Criterion E - Evaluation

Success Criteria	Was it Met?	Comments based on client feedback
The client will be able to access three different main options. One for adding spots, another for renting out spots, and another one for viewing the spots in his files.	Met	All three functions specified by my client were successfully incorporated, whereby using a 'while(true)' loop he is able to choose from any of the options listed as many times as he wished to do so.
The program has a menu, which allows the users to choose the options and exit whenever he wants.	Met	The main menu is coherently and accurately displayed, allowing my user to constantly choose any option and function he wants. The feedback given by my client regarding the Main Menu function was very positive (see appendix 9)
When adding a spot, the program will ask for the city of the spot and the available days for rental (weekday or weekends) to store each address in its specific place.	Met	Every time my user chooses the "Add Spot" function, the program asks for the city (Bogota or Cali) as well as the available day for rental (weekday or weekends). The address will be stored inside a specific ArrayList that meets both conditions (city and day).
Every time a spot is added, the list storing the addresses will be increased and the new address will be stored. On the other hand, whenever a spot is rented, the address will disappear from the file, for my client to keep track and know exactly which spots are currently available and dismiss those which have been rented out.	Met	Every time an address was added to the business, it was allocated to the specific ArrayList and file (depending on the conditions) and every time a parking space is rented out, the address will be deleted from the ArrayList and file, which stored this specific parking space.
When renting out a spot, the program will ask for the city where the client needs to park and the day (weekday or weekend) to show the available spots considering those specific conditions (city and day).	Met	When my client chooses the "Renting Out Spot" function, the program asks for the city (Bogota or Cali) and the day for rental (weekday or weekends). The program will only display the addresses contained specifically at the ArrayList which keeps track of these conditions.
When renting out a spot, the program will ask the number of	Met	The program asks for the day (weekday or weekend) and the number of days the parking space is going to

days, in order to calculate the prices which should be charged to the customer. (As suggested by the client, (see appendix 1)		be rented and consequently calculates the price.
The program will run in an infinite loop (While(true)) and therefore will allow the users to perform multiple tasks without ending or closing the program. Whenever the user wants to exit, an option will be provided and the program will break/terminate	Met	The program runs infinitely, allowing my client to perform as many tasks as he wished. Whenever he needs to end the program, the “Exit” option created can be used.
The code meets the coding complexity requirements (see appendix 2), to ensure a sufficient level of detail, functionality, and complexity, for the client to have a quality end product that meets his requirements and solves his problem.	Met	The code meets the requirements listed in appendix 2, allowing the program to be complex and detailed enough, in order to provide a functional and professional experience to my client.
The code uses imported libraries from java that can enrich the user experience. This might include Array List, File Output Stream, and Scanners.	Met	Libraries such as Scanner, Arraylist, and file input/output were used in the code, in order to enrich my program and add functionalities.
The code uses java serialization, in order to store each Arrallist containing the addresses on organized files, keep them updated and use them every time the program starts. This will allow my client to close the program at the end of every workday, and the next day when he starts the program he will have all of the addresses updated and organized thanks to the files which are keeping track of the data. Serialization will also help my user keep its data safe in case the program crashes, the data will still be kept safe and updated within the files created.	Met	The code uses java serialization, enabling my client to have files that organize and store all of the addresses in separate places. This feature allows my client to terminate the program and still have all the data stored, organized, and ready to use whenever he runs the program again. As my client stated, this was an essential and very practical feature to have in the program. (see appendix 9)

Further Improvements:

In my final meeting with my client (Appendix 9), we came up with ideas and suggestions for future improvements to the program.
The main additions or improvements suggested are the following:

- ***Graphical User Interface:*** With my client, we both agree that a big area of improvement for this program is related to the Graphical User Interface and the User Experience. By coding some GUIs, adding a text box, buttons, and visual elements, the program will become more user-friendly, providing a better experience, and making it easier to use by my client. As well, if we are able to add GUIs to this program, we can eventually convert this program into an App, enabling my client to easily use his phone in order to perform any task he needs related to his parking spot business.
- ***Password and Security Checks:*** Taking into account my client is saving different types of data, which includes; addresses, names, phone numbers, etc, a future improvement could be related to a password feature that keeps all of the information secure and out of reach to anyone who accesses the program. We could ultimately create specific user accounts, which securely store the data and can only be accessed by a personal and secret password created by the user.
- ***DataBases:*** For further improvements, my client suggested that we can evolve the serialization feature and make a DataBase. This will allow us to perform a wider variety of tasks and features. For example, we will be able to store each name, phone number, and email with the according parking spot address, hence creating a more complete and useful Database of clients. As well, with a Database, my client will be able to access all the information in any device he needs, rather than being limited to only using the device which has the serialization files in it.

Future extensions and functionalities:

- For future extensions and further functionalities, we could eventually expand this program to a marketplace platform, where users can either publish their parking spots and on the other hand, other users can easily access and rent a parking spot they need. It will become an intermediary platform, which connects users who are willing to rent their parking spots in exchange for money, with users who are looking for a parking spot in a certain city. In order to do this, I will potentially need to create a ‘MySQL’ database, which stores and constantly updates the user information, in order to properly and organize each parking spot availability. As well, using java importing libraries such as ‘javax.email’ and ‘javax.mail.internet’, I could eventually create a function that sends an email to a user every time someone is interested in renting their parking spot. Enabling them to get in touch, arrange days, payments, and any future details. I will basically be using the inspiration from the ‘Airbnb’ model, creating a simple but very functional platform, which allows users to rent and look for parking spots.