(All your JS code should be **unobtrusive**)

Problem 1

1. Write a JavaScript function that generates a temporary token using, as a seed, your AUBNet account (e.g. sr46) as follows:
    a. You convert each alphabet letter to its ASCII code and concatenate them together i.e. "sr" becomes "115114"
    b. Convert the numerical part to Hexadecimal i.e. "46" becomes "2E" to which you prepend "0x" to get "0x2E".
    c. Concatenate (a) and (b) and return the result e.g. "1151140x2E"

2. Create an HTML form containing a textbox whose id is "*netaccount*" where the user enters his/her AUBNet account and another one whose id is "*token*" in which you show the generated token.
3. Once the textbox "netaccount" loses focus, you should automatically initiate an asynchronous AJAX request, using XMLHTTPRequest or Fetch(), to check whether the generated token already exists in a text file called "*tokens.txt*"[1]. For this purpose, you need to create this file manually, place it in the same folder as your solution, and write in it some randomly created tokens. The file contains along with each token, the name of its owner. For example, the file could contain the following:

    Saeed Raheel: 1151140x2E
    Haidar Safa: 1041150x21
    …

4. If the token is not found, you should then show a green tick mark next to the textbox "token".
5. If the token is found (i.e. it is not available), you should show a red "X" mark next to the textbox "*token*" and change its background to red. You should also dynamically create a label reading "Conflicts with: " and a disabled textbox containing the name of the conflicting person and place them right underneath the textbox "*token*". These should be removed and everything set back to normal when a newly generated token is available again.
6. On the other hand, when the token is a not available (i.e. it is found in the text file), a button should be dynamically added to the form that reads "Help me!" that once pressed proposes an alternative token according to the following. Mind that the user has to repeatedly click the button to go from one of the below attempts to the other:
    a. Attempt 1: permutate his/her net account's alphabet characters and generate a token. For example, you should, in the case of my netaccount, generate a token from the value "rs46" and then check its availability using AJAX or Fetch() again. If it is not, notify the user about it as you did in (5) and wait until he/she clicks again.
    b. Attempt 2: permutate the numeric part of the net account i.e. "sr64", generate a token, and check for its availability using AJAX or Fetch(). If it is not, notify the user about it as you did in (5) and wait until he/she clicks the button again.
    c. Attempt 3: revert back to the original netaccount, append to it a numeric index (i.e. "sr46-1"), generate a token, and then check its availability and notify accordingly. If

it's not, wait for a new click, increment the numeric index, check, and then notify. The user can keep clicking the button as many times as needed until an available token is found. Every time the user clicks the button, you should show between parentheses the current numeric index you're considering e.g. "Help me! (1)" then if he/she clicks again "Help me! (2)", etc.

Problem 2

A) In this exercise, you'll use JavaScript to parse a JSON object stored locally within your HTML page as follows. The objective is to convert the content of this file into a DOM similar to the one shown on the right:

The JSON object contains an array of YouTube clips for Fayrouz (you may use the content of the json file attached with this hw). Each clip should be represented by its ID, its Title, and its Author. When the user clicks on any clip, a player similar to the one below should be shown. On top of the player, you should display the name of the clip and its writer.

Use the "youtube.js" file to help you generate the player. You will need to iterate over the clips in the array and render each one into the page as a list item.

Make sure that you use your browser developer tools to make debugging easier while working on this. Check for errors, and use console.log() to figure out how far your code makes it, and what the values of your variables are along the way.

B) Rather than using the array that is local to your HTML page, you need now to get the list of clips from an external JSON file using AJAX/Fetch()and turn it into same DOM as described in part (A).

Problem 3

Assume the following XML document:

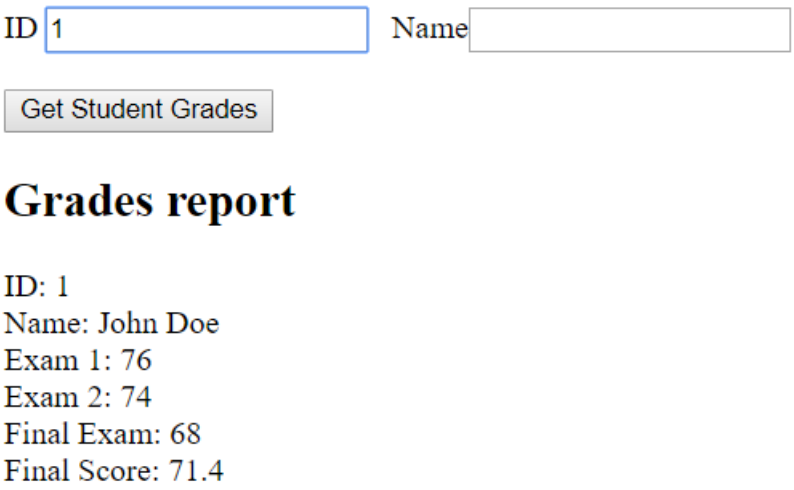
```
<?xml version="1.0"?>

<students>
    <student id="1">
        <name>John Doe</name>
        <exam1>76</exam1>
        <exam2>74</exam2>
        <finalexam>68</finalexam>
    </student>
    <student id="2">
        <name>Jane Doe</name>
        <exam1>81</exam1>
        <exam2>82</exam2>
        <finalexam>83</finalexam>
    </student>
    …
</students>
```

You should create an HTML page that allows the user to enter either the ID of a student or his/her name. Upon pressing a button "Get Student Grades", the request is sent via AJAX to XML file above. The result of your request should show on the HTML page the detailed info about the student in question and his/her final average. To compute the final average, you should create and get the weights of the exams from a JSON file named "weights.json" whose structure is as follows:

{exam1: 0.2, exam2: 0.3, finalexame: 0.5}

And then do the math as shown in the images below. The results should go into the div whose id is "divresults".

Run 1



Run 2

ID Name

Get Student Grades

Grades report

ID: 2
Name: Jane Doe
Exam 1: 81
Exam 2: 82
Final Exam: 83
Final Score: 82.3

You will submit via Moodle a compressed file called **AUBnetAccount-hw5.zip/rar**. The compressed file should contain three subfolders named **p1**, **p2**, and **p3**. Put in these folders all of the files related to your solution (HTML, CSS, JS, images, etc.).

Please do not place a solution to this assignment online on a publicly accessible web site. Doing so is considered a violation of the academic integrity policy.