Name: Chris Scarola

Description: System for librarians to keep track of various media (books, DVDs, video games) in a group of libraries e.g. CU has multiple libraries with different books, this could manage all of them. The idea is that the readers don't have access to this system, but the librarians handle all the books and checking out of books for the readers and can notify them if there are any overdue books.

Features Implemented:

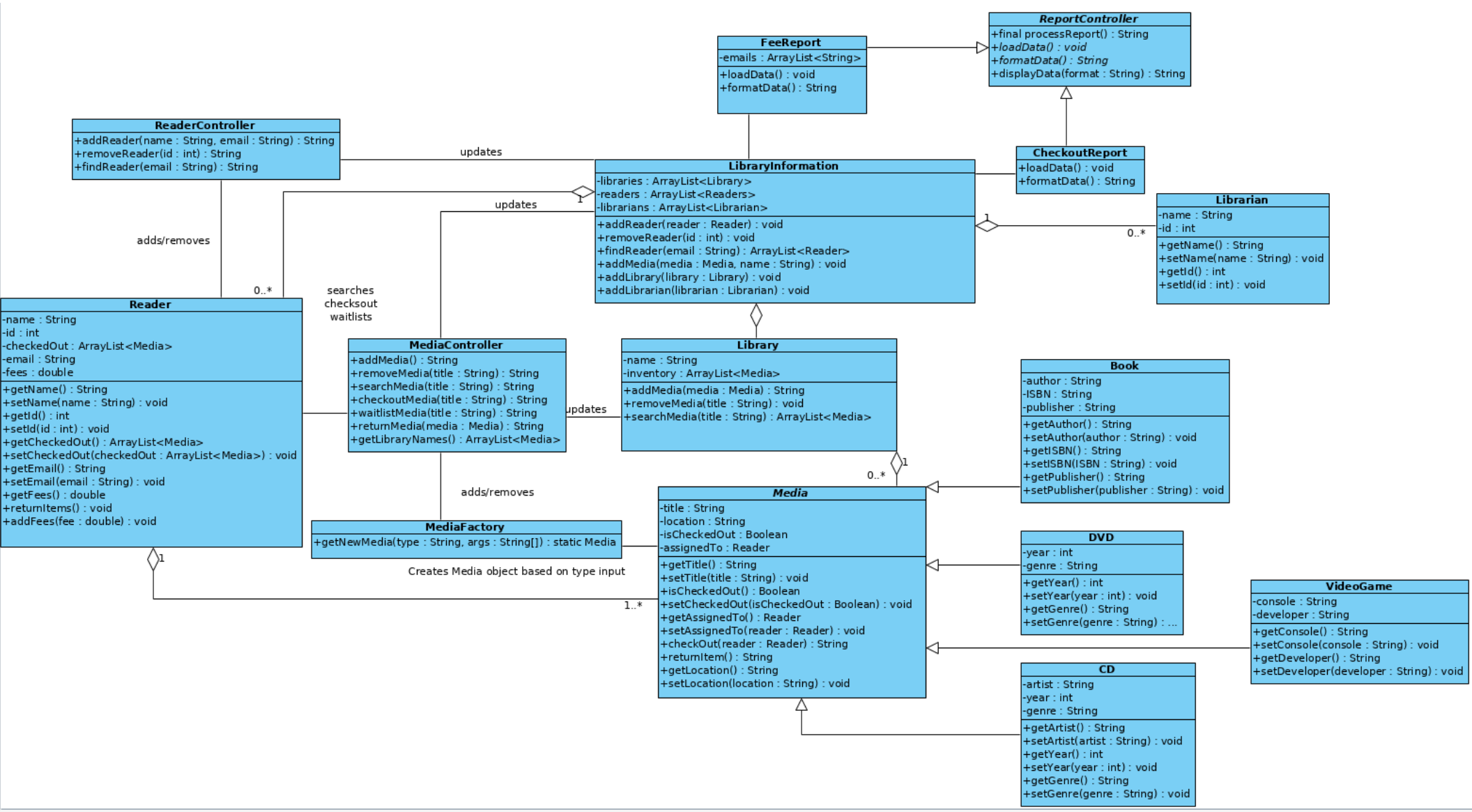
Use Case ID	Use Case Description
U-01	Librarians can add media
U-02	Librarians can remove media
U-03	Librarians can generate reports of overdue media
U-05	Librarians can generate reports of checked out media
U-06	Librarians can checkout books for a reader
U-07	Librarians can add readers
U-08	Librarians can remove readers
U-09	Readers can view checked out media to see return dates
U-10	Readers can check for overdue fees
U-11	Readers can search for media for where it is located in a library
U-13	Readers can self checkout media

Features Missing:

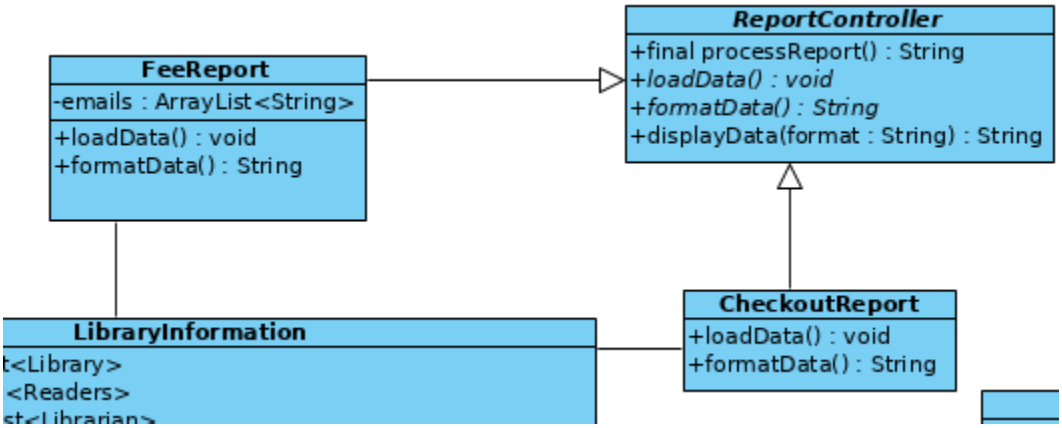
Use Case ID	Use Case Description
U-04	Librarians can notify readers of overdue media
U-09	Return dates missing, partial incomplete
U-10	Fees technically never calculated, partial incomplete
U-12	Readers can wait list media they are interested in checking out, if not available

Final Class Diagram:

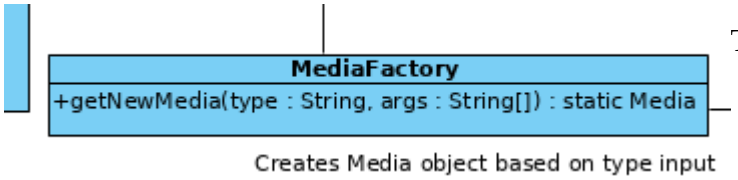
Some functions were removed for features not implemented. A media factory was added to make adding the children easier. Most of the design helped with building everything easier. I also was able to clean up the functions since I knew exaclty how they had to work once coding it. The features I did not implement seemed out of scope for the purpose of the class, learning patterns and planning. Since technically there is no login feature, it is hard to deal with the waitlist.


FeeReport
-emails : ArrayList<String>
+loadData() : void
+formatData() : String
ReportController
+final processReport() : String
+loadData() : void
+formatData() : String
+displayData(format : String) : String
ReaderController
+addReader(name : String, email : String) : String
+removeReader(id : int) : String
+findReader(email : String) : String
updates
LibraryInformation
-libraries : ArrayList<Library>
-readers : ArrayList<Readers>
-librarians : ArrayList<Librarian>
+addReader(reader : Reader) : void
+removeReader(id : int) : void
+findReader(email : String) : ArrayList<Reader>
+addMedia(media : Media, name : String) : void
+addLibrary(library : Library) : void
+addLibrarian(librarian : Librarian) : void
CheckoutReport
+loadData() : void
+formatData() : String
Librarian
-name : String
-id : int
+getName() : String
+setName(name : String) : void
+getId() : int
+setId(id : int) : void
1
0..*
adds/removes
updates
searches
checksout
waitlists
Reader
-name : String
-id : int
-checkedOut : ArrayList<Media>
-email : String
-fees : double
+getName() : String
+setName(name : String) : void
+getId() : int
+setId(id : int) : void
+getCheckedOut() : ArrayList<Media>
+setCheckedOut(checkedOut : ArrayList<Media>) : void
+getEmail() : String
+setEmail(email : String) : void
+getFees() : double
+returnItems() : void
+addFees(fee : double) : void
MediaController
+addMedia() : String
+removeMedia(title : String) : String
+searchMedia(title : String) : String
+checkoutMedia(title : String) : String
+waitlistMedia(title : String) : String
+returnMedia(media : Media) : String
+getLibraryNames() : ArrayList<Media>
Library
-name : String
-inventory : ArrayList<Media>
+addMedia(media : Media) : String
+removeMedia(title : String) : void
+searchMedia(title : String) : ArrayList<Media>
updates
Book
-author : String
-ISBN : String
-publisher : String
+getAuthor() : String
+setAuthor(author : String) : void
+getISBN() : String
+setISBN(ISBN : String) : void
+getPublisher() : String
+setPublisher(publisher : String) : void
adds/removes
MediaFactory
+getNewMedia(type : String, args : String[]) : static Media
Creates Media object based on type input
Media
-title : String
-location : String
-isCheckedOut : Boolean
-assignedTo : Reader
+getTitle() : String
+setTitle(title : String) : void
+isCheckedOut() : Boolean
+setCheckedOut(isCheckedOut : Boolean) : void
+getAssignedTo() : Reader
+setAssignedTo(reader : Reader) : void
+checkOut(reader : Reader) : String
+returnItem() : String
+getLocation() : String
+setLocation(location : String) : void
DVD
-year : int
-genre : String
+getYear() : int
+setYear(year : int) : void
+getGenre() : String
+setGenre(genre : String) : ...
VideoGame
-console : String
-developer : String
+getConsole() : String
+setConsole(console : String) : void
+getDeveloper() : String
+setDeveloper(developer : String) : void
CD
-artist : String
-year : int
-genre : String
+getArtist() : String
+setArtist(artist : String) : void
+getYear() : int
+setYear(year : int) : void
+getGenre() : String
+setGenre(genre : String) : void
1
0..*
1..*

Design Patterns:



The first pattern is the template method used to generate the various two reports. This patten worked here because the reports were similar enough that the steps were the same.



This factory was used to build media objects.

Learned:

I learned how to architect a program instead of winging it. There were still parts that I felt like I was making up, but it flowed better and I ended up with 2400+ lines of code pretty easily. Going through the design patterns was interesting, but seems overkill for this project, even if I used them.