Progress (20%):

1. **Fail (35-39%) Minimal progress, student has no working code.**
   The “CMSystem” project works well.

2. **Pass (40-49%) Student has working code to present but it has not progressed much beyond an introductory MVC tutorial (basic CRUD).**
   “~\CMSystem\Views\Annoucement_AnnouncementTable.cshtml” is a partial view of the announcement details.

3. **2:2 (50-59%) Student has implemented a web application but there are a number of bugs which would make the application unsuitable for release.**
   Line 80 in “~\CMSystem\Controllers\ AnnouncementController.cs” is on evidence of debugging.

4. **2:1 (60-69%) There is no evidence of bugs.**
   There is no bug in current using.

5. **Distinction (80%+) A marketable web application has been produced.**
   The project has proved to be a usable one.

Implementation (20%)

1. **Fail (35-39%) Minimal progress, student has no working code**
   The “CMSystem” project works well.

2. **Pass (40-49%) There is working code but the vast majority has been generated using MVC scaffolding, the student has only independently implemented a model definition.**
   Line145-161 of “~\CMSystem\Controllers\AnnouncementController.cs” is the evidence of independent implementation of delete confirming method.

3. **2:2 (50-59%) Evidence that the student has independently implemented a significant portion of the application.**
   Line 30-63, line 105-161 of “~\CMSystem\Controllers\EventController.cs ”, line 1-85 of “~\CMSystem\Views\Event\SingleEvent.cshtml”, line 137-196 of “~\CMSystem\Views\Event\Index.cshtml” is the evidence of independent implementation of creating new event in “Event” calendar and return the event information as a partial view when clicking the event on the calendar.

4. **2:1 (60-69%) The student has followed good practices both in terms of the MVC framework and the coding conventions for the languages used.**
   Line 33-38 in “~\CMSystem\Controllers\AnnouncementController.cs” is a method called “GetAnnouncement” which returns a list of announcements.

Line 41-40 using the lists from “GetAnnouncement” and post them to a partial view “_AnnouncementTable”, and return the partial view. And “_AnnouncementTable.cshtml” is an independent part in “~\CMSystem\Views\Announcement” and can be changed with other parts of index view no changes. These three parts is packed as objects and can be used in other methods.

5. **First (70%-79%) Each method the student has implemented is explained using comments and there is evidence that the student has used software tools (such as version control or unit testing) where appropriate**
   Line 31-32, and line 40 in “~\ CMSystem \ Controllers \ AnnouncementController.cs” is the evidence of using comments.
   Github is used as git repo, see folder .git.

Security (20%)

1. **Fail (35-39%) The web application contains one or more examples of a serious security flaw**
   There is no serious security flaw of the application.

2. **Pass (40-49%) There is no obvious cross site request forgery vulnerability.**
   in line 75 of “~\CMSystem\Controllers\AnnouncementController.cs”, “[ValidateAntiForgeryToken]” is added to Creating method to prevent CSRF attacking.
   Line 19 of “~\CMSystem\Views\Announcement\Index.cshtml” uses “@Html.AntiForgeryToken()” to prevent CSRF attacking

3. **2:2 (50-59%) There is no obvious cross site scripting vulnerability or over posting risk.**
   Line 28 of “~\CMSystem\Views\Event_SingleEvent.cshtml” contains “@(Html.AttributeEncode()” used to prevent CSS attacking.

4. **2:1 (60-69%) Steps have been taken to prevent cookie theft and SSL is used where appropriate**
   Line 24 of “~\ CMSystem \Web.config” contains “<httpCookies domain="" httpOnlyCookies="true"\>” to prevent cookie theft.

5. **First (70%-79%) Steps have been taken to mitigate the risks posed by error reporting**
   “~\CMSystem\Controllers\ErrorPageController.cs” and line 21-55 of “~\ CMSystem\Global.asax.cs” shows the evidence of mitigate the risks posed by error reporting.

6. **Distinction (80%+) As below with evidence of the consideration of security risks not formally taught during lectures.**
   Line 78 of "~\ CMSystem \ Controllers \ AnnouncementController.cs" using [Bind (Include="AnnouncementId, AnnouncementTitle, AnnouncementContent, AnnoucingTime, ExpiryTime")] to prevent over posting.

Working with Models (10%)

1. **Fail (35-39%) No database is used**
   Entity framework code first is used, so the database is automatically generated.

2. **Pass (40-49%) A database is created from a model definition using the code first convention.**
   The model files in "~\CMSystem\Models" show examples of models. After the definition (creating the models), and migration, the database is generated.

3. **2:2 (50-59%) Database migrations are enabled and the database is seeded using this package.**
   "~\CMSystem\Migrations" contains all migration files of database migration and updating.

4. **2:1 (60-69%) View specific models and data annotations are used effectively.**
   Line 16 of "~\CMSystem\Models\Announcement.cs" shows data annotations for "announcementTitle".
   The "~\CMSystem\Views\Annoucement\Index.cshtml" is a view based on Annoucement model.

5. **First (70%-79%) Client side validation is implemented.**
   Line 15 of "~\CMSystem\Models\Announcement.cs" using "[Required]" to implement client side validation.

6. **Distinction (80%+) Ajax is used to enhance the user experience with respect to interacting with the data.**
   "~\CMSystem\Scripts\Custom_Scripts\BuildTable.js" is an example of using ajax to get an partial view data and insert them in the html.

Identity (10%)

1. **Fail (35-39%) User accounts are not used in any way in application.**
   Line 39-52 of "~\CMSystem\Migrations\Configuration.cs" shows the user accounts creation.

2. **Pass (40-49%) Users can log in to the web application but it serves no purpose.**
   There are two roles of users: Customer and Member, as showing in line 26-37 of "~\CMSystem\Migrations\Configuration.cs". From line76-77 of "~\CMSystem\Controllers\ AnnouncementController.cs", only "member can create announcement".

3. **2:2 (50-59%) Different users have different levels of authorization.**
   From line76-77 of "~\CMSystem\Controllers\ AnnouncementController.cs", only "member can create announcement".

4. **2:1 (60-69%) A claim based authorisation system is implemented using role names. Good security practices have been followed with respect to authorisation.**
   Line 19 of "~\CMSystem\Controllers\ AnnouncementController.cs" shows a claim based authorization function using role names.

5. **First (70%-79%) A claim based authorisation system is implemented using the claims system in ASP.NET Identity.**
   Line 39-63 of "~\CMSystem\Migrations\Configuation.cs", and Line 19 of "~\CMSystem\Controllers\ AnnouncementController.cs" serves as an example of claim based authorization system.

Functionality (10%)

1. **Fail (35-39%) The application is a non-interactive web site**
   Users can "add, delete" announcement, comment, event.

2. **Pass (40-49%) The application only has the functionality stated in the basic requirements**
   "Memeber" users can "add, delete" announcement, event.
   "Customer" users can "add, delete" comments, and they can "sign up" events.
   After clicking events in "Event" view, there is progress bar showing how many people have signed up the events according.

3. **2:2 (50-59%) The web application has at least one additional working feature.**
   In line 33-37 of "~\CMSystem\Controllers\ AnnouncementController.cs", only those announcements of which announcing time between setting announcing time and expiry time can show on the screen.

4. **2:1 (60-69%) The web application uses JavaScript to improve the user experience.**
   Line 122-210 in "~\CMSystem\Views\Events\Index.cshtml" shows the

coding using JavaScript to improve the user experience.

5. **First (70%-79%) There is evidence of using an external web API or a library which is not included within ASP.NET.**
   Line 24 of "~\CMSystem\App_Start\BundleConfig.cs" loads "moment" library.
   Line 27 of "~\CMSystem\App_Start\BundleConfig.cs" loads "fullcalendar" library.
   Line 137-194 of "~\CMSystem\Views\Events\Index.cshtml" using functions of "fullcalendar" library.

6. **Distinction (80%+) As below but the features show evidence of creativity and independent research.**
   Line 4-34 in "~\CMSystem\Views\Events_SingleEvent.cshtml" shows a feature of progress bar which tells users how many people have signed up the events. According to "capacity" number set by event creator, if people signing up the events are full. The progress bar will turn into red and tell people the people are full. If there is no people signing up, the progress bar will not show on the screen and only sentence "Welcome to sign up" with the event details appear on the screen.

Presentation (10%)

1. **Pass (40-49%) The web application is unchanged from the original template**
   The style of application has changed.

2. **2:2 (50-59%) The web application visual style has changed from the original template**
   Line 1089-1095 of "~\CMSystem\Content\bootstrap.css" shows the changes for body content.

3. **2:1 (60-69%) All styling is done using CSS**
   All styling of Home and Annoucement view are done using "~\CMSystem\Content\bootstrap.css", and most part of Event view is done using "~\CMSystem\Content\fullcalendar.css".

4. **First (70%-79%) The web application layout changes appropriately as the browser window is resized.**
   Line 527 of "~\CMSystem\Content\fullcalendar.css" using "position: relative;" for resize handle.

5. **Distinction (80%+) The web application is unrecognisable as a ASP.NET project**
   There is no ASP.NET words from original template showing on the screen.