**MUSIC MANAGEMENT SYSTEM**

• As a group, we decided to design a music market management system. This program is basically divided into two different account types : **manager** and **customer**. Customer can create their own account after specifying the name and balance information, and then create their own song list through the program, enter the genres they love, get random song suggestions from customer’s favorite genres and make customer listen to the song which is suggested. Also, by buying them through their account they can listen to the songs. During these activities, the program can update balance and query the user's balance. Manager provides access to the program by entering a password and manages the Music Store as seller. Manager holds music and album information in it’s data base , then manages the inventory. Customers have their own balance. According to the purchased songs and albums, balance will be updated and can be seen in the program.

* ACCOUNT:

We can say that Account class is the class that forms the basic building block of work in the project. It takes on the role of **parent class**, which derives two other child class**: Manager** and **Customer**. **Account** class can read **Album** and **Song** data from a file and transfers the data to make the **Manager** class read and process the data. For the **Customer** class, with the help of **Album** and **Song** class constructors the data is initialized by default values, **Customer** gets and transfers the name and balance values for the user.

* CUSTOMER:

The Customer class is connected to the **Account** class as a **derived class** and indirectly to the **Manager** class through the customer's data, allowing customer to listen to the songs and albums they want by purchasing and making their own balance account. In addition, it is also possible to change the account type before the program is finished (User can switch between manager account and customer account.)

* MANAGER:

The **Manager** class first creates a new password and name. By connecting to the Song, Album and Account classes (with the help of main code), it receives music and album information. Manager is able to update stocks. Also stocks and balance of the store are updated automatically when customer buys a new item. The manager's own balance account is defined as $ 1000 at the beginning. Manager can view balance of the store if desired. It can also suggest random songs to the customer according to the favorite genres of the customer. In addition, it is possible to change the account type before the program ends. User cannot access to the manager account without entering true password (User can switch between manager account and customer account.)

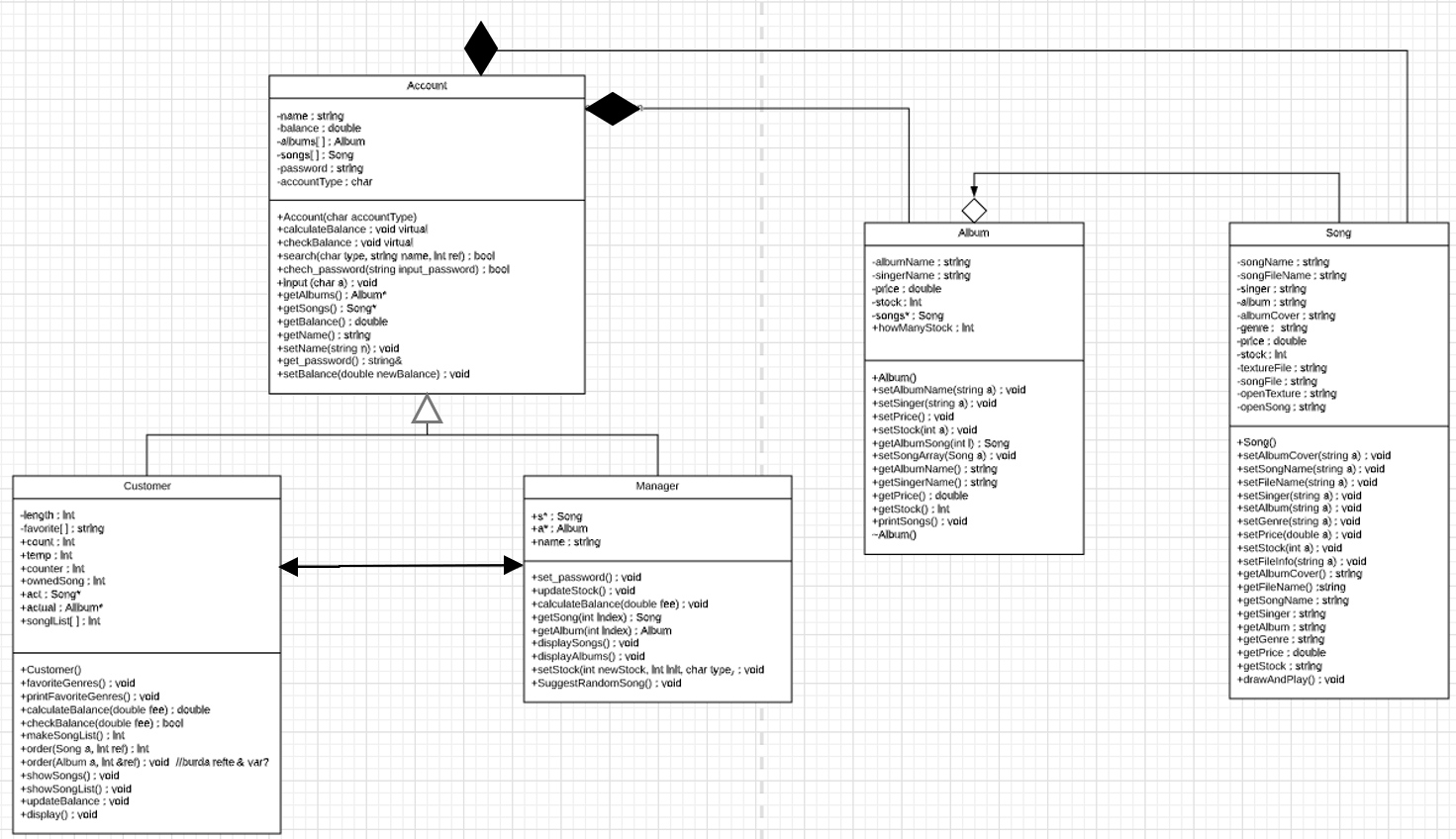
* ALBUM:

Album class is a class in which we assign all information for an album and allow you to contact the desired location.

. SONG :

Song class provides information of the songs that program contains. This class has differences from the other classes. This class uses an API called SFML. With the help of SFML, program can pop-out a window and play music.

UML CLASS:



**Conclusion**

-Album has aggregation with song

-Account class has composition relation with album class and song class

-Customer and Manager are inherited from Account class.

-Communication between Customer and Manager is provided from

Source.cpp(int main) file

-During in the runtime process of the program, we get a lot of debug assertion failure errors. We have lost loads of time so we could not develop the program into a much more extensive form. When we debugged the program, we noticed that, this ‘Debug Assertion Failed’ error is originated from destructors. Due to this fact, we could not implement destructors into the program.

-We could have done an Graphical User Interface but we didn’t have much time.

-We could have used exceptions and templates.