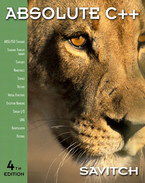
** COSC 1320**

**Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Seat # \_\_**

**Estimated Hours 1**

**Actual Hours**

**TA (Textual Analysis) for UML USE CASE Diagram MODEL**

**Programming Assignment 4:**

**(10 points)**

**(Due date found in the COSC 1320 BB Calendar!)**

**You must use Microsoft WORD program. Insert the TEMPLATEs for Actors and Use Cases in this Requirements Document.**

*Please have Step 1, 2, 3, and 4 in this order.*

PLEASE use the “TA for UML USE CASE Diagram MODEL Movie Company System.doc” AS TEMPLATE.

(Do not Show STEP 1,…; do not change Line Numbers; do not change Number of Pages)

Any DIAGRAM that is NOT the result of CUT and PASTE

WILL BE IGNORED. (YOU WILL GET ZERO POINTS)

**Requirements Analysis**: Construct the **UML USE CASE** Diagram MODEL

ProgrammingAssignment4 **software product** **for a Video Store**

For a family or an individual, a favorite place to go on weekends or holidays is to a video store to rent movies. A new video store in your neighborhood is about to open. However, it does not have a program to keep track of its videos and customers. The store managers want someone to write a program for their system so that the video store can function. The program should be able to perform the following operations:

UC1: DisplayInventory

1. **Display** a list of **all** of the videos in the store (full information).

2. **Search** a video **- by title** and show the details of **a** particular video (title, stars (up to two!), producer, director, productioncompany, copiesinstock).

UC2: SearchInventory

3. **Rent** a video.

UC4: DisplayCustomers

UC3: RentVideo

4. **Display** a list of **all** of the store’s customers (full information).

5. **Search** a customer by name and display the details of **a** particular customer **– by last name** (firstname, lastname, id).

UC5: DisplayCustomerInfo

6. **Display** a list of all of the videos rented by **a** particular customer **– by last name** (firstname, lastname, title, title).

UC6: DisplayCustomerRecord

ProgrammingAssignment4 should illustrate the **O**bject **O**riented **P**aradigm methodology and, in particular, **inheritance** and **overloading**.

There is a need to maintain the following lists (**cannot use STL Classes**):

 A list of all of the videos in the store

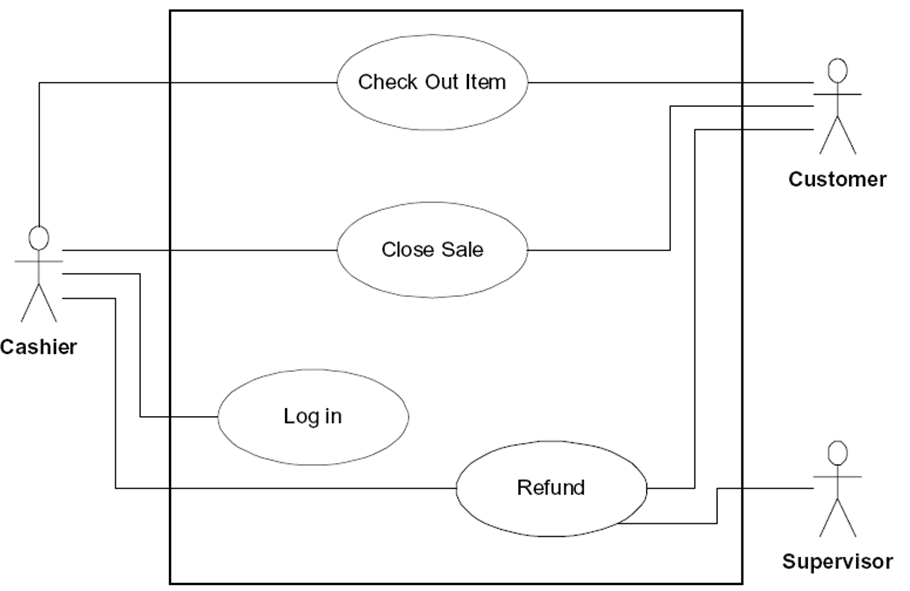
 A list of all of the store’s customers

 Lists of the videos currently rented by the customers

UC7: ReadFile

Provide **member functions** to load the video store collection, customer information & customer rental information from the file “**Programming Assignment 4 Data.txt**” (FILE INPUT) into your **ProgrammingAssignment4** **APPLICATION** memory (**please DO NOT read from File and write to File skipping reading into the program memory**) and write it back out to the **SAME** data file (FILE OUTPUT) when exiting from **ProgrammingAssignment4** **APPLICATION** **“Programming Assignment 4 Data.txt”**.

UC8: WriteFile



**A1: cashier**

The data in the input file **“Programming Assignment 4 Data.txt”** is in the following format:

**number of videos**

video title ( that is, the name of the movie)

movie star1

movie star2

movie producer

movie director

movie production co.

number of copies

. . .

**Customer**’s first name**,**last name**,**and id

. . .

**Customer** id**,**video title**,**video title

A sample input file “**Programming Assignment 4 Data.txt**” follows:

**3**

Titanic

Kate Winslet

Leonardo DiCaprio

Cameron

Cameron

20th Century Fox

**2**

Jerry MacGuire

Tom Cruise

Renne Zellwegger

Brooks

Crowe

United Artists

**18**

Rain Man

Dustin Hoffman

Tom Cruise

Johnson

Levinson

United Artists

**2**

**Donald,Duck,1**

**Mickey,Mouse,2**

**Minnie,Mouse,3**

**Goofy,Dog,4**

**4,Rain Man,Titanic**

**1,Jerry MacGuire**

Create a **ProgrammingAssignment4.cpp** that contains the **main ()**.

**Class**es must start with UpperCase Letter.

**Class**es must have **.h** and **.cpp**.

One **Class** per File Name that matches the **Class** Name.

**You cannot use any DATA STRUCTURES STL Classes! Use arrays.**

**One Class per .h &.cpp file** (you will lose 15 points if not)**!**

**Must use Inheritance** (you will lose 20 points if not)**!**

**Each Class MUST HAVE a toString() member function**(you will lose 20 points if not)**!**

**Is toString() member function used** to display customers, videos, or rented videos by customer (you will lose 10 points if not)**!**

Each .h & .cpp file must have these 4 lines of comments (you will lose 5 points if not):

// COSC 1320 Summer 2015

// Name:

// Programming Assignment 4

// This is my own work; I will not post

DisplayInventory

SearchInventory

RentVideo

DisplayCustomers

DisplayCustomerInfo

DisplayCustomerRecord

**Controller**

**UC 1**

**UC2**

**UC3**

**UC4**

**UC5**

**UC6**

**UC7**

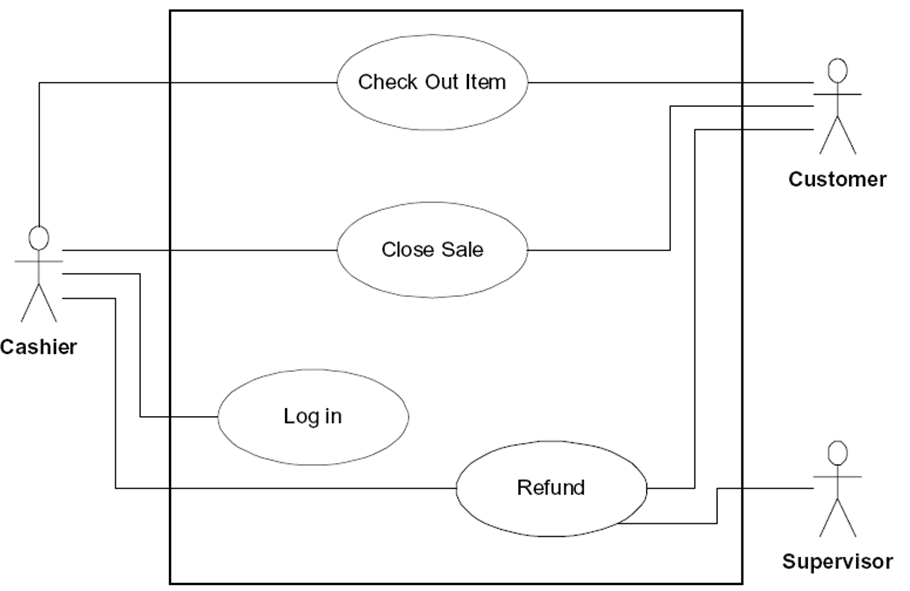
**UC8**

ReadFile

WriteFile

UC1 DisplayInventory Description

A1: **Cashier**



|  |  |
| --- | --- |
| Name: | DisplayInventory |
| Actor: | Cashier |
| Description: | This use case describes the process used by Cashier to display the current inventory |
| Successful Completion: | Cashier requests to display the inventory   1. Controller loops through inventory and displays the contents. |
| Alternative: |  |
| Pre-Condition: | Cashier requests to display the inventory |
| Post-Condition: | Inventory is displayed |
| Assumptions: | None |

UC2 SearchInventory Description

|  |  |
| --- | --- |
| Name: | SearchInventory |
| Actor: | Cashier |
| Description: | This use case describes the process used by Cashier to search the inventory for a particular title |
| Successful Completion: | Cashier requests to search the inventory for a particular title   1. Controller loops through inventory and displays a match with a provided string 2. Asks Cashier if they want to rent the video |
| Alternative: | Lets Cashier know that the movie is not listed |
| Pre-Condition: | Cashier requests to search the inventory for a movie |
| Post-Condition: | Movie info is displayed or notice is displayed |
| Assumptions: | None |

UC3 RentVideo Description

|  |  |
| --- | --- |
| Name: | RentVideo |
| Actor: | Cashier |
| Description: | This use case describes the process used by Cashier to rent a video to a Customer |
| Successful Completion: | Cashier requests to rent a video to a Customer   1. Controller prompts for Customer ID 2. Controller loops through custArr for object that matches ID 3. Controller calls addRental() on matched object |
| Alternative: | Cashier requests Adding Disc to artistName   1. Controller prompts for Customer ID 2. Controller loops through custArr for object that matches ID 3. Controller calls addRental() on matched object 4. Customer cannot rent any more movies |
| Pre-Condition: | Cashier requests to rent video to Customer |
| Post-Condition: | Video is rented to customer or not |
| Assumptions: | None |

UC4 DisplayCustomers Description

|  |  |
| --- | --- |
| Name: | DisplayCustomers |
| Actor: | Cashier |
| Description: | This use case describes the process used by Cashier to list all the customers. |
| Successful Completion: | Cashier requests to list all the customers   1. Controller displays all the Discs, discCatalogNumber, title, artistName |
| Alternative: |  |
| Pre-Condition: | Cashier requests to list all the customers |
| Post-Condition: | Customers are displayed |
| Assumptions: | None |

UC5 DisplayCustomerInfo Description

|  |  |
| --- | --- |
| Name: | DisplayCustomerInfo |
| Actor: | Cashier |
| Description: | This use case describes the process used by Cashier to display the full information of a particular customer |
| Successful Completion: | Cashier requests customer information by last name   1. VideoStore prompts for a last name 2. Displays information of customers that have that last name |
| Alternative: | Cashier requests customer information by last name   1. VideoStore prompts for a last name 2. No match is found and the Cashier is alerted. |
| Pre-Condition: | Cashier requests customer information |
| Post-Condition: | Customer information is displayed. |
| Assumptions: | None |

UC6 DisplayCustomerRecord Description

|  |  |
| --- | --- |
| Name: | DisplayCustomerRecord |
| Actor: | Cashier |
| Description: | This use case describes the process used by Cashier to display all current rentals by a customer |
| Successful Completion: | Cashier requests a customer’s rentals   1. Controller prompts Cashier for an ID number 2. Loops through custArr and displays rentals from Customer with provided ID |
| Alternative: |  |
| Pre-Condition: | Cashier requests to display rentals by ID |
| Post-Condition: | Customer’s rentals are displayed |
| Assumptions: | None |

UC7 ReadFile Description

|  |  |
| --- | --- |
| Name: | ReadFile |
| Actor: | none |
| Description: | This use case is called automatically upon startup |
| Successful Completion: | Program starts   1. “Programming Assignment 4 Data.txt” is read |
| Alternative: | If file does not exist, program exits |
| Pre-Condition: | Cashier requests List Tracks for Disc |
| Post-Condition: | Inventory and user info is stored within the program |
| Assumptions: | None |

UC8 WriteFile Description

|  |  |
| --- | --- |
| Name: | WriteFile |
| Actor: | Cashier |
| Description: | This use case describes the process used by Cashier to write data to a file and exit the program |
| Successful Completion: | Cashier requests to a file and exit the program   1. Controller writes data to “Programming Assignment 4 Data.txt” |
| Alternative: |  |
| Pre-Condition: | Cashier requests to a file and exit the program |
| Post-Condition: | Tracks for Disc displayed |
| Assumptions: | None |