in the videos folder, named “TESTING 2B – MULTIPLE MOVIES”. Note that in the shell, each rating when you press a radio button prints out twice because I am changing the rating attribute for the relevant movie classes and want to make sure that it is the same as the rating assigned to the rating variable. After this testing, I was done with the multiple movies component. |  |
| Here is the testing plan for the next component, setting up the search frame. This component will not actually have any functionality for the searching, so for testing purposes the rating that is meant to be searched for will just be printed out into the shell. | Test plan for search frame:   * Select to search for movies with no rating   + Prints out no rating (it cannot actually search yet) * Select to search for movies with a rating of 1   + Prints out 1 in the shell * Select to search for movies with a rating of 2   + Prints out 2 in the shell * Select to search for movies with a rating of 3   + Prints out 3 in the shell * Select to search for movies with a rating of 4   + Prints out 4 in the shell * Select to search for movies with a rating of 5   + Prints out 5 in the shell |
| In version 5 I create the search frame. At the moment this frame can’t actually search for movies, but when you select a rating to search for and press go, it will print out into the shell. This is mostly set-up for when I create the summary frame, and this will prompt an actual search for movies. Unlike the other frames, this is only a local variable because it will be visible on both screens and doesn’t need to be hidden. |  |
| Once I had finished the search frame, I followed the testing plan I made for it. The video for this is in the videos folder, named “TESTING 3 – SEARCH FRAME”. Next, I had to combine the two frames I’ve made into one program. |  |
| In version 6, I combined both the rating frame and the search frame into one program. I put the variables needed for both frames at the top, then all the code for the rating frame GUI, then all the code for the search frame GUI in the init method of the MovieRaterGUI class.  Everything still functioned as before when I moved them into this program, but I had to make a few formatting changes. I changed the background colour of the searching frame to a light grey to differentiate it from the rating frame, and I added some padding – most notably between the two frames – to space things out. |  |
| I tested that the combined frames worked as expected. There was no testing plan needed for this, as it is the same functionality as in previous tests. The video is in the videos folder named “TESTING 4 – COMBINING FRAMES”. Next up was to get the search to actually work. |  |
|  |  |
|  |  |