**COSC 4351**

**Name: \_\_\_\_\_\_\_\_Saim Ali\_\_\_\_\_\_\_\_\_\_\_\_**

**UML Modeling – WHAT Practice**

Apply the example from UML Modeling TUTORIAL to these requirements. Please complete as much as possible. Consider it the first iteration.

You must use Microsoft Word Case Tool.

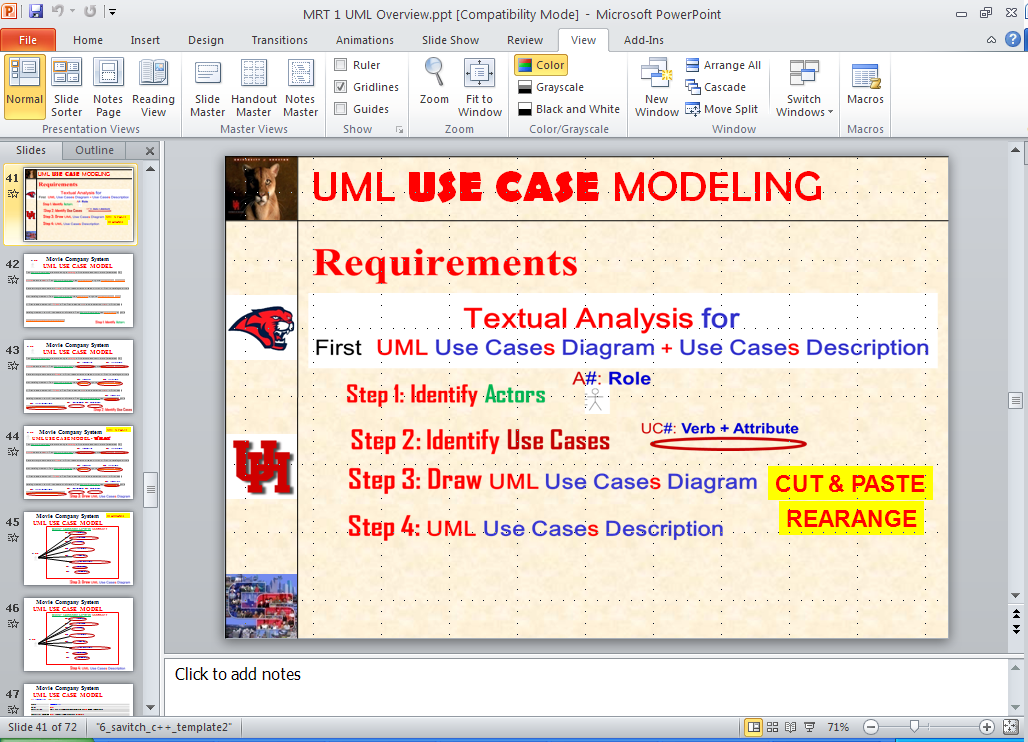
1. (50 points)

**UML USE CASE Modeling**

**Perform “compilable” Textual Analysis with 4 steps for REQUIREMENTS Workflow.**

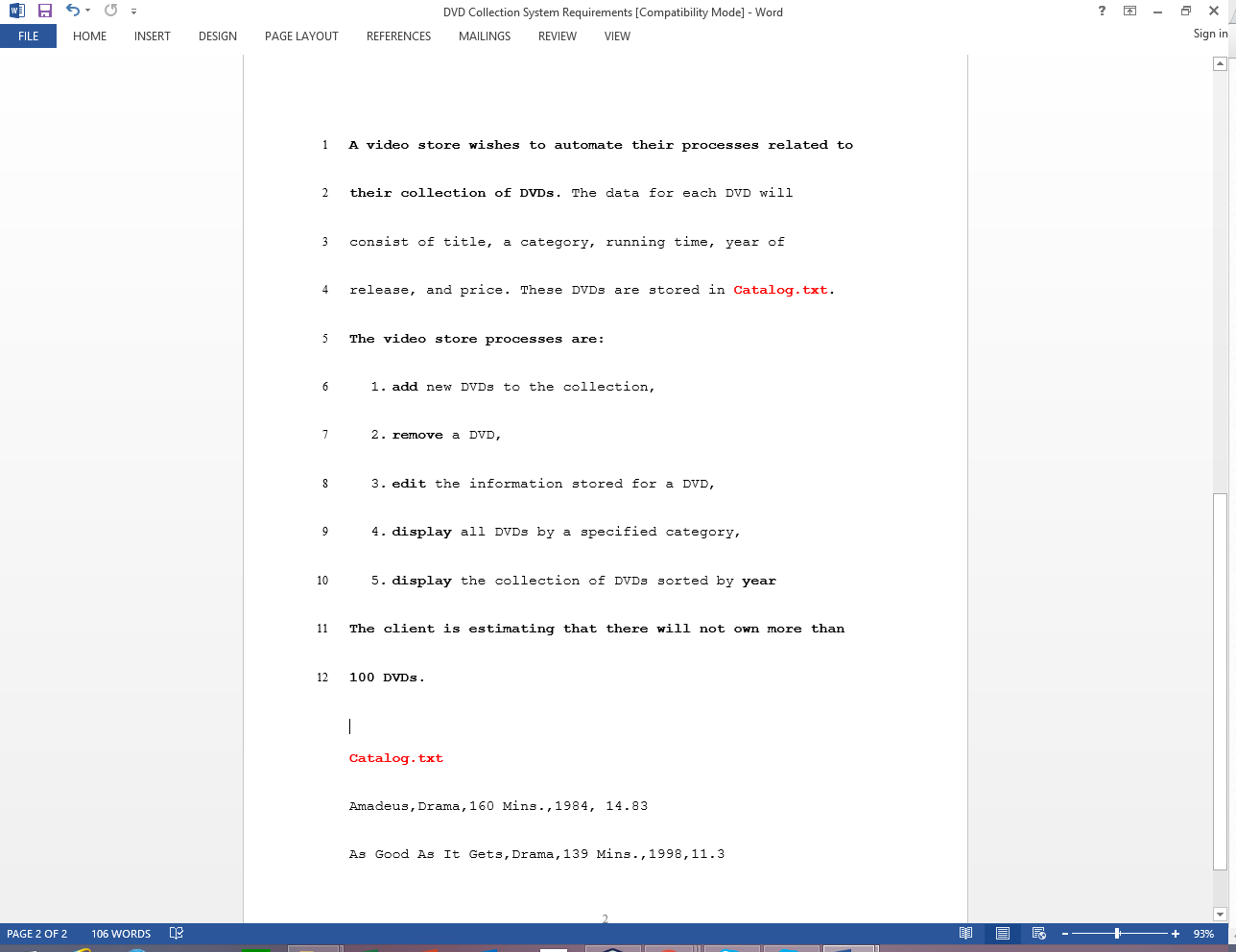
**You only need to show addDVD templated Use Case Description.**

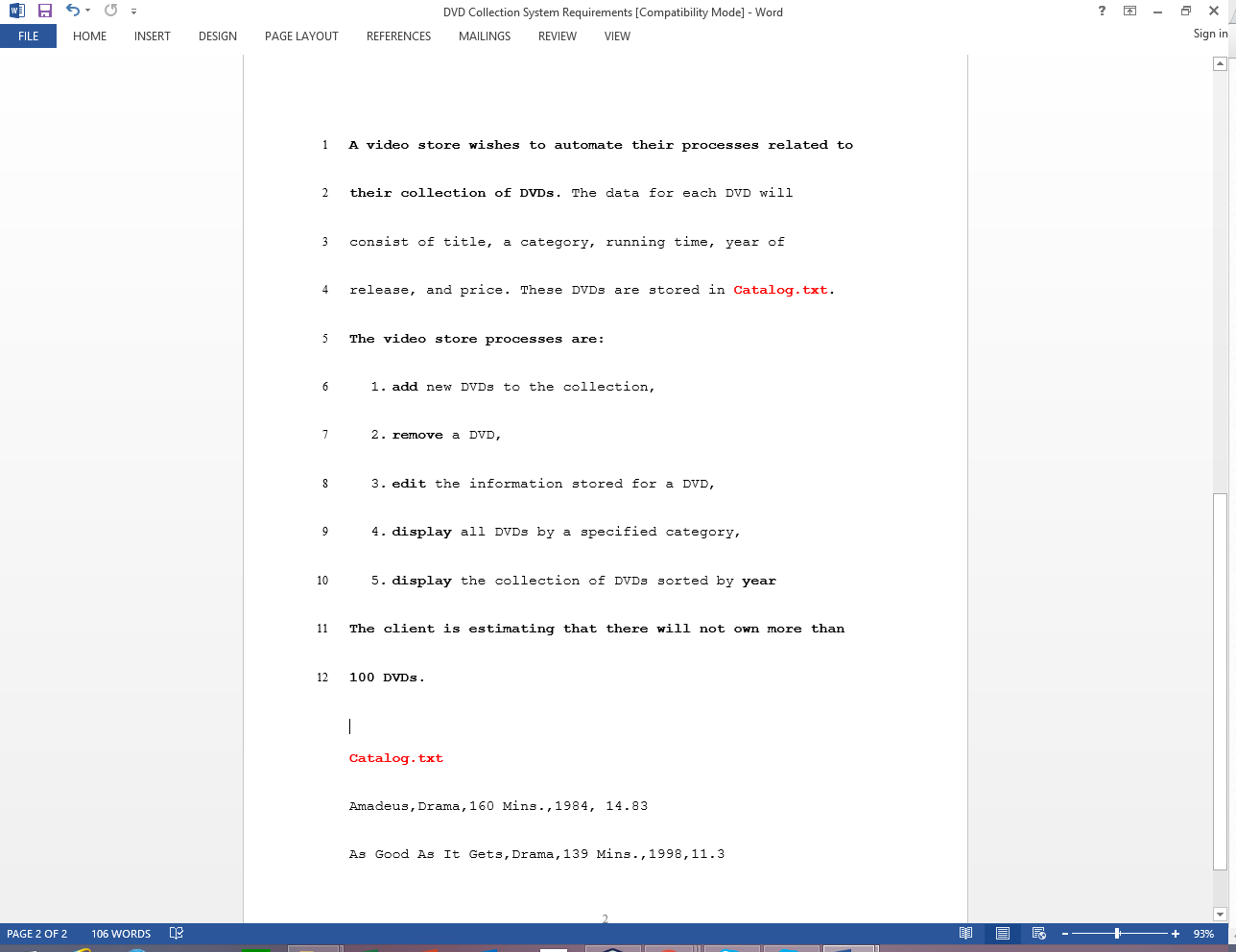
**(NOTE: Choose the Use Case name following the naming convention for “C” modules and append UC before it))**



**(Please see the UML Models EXAMPLES on the last pages.**

**DVD Collection Application Requirements:**

**Answer:**



UC3: **UCEditInfo**

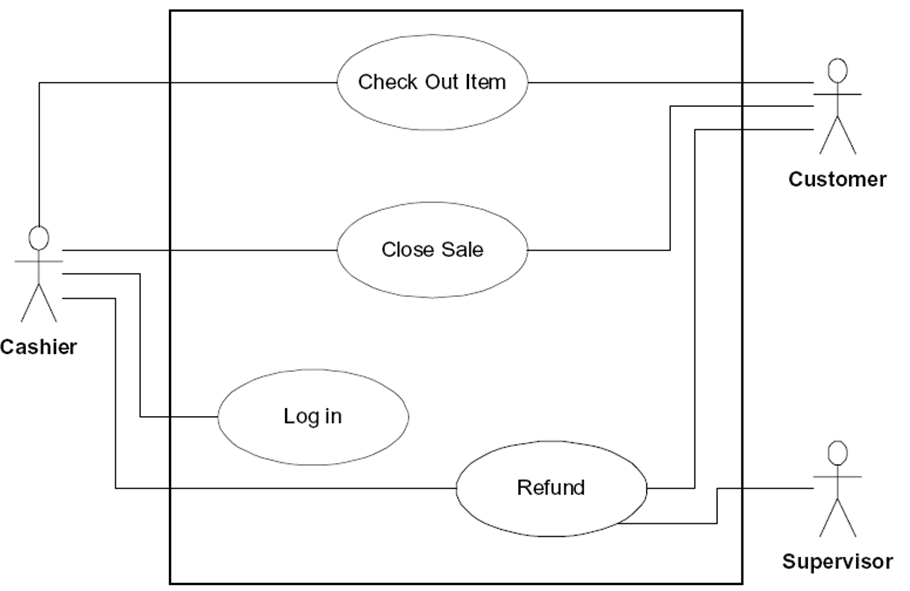
UC2: **UCRemoveDVD**

UC1: **UCaddDVD**

UC5: **UCDisplayByYear**

UC4: **UCDisplayByCat**

**UML DIAGRAM**

****

UC5

UC4

UC3

UC2

UC1

UC5: **UCDisplayByYear**

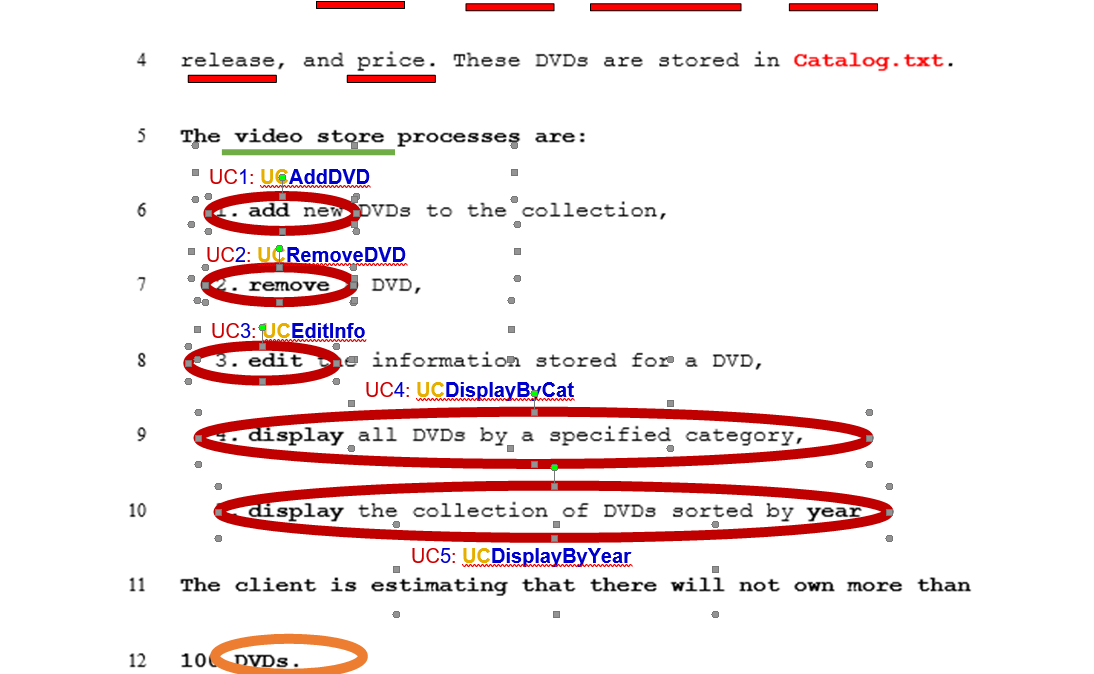
UC4: **UCDisplayByCat**

UC3: **UCEditInfo**

UC2: **UCRemoveDVD**

UC1: **UCaddDVD**

**A1: User**

****

**UML USE CASE  
  
UC1 UCaddDVD Description**

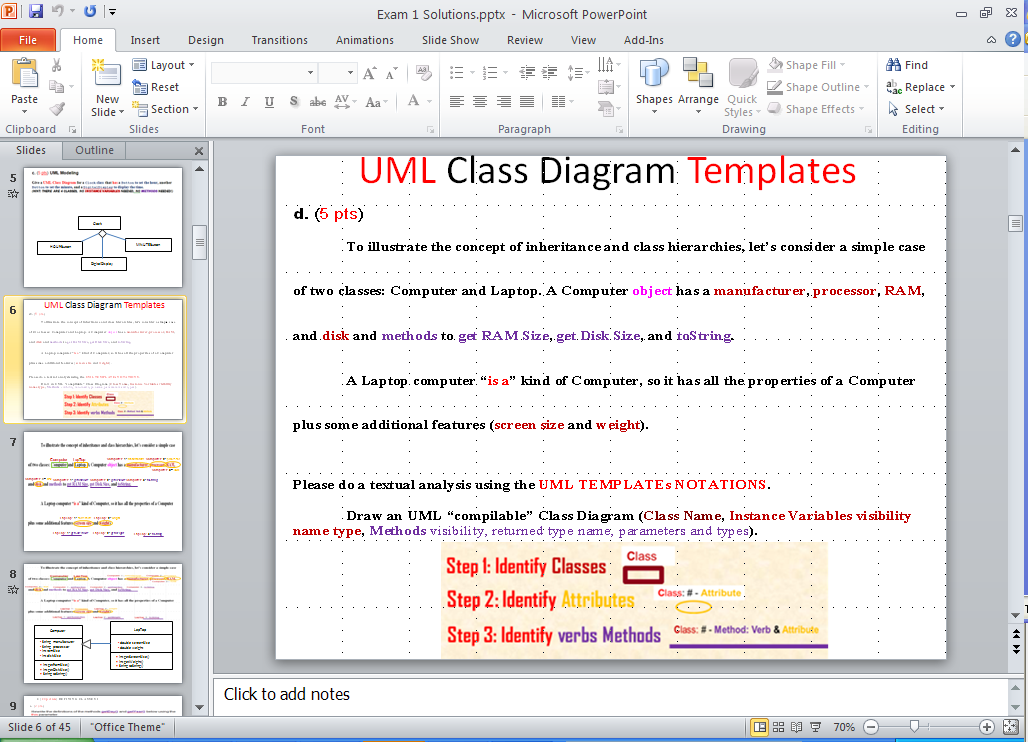
|  |  |
| --- | --- |
| **Name:** | **UCaddDVD** |
| **Actor:** | **Client** |
| **Description:** | **This use case describes the process used by Client to Add another DVD** |
| **Successful Completion:** | **Client requests Adding Artist by DVD title**   1. **Video Store checks to see if not already there and there is room in the DVD array** 2. **DVD with DVD title is added at the end of the DVD array and successful message is sent to the Client** |
| **Alternative:** | **Client requests Adding DVD by DVD title**   1. **Video Store checks to see if not already in the DVD array and there is no room** 2. **If DVD title either ALREADY present or NO MORE ROOM MAX-DVD title with DVD title is NOT added and Unsuccessful message is sent to the Client** |
| **Pre-Condition:** | **Client requests Add DVD** |
| **Post-Condition:** | **DVD with DVD title Added successfully or Unsuccessfully** |
| **Assumptions:** | **None** |

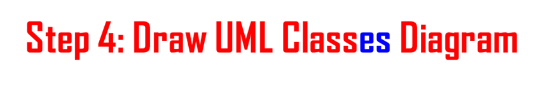
**UML MVC CLASS Modeling**

2. (50 points)

**Perform “compilable” Textual Analysis with 3 steps (1, 2, and 4) for ANALYSIS Workflows UML MVC CLASS DIAGRAM MODEL (you must use MVC design pattern).**

**(You do not need to show the methods).**





**Please see the UML Models EXAMPLES on the last pages.**

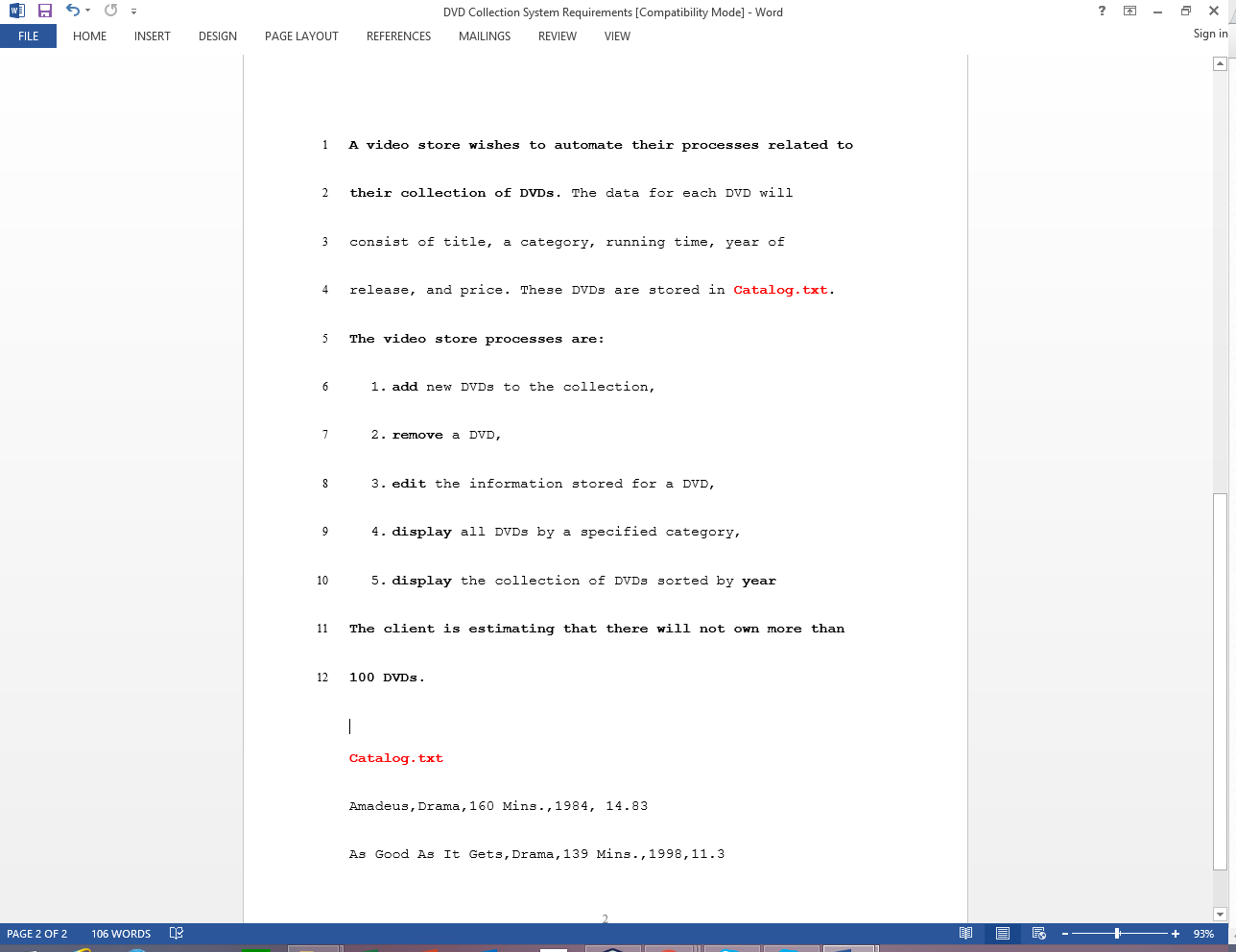
**Artist 1: name : String**

**Catalog**

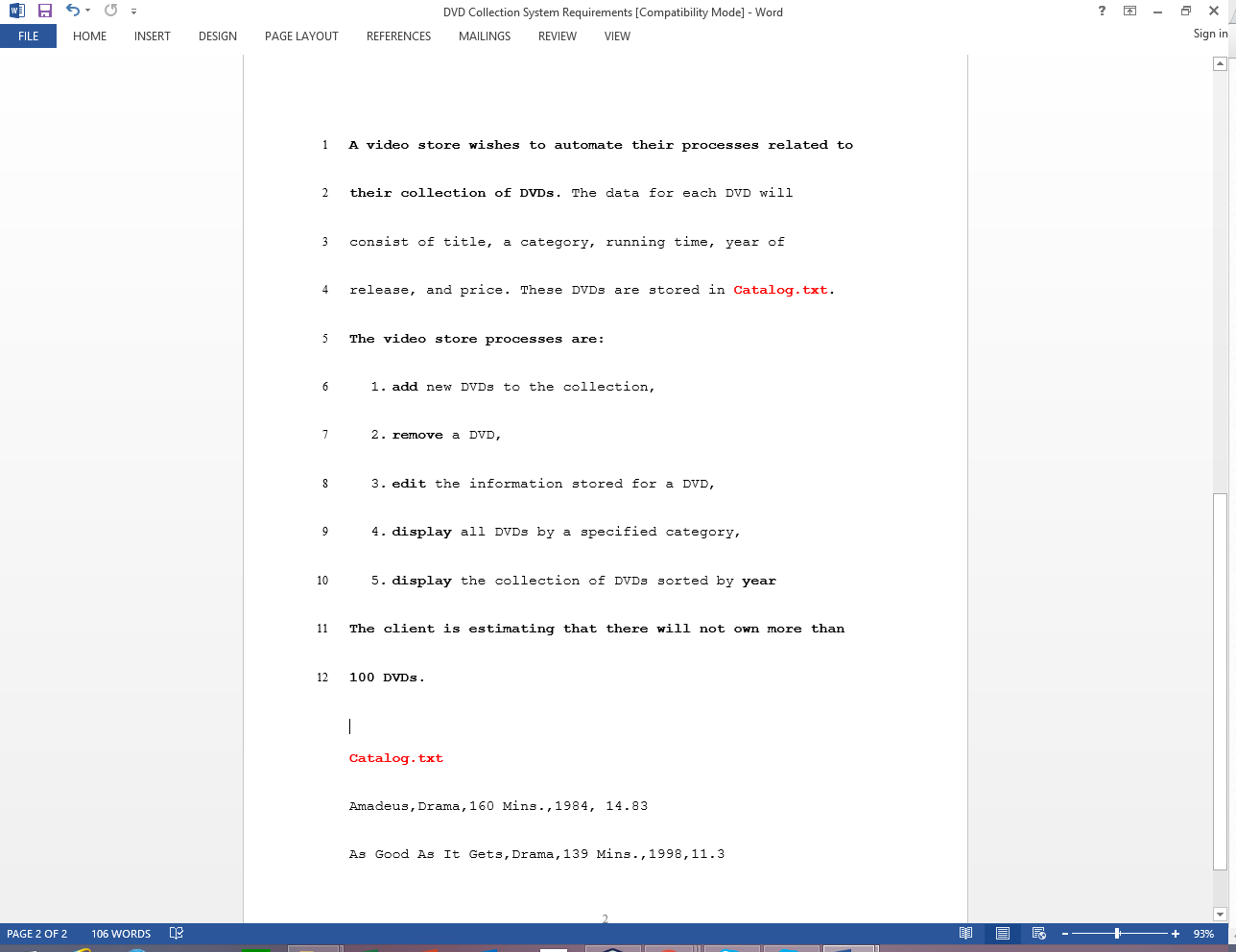
**Controller 1: UCAddArtist(**String artist**);**

**Catalog 1: addArtist(**String artist**);**

|  |
| --- |
| **Controller** |
| - **catalog**: **Catalog** |
| + **UCAddArtist**(String artist): void // UC1  + **UCListArtists**(): void // UC2  + **UCAddDisc**(String artistName, String discCatalogNumber, String discTitle): void // UC3  + **UCListDiscs**(): void // UC4  + **UCAddTrack**(String discCatalogNumber, String trackTitle, double time): void // UC5  + **UClistTracksForDisc**(String discCatalogNumber): void // UC6  + **UCrestoreCatalog**(): void // UC7  + **UCsaveCatalog**(): void // UC8 |

**DVD Collection Application Requirements:** 

**Answer:**



**Collection**

**DVD**

**DVD6: MAX\_DVD:int**

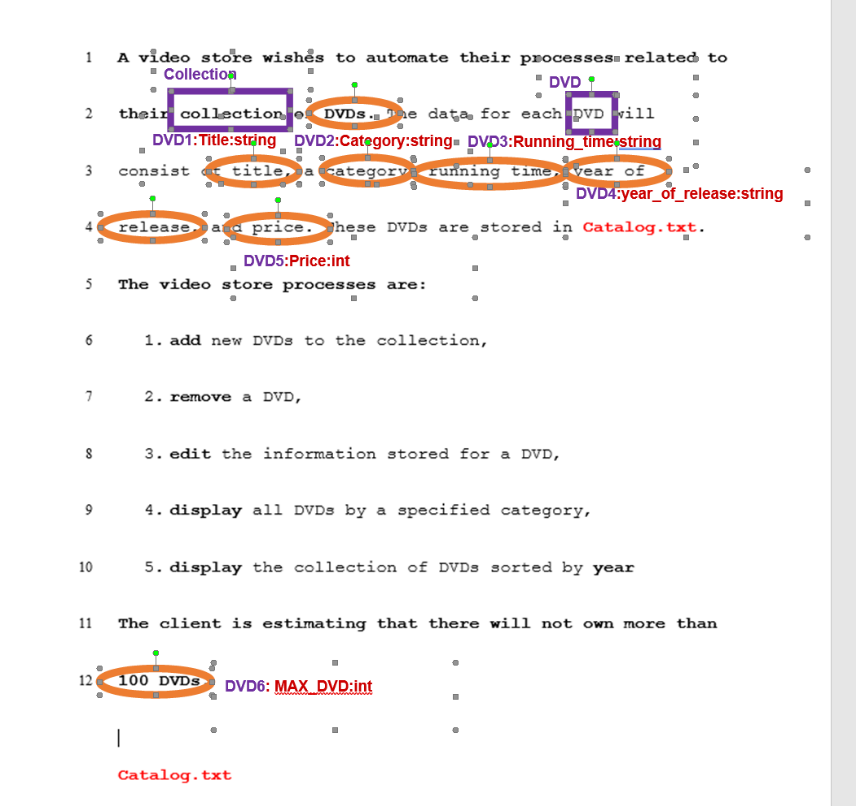
**DVD1:Title:string**

**DVD2:Category:string**

**DVD3:Running\_time:string**

**DVD4:year\_of\_release:int**

**DVD5:Price:double**



**Client**

“Has a”

“Uses”

|  |
| --- |
| **VideoStoreSystem** |
| * **control: Controller** * **view**: **View**   +title: **String**  +category: **String**  +running\_time:: **String**  +year\_of\_release: **int**  +price: **double** |
| + main ( ) |

|  |
| --- |
| **Controller** |
| - **catalog**: **Collection** |
| + **UCaddDVD**(String title): void // UC1  + **UCRemoveDVD**(String title): void // UC2  + **UCEditInfo**(String title): void // UC3  + **UCDisplayByCat**(String Category): void // UC4  + **UCDisplayByYear**(int Year): void // UC5 |

|  |
| --- |
| **Collection** |
| -**MAX\_DVD = 100 //1**  -**MAX\_CLIENT = 1 //2**   * **DVDS**: **DVD[MAX\_DVD] //3** * **clients**: **Client[MAX\_Client] //4**   #numberOfDVD: **int //5**  #numberOfclient: **int //6** |
| + **addDVD**(String title): void // 1  + **RemoveDVD**(String title): void // 2  + **EditInfo**(String title): void // 3  + **DisplayByCat**(String category): void // 4  + **DisplayByYear**(int year): void // 5 |

|  |
| --- |
|  |
| -name: string //1 |
|  |

**PSUEDO FOR FUNCTIONS:**

class DVDController

private List<DVD> DVDs

function DVDController()

DVDs = new List<DVD>

function RestoreCatalog(filePath)

lines = Read all lines from file at filePath

foreach line in lines

fields = Split line by comma

title = fields[0].Trim()

category = fields[1].Trim()

runningTime = int.Parse(fields[2].Replace(" Mins.", ""))

year = int.Parse(fields[3])

price = decimal.Parse(fields[4])

dvd = new DVD(title, category, runningTime, year, price)

DVDs.Add(dvd)

function SaveCatalog(filePath)

using writer = new StreamWriter(filePath)

foreach dvd in DVDs

writer.WriteLine(dvd.Title + ", " + dvd.Category + ", " + dvd.RunningTime + " Mins., " + dvd.Year + ", " + dvd.Price)

function AddDVD(title, category, runningTime, year, price)

dvd = new DVD(title, category, runningTime, year, price)

DVDs.Add(dvd)

function RemoveDVD(title, category)

dvdToRemove = null

foreach dvd in DVDs

if dvd.Title == title and dvd.Category == category

dvdToRemove = dvd

break

if dvdToRemove != null

DVDs.Remove(dvdToRemove)

function EditInfo(title, category, runningTime, year, price)

foreach dvd in DVDs

if dvd.Title == title and dvd.Category == category

dvd.RunningTime = runningTime

dvd.Year = year

dvd.Price = price

break

function DisplayByCat(category)

dvdsInCategory = new List<DVD>

foreach dvd in DVDs

if dvd.Category == category

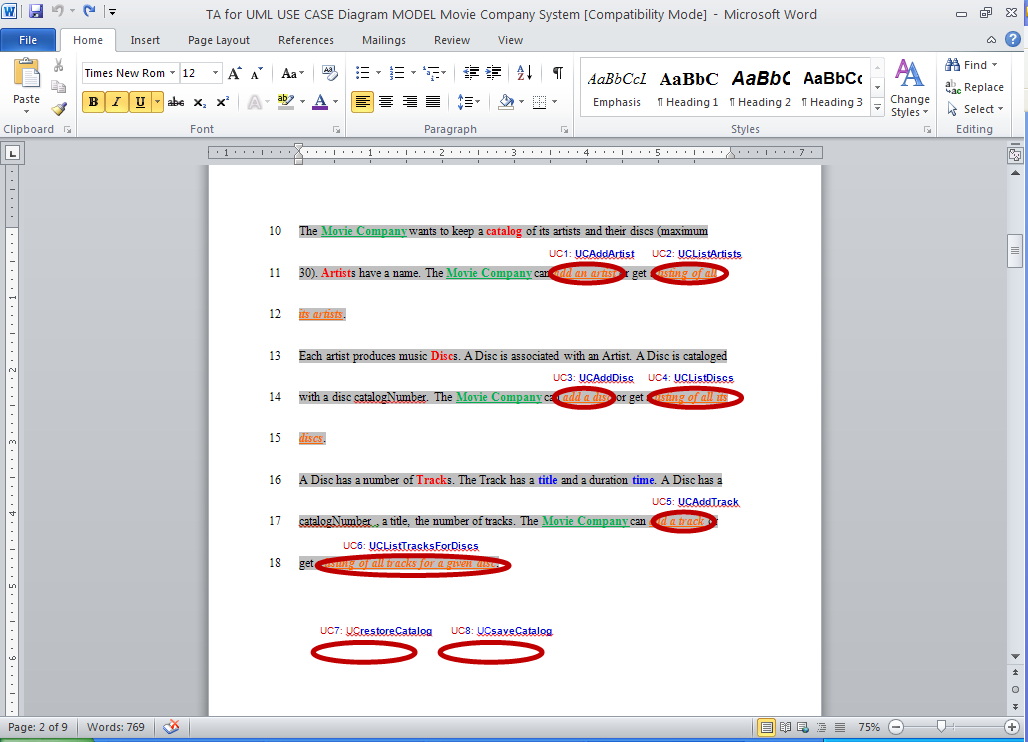
dvdsInCategory.Add(dvd)

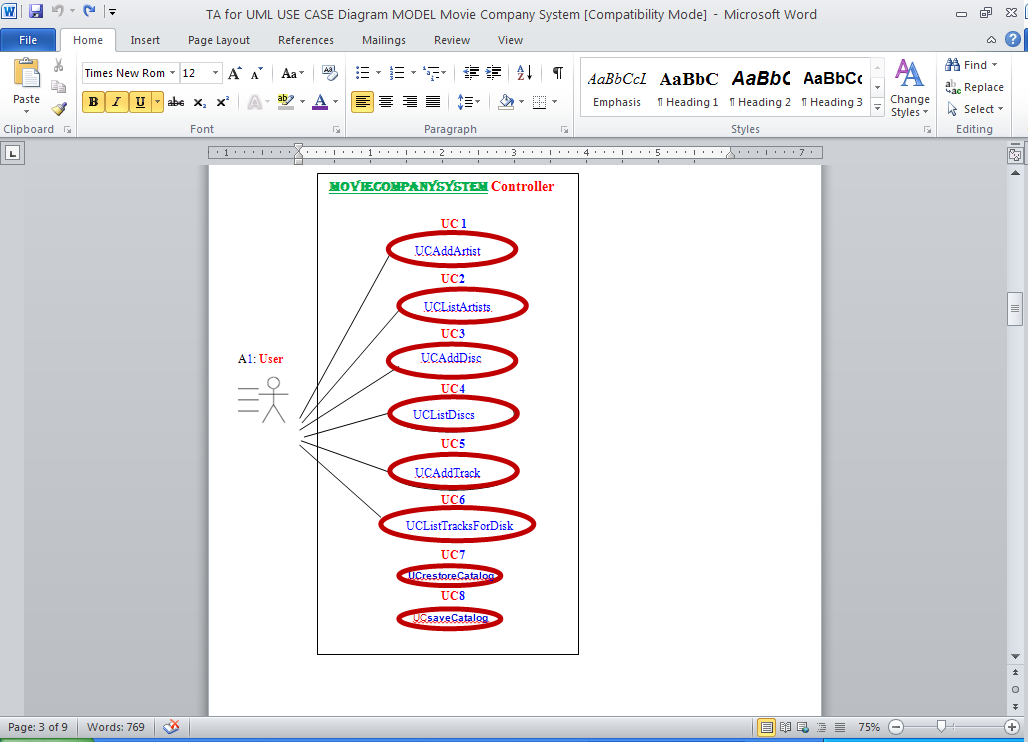
return dvdsInCategory

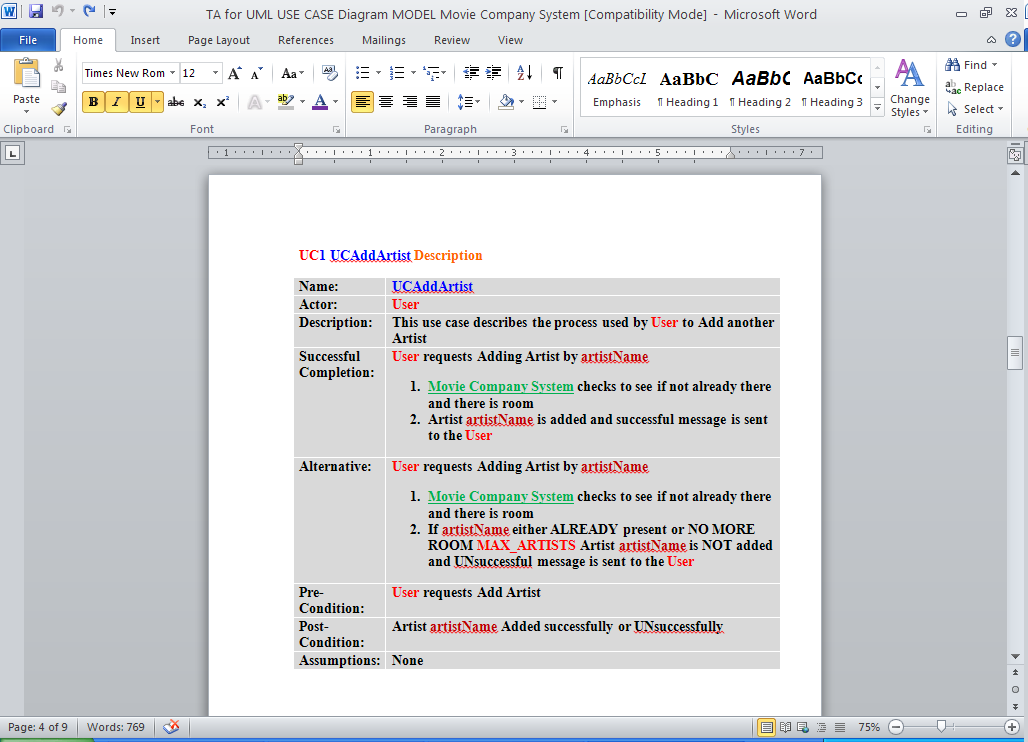
function DisplayByYear()

return DVDs.OrderBy(dvd => dvd.Year).ToList()

**Example UML USE CASE Model:**







**Example UML MVC CLASS Model:**

