# Syllabus

## General Info

**Course** CS325 Web Application Development
**Instructor** Charilaos Skiadas (skiadas at hanover dot edu)
**Term** Winter 2016-2017
**Office** LYN 109 and SCH 121C
**Office Hours** MWF 1pm-2:30pm, and by appointment
**Book** ACM-provided books and other online sources[1]
**Websites** Notes[2]
**Class times** TR 12:20pm-2:05pm in LYN120A

## Course Description

This course focuses on some of the fundamentals of software development, with a distinct emphasis on the development of rich large-scale web applications. Along the way, you will become skilled in Javascript, the programming language for the web.

The inception of the World Wide Web has ushered in a new era in communications, with applications and website being accessed from all parts of the world. Web applications play a pivotal role in this new world, with their ability to bring people together and to allow them to exchange goods and ideas in an unparalled scale. We live in a world where common desktop tasks such as word processing and spreadsheet calculations are now all done "in the cloud", leading to the software-as-a-service business model of today's web applications.

Web applications, along with Mobile applications, are a dominant sector in today's marketplace. In addition to all the normal challenges presented in software development, both kinds of applications run within a specific platform and environment, with limited capabilities and new challenges not faced by desktop applications. They also are subject to additional security questions. We will be examining these issues as the term progresses.

Large-scale applications contain at a minimum tens of thousands of code lines, and possibly hundreds of thousands. And a lot of that code is interconnected, so changes in one place might have ramifications all over the code base. Most of these projects also last for many years, and involve many revisions of the code, often by different programmers. Managing to work with such large pieces of code requires discipline and consistent use of practices aimed at managing as well as reducing the complexity of your code. Therefore a considerable part of the course is devoted to software development practices:

- using a version control system to maintain ever-evolving code.

[1] http://skiadas.github.io/WebAppsCourse/site/links.html
[2] skiadas.github.io/WebAppsCourse/site/

- employing extensive test suites to ensure more stable code.
- documenting your code so that both your users and future maintainers can use your application.
- modularizing your code base into manageable and reusable components.
- employing design patterns to create a common frame of reference.
- following the Pair Programming discipline which enables programmers to learn from each other.

These are all essential skills transferable to other programming languages and environments. We will also be focusing on a number of issues related specifically to Javascript programming and Web Applications, including:

- Core language features and patterns.
- Interacting with a web page and a web server.
- Learning about the various approaches to handling the asynchronous nature of user interaction and network communication.

A large part of the course will be a project that you will be working on in collaboration with one other student. This will give you the opportunity to practice the above principles in a real project, as well as giving you the satisfaction of having created a deliverable project of your own.

## Course Components

### Reading Assignments

In the class schedule page[3] you will find, for each class day, a list of links to reading assignments. Your homework and lab work will require you to have a solid understanding of the material covered there, so I strongly encourage you not to get behind.

### Class Attendance

You are expected to attend every class meeting, including labs. You are only allowed to miss 3 classes without excuse. From that point on, every unexcused absence will result in a reduction of your final score by one percentage point, up to a total of 5 points. Excused absences should be arranged in advance, and backed by appropriate documentation. Emergencies will be dealt with on an individual basis. There are very few reasons that would qualify as an excuse for an absence.

### Quizzes

You will have quiz about once a week, based on the material covered that week. You are expected to work on these on your own. Whenever there is code involved, make sure to have tested the code in the Console before filling in your answer.

[3] skiadas.github.io/WebAppsCourse/site/schedule.html

### Labs

About once a week you will have a slightly longer and more thought-provoking lab assignment. You will work on these labs with the same person that will be your collaborator for the project. It is expected that you will work following a pair-programming paradigm, where you both work on the same computer, one person typing and another thinking about the code, trading places ever so often.

### Exams

There will be two equal-weight exams, one midterm (TODO: Date) and one during finals week, focusing on both the theoretical aspects of the course and some code snippets.

### Project

A large part of your grade will be the term project that you will need to work on with one partner. The end result will be a fully-functioning Web Application and the GitHub project used to maintain it. This will also include documentation and an extensive test suite for your application.

### Getting Help

- You should never hesitate to ask me questions. I will never think any less of anyone for asking a question. Stop by my office hours or just email me your question, which has the great benefit of forcing you to write it down in clear terms, which often helps you understand it better.
- There are lot of online resources that are linked from the reading assignments and the site's links page. You are free and encouraged to use those resources. What you are NOT allowed to do is blindly copy-paste a code solution from some resource.
- You are allowed, and in fact encouraged, to work together and help each other regarding the notes and understanding the material.
- You may discuss the ideas behind the quizzes with others, but only after you have spent some time trying them on your own. The submitted work must be your own! So even though you may talk to others about the problems, when you sit down to write the answers you should be on your own.

## Grading

Your final grade depends on class attendance, homework, labs, project and the final, as follows:

| Component | Percent |
|---|---|
| Attendance | 5% |
| Quizzes | 15% |
| Labs | 15% |
| Code Reviews | 10% |
| Project Completion | 15% |
| Project Rubric | 20% |
| Final Exam | 20% |

This gives a number up to 100, which is then converted to a letter grade based roughly on the following correspondence:

| Letter grade | Percentage Range |
|---|---|
| A, A- | 90%-100% |
| B+, B, B- | 80%-90% |
| C+, C, C- | 70%-80% |
| D+, D, D- | 60%-70% |
| F | 0%-60% |

## Objectives

This is a list of the learning objectives for the course, split into categories.

### Software development

- Maintain a version-controlled repository of your code.
- Manage your project and code evolution via management tools like issues, milestones, labels, and commits.
- Improve the reliability of your code via both unit tests and integration/functional tests.
- Maintain clean and readable code via linting tools and coding standards.
- Organize code into self-contained interacting modules.
- Reuse code from existing libraries.
- Work with teammates on the same code and learn the value of pair-coding
- Perform code reviews on other teams' code and learn from their code reviews.
- Use diagrams and other tools to design your application.

### Web Applications

- Multi-page vs Single-page applcations and their tradeoffs.
- Basics of web communication (HTTP, XHR)

- Various web/data formats, their use and roles (HTML, CSS, JavaScript, JSON, XML).
- Constraints of working in the web/browser platform.
- Built-in Javascript functionality
- Design patterns with Javascript objects and functions
- Basic web page elements (creating, formatting and interacting)
- Using jQuery for more elaborate interfaces
- Asynchronous/Reactive models of operations (Events, PubSub, Promises)
- Basic frameworks for GUIs (MVC, Flux)
- Modular programming in Javascript

## Schedule

A rough schedule of the topics covered. See the detailed schedule[4] for more details.

**Week 1** History of the Web and Javascript. EcmaScript 5 vs 6.

Components of a Web page: HTML, CSS, Javascript.

Multi-page apps vs Single-page apps.

Modes of Javascript operation: In browser vs node.js

Basic Javascript Language constructs: Local/Global variables, arrays, objects, strings, functions.

**Week 2** Version control, Git and GitLab. Typical workflow.

Functions as values.

Function closures. Patterns of using function closures.

More work with arrays.

**Week 3** Writing tests: In browser and in node.

Automating tasks (Grunt).

Linting and code standards.

Objects in Javascript. Use of "this". Classes.

**Week 4** Interacting with a page. DOM, jQuery.

Events. The event loop. Timers.

**Week 5** Modular design patterns in Javascript.

Basic design patterns.

**Week 6** MVC patterns.

Midterm.

[4] schedule.html

**Week 7** Using templates.

XHR.

Project proposals due, discussion.

**Week 8** Components, React.js

Flux paradigm.

**Week 9** Graphical modeling of applications.

Project Models. Discussion.

**Week 10** Work on projects.

Code reviews.

**Week 11** Security concerns.

**Week 12** TBA

**Week 13** Work on projects.

Code reviews.

**Week 14** Wrap-up.

Work on projects.