Hello and welcome to this presentation on CRC Cards. I, James Moran, will be giving this presentation, let’s begin.

First off, what are CRC Cards? They are Class Responsibility Collaborator Cards, where the class identifies a certain entity, component or service of the system, the responsibilities are what the class is to keep track of, or the actions it can perform and the collaborators, are other classes that are related to/derive from or responsible for, certain aspects of this class.

Here is an example of their usage in the Game Café System. First, there is a Member class, whom can make bookings and they obtain eSports event tickets. Second, comes the Booking class, which has a list of booked Hardware, date and time, duration, price and is owned by a member or non-member. Third, comes the ESportsEvent class, which has a date and time, number of tickets and is owned by a Member. Fourth, is the Hardware class, which has a name and compatible software. Last is the Software class, with a title, game type, single or multi-player and a PEGI rating.

From the first pass, it is now possible to determine the properties/methods of a class, given that a general overview of their responsibilities has been provided. Note that each class has an ID (which is used as a primary-key, in the project’s database). Considering the Member class, The Game Café wants to know the details for a Member (e.g. Membership Type). A Member is linked to the Booking class (as they can make multiple bookings) and the eSports-Event class (as they can also purchase tickets for an eSports Event). The Booking class will store all of the relevant details for a Booking (such as which Member has made that Booking). Hence, they are linked to the Member class, as well as the Hardware class (as Member’s can choose a particular platform for their Booking). Considering the Hardware class, there is a Software class, for the Software that runs on that piece of Hardware (if appropriate). This class stores the appropriate details for that piece of Software.

There are potential drawbacks of using CRC-Cards though, such as low-cohesion and high-coupling (where a class’s role is not clearly defined, relying on multiple other classes). Macho classes (where one class handles most of the project’s functionality, causing other classes to have a very niche role in the project, or even an unnecessary role). Last off, is the potential for no clear role to be defined for a class (related to macho classes, but also including classes with low-cohesion).

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