Assessment 1

Name: Sahand Sabour

Student ID Number: 1614650

E-mail: [Sahand.sabour16@student.xjtlu.edu.cn](mailto:Sahand.sabour16@student.xjtlu.edu.cn)

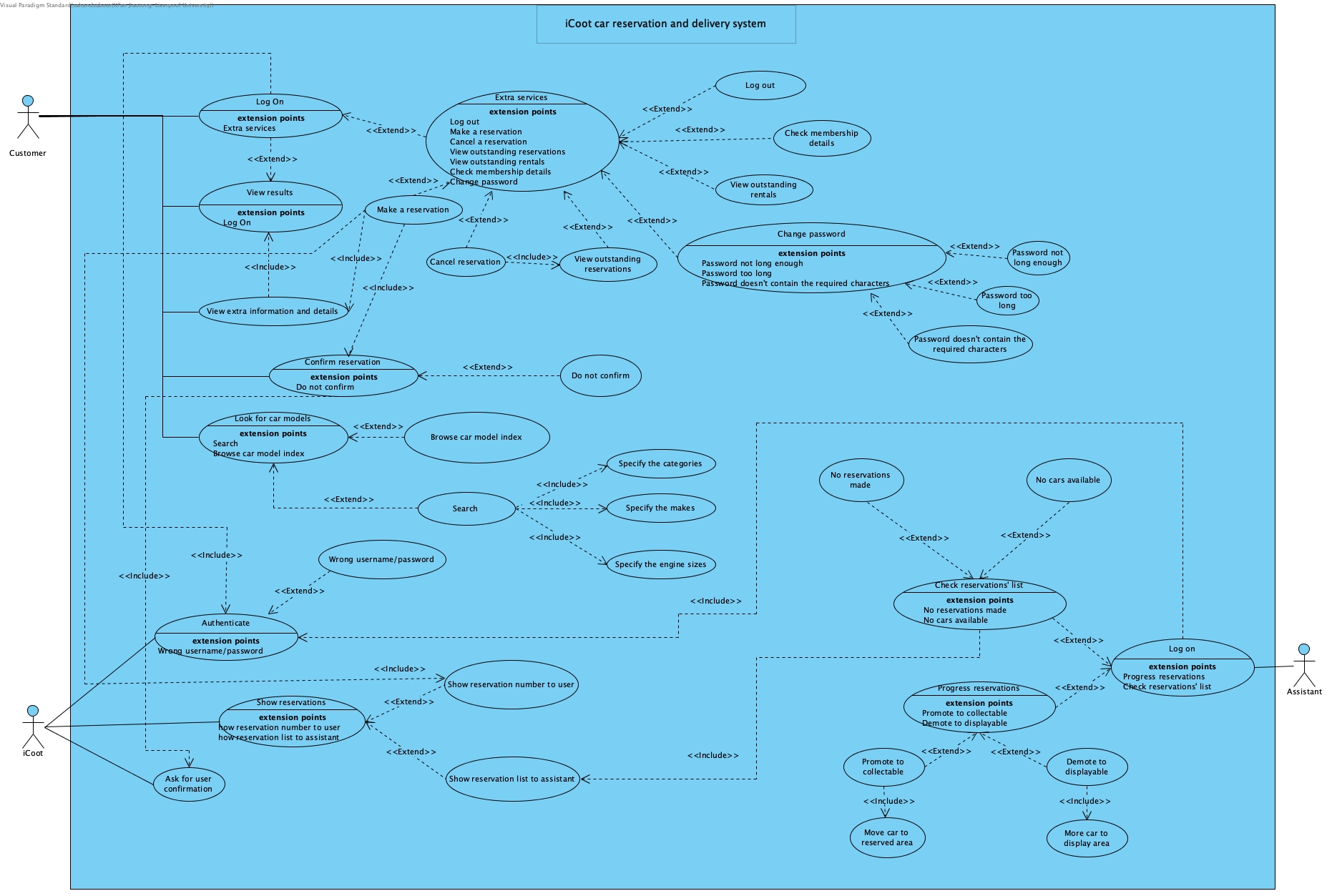
Computer Science and Technology (CST)

Software Engineering I – CSE207

Lecturer: Kaiyu Wan

Submission deadline: THU 2018/10/25 11PM

a) UML use-case diagram for the Coot software:



UML use-case description for the Coot software:

|  |  |  |
| --- | --- | --- |
| iCoot system: look for car models | |  |
| Actors | Customer | |
| Description | A customer should be allowed to either browse the CarModel index or search the database to look for their intended inquiry. In the latter case, the customer is required to specify a category, makes and engine sizes. The costumer would then be shown a collection of matching car models along with basic information regarding that car such as car model name. The user should also be able to view extra information such as description and advert if they choose to do so. | |
| Data | Either the CarModel index or specified categories, makes and engine sizes, which the user inputs. | |
| Stimulus | User command issued by the customer who is looking/searching for a car. | |
| Response | A list of cars with detailed information according to the inquiry. | |
| Comments | The customer needn’t be a member to access this functionality. | |

|  |  |
| --- | --- |
| iCoot system: Log on (as a customer) | |
| Actors | Customer |
| Description | A customer that has become a member should be able to log on and gain access to the extra features that the platform provides. After the user logs on, the input information is authenticated by the system, and the customer can continue as a member if the authentication is successful. |
| Data | Customer’s log-in information. |
| Stimulus | User command issued by the customer who wishes to log on as a member and access the extra services. |
| Response | Extra functionalities are provided for the customer to choose from. |
| Comments | The customer must have become a member prior to this stage to be granted the permission of logging on to the system. |

|  |  |
| --- | --- |
| iCoot system: Extra services | |
| Actors | Customer (who has logged on as a member) |
| Description | A member should be allowed to access a number of extra features including making a reservation, cancelling a reservation, etc. The member should also be able to change their passwords according to their desire and also be able to log out of the system once their work with the system has been completed. |
| Data | The data can vary based on the function that is chosen by the member. The detailed data used for functions excluding changing password and logging out is mentioned respectively below. As for the mentioned two, the old and a new password is considered as the required data for the prior and for the later, no data is required. |
| Stimulus | User command issued by the member who wishes to use extra services. |
| Response | The response corresponds to each of the chosen functions and therefore, it would be discussed in more detail in the later use cases. The response for the excluded function shall either be a change in personal information and/or a message displayed to the user. |
| Comments | Some security measures should be predicted for the password input. |

|  |  |
| --- | --- |
| iCoot system: Make a reservation | |
| Actors | Customer (who has logged on as a member) |
| Description | A member should be allowed to make a reservation on a desired vehicle. It must be checked that the user is viewing the details regarding their chosen car in order to make a reservation. If a car is moved to the reserved for the user, the user is expected to receive the vehicle within a specified period of time. Failure to collect the reserved car would result in a fine. |
| Data | Payment methods and information may be included in this section. |
| Stimulus | User command issued by the member who needs a car and wishes to make a reservation. |
| Response | A message asking for user confirmation and another showing the reservation number and an indication that an assistant would be in touch when a car is available. Both of the mentioned messages would prompt on the screen. |

|  |  |
| --- | --- |
| iCoot system: Cancel a reservation | |
| Actors | Customer (who has logged on as a member) |
| Description | The member should be able to cancel the already made reservations. The user must be viewing their outstanding reservations in order to proceed with their cancellation. After the cancellation request is issued, the user may be charged with a cancellation fine. |
| Data | The reservation number. |
| Stimulus | User command issued by the member who, for an unknown personal/general reason, wishes to cancel their outstanding reservation. |
| Response | Removal of the mentioned reservation from the reservations’ list. |
| Comments | The member may be fined depending on the time remaining to collection or the time passed from the reservation. |

|  |  |
| --- | --- |
| iCoot system: Checking membership details | |
| Actors | Customer (who has logged on as a member) |
| Description | A member should be able to check (and change) their membership details such as username, email, subscription start and end date, etc. |
| Data | User information (if change of information is allowed) |
| Stimulus | User command issued by the member who wishes to view their membership details. |
| Response | List of user information. |

|  |  |
| --- | --- |
| iCoot system: View outstanding reservations | |
| Actors | Customer (who has logged on as a member) |
| Description | The system should allow members to view their outstanding reservations. This allows members to check their reservations and make cancellations when needed. |
| Data | No data transfer. |
| Stimulus | User command issued by the member who wishes to view the reservations they have already made. |
| Response | List of outstanding reservations. |

|  |  |
| --- | --- |
| iCoot system: View outstanding rentals | |
| Actors | Customer (who has logged on as a member) |
| Description | The customer should be allowed to view their outstanding rentals as another feature of the extra services. In this section, the user would be provided the list of their previous successful reservations that lead to car rentals and may choose to use that information for a new reservation. |
| Data | No data entry. |
| Stimulus | User command issued by the member who wishes to view their previously rented car history (outstanding rental history). |
| Response | List of outstanding rentals. |
| Comments | There was no further description of this function in the provided problem description and the mentioned function was assumed to serve as a record of previous uses of the system accordingly. |

|  |  |
| --- | --- |
| iCoot system: Log on (as an assistant) | |
| Actors | Assistant |
| Description | The assistants should be able to log on to the system and use the system in order to be involved in the life cycle of reservations, which includes moving cars to and from reservation area alongside other functionalities. |
| Data | Assistant’s log-on information |
| Stimulus | An assistant who wants to do their job. |
| Response | The features that are specified for the assistants’ use. |
| Comments | The assistants must be given the specified authority beforehand in order to be able to log on to the system. |

|  |  |
| --- | --- |
| iCoot system: Check reservations’ list | |
| Actors | Assistant |
| Description | The system should provide each assistant with a list of the members’ reservations after they have logged on to the system. The assistants would then use this list to look for a suitable car. |
| Data | No data entered. |
| Stimulus | User command issued by the assistant who wishes to check the reservations in order to potentially progress members’ reservations. |
| Response | List of reservations. |
| Comments | No comment. |

|  |  |
| --- | --- |
| iCoot system: Progress reservations | |
| Actors | Assistant |
| Description | The assistant must be able to both demote a car to displayable and promote it to collectible via the system. The assistant makes these decisions based on the conclusions made from viewing the members’ reservations. If the assistant chooses to promote a car to collectible, they also have to move the car to the reserved area. If the assistant chooses to demote a car to displayable, they would have to move the car to the displayable area. |
| Data | Choosing between the mentioned demotion or promotion |
| Stimulus | User command issued by the assistant who wishes to do their responsibility and progress members’ reservations. |
| Response | Change in both the cars’ and their reservations’ status. |
| Comments | No comment. |

|  |  |
| --- | --- |
| iCoot system: Show reservations | |
| Actors | iCoot |
| Description | The iCoot system must be able to show reservations’ information to both the members and the assistants. |
| Data | No data entered. |
| Stimulus | Automated system command. |
| Response | List of reservations organized based on type of the user. |
| Comments | No comment. |

|  |  |
| --- | --- |
| iCoot system: Authenticate | |
| Actors | iCoot |
| Description | The iCoot system must be able to authenticate user information when members and/or assistants are logging on to grant them access to their control panels. |
| Data | No data entered. |
| Stimulus | Automated system command. |
| Response | List of reservations organized based on type of the user. |
| Comments | No comment. |

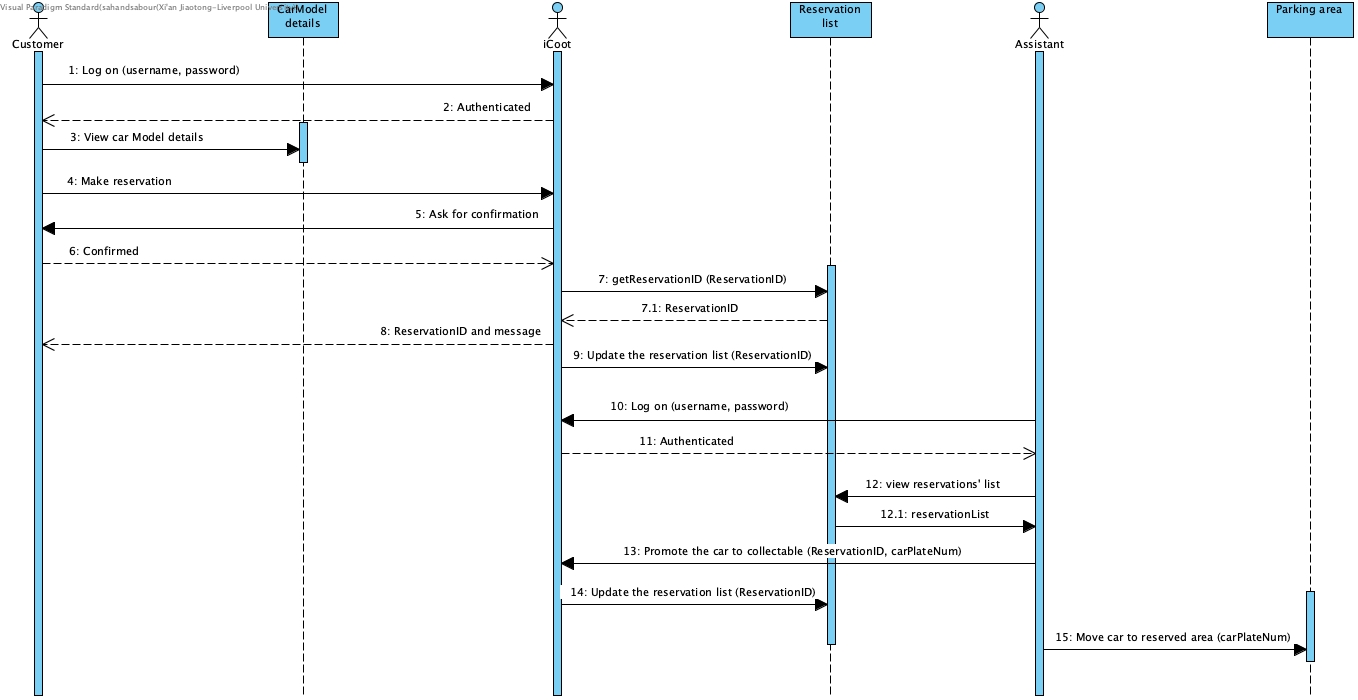
b) The main scenario for the “Make Reservation” use case:

|  |  |
| --- | --- |
| Use-case name | Make reservation |
| Participating actors | Customer |
| Flow of events | 1. Customer enters the personal information and tries to log on to the system. 2. The system (iCoot) authenticates and with successful authentication, the customer would be considered a member. 3. The member views the details of a CarModel. 4. The member chooses to make a reservation. 5. The system (iCoot) displays a message that asks for member’s confirmation and alerts the member that a failure to collect a reserved CarModel would result in a fine. 6. Member chooses to continue and confirms the displayed message. 7. The system (iCoot) processes the reservation and saves it in the system.    1. The system (iCoot) gets the reservation number from the database once it is saved. 8. The system (iCoot) displays the reservation number and a message indicating that an assistant will be in touch when a car is available. 9. The system (iCoot) updates the reservations’ list. 10. Assistant logs on to the system. 11. The system (iCoot) authenticates. 12. The assistant asks for the list of reservations.     1. The assistant receives a list of reservations. 13. The assistant chooses to promote the chosen car to collectable since that car is available. 14. The system updates the reservations’ list. 15. The assistant moves the collectable car to the reserved area. |
| Entry condition | A customer who would like to reserve a car. |
| Exit condition | A message showing the reservation has been made in addition to a car moved to the reserved area and prepared for pick-up. |
| Quality requirements | - The user authentication time should take no more than 5 seconds. In addition, a loading screen must be displayed meanwhile to indicate that the process is ongoing.  - There must be an assistant logging onto the system for at most every half an hour to progress incoming reservations if any reservations were made. This is assumed to be sufficient waiting time if having assistants online for 24 hours per day is not a possible option for the system implementation.  - The response time for each query in the system (such as getting reservations’ list, making reservations, etc.) should take no longer than 10 seconds, depending on the user’s internet connection bandwidth. As mentioned before, a loading bar should also be added to these sections to indicate how much progress the ongoing process has made thus far.  - The details regarding each car model should include detailed information regarding the vehicle’s specs and features. The mentioned information should also be accompanied with several recently taken photos of the chosen car.  - The database used for saving the reservations should be well-organized. In addition, the names for columns should be carefully chosen to prevent any further misunderstandings. |

The exceptional scenario for the “Make Reservation” use case:

|  |  |
| --- | --- |
| Use-case name | Make reservation |
| Participating actors | Customer (who has logged on as a member) |
| Flow of events | **Exceptional scenario 1:**   1. Customer enters the personal information and tries to log on to the system. 2. The system (iCoot) displays an error for incorrect log-in information (incorrect username/password).   **Exceptional scenario 2:**   1. Customer enters the personal information and tries to log on to the system. 2. The system (iCoot) authenticates and with successful authentication, the customer would be considered a member. 3. The member views the details of a CarModel. 4. The member chooses to make a reservation. 5. The system (iCoot) displays a message that asks for member’s confirmation and alerts the member that a failure to collect a reserved CarModel would result in a fine. 6. The member chooses not to continue and doesn’t confirm.   **Exceptional scenario 3:**   1. Customer enters the personal information and tries to log on to the system. 2. The system (iCoot) authenticates and with successful authentication, the customer would be considered a member. 3. The member views the details of a CarModel. 4. The member chooses to make a reservation. 5. The system (iCoot) displays a message that asks for member’s confirmation and alerts the member that a failure to collect a reserved CarModel would result in a fine. 6. Member chooses to continue and confirms the displayed message. 7. The system (iCoot) processes the reservation and saves it in the system.    1. The system (iCoot) gets the reservation number from the database once it is saved. 8. The system (iCoot) displays the reservation number and a message indicating that an assistant will be in touch when a car is available. 9. The system (iCoot) updates the reservations’ list. 10. Assistant logs on to the system. 11. The system (iCoot) authentication fails.   **Exceptional scenario 4:**   1. Customer enters the personal information and tries to log on to the system. 2. The system (iCoot) authenticates and with successful authentication, the customer would be considered a member. 3. The member views the details of a CarModel. 4. The member chooses to make a reservation. 5. The system (iCoot) displays a message that asks for member’s confirmation and alerts the member that a failure to collect a reserved CarModel would result in a fine. 6. Member chooses to continue and confirms the displayed message. 7. The system (iCoot) processes the reservation and saves it in the system.    1. The system (iCoot) gets the reservation number from the database once it is saved. 8. The system (iCoot) displays the reservation number and a message indicating that an assistant will be in touch when a car is available. 9. The system (iCoot) updates the reservations’ list. 10. Assistant logs on to the system. 11. The system (iCoot) authenticates. 12. The assistant gets the list of reservations.     1. There are no reservations (empty reservations’ list). |
| Entry condition | A customer who would like to reserve a car. |
| Exit condition | Message showing false log-in information or no available cars. |
| Quality requirements | - The response time for the false information errors should take no longer than 2 seconds.  - If there are no available cars, the query to get the list of available cars should take less than a second (just return false rather than an empty list). |

c) The UML sequence diagram for the major (main) scenario:



The UML sequence diagram for the exceptional scenarios:

