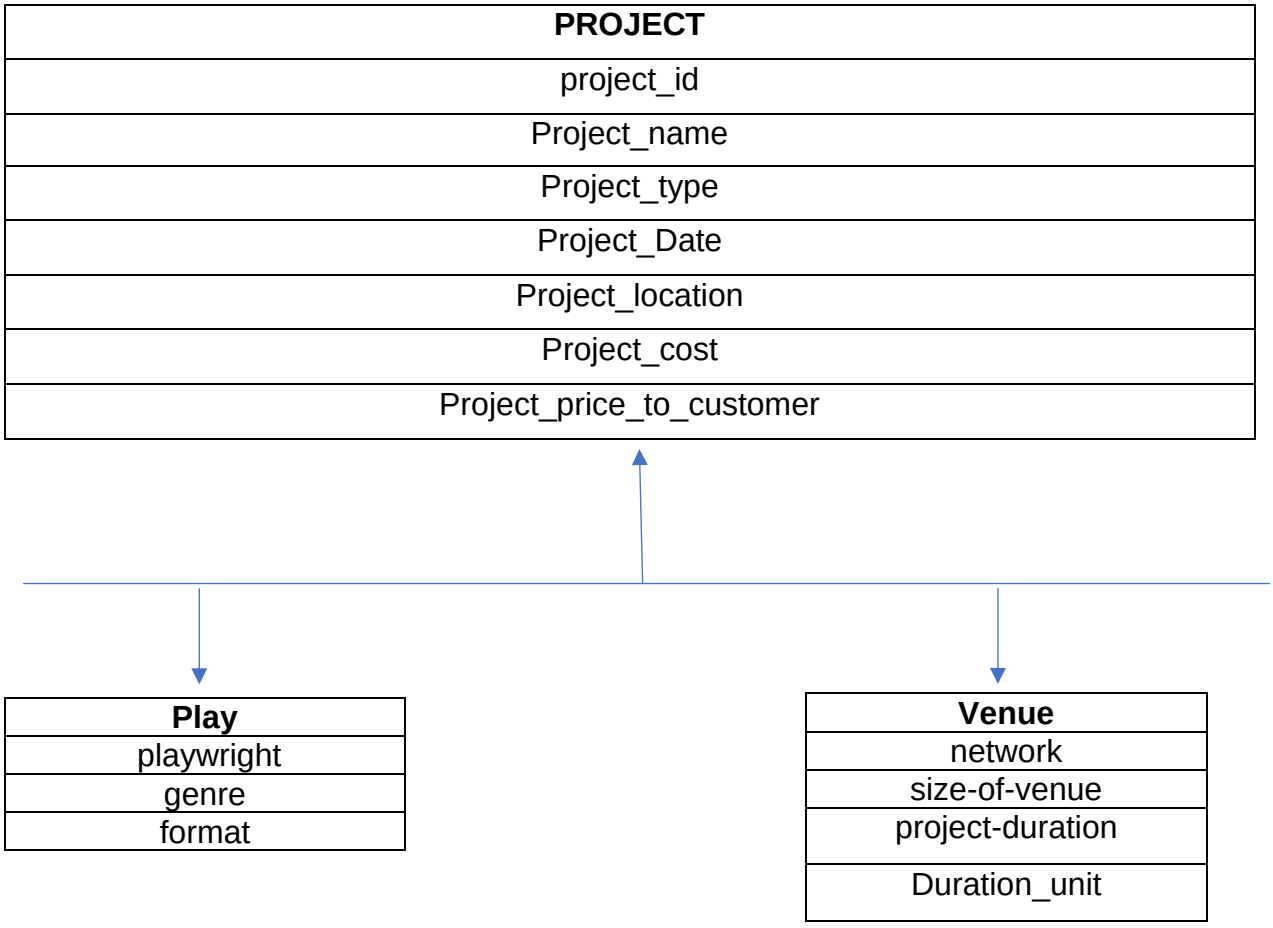
Assignment 2

Report

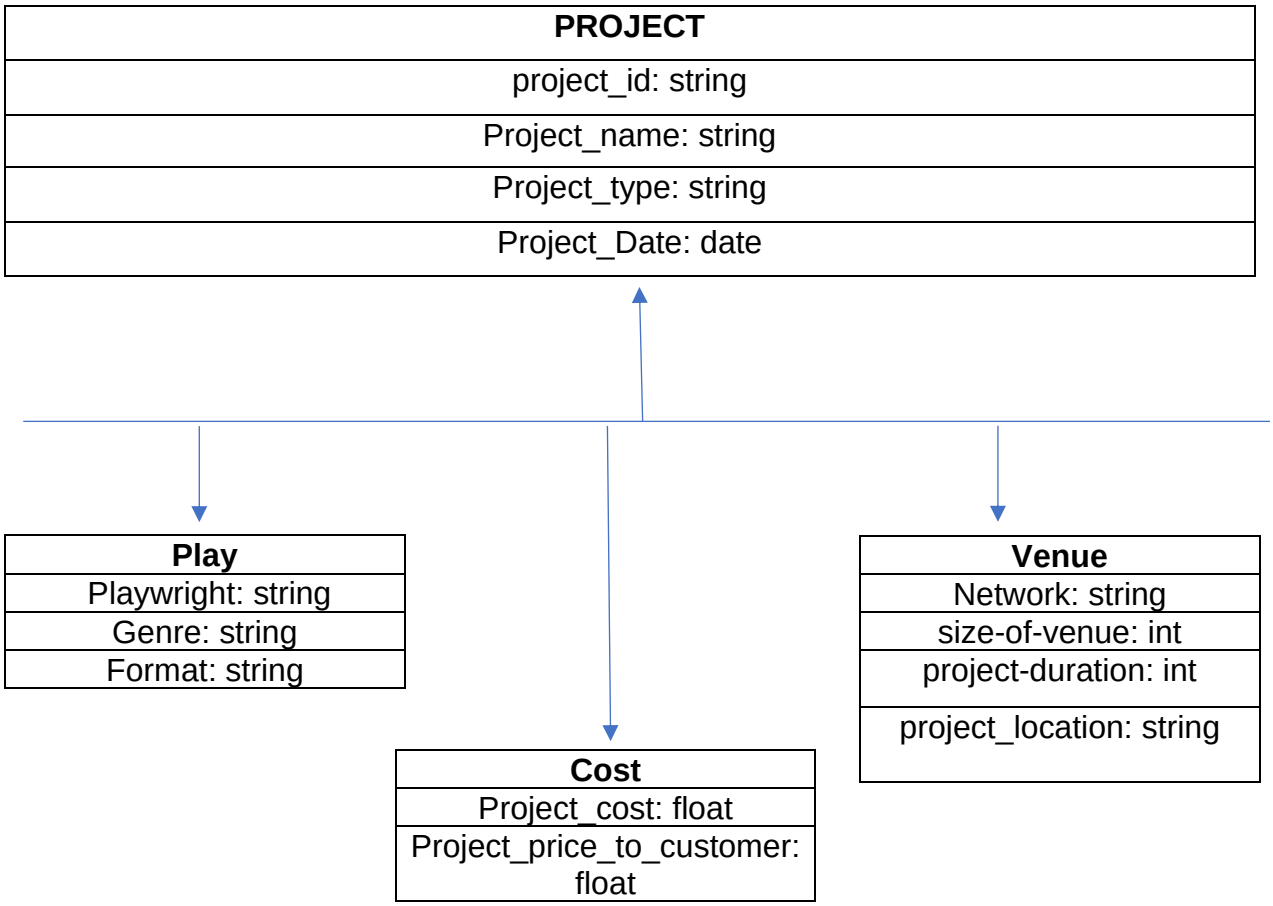
A. Design

i. UML class diagram

UML CLASS DIAGRAM 1:



UML CLASS DIAGRAM 2:



ii. Justification for design

I would use the second diagram. This is because this diagram is more clear, complete, and simple. The first diagram does not have another Cost class table whereas, the second diagram does. This class diagram conveys the key concepts and relationships that is easy to understand and maintain. This diagram also mentions the data type for each attribute which makes it easier for me. In the first diagram, the Project class has a lot of attributes.
This makes the diagram very cluttered and therefore, in the second diagram I managed to reduce the attributes from the Project class. I made 3 class tables: Play, Venue and Cost. The

However, both diagrams do the job but the second one makes it easier and lot better for me to develop the program.

B. Post-documentation

So, I decided to make a few java classes referring from my UML diagram. For the ‘Project’ class it has four private instance variables (project_id, project_name, project_type, and project_date). Also, a constructor for this class that takes in four arguments (id, name, type, and date) and assigns their values to the instance variables. I also set getter methods (getProject_id(), getProject_name(), getProject_type(), and getProject_date()). Finally, I have defined a method called "getProject()" that concatenates the values of all the instance variables and returns the result as a string.

I have taken an OO approach to program the classes, using inheritance. I created another class called ‘Play’ which is similar to other other classes I have made. The "Play" class inherits all the instance variables and methods of the "Project" class. I have defined four private instance variables (playwright, genre, format, and price) that are specific to the "Play" class.
You have also defined a constructor for the "Play" class that takes in twelve arguments, including the four arguments required by the "Project" constructor. Inside the constructor, I called the "super" keyword to invoke the constructor of the “Project” class and pass in the required arguments.
Furthermore, I defined getter and setter methods for the "Play" class instance variables, as well as a new version of the "getProject()" method that concatenates the values of all the instance variables and returns the result as a string.

For the ‘Cost’ class and ‘Venue’ class I have done the same thing except the instance variables used for both class are different. For ‘Cost’ I used (cost and price), for ‘Venue’ I used (network, size_of_venue, project_Duration, project_location).

To manage a list of "Project" objects and provide functionality to generate a summary report based on the data in that list. I made a “ProjectManager” class. This class has a list of "Project" objects as one of its instance variables. I have defined a constructor for the "ProjectManager" class that initializes this list as an empty ArrayList. I created a method called "addProject" that takes in a "Project" object and adds it to the list of projects. Furthermore, creating a method called "generateSummaryReport" that calculates various statistics about the projects in the list and returns a summary report as a string.

I then created the GUI class “ProjectGUI” using Java Swing library. It defines ‘ProjectGUI’ class that extends ‘JFrame’ to create text fields, labels and buttons. Moreover, The “ProjectGUI” class also defines a “ProjectManager” object that is used to manage the projects added by the user. The “addButton” and “summaryButton” buttons have event listeners attached to them that allow the user to add new projects and generate a summary report respectively. However, the GUI code did not work.