Logan Monaco

Professor Arias

Software Development 1

<u>Movie Reservation Application Project</u>

<u>Abstract:</u>

The program I have written is an online movie reservation system meant to be used by a small movie store that is looking to provide its customers with a convenient way to reserve a movie before going to the store in person. The system allows users to create an account with the store that will allow them to freely reserve movies under those credentials. The system also has a functionality in which it can tell users who already have a movie rented how much time they have left on the rental. If they happen to be over that time, it will calculate the total cost they will pay upon returning the movie that day. The system offers users the options through a menu and at any point the user can close the application from that menu when they have finished.

<u>Introduction:</u>

I have created this system because I remember the exciting feeling I used to have as a child when my parents would let me go to the local Blockbuster store and rent a movie or game. However, this event could lead to disappointment as movies you intend on renting may not be in stock upon arriving at the store. Being someone who loves movies and spends a lot of his free time renting movies online and watching shows on Netflix, I couldn’t think of a better motivation to write a program than this one. The program is an efficient way of reserving movies to prevent wasted trips to the store, but more importantly, it prevents the store from losing out on potential paying customers since people will not be turned away as often if they use this system. The system also monitors the age of the person reserving the movie and what the total number of copies of a movie are available. This way the system prevents people from reserving movies that are not within their age group as to avoid conflict with state and federal law and it also keeps people from renting movies that aren't represented by a physical copy in the store.

Detailed System Description:

The system begins in the Project2 class and creates multiple data fields for the user's info including username, password, account id, and date created. The first three specifically pertain to the information of the user's account and will be unique for every person who uses the interface. The fourth data field is to create a new date to represent the time at which the account was formed. A scanner is then used to gather the first two items from the users and the account Id is gathered by generating a random number between 0 and 5000 and taking the floor of that number. Once all of these data fields are found they are printed to the user and the user is then directed to the systems main menu through the displayMainMenu() method. This method creates a menu that uses a switch statement to offer the user multiple options including calling the reserveMovie method, calling the checkRental method and quitting the application. If reserveMovie is chosen, the user will be prompted with a question about their age, stored in the variable year which is later used to justify the rental of R rated movies. Before the user goes to rent a movie, the system creates an array which contains integers, each representing a specific movie and the quantity currently found in the store. It will also create variables under the id movie# from 0 to 5 which keep track of the total number of each movie the user has reserved while using the program. The method then prompts the user to select a movie to reserve by choosing a specific number that corresponds to the movie in the array. These numbers are contained in a switch statement which calls different cases which are specific to the movie that it corresponds to in the array. For all movies that are not rated R, the cases first do a check on whether the spot in the array has a value over 0, which would indicate there is a copy available. If it passes this, it will ask the user whether they would like to reserve a copy of that movie by choosing 1 for yes and 0 for no. There is then an if statement checking if they chose 1 and if they did, it will subtract 1 from the spot in the array and add 1 to the corresponding movie# variable to be printed once the user is finished. Finally, it will send the user back to the reserveMovie method until they choose to terminate the method and return to the main menu. If the system doesn't find a value above 0 in the array during the first check, it will kick out a message saying there are no copies available and send the user back to reserveMovie. Likewise, if the user chooses not to reserve the movie they select, the system will say they didn't choose a movie and will return them to the start of reserveMovie. The main difference between an R rated movie is the check during the second if statement which also checks to see if the user is 18 years or older. If they are not, it will not satisfy the condition and the response will say that either they didn't choose a movie, or they were too young to do so. This switch goes up till 6 which is not a movie but instead a way of exiting the method. Not only does it exit the method, it also takes the variable of type movie# and returns those values to show the user how much of each movie they have reserved while using the method. Once showing the user this, it will call the displayMainMenu method which allows the users the option to quit or call checkRental. If checkRental is selected, an integer named days will be initialized at 0 and the user will be prompted to enter the number of days they have had their movie rented for. If the number is below 30, indicating that the user has had it for less than 30 days, which is the stores policy, then it will simply take 30 and subtract the variable days from it and return that value. If the value happens to be above 30, the program will move to the next else if statement which checks for this. If it finds it to be true, it will create a daysOverThirty variable and an overdueFee variable to be used in a formula. Then there is another check to see if the variable daysOverThirty is 10 or below. This check is done because the growth of the overdueFee stops once ten days late is hit and stays at 50 dollars from that point on. This is reflected in the else statement to this if statement which will tell the user that the rental is over 10 days late and will be 50 dollars from that point on. However, if the daysOverThirty variable is 10 or less, it will calculate the overdueFee based on the formula 4n + 10 assuming after 30 days late there will be an initial 10-dollar fee. It will then print out the variable overdueFee to the user, so they know how much they can expect to pay when they get to the store. If both checks on the number days fails, then the system will assume the user entered something invalid that was most likely not a number. For this, there is an else statement which kicks out this error message and returns the user to the start of checkRental. Once the user has gotten the info about their rental, all of the options end with displayMainMenu() to bring them back to where they can choose more options. This covers all the intricacies within this application. For the most part, this application works in a way that all the calculations/method work in the rentMovie class while the Project2 class takes care of the initial interaction with the user and the setup of their account.

Requirements:

The problems this system is addressing tend to be convenience rather than need. The problems it solves are also beneficial to the store and it total profits. It resolves the issue of needing to get to the store super early for a new release since users can simply reserve the movie on their account without having to worry about going to the store and finding out that no more copies are available. Another problem this solves is if children want to reserve a movie but can go to the store themselves, they cant do it online and when their parents are available they can have them drive them there without wasting any time or gas money. As for the checkRental feature, customers of the store will be able to keep track of their remaining rental time so they don't forget and even if they do, they can refer to the program to find out how much they can expect to pay upon getting to the store. This will solve the problem of customers complaining that they weren't aware of the stores rental policy or how much they would have to pay and if its even an accurate amount of money. Finally, it solves something that's less a problem but more of a competing with the competition kind of thing, that being the ability to have customers keep an account with the store online. Being able to keep track of how many customers you have and what they are doing is invaluable to any company.

Literature Survey:

Many more advanced version of this software have been made and paired with already existing databases to create movie reservation systems, usually for large movie theatre companies. Since things like Blockbuster are a thing of the past, most people are more concerned with reserving tickets and renting things on the spot on their computer. Most programs out there from what I could see have features like sales done right in the software, email notifications sent after certain conditions are met, and auto generated tickets printed at the end of the program. One specific version I found was done by PHP Jabbers that is available for purchase if any company wishes to utilize the features it offers.

User Manual:

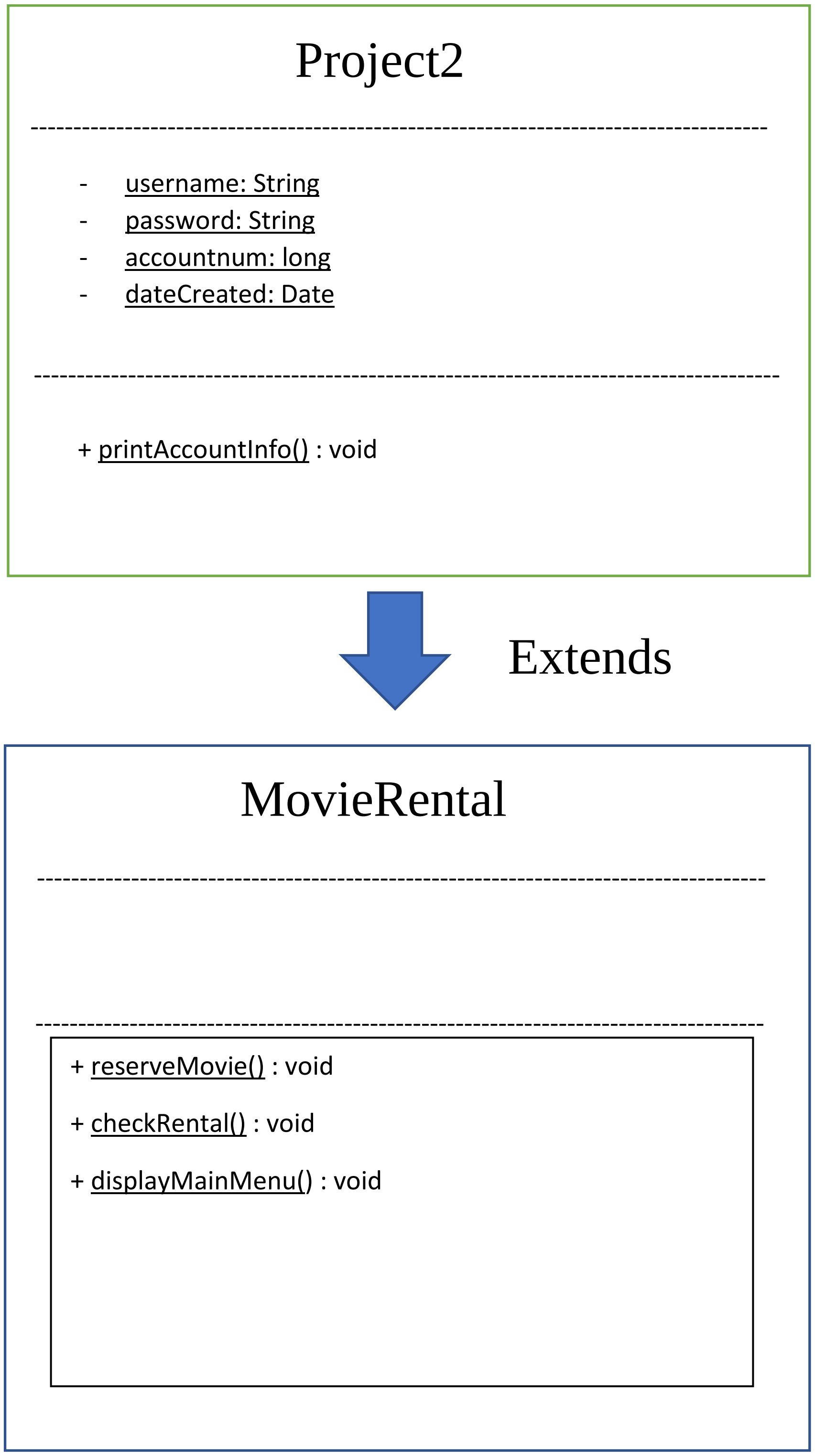
To use the system, simply put in your desired username and password and the system will tell you what your account number, username, and password are as well as the current date. You will then be prompted with a menu that offers you three choices, reserve a movie, check a rental, or quit. If you choose reserve movie, the system will have you input your birth year and then ask you to choose the movie you wish to reserve. After choosing it you will find out if there are copies available. If there are copies, you can then select if you want to reserve the movie or not which will place 1 copy under your account and send you back to the beginning. Otherwise, the system will either tell you that there are no copies or you didn't select yes. In the case of an R rated movie, you will not be able to rent the movie if you are under 18 years old. If you chose to check on a rental at the main menu, the system will bring you to a screen that asks you how many days you have had your rented movie out for. If its under 30 days, you will find out how many days you have left. If its over 30, you will be told how much money you will need to pay the store to return the movie. Completing this transaction will send you to the main menu again. Finally, choosing quit will exit the program and all your transactions will be done on your account.

<u>Conclusion:</u>

The system accomplishes the goal of making the transactions done at the movie rental store smoother and more efficient than they would be without this system in place. It puts customers at ease knowing that the store has movies put on hold specifically for them protected under their own account info that only they know. It also prevents any arguments or legal problems from occurring as customers who have rented movies will be well aware of what the status of their rental is and how much they can expect to hypothetically pay if they don't bring the movie back within the time constraints.

<u>References/Bibliography:</u>

I did not consult many documents or source code while doing this project but one thing I did find myself using a lot as a reference was a problem from one of our labs which involved making an account and doing ATM transactions like making a savings account and checking account class as well as depositing and withdrawing cash from each.


Project2
- username: String
- password: String
- accountnum: long
- dateCreated: Date
+ printAccountInfo() : void
Extends
MovieRental
+ reserveMovie() : void
+ checkRental() : void
+ displayMainMenu() : void