Project 1, Week 2 Instructions and Rubric

The entire first project upon completion will be a simple to-do application where users can add and remove a to-do and upon navigating away from the page their to-dos will be saved and restored.

This week's goal is to, with your team, create the back-end portion for your app and connect it to your front-end.

The back end must consist of the following requirements:

- Item Creation: **20pts**
- Fetch Items: **20pts**
- Update Item: **20pts**
- Delete Item: **20pts**
- Front end integration: **20pts**

1) **Item Creation**

   This back-end route must create a new item in your todo list on your server side data structure using the POST method.

   Example:

   <u>client request:</u>

   POST /todo/create

   { content: "eggs" }

   <u>server response:</u>

   ```
   {
           result: "success"
   }
   ```

2) Fetch List

This route must return all items in the list stored on the server side via the GET method.

Example:

client request:

```
GET /todo/read
```

server response:

```
{
        result: “success”,
        data: [
                “eggs”,
                “milk”,
                “cheese”
        ]
}
```

3) Item Update

This route must update an item on the list on the server side via the PUT method.

Example:

client request:

```
PUT /todo/update
{
        item_number: 3,
        content: “spaghetti”
}
```

server response:

```
{
        result: “success”
}
```

4) Item Delete

This route must delete an item from the list on the server via the DELETE method.

Example:

client request:

```
DELETE /todo/delete
{
	item: 2
}
```

server response:

```
{
	result: “success”
}
```

5) Front End Integration

The “/” route should return your html page.

Now connect your front end user interface to the back end by making HTTP requests from JavaScript. You should have buttons and input fields that allow all of the actions implemented above. The changes you make on the back end must also be reflected on the front end. This means that you should update your page after confirming that the request you made was successful.

Things to watch out for:

- Points will be deducted if any endpoints returns a 500 (server error) so be sure to test for edge cases and handle them accordingly
- Points will be deducted if you update your front end and the back end has not been changed. (e.g. if /todo/delete fails and you remove an item from your page)
- Points will be deducted if todo list is not stored in a session variable

Resources for Back End Web Development:

http://flask.pocoo.org/docs/0.12/quickstart/
http://api.jquery.com/jquery.ajax/

Note: ***Do not copy the above code examples***, we are checking for plagiarism but those 3 resources have everything you need to build this portion of the project out.