CSCI 39548 Practical Web Development

Course Description

The class will offer a survey the topics below, combined guest lectures and workshops, and in-class hands on experience.

- Todo web application to facilitate web 1.0 understanding
- Developing software in teams using a simplified Scrum framework and Github
- Discussing and presenting software architecture.
- Deploying applications into the cloud

Learning Outcomes

1a, 2b, 3a, 3d

Instructor

Jesse Greenberg (jg4845@hunter.cuny.edu) and James Lin (jl6851@hunter.cuny.edu)
Office Hours: By appointment

Course Materials

There will be no textbook for this course. Git, and Github will be required. Laptops are not required, but strongly suggested.

Deliverables

- Individual
  - Todo Web Application
  - Individual Project Proposal
  - Team Revenue Breakdown
- Group
  - Architecture Review
  - At least 4 Check-Ins
  - Term Project
  - Final Team Presentation

No work will be accepted late.

Course Schedule

- Maintained as a dynamic spreadsheet since dates are all liable to change

Grading

- Individual Project - 5%
  - Part 1
  - Part 2
- Project Proposal - 5%
- Architecture Presentation - 10%
- Check-Ins - 10% (total)
- Final Presentation - 10%
- Term Project - 60%

Attendance Policy

Code submission

Individual projects will be stored on github. The link will be submitted via blackboard before the due date.

For the group project, we will create github repositories for each team. The master branch will require pull requests. Pull requests will require passing build before they can be merged into master.

Hunter College Policy on Academic Integrity

Hunter College regards acts of academic dishonesty (e.g., plagiarism, cheating on examinations, obtaining unfair advantage, and falsification of records and official documents) as serious offenses against the values of intellectual honesty. The college is committed to enforcing the CUNY Policy on Academic Integrity and will pursue cases of academic dishonesty according to the Hunter College Academic Integrity Procedures. Special attention is given to CONTRACT CHEATING (this is where students have work completed on their behalf which is then submitted for academic credit).

ADA Compliance

In compliance with the American Disability Act of 1990 (ADA) and with Section 504 of the Rehabilitation Act of 1973, Hunter College is committed to ensuring educational parity and accommodations for all students with documented disabilities and / or medical conditions. It is recommended that all students with documented disabilities (Emotional, Medical, Physical and/or Learning) consult the Office of AccessABILITY located in Room E1124 to secure necessary academic accommodations. For further information and assistance please call (212-772-4857)/TTY (212-650-3230).

Hunter College Policy on Sexual Misconduct

In compliance with the CUNY Policy on Sexual Misconduct, Hunter College reaffirms the prohibition of any sexual misconduct, which includes sexual violence, sexual harassment, and gender-based harassment retaliation against students, employees, or visitors, as well as certain intimate relationships. Students who have experienced any form of sexual violence on or off campus (including CUNY-sponsored trips and events) are entitled to the rights outlined in the Bill of Rights for Hunter College.

1. Sexual Violence: Students are strongly encouraged to immediately report the incident by calling 911, contacting NYPD Special Victims Division Hotline (646-610-7272) or their local police precinct, or contacting the College's Public Safety Office (212-772-4444).
2. All Other Forms of Sexual Misconduct: Students are also encouraged to contact the College'sTitle IX Campus Coordinator, Dean John Rose (jtrose@hunter.cuny.edu or 212-650-3262) or Colleen Barry (colleen.barry@hunter.cuny.edu or 212-772-4534) and seek complimentary services through the Counseling and Wellness Services Office, Hunter East 1123.

CUNY Policy on Sexual Misconduct Link:
http://www.cuny.edu/about/administration/offices/la/Policy-on-SexualMisconduct-12-1-14-with-links.pdf

Syllabus Change Policy

This syllabus is a guide for the course and is subject to change. Announcements will be made on CUNY Blackboard and by email regarding the changes.

Week	Plan	Sprint #	Notes
	* Intro to class		
	* How does the web work?		
	* Client-Server		
	* Web 1.0 vs Web 2.0		
	* Introduce Term Project		
29-Jan	* Introduce Solo Project (PTDL)		
	~~* Walk through examples for Node~~		
	~~* Explore term project ideas~~		
5-Feb	~~* Workshop time~~		James was sick :(
12-Feb			
	*** Term Project Presentation 1**		
	* Start forming groups		
19-Feb	* Scrum	0	James Out
	* Web Review		
	* Node / MVC examples		
	* Groups formed		
	* Introduce architecture presentation		
	** Assignment: http://bit.ly/394r7Cn		
26-Feb	** Example: http://bit.ly/32vVsXT	1	
	*** Web 1.0 PTDL Due on 3/5 at 11:59 pm**		
	* API-DB Architecture Overview		
	* UI-FE Architecture Overview		
	* Brief Node-SQL Walkthru (workshop)		
4-Mar	* Brief React Walkthru (workshop)	2	
	*** Check-in 1 with Draft Architecture (in-person or remote)**		
11-Mar	* Open Working + Q&A	3	
			NO CLASS (will be
	*** PTDL-ext Due on 3/19 at 11:59 pm**		available online for
18-Mar	* Open Working + Q&A	4	questions per usual)
	*** Architecture Presentation**		
25-Mar	* Open Working + Q&A	5	**** REMOTE ****

Week	Plan	Sprint #	Notes
	* React (with hooks) Tutorial		
1-Apr	* To-Do API Code Review	6	** **REMOTE** **
	***** Check-in 2**		** TUESDAY **
7-Apr	* Open Working + Q&A	7	** **REMOTE** **
	* File Hosting and CDN using AWS S3 and Cloudfront		
15-Apr	* Open Working + Q&A	8	** **REMOTE** **
	***** Check-in 3**		
22-Apr	* Open Working + Q&A	9	** **REMOTE** **
	* Docker, Kubernetes, and Microservices		
29-Apr	* Open Working + Q&A	10	** **REMOTE** **
	***** Final Check-in**		
6-May	* Final Presentation prep	11	** **REMOTE** **
	***** Term Project Due**		
13-May	***** Final Presentations**		** **REMOTE** **

Project 1 Part 1 – Persistent Storage To-Do App Using API

I have broken part 1 of this assignment up into multiple parts (see bottom). I STRONGLY suggest getting started right away and getting as far as you can so that you have time to bring questions to class. You are not required to do all of the multiple parts if you are confident diving right in.

Overview

Submission + Due Date:

- Email the link to your hosted heroku app to James (jl6851@hunter.cuny.edu) and Jesse (jg4845@hunter.cuny.edu)
- Email the link to your Github repo to James and Jesse
- Due Wed 3/5 11:59 PM, late work will NOT be accepted

Goals:

- Get you up-to-speed with very basic web programming, things like:
    - MVC web framework to build a web application
    - HTML
    - HTTP
    - Integrating with a basic REST API using JSON
    - Session and cookies

Assignment Requirements:

- Create a (clunky) to-do list web app (website) that can:
    - Create a new user
    - Log in as a user (no password required, just have someone type in their name)
    - Be able to see and interact with their to-do items (and ONLY THEIRS, user should NOT be able to see other to-do items)
        - Create new item(s)
        - See their item(s)
        - Mark item(s) as done
        - Delete item(s) altogether
    - IMPORTANT: the data should be persistent, if I restart your app, the data should still be there
    - Log out

Other Requirements:

- Source code on your Github account
- Hosted on Heroku
- Use our provided API

NOT Required:

- Does NOT need to be a single page app. You are actually strongly encouraged to build a clunky, multi-page application (think old internet websites)
- Does NOT need to be beautiful (good old fashioned html is fine, no need for any fancy CSS or JS)

Grading:

- 3: all assignment criteria and requirements met
- 2: over 50% of assignment criteria and requirements met
- 1: <= 50% of assignment criteria and requirements met
- 0: 0% of assignment criteria and requirements met OR no submission OR late submission

Helpful Resources (not in any particular order):

- HTML Tutorial
- HTML Cheat Sheet
- MDN HTTP Docs
- JSON Docs
- Express Docs

Assignment Broken Down into Parts

To help students with less web programming background, I broke the assignment down into various parts that you can complete in sequence. You will only be graded on the final result (the stuff in Assignment Requirements and Other Requirements above), not be graded on any of these pieces

Part 0 (optional) - JavaScript

All of you should be familiar with programming in at least one language. You can complete this first assignment in almost any modern language that you know (including C++). However, JavaScript is an important part of the web stack in browsers, and increasingly so in other contexts as well, such as back-end, scripting, testing, and mobile. As such, I would recommend at least familiarizing yourself with JS, even if you choose to do this project in another language.

Learning a “entirely new programming language” may seem daunting, but many of the basic concepts of programming map well across languages, e.g. variables, conditional statements, functions, certain data structures, etc. I often like to first identify familiar operations from a known language and map them to the new one (printing to the console is usually a good place to start). From there, you can slowly branch out to less familiar or novel parts of the new language (closures, functional programming, and asynchronous programming probably being the most jarring in JS for someone coming from C++).

We are not here to tell you how to learn. Please feel free to explore different methods that will work for you. I personally like to play around the language with small examples and programs with someone familiar with the language, but you may prefer going through an online tutorial or reading a textbook.

Part 1a - Static HTML Page

Build a static HTML page that can be opened in your browser. I suggest you make a very basic profile of yourself containing at least:

- 1 image
- 5 links
- 1 list

As a pretty minimalist example, you can use my college roommate's attempt at this. You can feel free to make something this ugly, or have some fun and expand on this to learn more HTML. I expect that most of you will use a web browser to render your HTML file locally (ask us if you don't know how to do this).

Host your static HTML page on github pages. If you don't know how to do this, follow the instructions here.

Part 1b - Express to Serve Static HTML

Take the HTML page that you created and use Express to actually serve it to a user on a given route. Now you have a server running on your local machine, which your browser is making a request to, receiving your HTML file from, and rendering that file.

Part 1c - Express to Template Data from JSON File

Create a file called *data.json*. You should use the following content for the JSON file:

```
[
	{
		"name": "eat",
		"completed": false
	},
	{
		"name": "sleep",
		"completed": false
```

```
    },
    {
        "name": "poop",
        "completed": true
    }
]
```

First, use Node's fs module to read the contents of the file. Then, create a template that will display this data as a list. When an item is completed, you should show the item as ~~struck through~~. Otherwise, you should display the name like normal. Your page should adjust if you change the JSON file content (e.g. toggle completed, add a new item, etc) and then restart your app. I would recommend exploring the handlebars templating system, with the express-handlebars project being helpful in connecting handlebars and express.

Part 1d - Submit Data to Express App Using HTML Form

Look up how to submit data using an HTML form. Using a form, send data to your Express app and print the data to your terminal.

Part 1e - Use Node to Make HTTP Requests to Our API

Our API is hosted at https://hunter-todo-api.herokuapp.com . See the documentation here.

I encourage you to use the axios module to make HTTP requests to our API. Try to use the API to:

1) Create a new user
2) Authenticate as the user you created
3) Create a new item as that user
4) See all items of that user
5) Change some data on an item
6) Delete an item

Part 1f - Create a Register, Login, and Logout Page

Using the skills you picked up with HTML forms and the requests module, create your register, login, and logout page. You will need to use /auth and /user services, and will need to do some research on cookies, mainly how to set them and unset them.

Part 1g - Finish the To-Do App!

Extend your work above to also do everything from your to-do app! By this point, you are almost completely done.

Part 1h - Host Your App on Heroku

Follow the instructions here to host your app on Heroku.

Project 1 Part 2 – Extending Your Persistent Storage To-Do App

You have TWO choices for extending your persistent storage to-do app. You should choose to implement ONE of the options (not both).

1) Change the back-end of your application to hit your own implementation of the API rather than the API that I built. The look-and-feel of your app can remain exactly the same.
2) Leverage your work with the API I built to change your app into a single page application using a front-end framework (we will support React).

Overview

Submission + Due Date:

- Email the link to your hosted heroku app to James (jl6851@hunter.cuny.edu) and Jesse (jg4845@hunter.cuny.edu)
- Email the link to your Github repo to James and Jesse
- Due Wed 3/19 11:59 PM, late work will NOT be accepted

Goals:

- Get you up-to-speed with an aspect of modern web programming, either on the front-end or the back-end, using something you are already familiar with.

Assignment Requirements:

- Exactly the same assignment / requirements as the first EXCEPT using either extended front-end or back-end. Extended front-end should act as a single page application. Extended back-end should not use my API.
- Create a to-do list web app (website) that can:
    - Create a new user
    - Log in as a user (no password required, just have someone type in their name)
    - Be able to see and interact with their to-do items (and ONLY THEIRS, user should NOT be able to see other to-do items)
        - Create new item(s)
        - See their item(s)
        - Mark item(s) as done
        - Delete item(s) altogether
    - IMPORTANT: the data should be persistent, if I restart your app, the data should still be there
    - Log out

Other Requirements:

- Source code on your Github account
- Hosted online
- Pick option 1 or option 2

NOT Required:

- Does NOT need to be beautiful (good old fashioned html is fine, no need for any fancy CSS)

Grading:

- 3: all assignment criteria and requirements met
- 2: over 50% of assignment criteria and requirements met
- 1: <= 50% of assignment criteria and requirements met
- 0: 0% of assignment criteria and requirements met OR no submission OR late submission

Helpful Resources (not in any particular order):

- React Tutorial 1
- React Tutorial 2
- AWS Educate (student account)
- MySQL Tutorial 1
- MySQL Tutorial 2
- MySQL Python Connector Examples