# Assignment: Single Page Architecture

In this assignment we will use the jQuery show and hide methods to create a simple application. This application will retrieve a dailyquote from an API and show it to a user. The user then has the option to share that quote with a friend via an SMS text message.

Make note of the following URLs.

|  |  |
| --- | --- |
| **URL** | **Notes** |
| <https://misdemo.temple.edu/dailyquote/> | GET from this URL to get an JSON response containing a fun and/or inspirational quote. Each GET requests results in a new/different quote.  **IMPORTANT:** In this specific assignment is important that you put the "/" at the end of the URL as shown here. |
| <https://textbelt.com/text> | POST to this URL to send a text message. You must pass “key” and “message” and “phone” as URL encoded data. This API feature returns confirmation data in a JSON array.  **IMPORTANT:** This URL should ***not*** end in a forward slash. |

Recall the basic structure of the jQuery $.getJSON() method.

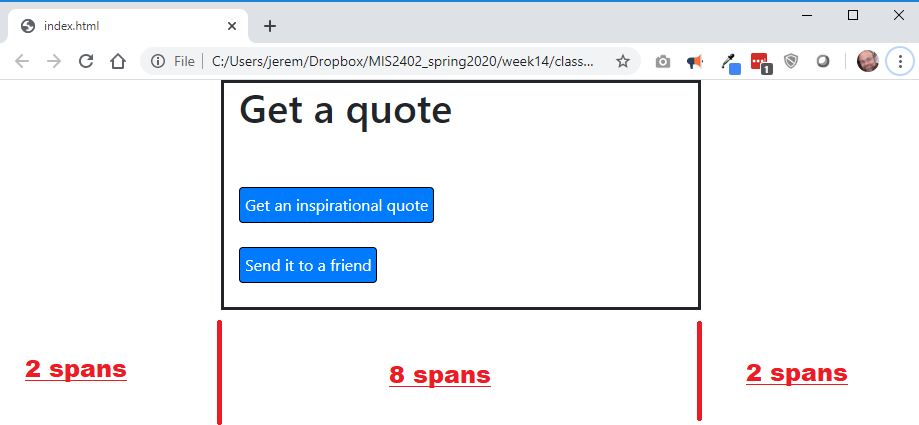
$.getJSON( *someurl* , *serialized\_data* , function(data){ } );

Recall the basic structure of the jQuery $.post() method.

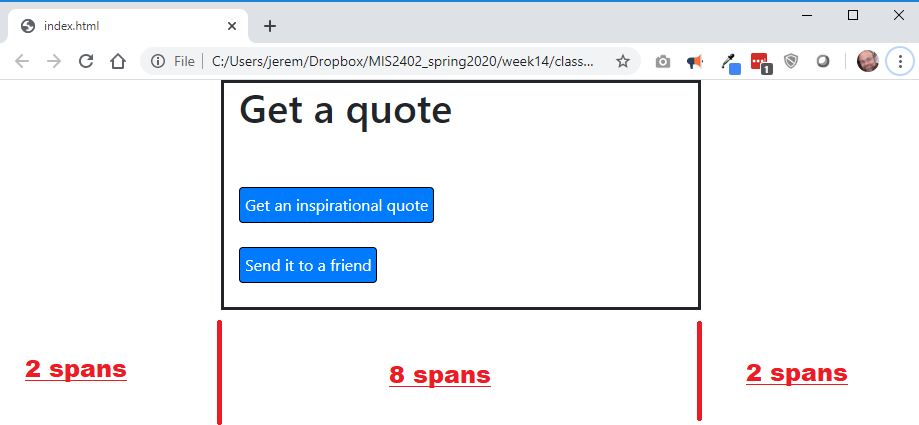
$.post( *someurl* , *serialized\_data* , function(data){ } );

## Instructions

1. Start with demo\_quotes.zip. Set up your project in VS code.
2. Notice that the HTML portion of this page has a Bootstrap container div, a Bootstrap row div, and several Bootstrap column div tags. Specifically, there is:
   1. A 2 span column on the left
   2. A 2-span column on the right
   3. In between the left and right columns there are two 8-span div tags identified by “div-quote” and “div-sms”. For the sake of clarity, a CSS style has been added to these div tags to give them a visible border.
3. Down in the <script> tag there are show and hide commands that are commented out. Uncomment them and see what happens.
4. You should now see an image like what is show below:



1. Note that the width of each portion of the page is measured in bootstrap “spans”. The full width of the page equals 12 “spans”. The red text added to the following screenshot illustrates this.



1. By using hide and show you can control which of the 8-span div tags is displayed in response to user driven events (like button clicks.)

In the <script> tag, edit the click event handlers for # btn\_show\_form and # btn\_show\_quote to show / hide the 8-span div tags. This gives the illusion of moving from “page” to “page”. But really, everything is in a **single** page, index.html. Before you move on to the next step, satisfy yourself that this is working. Test it!

1. Now complete the supporting function get\_the\_quote(). This function should make an GET request, using getJSON, against <https://misdemo.temple.edu/dailyquote>

* Be sure to console.log the data you get back from the API.
* Looking at the data you get back, you should be able to put the text of the quote into two places.
  + The div tag “quote”
  + The input tag “message”

**CONTINUED…**

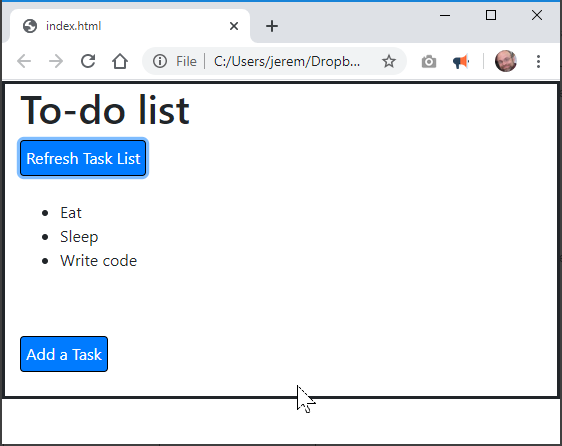
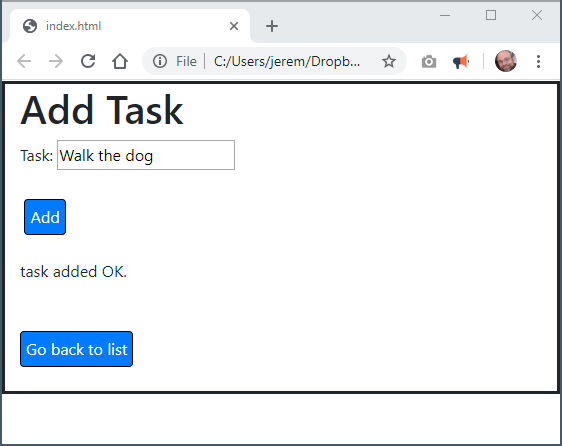
1. Now complete the supporting function send\_the\_text(). This function should serialize the form that holds the text message tags, and send it to textbelt.com
2. Upload your work. Test it.

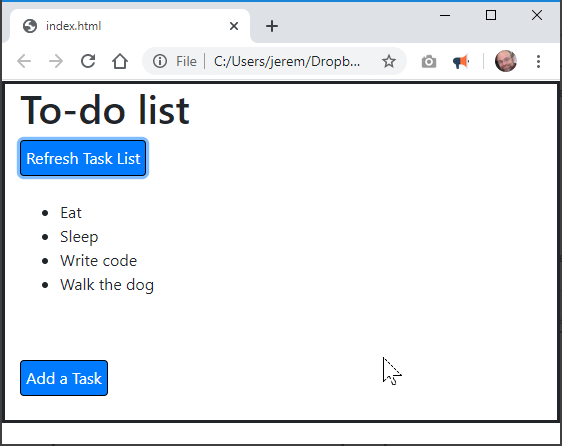
## On your own

1. Start with assignment21\_spa.zip. Set up your project in VS code.
2. Find the hidden input tag named task\_owner. Change the value of that from “Eustace” to your own name. That way, your “to do” list will be different from everyone else’s.
3. Uncomment the show / hide jQuery commands that run when the index.html page loads.
4. Create click event handlers to allow the user to show and hide the div tags that hold useful content.
5. Create add\_task() and get\_task() functions using $.post() and $.getJSON(). These are the API URLS you will need to use:  
     
   https://misdemo.temple.edu/todo/addtask  
   https://misdemo.temple.edu/todo/tasks

In this specific assignment use of **https:**// is advisable.

When you are done, you should have a working “to do” list.



## How will this assignment be graded?

## Only the “On Your Own” portion of the assignment will be graded!!!

|  |  |
| --- | --- |
| Item | Point Value |
| All files / folders uploaded correctly | 10 |
| Successfully implemented SPA | 30 |
| Wrote the code necessary to POST data to the addtask endpoint. Works. | 20 |
| When a task is added, a confirmation message in the <div> tag is provided. | 20 |
| Wrote the code necessary to GET data from the tasks endpoint. Looped through the data returned by the API and appended each task to the existing unordered list. | 20 |