Course:	**INFO6145 JavaScript**
Professor:	**Darryl Bedford**
Project:	**Project #1 – Major League Baseball Form**
Due Date:	**Friday, June 21, 2024 11:30pm**
Submitting:	**Please see the last page for instructions**

How will my project be marked?

- This project counts for 15% of your final grade and will be evaluated using the following grid:

Marks Available	**What are the Marks Awarded For?**	**Mark Assigned**
2	Good coding style including proper indentation and use of variable and object naming conventions and suitable comments.	
3	Display page correctly and with good styling.	
2	Request and retrieve the first JSON string from the MLB site using selected values from the 3 pulldowns.	
2	Enable the user to select the properties for the “next” and “previous” games and properly display the team names for the currently selected game. This also requires the retrieval of the second JSON string from the MLB site, displaying the first “highlight name” for the currently selected game.	
2	Retrieve the second JSON string from the MLB site, displaying the first “highlight name” for the currently selected game each time the game selected changes for any reason.	
2	Enable the user to select the properties for the “next” and “previous” highlight object, properly displaying the first highlight name for the currently selected game.	
1	Allow the user to run the “MP4” video for the currently selected highlight.	
1	Proper zipped submission.	
15	Total	

Project Description

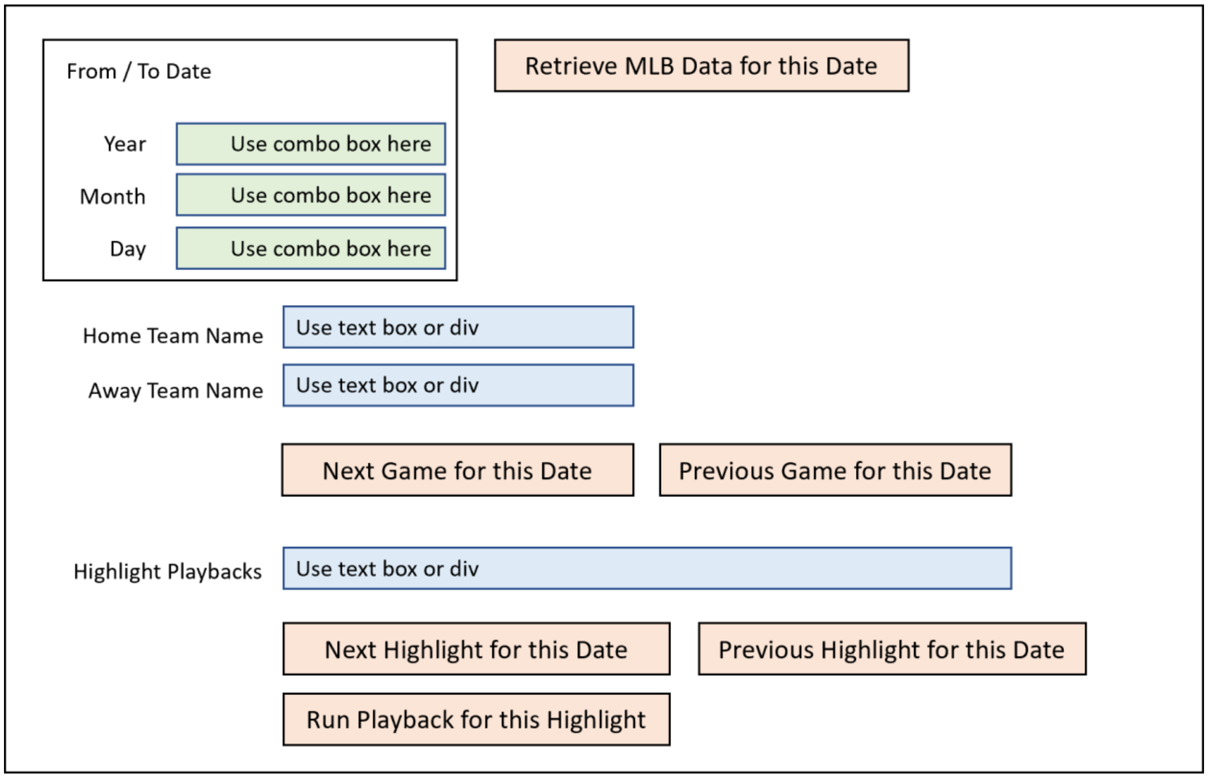
This project is based on the MLB Demo example that’s been provided to you recently. The demo page examples contain the JavaScript code required to retrieve data (JSON format) from the Major League Baseball site on the world wide web. For the purpose of this project, retrieval is based on a set of games for a specific day and requires a year, month and day to be specified. In addition, the MLBDemo2.html example shows how to use an anonymous function as the target for an asynchronous callback and should be the approach taken for the project solution.

The overall objectives for this project are:

1. Retrieve a set of baseball schedule data for **an entered date that has already passed**.
2. Provide the user with the opportunity to display the team names (home and away) for any of the games that have been played on the selected day using a pair of "next" and "previous" buttons.
3. Also, provide the user with the ability to display a single "highlight headline playback" from the set of headlines for a single game using a different pair of "next" and "previous" buttons.
4. Finally, allow the user to run the "mp4" video associated with the headline. This should be done within a new browser tab.

Specific Requirements:

Create a basic user interface that looks somewhat like the following. You should use your own colours and fonts to create something that's professionally attractive.



When the page loads, the user will need to select a year, month and day from 3 <select> pulldown lists:

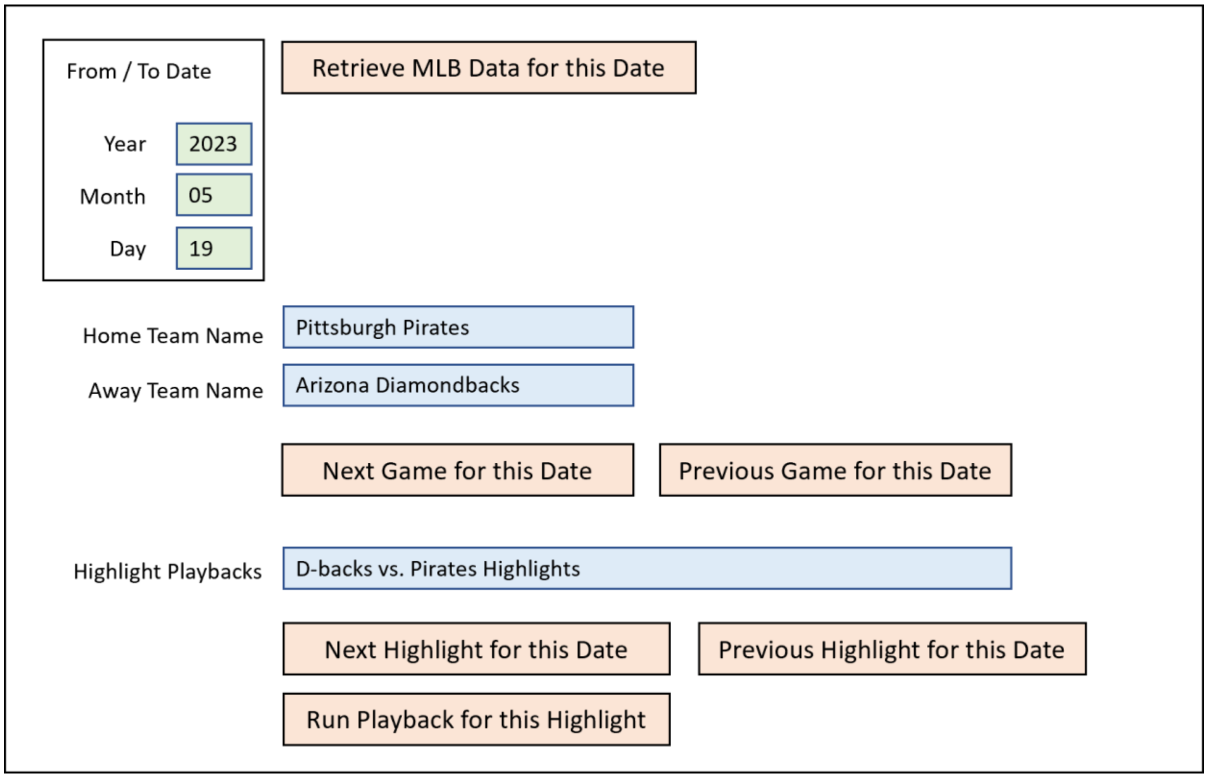
The Year pulldown should have 5 options: 2020, 2021, 2022, 2023, and 2024.

The Month pulldown should have 12 options: 01, 02, 03 … 12

The Day pulldown should have 31 options: 01, 02, 03 … 31

We won't worry about validating any of the selected dates.

When the user clicks the "Retrieve MLB Data for this Date" button, use an AJAX call to the MLB online site to retrieve the JSON schedule for the specified day.
Here's an example of what the previous page design would look like if we used May 19, 2023 for our input date.


From / To Date
Year
2023
Month
05
Day
19
Retrieve MLB Data for this Date
Home Team Name
Pittsburgh Pirates
Away Team Name
Arizona Diamondbacks
Next Game for this Date
Previous Game for this Date
Highlight Playbacks
D-backs vs. Pirates Highlights
Next Highlight for this Date
Previous Highlight for this Date
Run Playback for this Highlight

During our in-class session, we referenced the GitHub site which provides information about using the MLB API:

https://github.com/brianhaferkamp/mlbapidata

In order to succeed with this project, you'll need to make use of two of the URLs we worked with:

http://statsapi.mlb.com/api/v1/schedule/games/?sportId=1&startDate=2023-05-19&endDate=2023-05-19

https://statsapi.mlb.com/api/v1/game/[gamePk]/content

The first URL will allow you to retrieve JSON data which contains an array (among other things) where you'll find the team name information and the "gamePk" property. Once you have the "gamePk" property you can retrieve the detailed game information (including highlight videos, etc.) using the second "statsapi.mlb.com" url above. We don't have any reason to display the "gamePk" property on our web page, but it's needed behind the scenes.

Also, notice that the “gameID” property which appears in the GitHub site has been replaced with “gamePk”.

In order to get started, I would recommend that you create an HTML page that represents your visual design. Include a set of HTML controls along with empty event handlers.

Notice that your HTML <select> list controls don’t require event handlers because you can extract the year, month and day values within your “Retrieve MLB Data for this Date” button event handler. See the “ControlsReference.html” example page if you need to remember how to extract <select> values.

It’s not required that you duplicate perfectly the design shown above, but it is necessary to duplicate the functionality. Your design will need to make use of the MLB API in two stages:

Once you have your initial web page complete, try to embed the year, month and day into the “season schedule" url at the beginning of your JavaScript event handler for the “Retrieve MLB Data…” button. You’ll need to do some string processing using concatenation or the “replace” method to get the job done.

http://statsapi.mlb.com/api/v1/schedule/games/?sportId=1&startDate=2023-05-19&endDate=2023-05-19

Next, using the MLBDemo2.html example retrieve your first JSON string and use it to create a JavaScript object such as we did with “jsObject.”

Now you should be able to use the tools within the Chrome debugging environment to drill down on the information you need. In order to help everyone get started, I’ve provided some initial examples to demonstrate some basic ideas:

```
var gamePk = jsObject.dates[0].games[0].gamePk;
var homeTeam = jsObject.dates[0].games[0].teams.home.team.name;
```

One thing to keep in mind is that the dates[] array in our application will always have one element only because our “endDate” is the same as our “startDate.”

In contrast, the games[] array will typically have more than one entry because there will be multiple games played on a single day. As such, you’ll need to keep an array index for the “games” array which you can increment or decrement with your “Next Game…” and “Previous Game…” buttons.

Important:

Each time you change the game you're looking at, you'll need to automatically issue another AJAX GET request using the correct "gamePk" value and the "Individual Game API" url. The "[gamePk]" will have been replaced by something like "718118"

When you get back your "game" data, you need to drill down through "highlights.highlights" to the "items" array where you'll find the "headline" property. Finally, you'll need to drill down again from the "items[x]" level until you get to the "playbacks" array. Note that we're only ever interested in the first element of the "playbacks" array because it's the one with the "MP4" url.

Remember that it's the "items[]" array that you want to navigate through with "Next Highlight…" and "Previous Highlight…"

When you get all this to work, the last part is relatively easy, because you'll be given the full url to the "MP4" video you want to play and it can be launched from JavaScript with something like:

```
window.open(url, '_blank').focus();
```

Here's a stackoverflow link if you need more detail:

https://stackoverflow.com/questions/4907843/open-a-url-in-a-new-tab-and-not-a-new-window

*** End of Requirements ***

How should I submit my project?

Electronic Submission:

Submit your program files to the *Info6144 "Project 1"* electronic submission folder in *FanshaweOnline*. These files should be submitted as a single "zip" file containing your web application's complete website.

I strongly recommend that you test your own submission to ensure that nothing has been missed.

Professor's Note: Project Standards are Stricter

The standards we use for marking projects tend to be more strict than when we mark labs. Refer to the rubric and the policy references below.

Submit your project on time!

Project submissions must be made on time! Late projects will be subject to divisional policy on missed test and late projects. In accordance with this policy, no late projects will be accepted without prior notification being received by the instructor from the student.

Academic Integrity Policy

It is considered cheating to submit work done by another student or from another source. Helping another student cheat by sharing your work with them is also not tolerated. Students are encouraged to share ideas and to work together on practice exercises, but any code or documentation prepared for a project must be done by the individual student. Penalties for cheating or helping another student cheat may include being assigned zero on the project with even more severe penalties if you are caught cheating more than once. Just submit your own work and benefit from having made the effort on your own.