Hello and welcome to this presentation on the Model View Controller Pattern (MVC). I, James Moran, will be giving this presentation, let’s begin.

What is the MVC? It separates modelling of the logical-framework, visual-presentation and handling user-input, into three separate design aspects for a project. The Model handles the business-logic of the application, responding to requests for information from the View, given the current state, as well as any commands to change state (such as the user clicking on a button in the View), from the Controller. The View is the visual aspect for the application, in the case of the Game Café, showing a form with various controls, that show information to the user, garnered from the Model (such as Membership Information) and allow traversal through the application (via form navigation buttons, to access a certain part of the system).

This is how Use-Cases are utilised in the Game Café, as shown in this Use-Case Index-Grid (for that of a text-overview of each Use-Case). For a Game Café Member, they can make a Booking for themselves, make a Booking for a Non-Member and get a ticket for an eSports Event. For a Game Café Staff Member, they can manage Membership Information, manage eSports Event Information, manage Hardware and Software information, associate Bookings with Members and associate eSports Event Tickets with Members. The Complexity relates to how difficult it is to implement that Use-Case in the system and the Priority is for identifying which Use-Cases should be implemented first (for a certain Primary Actor).

Moving on from the Index-Grid in the previous slide comes a UML representation of these Use-Cases, as a Use-Case-Diagram. This breaks down the components for some of the Use-Cases (e.g. Adding/Updating/Maintaining/Viewing Current Membership Information), as well as the use of <<include>> connections between Activities, indicating a link between one Actor’s Use-Case and another Actor’s Use-Case (e.g. a Game Café Member can purchase an eSports Event Ticket, that a Game Café Staff Member would want to associate with the respective Game Café Member).

The advantages of Use-Case Modelling, are detailed here, with the Advantages being that of a User-Centred Technique (ensuring that the correct system is developed for the User), easy for the Customer/User to understand (as they are composed in a natural-language form, providing an excellent way to communicate with them). As well as providing an Objective Means to track the project (by deriving earned value from the implementation and delivery of them). The potential disadvantages of their use though, are that of Potential Redundant Classes (as a Use-Case can span across the use of multiple classes, given Object-Orientated-Design), the use of a different paradigm to the Object-Model (with unclear mapping between the Use-Case Model’s structure and the Object-Model’s network structure) and Poor Scalability (not providing enough Use-Cases for an adequate specification or using too many, leading to the functional decomposition of objects and classes).

This slide details the references used in this presentation, thank-you for watching and goodbye for now.