**Report**

The solution is currently in an unfinished state and a lot of the requirements remain incomplete, development was largely held back due to my own misunderstanding of business requirements and inability to estimate the time it would take to implement features. A lot of time was spent developing control of the Catering API through the MVC application, this included mapping DTO object on both sides. I understand now that this was not necessary to do, however I feel a lot was learned from this process regarding understanding GET and POST requests to an API. The service layer also aids in keeping the methods within the controller minimal.

View models are implemented well, with view models in place for guests, events, food Items and menus, these also feature data annotations that improve useability and prevent bad data from being entered. There is still room for improvement as the staff entity also requires a view model and validation on the current models is not ideal, for instance although the data must be a date or time, any date or time can be entered when creating a new Event, including in the past.

An API call to the venues API is made successfully through the event controller, the Create method uses a private method to retrieve a list of events in a string format and displays these in the event view model.

A Problematic stage of development was using AJAX to update the staff Boolean property “Attended”. This is within the Event menu in the MVC application and the manage Guests sub menu. The main issue was in validating whether a checkbox was clicked and obtaining the value from this checkbox, the difficulty was compounded by the number of checkboxes being dynamic, it is dependant on the number of Guests attending an event. More information can be found in my development notes.

The only current security implemented is having cross origins enabled with only the venues API given access. If more development time was available, I would implement user access control, these steps would include adding role services, requiring authenticated users, and creating a class to manage the users.

If I was to undergo this project again, I would take time at the beginning to properly map out the development and provide an estimate of how long each of the features would take to implement, this could be done by using an application such as Trello to properly plan the project before I start development.