```
1   //all of the user-defined Angular directives
2   angular.module('ClassMaticApp.directives', []).
3   directive('loginform', function() {
4       //directive to display the login form
5       //bound to the list of classes, terms, years
6       //also bound to the login model and function to handle authentication
7       return {
8           scope: false,
9           replace: true,
10          restrict: 'E',
11          template: '' +
12              '<form ng-submit="sendLogin()">' +
13                  '<table style="background-color: #66C166; color: white; padding:
                    30px; padding-left: 100px; padding-right: 100px; border-radius:
                    10px; border: 1px solid white" cellpadding="20" cellspacing="0"
                    border="0" align="center">' +
14                      '<tr>' +
15                          '<td colspan="2">' +
16                              '<h2>ClassMatic Login</h2>' +
17                              '<span ng-if="error" style="color:
                                #FFCCCC"><br>Incorrect login credentials, or you are not
                                registered for courses</span>' +
18                              '<span ng-show="loading"><img src="loading.gif"
                                style="max-width: 40px"></span>' +
19                          '</td>' +
20                      '</tr>' +
21                      '<tr>' +
22                          '<td>Banner ID:</td>' +
23                          '<td>' +
24                              '<input class="loginctrl" type="text"
                                ng-model="login.sid">' +
25                          '</td>' +
26                      '</tr>' +
27                      '<tr>' +
28                          '<td>PIN:</td>' +
29                          '<td>' +
30                              '<input class="loginctrl" type="password"
                                ng-model="login.pin">' +
31                          '</td>' +
32                      '</tr>' +
33                      '<tr>' +
34                          '<td>Term:</td>' +
35                          '<td>' +
36                              '<select ng-model="login.term" ng-options="term for term
                                in terms" ng-init="buildTerms()" style="float: left;
                                width: 45%"><option ng-if="false"></option></select>' +
37                              '<select ng-model="login.year" ng-options="year for year
                                in years" ng-init="buildYears()" style="float: right;
                                width: 45%"><option ng-if="false"></option></select>' +
38                          '</td>' +
39                      '</tr>' +
40                      '<tr>' +
41                          '<td colspan="2">' +
42                              '<input ng-show="false" class="button" type="submit"
                                value="Submit">' +
43                          '</td>' +
44                      '</tr>' +
```

```
45                  '</table>' +
46              '</form>',
47          controller: function ($scope) {
48          }
49      };
50  }).
51  directive('viewer', function ($location) {
52      //directive for inline file previewing
53      //has the capability to preview images, video, audio, pdfs, and text-based documents
54      //uses the Google viewer for documents, browser functionality for all others
55      return {
56          restrict: 'E',
57          scope: false,
58          replace: true,
59          link: function (scope, element, attrs) {
60              //builds a link to the server where this HTML is currently being served from, in the same format expected for downloading files
61              var link = $location.protocol() + "://" + $location.host() + ':' + $location.port() + '/download?active=' + attrs.active + '&hash=' + attrs.hash;
62              //relies on the onerror events of individual embedded elements to remove itself from the DOM
63              //successful loading removes the other embedded elements from the DOM
64              element.html('<div style="background-color: #F8F8F8; border: 1px solid #D8D8D8; padding: 20px; text-align: center; border-radius: 20px"><div style="padding: 0px; line-height: 0px; display: none">' +
65              '<video controls preload = "auto" style="max-height: 500px; max-width: 500px" src="' + link + '" onerror="if ($(this).siblings().length == 1) { $(this).parent().css(\'display\', \'\'); } $(this).remove();" oncanplay="if (this.duration > 0) { $(this).siblings().remove(); $(this).parent().css(\'display\', \'\'); } else { if ($(this).siblings().length == 1) { $(this).parent().css(\'display\', \'\'); } $(this).remove(); }"></video>' +
66              '<img style="max-height: 400px; max-width: 400px" src="' + link + '" onerror="if ($(this).siblings().length == 1) { $(this).parent().css(\'display\', \'\'); } $(this).remove();" onload="$(this).siblings().remove(); $(this).parent().css(\'display\', \'\')">' +
67              '<iframe src="https://docs.google.com/gview?url=' + encodeURIComponent(link) + '&embedded=true" style="width:600px; height:500px;" frameborder="0" onload="if ($(this).siblings().length == 0) { $(this).parent().css(\'display\', \'\'); }"></iframe>' +
68              '</div></div>');
69          }
70      };
71  }).
72  directive('dragAndDrop', function($rootScope) {
73      //directive for dragging and dropping onto a container
74      return {
75          scope: false,
76          restrict: 'A',
77          link: function($scope, elem, attr) {
78              //bind to dragenter to apply green highlighting to folder titles
79              elem.bind('dragenter', function(e) {
80                  //don't let this event pass on to the default handlers
81                  e.stopPropagation();
```

```
82                  e.preventDefault();
83                  if ($scope.path && !$scope.student) {
84                      elem.filter('.titlebar').css('background-color', '#66C166');
85                  }
86              });
87              //bind to dragleave to remove green highlighting on folder titles
88              elem.bind('dragleave', function(e) {
89                  //don't let this event pass on to the default handlers
90                  e.stopPropagation();
91                  e.preventDefault();
92                  if ($scope.path && !$scope.student) {
93                      elem.filter('.titlebar').css('background-color', '#F1F1F1');
94                  }
95              });
96              //bind to dragover to provide a secondary mechanism for applying green
                highlighting on folder titles
97              elem.bind('dragover', function(e) {
98                  //don't let this event pass on to the default handlers
99                  e.stopPropagation();
100                 e.preventDefault();
101                 if ($scope.path && !$scope.student) {
102                     elem.filter('.titlebar').css('background-color', '#66C166');
103                 }
104             });
105
106             //auxiliary function to dynamically create a move link and click it for
                when a drag-drop action occurs
107             var moveFile = function (elem, source) {
108                 var hash = source.split("/")[source.split("/").length - 1];
109                 var destination;
110                 if ($scope.path && $scope.path !== "") {
111                     if (!hash) {
112                         destination = $scope.path + source.split("/")[source.split(
                            "/").length - 2] + "/";
113                     } else {
114                         destination = $scope.path + hash;
115                     }
116                 } else {
117                     if (!hash) {
118                         destination = source.split("/")[source.split("/").length - 2]
                             + "/";
119                     } else {
120                         destination = hash;
121                     }
122                 }
123                 if (destination.substring(0, source.length) !== source) {
124                     var link = document.createElement('a');
125                     link.href = "move?active=" + angular.element($(elem)).scope().
                        activeClass + "&source=" + source + "&destination=" + destination;
126                     document.body.appendChild(link); //must be in DOM to work in
                        Firefox
127                     link.target = "hidden-iframe";
128                     link.click();
129                     document.body.removeChild(link);
130                 }
131             }
132
```

```
133             //bind to the drop action
134             elem.bind('drop', function(e) {
135                 //don't let this event pass on to the default handlers
136                 e.stopPropagation();
137                 e.preventDefault();
138                 if ($scope.path) {
139                     elem.filter('.titlebar').css('background-color', '#F1F1F1');
140                 }
141                 //check to make sure we're not in student mode
142                 if (!$scope.student) {
143                     var go = true;
144                     //if there is text data associated with the drop event, then
                        interpret as a move and return
145                     if (e.originalEvent.dataTransfer.getData("text") && e.
                        originalEvent.dataTransfer.getData("text") !== undefined) {
146                         moveFile(elem, e.originalEvent.dataTransfer.getData("text"));
147                         return;
148                     }
149                     //if there are already files that have been dropped onto the
                        interface, then alert the user they will be overwritten
150                     if ($rootScope.fields.droppedFiles.length > 0) {
151                         go = confirm("This will overwrite your previously dropped
                            files.");
152                     }
153                     //if the user accepted the overwrite or there was no conflict,
                        parse the files and reflect it in the interface
154                     if (go) {
155                         $rootScope.$apply(function () {
156                             //clear the old files that had been dropped
157                             $rootScope.fields.droppedFiles = [];
158                             var dropped = e.originalEvent.dataTransfer.files; //no
                                originalEvent if jQuery script is included after angular
159                             for (var i in dropped) {
160                                 //if the file has a type or a size that isn't a
                                    multiple of 4096 or is larger than 4096*3, then it
                                    is a file and not a folder
161                                 //we don't want to handle folders as that
                                    functionality is not available in any browsers
                                    outside of Chrome
162                                 if (dropped[i].type || (dropped[i].size && (dropped[i
                                    ].size % 4096 !== 0 || dropped[i].size / 4096 > 3))) {
163                                     $rootScope.fields.droppedFiles.push(dropped[i]);
164                                 }
165                             }
166                             if ($rootScope.fields.droppedFiles.length > 0) {
167                                 //if we're not in the upload pane already then open
                                    it and overwrite the folder
168                                 if ($rootScope.fields.upload == false) {
169                                     $rootScope.fields.upload = true;
170                                     if ($scope.path && $scope.path !== "") {
171                                         $rootScope.fields.folderName = $scope.path.
                                            substring(0, $scope.path.length - 1);
172                                     } else {
173                                         $rootScope.fields.folderName = "/";
174                                     }
175                                 } else {
176                                     //if the upload pane is already open and we
```

```
                                    drag-dropped onto no path, keep what was
                                    there... but if we dropped on a custom folder,
                                    take that instead
177                                 if ($scope.path && $scope.path !== "") {
178                                     $rootScope.fields.folderName = $scope.path.
                                        substring(0, $scope.path.length - 1);
179                                 }
180                             }
181                         }
182                     });
183                 }
184             }
185         });
186     }
187 };
188 }).
189 directive('uploadForm', function($rootScope) {
190     //directive that controls the upload form
191     return {
192         scope: false,
193         restrict: 'A',
194         template: '' +
195             '<table ng-style="{backgroundColor : (fields.loading && \'#FF9999\') ||
            \'transparent\'}" style="width: 330px; padding: 15px; border: 1px solid
            #909090" cellpadding="10" cellspacing="0" border="0" align="center">' +
196                 '<tr>' +
197                     '<td valign="top">Folder:</td>' +
198                     '<td>' +
199                         '<span ng-bind="fields.folderName | humanreadable"></span>' +
200                         '<input type="text" ng-show="false" id="folder"
                            ng-model="fields.folderName" name="folder">' +
201                     '</td>' +
202                 '</tr>' +
203                 '<tr>' +
204                     '<td valign="top">File(s):</td>' +
205                     '<td>' +
206                     //ng-show="!(fields.droppedFiles.length > 0) || this.value !==
                        \'\'"
207                         '<input style="width: 100%" id="file"
                            ng-disabled="fields.loading" type="file"
                            name="{{activeClass}}"
                            ng-required="!(fields.droppedFiles.length > 0)"
                            multiple="multiple">' +
208                         '<p><span ng-if="fields.droppedFiles.length > 0"
                            style="color: red"><b>+</b></span></p>' +
209                         '<div style="width: 100%"
                            ng-show="fields.droppedFiles.length"><span style="color:
                            red"><b>{{fields.droppedFiles.length}} file(s)
                            drag/dropped</b> <img style="float: right; max-height: 20px"
                            ng-src="x.png" ng-click="fields.droppedFiles =
                            []"></span></div>' +
210                     '</td>' +
211                 '</tr>' +
212                 '<tr>' +
213                     '<td valign="top" align="left" colspan="2">' +
214                         '<label><input ng-model="futureReveal"
                            ng-disabled="fields.loading" type="checkbox"></input> Future
```

```
                            Reveal</label><br />' +
215                         '<p>' +
216                             '<div ng-show="futureReveal"><span style="padding-right:
                                10px">Date: </span><input style="width: 170px"
                                ng-required="futureReveal" ng-model="revealTime"
                                id="datetimepicker" type="text"
                                name="reveal"></input><script
                                type="text/javascript">$("#datetimepicker").AnyTime_picker
                                ({' +
217                                 'format: \'%m/%e/%Y %h:%i:%s %p\',' +
218                                 'earliest: new Date(),' +
219                             '});</script></div>' +
220                         '</p>' +
221                     '</td>' +
222                 '</tr>' +
223                 '<tr>' +
224                     '<td colspan="2" align="center" style="padding-top: 20px">' +
225                         '<input style="width: 60%; height: 40px" type="submit"
                            ng-disabled="fields.loading" value="Submit"></input>' +
226                     '</td>' +
227                 '</tr>' +
228             '</table>',
229         link: function($scope, elem, attr) {
230             //when the form is submitted, take over that event
231             elem.bind('submit', function(e) {
232                 //$(elem).children('table').css('background-color', '#FF9999');
233                 e.preventDefault();
234                 $scope.$apply(function () {
235                     if ($scope.futureReveal) {
236                         $scope.revealTime = new Date($scope.revealTime).getTime();
237                     } else {
238                         $scope.revealTime = "";
239                     }
240                 });
241                 oData = new FormData(this);
242                 $scope.$apply(function () {
243                     $scope.futureReveal = false;
244                 });
245                 $rootScope.$apply(function () {
246                     $rootScope.fields.loading = true; //must be applied after the
                        form data is grabbed since disabling the file input keeps it
                        from actually uploading
247                 });
248                 for (var i = 0; i < $rootScope.fields.droppedFiles.length; i++)
249                 {
250                     //loop through all of the dropped files and append them to the
                        formdata
251                     oData.append($scope.activeClass, $rootScope.fields.droppedFiles[i
                        ]);
252                 }
253                 $rootScope.$apply(function () {
254                     //clear out the dropped files before uploading this batch so if
                        the user drops more on, those can be queued up for the next
                        round after this is finished
255                     $rootScope.fields.droppedFiles = [];
256                 });
257                 //we're sending the data to the server using XMLHttpRequest
```

```
258                 //uploading to the /upload endpoint
259                 var oReq = new XMLHttpRequest();
260                 oReq.open("post", "upload", true);
261                 oReq.onload = function(oEvent) {
262                     if (oReq.status == 200) {
263                         //$(elem).children('table').css('background-color',
                            'transparent');
264                         $rootScope.$apply(function () {
265                             $rootScope.fields.loading = false;
266                         });
267                         try {
268                             if ($rootScope.fields.droppedFiles.length == 0) {
269                                 $rootScope.$apply(function () {
270                                     $rootScope.fields.upload = false;
271                                 });
272                             }
273                             document.getElementById('file').value = '';
274                             $scope.revealTime = "";
275                         } catch (e) {}
276                     } else {
277                         alert("There was an error uploading your file");
278                     }
279                 };
280                 //send the data
281                 oReq.send(oData);
282 
283             });
284         }
285     };
286 }).
287 directive('folder', function(RecursionHelper) {
288     //directive that renders folders which may recursively contain other files and
        folders
289     return {
290         scope: false,
291         restrict: 'E',
292         template: '' +
293             '<ul drag-and-drop class="example-animate-container"
                ng-style="{borderLeft : ((object.files || object.folders) && path &&
                \'1px solid #909090\') || \'\'}">' +
294                 '<li draggable="true" ondragstart="(function(event){
                    event.dataTransfer.setData(\'text\',
                    angular.element($(event.target)).scope().path +
                    angular.element($(event.target)).scope().file.hash); })(event);"
                    class="animate-repeat" style="position: relative; border-bottom: 1px
                    solid #909090; text-align: right" ng-repeat="file in object.files |
                    filter: {name: searchterm} | orderBy: \'-date\' track by file.hash"
                    ng-if="!student || currentDate >= file.reveal">' +
295                     '<a draggable="false" style="float: left"
                        ng-href="download?active={{activeClass}}&hash={{path +
                        file.hash}}" ng-bind="file.name"></a><div style="display:
                        inline-block; padding-left: 20px"><div style="display:
                        inline-block; padding-right: 20px"><a draggable="false" href="#"
                        ng-click="preview = !preview">{{preview ? "close" :
                        "preview"}}</a></div><div ng-if="!student" style="display:
                        inline-block; padding-right: 20px"><a draggable="false"
                        ng-href="delete?active={{activeClass}}&hash={{path +
```

```
                        file.hash}}" target="hidden-iframe">delete</a></div><div
                        ng-if="currentDate >= file.reveal" style="display: inline-block;
                        width: 65px">{{file.date | date:"M/dd/yy"}}</div><div
                        ng-if="currentDate < file.reveal" style="display: inline-block;
                        width: 65px; color: red">{{file.reveal |
                        date:"M/dd/yy"}}</div><div ng-if="currentDate >= file.reveal"
                        style="display: inline-block; width: 80px">{{file.date |
                        date:"h:mma"}}</div><div ng-if="currentDate < file.reveal"
                        style="display: inline-block; width: 80px; color:
                        red">{{file.reveal | date:"h:mma"}}</div></div><br />' +
296                     '<div ng-if="preview" style="padding-bottom: 10px"><viewer
                        hash="{{path + file.hash}}"
                        active="{{activeClass}}"></viewer></div>' +
297                 '</li>' +
298                 //ng-if="folder.files || folder.folders"> will hide empty folders by
                    default
299                 '<li ng-class="divClass" class="animate-repeat noselect"
                    style="padding-left: 20px" ng-repeat="(name, folder) in
                    object.folders | folderfilter:searchterm"' +
300                     '<div class="noselect">' +
301                         '<p></p>' +
302                         '<div drag-and-drop draggable="true"
                            ondragstart="(function(event){
                            event.dataTransfer.setData(\'text\',
                            angular.element($(event.target)).scope().path); })(event);"
                            class="titlebar" ng-style="{minWidth : (280 + (14 *
                            path.split(\'/\')[path.split(\'/\').length - 2].length)) +
                            \'px\'}" style="background-color: #F1F1F1; border: solid 1px
                            #909090; border-radius: 10px" ng-click="foldershow =
                            !foldershow"><img draggable="false" src="folder.png"
                            style="max-height: 40px; padding-top: 5px; padding-right:
                            5px; padding-left: 10px"> <span style="vertical-align: top;
                            position: relative; top: 7px; left:
                            5px">{{name}}</span><span style="float: right"><button-group
                            style="position: relative; bottom: 3px; right:
                            10px"></button-group><img draggable="false"
                            ng-src="{{foldershow || searchterm ? \'expand.png\' :
                            \'collapse.png\'}}" style="padding-top: 5px; max-height:
                            35px"></span></div></div>' +
303                         '<folder ng-show="!foldershow || searchterm" data="folder"
                            path="name"></folder>' +
304                     '</div>' +
305                 '</li>' +
306             '</ul>',
307         compile: function(element) {
308             //pull in the RecursionHelper service to handle rendering recursive
                content
309             return RecursionHelper.compile(element, function(scope, iElement, iAttrs,
                 controller, transcludeFn){
310                 scope.$watch(iAttrs.data, function (value){
311                     if (value) {
312                         //watches the attributes on this directive specified by the
                            HTML, and whether those objects are changing and reapplies
                            it here
313                         scope.object = value;
314                     }
315                 });
```

```
316                 scope.$watch(iAttrs.path, function (value){
317                     //recursively at each level down inherits the path from the
                        previous scope and redefines it again at this scope with the new
                        value appended
318                     if (value && scope.path !== undefined) {
319                         scope.path += value + "/";
320                     } else {
321                         scope.path = "";
322                     }
323                 });
324             });
325         }
326     };
327 }).directive('buttonGroup', function($rootScope) {
328     //directive to specify a common group of buttons for administrative purposes
329     return {
330         scope: false,
331         restrict: 'E',
332         template: '' +
333             '<span ng-if="!student"><a draggable="false" class="button" href=""
                ng-click="uploadFolder($event)">Upload</a> <a draggable="false"
                class="button" target="hidden-iframe"
                ng-href="newfolder?active={{activeClass}}&path={{path + newname}}"
                ng-click="newFolder($event)"><img draggable="false" style="max-height:
                30px; position: relative; top: 9px" src="newfolder.png"></a> <a
                draggable="false" ng-if="path" class="button" target="hidden-iframe"
                ng-href="deletefolder?active={{activeClass}}&path={{deletepath}}"
                ng-click="deleteFolder($event)"><img draggable="false"
                style="max-height: 30px; position: relative; top: 9px"
                src="delete.png"></a></span>',
334         controller: function ($scope, $rootScope) {
335             //function to handle uploading to a folder; pops open the upload form in
                this context
336             $scope.uploadFolder = function (e) {
337                 e.stopPropagation();
338                 $rootScope.fields.upload = true;
339                 if ($scope.path && $scope.path !== "") {
340                     $rootScope.fields.folderName = $scope.path.substring(0, $scope.
                        path.length - 1);
341                 } else {
342                     $rootScope.fields.folderName = "/";
343                 }
344             }
345             //function to handle creating a new folder in the interface
346             $scope.newFolder = function (e) {
347                 e.stopPropagation();
348                 $scope.newname = prompt("Please enter a name for your new folder: ");
349                 if ($scope.newname) {
350                     //strips non alphanumeric characters from new folder names
351                     $scope.newname = $scope.newname.replace(/([^a-z 0-9]+)/gi, '');
352                     $rootScope.fields.upload = true;
353                     $rootScope.fields.folderName = $scope.path + $scope.newname;
354                 } else {
355                     e.preventDefault();
356                 }
357             }
358             //function to handle deleting a folder
```

```
359                 $scope.deleteFolder = function (e) {
360                     e.stopPropagation();
361                     //confirms with the user before actually deleting
362                     if (!confirm("This will delete the folder and all of its contents,
                        and cannot be undone.  Are you sure?")) {
363                         //if the user doesn't want to commit to deleting the folder, the
                            link is prevented from loading
364                         e.preventDefault();
365                     } else {
366                         if ($scope.path && $scope.path !== "") {
367                             //constructs a deletepath for the dynamic destination of the
                                ahref url to load into the hidden iframe to actually perform
                                the deletion
368                             $scope.deletepath = $scope.path.substring(0, $scope.path.
                                length - 1);
369                         } else {
370                             $scope.deletepath = "/";
371                         }
372                     }
373                 }
374             }
375         };
376     });
```