Hello and welcome to this presentation on Robustness and Sequence Diagrams, I, James Moran, will be giving this presentation on their use in the Game Café, let’s begin.

What are Robustness and Sequence Diagrams? For Robustness Diagrams, they are a graphical way to depict use cases, that is also a ‘sanity check’ on Use-Cases (to make sure the Use-Cases match-up to what functionality the User would want from the system), that also allows the team to uncover new classes, which would not have been previously identified for the system. Sequence Diagrams allow a team to design the system in detail, considering the methods/functions the classes in the system will have and the order that these methods/functions are executed.

Considering the Game Café as an example, the Model has a representation of the Database and its tables, for the information that is stored about certain aspects of the Game Café (e.g. for Members, Bookings and Hardware available at the Game Café). There is a utility class (IDUpdateSystem), to handle updating the ID for each of these objects, whenever a new entry is added to the database. Next comes the View of the system, allowing the user to interact with the Model and Controller (by either traversing between forms or viewing information). This leads into the Controller, with the logic for the form visual-aspect, along with event-handlers for certain controls on this form, as well as event-handlers for the form itself (such as when the form is closing). The logic for these events, is contained within a class for the form, given what will occur when these events are raised (e.g. asking the User to confirm their choice).

Advantages and disadvantages of the MVC Design-Pattern, are detailed here, with the Advantages being that of Supporting Multiple Views (allowing the user to change the appearance of the system’s forms, as the view is separated from the Model) and Accommodating for Change (most notably, that of changing User-Interface Requirements, for if the User wants to view the application on another device). The potential disadvantages are that of Complexity (introducing new levels of indirection to the project, as well as taking into account the Event-Driven systems for User-Interface code, which can become more difficult to debug) and the Cost of Frequent Updates (even with decoupling between the Model and the View, as the developers of the Model should at least consider the View, when making changes to the system Model).

This slide details the references used in this presentation, thank-you for taking the time to watch this presentation and goodbye for now.