Welcome to CSC309!

Programming on the Web

Amir H. Chinaei, Spring 2017

ahchinaei@cs.toronto.edu
http://www.cs.toronto.edu/~ahchinaei/

Office hours: M 3:45-5:45 BA4222

today

- course outline (bird's-eye view)
  - survey 1
  - what this course is about
- logistics
  - course organization, information sheet
  - project, assignments, grading scheme, *etc.*
- introduction to
  - web application design

survey 1

- in survey 1, you provide us with:
  - your UTorID,
  - your GitHub username, and
  - your familiarity with technologies related to this course
- before completing the survey
  - make sure you have a GitHub username,
    - *if you don't, sign up here: https://github.com/join*
  - *and, get a student developer pack here: https://education.github.com/pack*
- if you have the GitHub username and UTorID
  - complete the survey here: *https://goo.gl/forms/1sovPcFA1dLEo1k42*
    (deadline: Jan 09)

what is this course about?

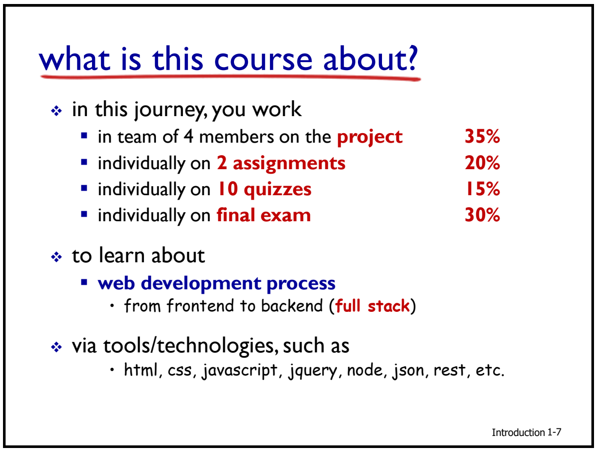
- developing a ***fully-fledged*** web application
  - from *definition* to *implementation* to *demonstration*

jan 20 …feb … march… march 31

what is this course about?

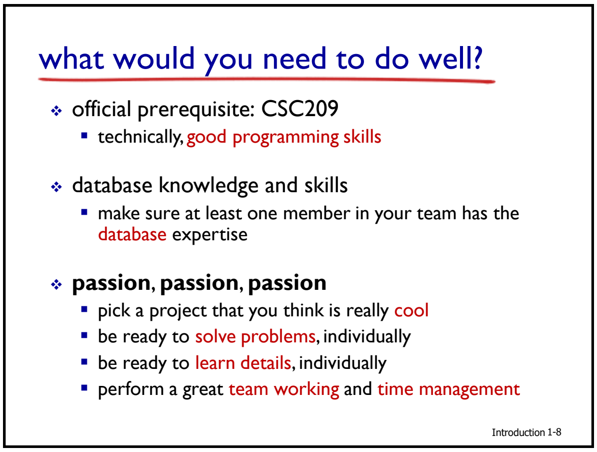
- developing a ***fully-fledged*** web application
  - from *definition* to *implementation* to ***demonstration***

APRIL 05



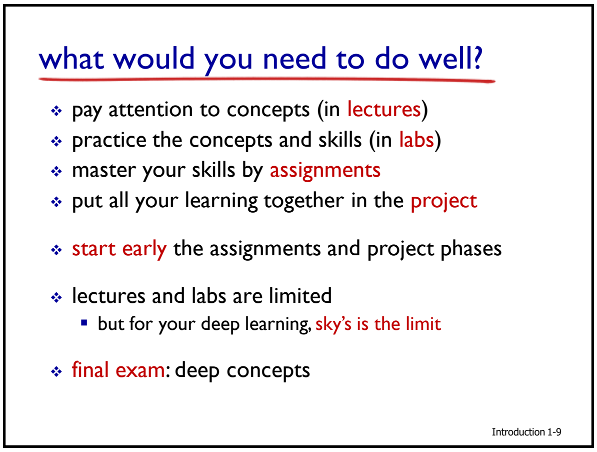
what is this course about?

- in this journey, you work
  - in team of 4 members on the **project** **35%**
  - individually on **2 assignments** **20%**
  - individually on **10 quizzes** **15%**
  - individually on **final exam** **30%**
- to learn about
  - **web development process**
    - from frontend to backend (**full stack**)
- via tools/technologies, such as
  - html, css, javascript, jquery, node, json, rest, etc.



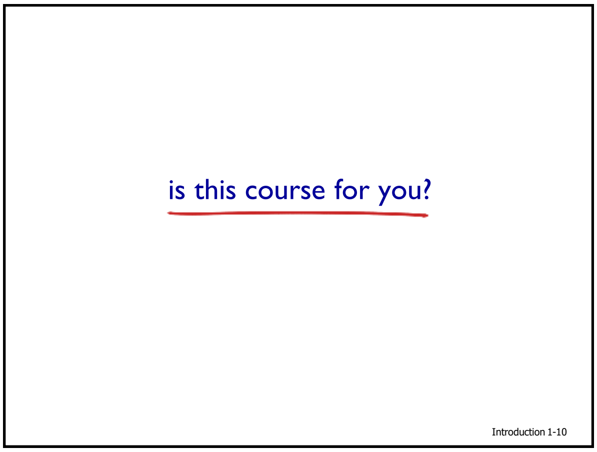
what would you need to do well?

- official prerequisite: CSC209
  - technically, good programming skills
- database knowledge and skills
  - make sure at least one member in your team has the database expertise
- **passion, passion, passion**
  - pick a project that you think is really cool
  - be ready to solve problems, individually
  - be ready to learn details, individually
  - perform a great team working and time management

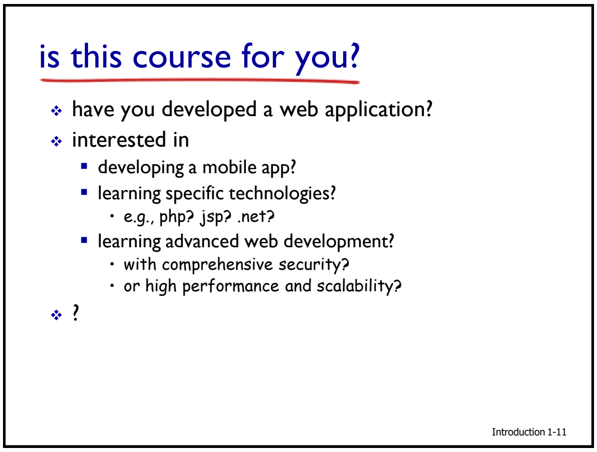


what would you need to do well?

- pay attention to concepts (in lectures)
- practice the concepts and skills (in labs)
- master your skills by assignments
- put all your learning together in the project
- start early the assignments and project phases
- lectures and labs are limited
  - but for your deep learning, sky's is the limit
- final exam: deep concepts

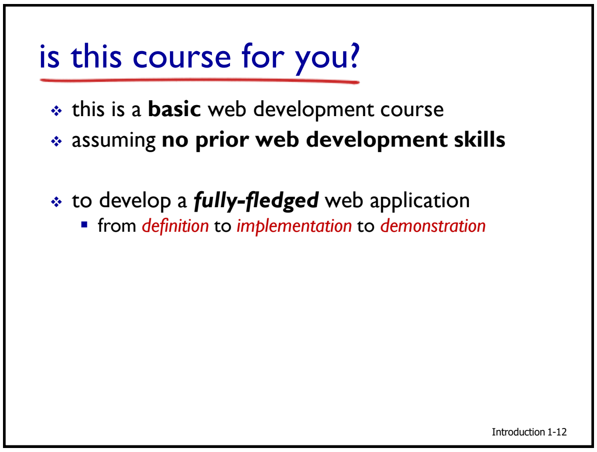


is this course for you?



is this course for you?

- have you developed a web application?
- interested in
  - developing a mobile app?
  - learning specific technologies?
    - e.g., php? jsp? .net?
  - learning advanced web development?
    - with comprehensive security?
    - or high performance and scalability?
- ?



is this course for you?

- this is a **basic** web development course
- assuming **no prior web development skills**
- to develop a ***fully-fledged*** web application
  - from *definition* to *implementation* to *demonstration*

student complaints in the past

- "I didn't learn the technology I wanted to learn"
- "It was too basic"
- "It was too advanced"
- "I had to learn everything myself on stackoverflow"
- "Lectures were useless"

course web page

- for important information on
  - lecture and lab time/location/material
  - contact information of course staff
  - office hours
  - project/assignments/more readings
  - deadlines and evaluation
  - communication and announcements
  - …
- follow the course web page, regularly
  https://csc309-spring2017.github.io/

discussion board

- we use *discourse*
  - https://bb-2017-01.teach.cs.toronto.edu/c/csc309

let's start with
web application design

principle of layering

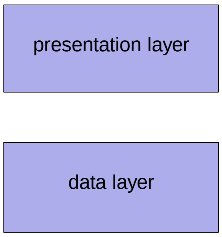
- dividing the application to two+ groups of classes
  - that are functionally or logically related
- such that each layer demonstrates cohesion
- and the dependency among classes is minimized

- **advantages:**
  - modularity, maintainability, reusability

- **disadvantages:**
  - reduced performance

2-layer architecture

- simple application functionality

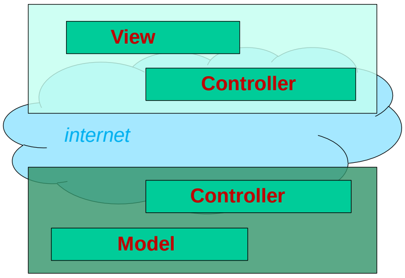


mvc

- The **model** tier
  - represents the **data and logic**
- The **view** tier
  - represents the **user interface**
- The **controller** tier
  - connects and coordinates—**controls**—activities between the view and model

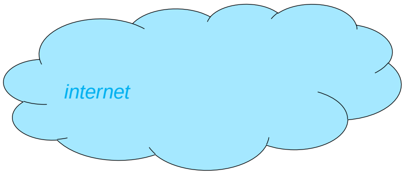
model-view-controller

- **MVC** is a 3-layer pattern



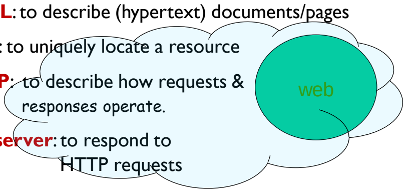
internet & services

- **Is Internet = WWW ?**



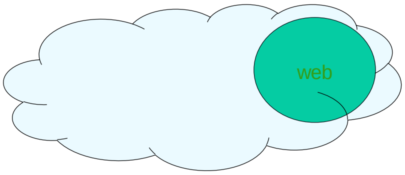
www = web

- it's an information space system—based on request & response—with the following features:
  - **HTML**: to describe (hypertext) documents/pages
  - **URL** : to uniquely locate a resource
  - **HTTP**: to describe how requests & responses operate.
  - **web server**: to respond to HTTP requests
  - **web browser**: to make HTTP requests from URLs and render/display the HTML document received



www

- in CSC309, we develop applications on top of this system.
- That's why they are called web application



we start with HTML

html

- **HyperText MarkUp Language**

  it's used to describe the **content and structure** of information in a document (web page)

- **general syntax:**

  <element>content</element>

- **example:**

  <h2>CS is COOOOL</h2>

- **html5** supports multimedia, semantic formatting, cross-mobile applications, and JS APIs.

css

- **Cascading Style Sheets**
  - it's used to describe the appearance of information
  - it can be embedded in HTML document
    - using the <style> element, or
    - placed in separate .css file
- **example:**

```
h2 {
    color: blue;
    text-align: center;
}
```

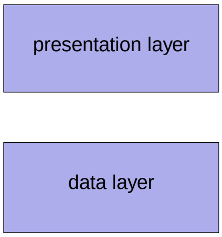
let's start with web application design

principle of layering

- dividing the application to two+ groups of classes
  - that are functionally or logically related
- such that each layer demonstrates cohesion
- and the dependency among classes is minimized
- **advantages:**
  - modularity, maintainability, reusability
- **disadvantages:**
  - reduced performance

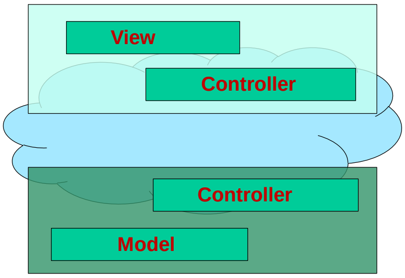
2-layer architecture

- simple application functionality



model-view-controller

- **MVC** is a 3-layer pattern


View
Controller
Controller
Model