**ASSIGNMENT I**

**Objective:**

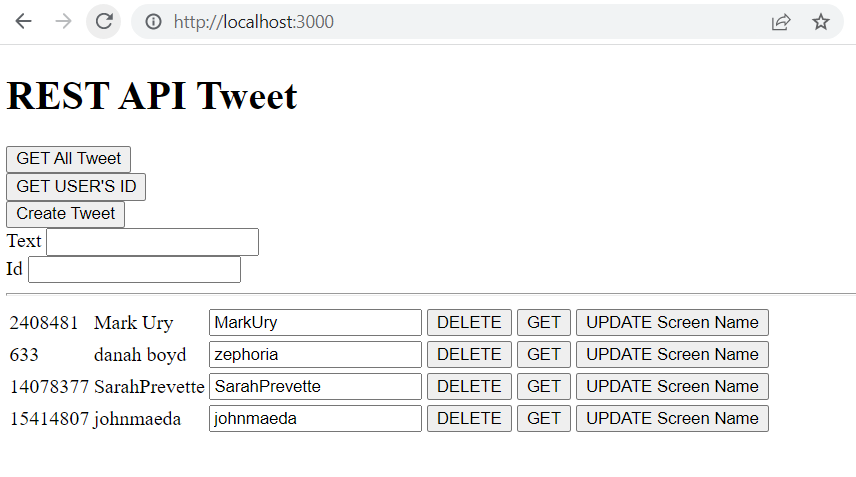
Analyze a simple JSON file favs.json by using RESTful API. The JOSN file contain the tweet and their information include time created text, user’s id, name and screen name. We need to write a RESTfull API to implement the following:

* Get all tweets (created time(“created\_at”), and tweet text (“text”)) in the file
* Get all users’ ID
* Get the details of tweet (text, created time), given ID
* Updated a screen\_name (given name and new screen\_name)
* Delete a tweet (given ID)

The system will consist of two main components: the front end, which is index.html file and the back end, which are server.js and script.js files implementing the REST APIs. The webpage uses local host port 3000. Its display input text fields and buttons that trigger the proper HTTP which are GET/PUSH/PUT/DELETE, request on the server.js then display the retrieve data.

**Server.js:**

* Using express to implement the HTTP request such as GET/PUSH/PUT/DELETE.
* Using readFile() function to read given favs.son file the store it in fileTweet array to implement.
* app.get(‘/fileTweet’, function()): get all tweet information include created time and text.
* app.get(‘/fileTweet/:id’, function()): get tweet with given id.
* app.post(): Get the input from front end to create a new tweet. The inputs are text, and id
* App.put(): Updated the screen\_name of user given the name, the information won’t change when webpage is refreshed.
* App.delete(): delete the tweet given user’s ID.

**Index.html:**

Create and design the webpage include text field, input field and button to trigger the HTTPs from backend. Above is how the webpage look like:

* GET All Tweet button: display all tweet include their text and time created.
* GET USER’S ID button: display all user’s identity as picture above.
* Create Tweet: get input from the Text, and Id input field to create a new tweet.
* DELETE button: delete a tweet given the user’s id at first column.
* GET button: get the tweet information given the user’s id at the first column.
* UPDATE Screen Name button: get a new screen name in text field then update it. Once updated, the information won’t change if webpage is refreshed.

**Script.js:**

* Using ajax to communicate the object between back end (server.js) and front end (index.html).
* ShowID() and ShowTweet(): using tbody.append() to show information following the column
* ShowID(): create buttons at the end of row for PUT and DELETE, assign their class for implementing.
* CREATE/POST: get the input from client, convert it to json string to implement at back end.
* PUT: allocate the tweet that change the screen name by find the corresponding user’s name by using closest() and find() function. Send the input and id to back end for implementation. Call get user’s id button at the end to show new name is updated.
* DELETE: allocate the tweet that delete button is clicked by find the corresponding user’s id by using closest() and find() function. Send the input and id to back end for implementation. Call get user’s id button at the end to show tweet and id have delete.