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. 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. Starting with the Member class, there is a Member-ID (all classes for this project have an ID, as a primary-key to identify them in a Database-table), a Membership-Type, a Member-Age-Group and a list of Bookings by that Member. These details are required, as expressed in the first sprint’s set of User Stories. For the Booking class, there is a Booking-ID, a set of Hardware for that Booking, a date and time, a duration, a price, an owner and the Member-ID as a foreign key (to show that Members own Bookings). For the ESportsEvent class, there is a date and time for the event, the remaining quantity of tickets and an owner of the respective ticket. For the Hardware class, there is a Hardware-ID, Hardware-Name and a list of compatible software. For the Software class, there is a Software-ID, Game-Title, Game-Type, Single-Or-Multi-Player Player-Number and a Game-Age-Rating.

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.