CS144: Web Applications – Spring 2021

http://oak.cs.ucla.edu/classes/cs144/

Time and Place

- **Hours:** Monday and Wednesday, 10-11:50 AM
- All lectures are conducted live on Zoom.
- Questions can be asked during lecture via sli.do.
- Video recordings of lectures will be available on CCLE.

Instructor

- **Name:** Junghoo "John" Cho
- **Email:** cho@cs.ucla.edu

- **Office hour:** Tuesday 10-10:30 AM, 3:30-4 PM

TAs

- Song Jiang (songjiang@cs.ucla.edu): Office hours - Mon 12:30-2:30 PM
- Boyang Fu (boyang1995@g.ucla.edu): Office hours - Thu 2-4 PM
- Lingxiao Wang (lingxw@ucla.edu): Office hours - Wed 1-3 PM

Discussion Sessions

- Lecture Discussion: Friday 10 AM
- Project Discussion: Friday 11 AM

Course Description

Developing today's Web applications requires knowledge of a number of diverse topics, including the basic Web architecture, core Web standards (such as HTTP, HTML, CSS), JavaScript, asynchronous and functional programming, internet security, and cluster-based scalable architecture. Traditionally, these topics have been taught in different subdisciplines of computer science, so students had to take a fair number of courses to learn the basic concepts necessary to build effective and safe Web applications. The goal

of this class is to teach students the most important concepts and give them first-hand experience with the basic tools for developing Web applications.

The topics covered in the class include:

- Core Web standards, such as HTTP, Unicode, HTML, JSON, and CSS
- Javascript programming
- Web programming paradigms, including functional programming, asynchronous programming, and MVC
- Web-site architecture
- Web-site scalability
- Web security

To help students digest the materials learned in the class, we will assign a quarter-long class project (which will be divided into multiple submissions), in which students have to build a Web site that allows users to write and publish blogs (a.k.a., mini WordPress) The software tools and development environment will be provided on the class Web site.

Prerequisites

CS143 is a required prerequisite to this class. In particular, students must know:

- Relational databases
- Java programming
- Unix command-line interface
- Basic HTML
- Basic networking (TCP/IP)
- Basic data structures and algorithms (sorting, hashing, tree, etc.)

Students should have access to a computer on which they can install software packages.

Books

The class does not have a required textbook, but students may find the following books helpful for reference and in-depth learning:

- JavaScript: The Definitive Guide by David Flanagan
- CSS: The Definitive Guide by Eric Meyer and Estelle Weyl

- Angular 5: From Theory To Practice by Asim Hussain
- Web Application Architecture by Leon Shklar and Rich Rosen
- Designing Data-Intensive Applications by Martin Kleppmann
- Foundations of Security by Neil Daswani, Christoph Kern, Anita Kesavan

We will also provide pointers to relevant online/offline materials as the class progresses.

Piazza

All students must join and utilize the CS144 group at Piazza by registering at https://piazza.com/ucla/spring2021/cs144. Piazza will be the primary channel for students to ask course and project-related questions and for others, including the TA, to answer them. Note that some of your questions may have already been discussed and answered by others, so please search the board first before asking a question. When you join Piazza, you may choose to receive email notifications for new messages or just to read them on the board. *You are responsible for all your posts to Piazza.* Thus, please do NOT post any content that might be considered as a violation of honor codes, such as your source code to the project. If you have any doubt or concern, please **ASK** the TA/lecturer before posting it.

GradeScope

All course assignments will be administered via GradeScope. Please ensure that you are registered at CS144 on GradeScope.

Grading

The final grade will be assigned 100% based on projects this quarter.

Project

Your project is to build an online Web site that allows users to post blog entries written in markdown. Over the quarter, your will need to implement the Web site twice, once using a more "traditional" stack based on MySQL and Apache Tomcat, and once more using a "modern" stack based on MongoDB, Node.js, Express, and Angular (a.k.a.

MEAN stack). Through this project, you will get familiar with the following software tools:

- Docker
- MySQL
- JDK
- Apache Tomcat
- MongoDB
- Node.js
- Express
- Angular
- Locust

System and Programming Issues

In order to minimize the installation and configuration hassle, we will provide "Docker container images" that include all necessary software packages needed for the project. All your development should be done inside a "virtual environment" (a.k.a. *container*) created by our image. In order to run the provided container, you will need a fairly-equipped machine, with at least 4GB of main memory and 10GB of free hard disk space.

The first two projects will be done primarily in Java on Apache Tomcat, while later projects will be done in Javascript on the MEAN stack. Even now, Java is one of the most popular methods used in the enterprise setting for server-side projects, so we want our students to be familiar with the language. Then in later projects, students will primarily develop in Javascript on the MEAN stack, a Web development stack that has become hugely popular in the last few years.

We will assume that students are already proficient with SQL and Java or able to learn them quickly. For those of you who haven't done much Java programming before, please note that the syntax and the programming model of Java is very similar to C++. As long as you are a proficient C++ programmer, even without much prior Java experience, we expect that you can finish our first projects in a reasonable amount of time. We do **NOT** assume any experience in Javascript programming. Later projects that are based on Javascript will be done only after we learn Javascript in the class.

Project Parts

The project is broken into five parts and you will have roughly two weeks for each project part except Project 5.

- Part 1 (System Setup and Warm-up): In this part, you will have to set up a provided "container" on your computer and interact with MySQL to populate data and query them. You will also have to implement a simple Java program to brush up on Java programming.
- Part 2 (Markdown Editor on Tomcat): In this part, you will need to implement a simple online markdown editor using MySQL and Tomcat.
- Part 3 (Blogging Server Using MongoDB and NodeJS): Now you will have to implement a Blogging server using the MEAN stack. In this part, you write the server-side program that saves and publishes blog posts written by the markdown-editor client that will be built in Part 4. The server-side program will be developed using MongoDB, NodeJS, and Express.
- Part 4 (Markdown Editor Using Angular): In this part, you will use an advanced version of the markdown editor client using Angular. All entries saved by the editor will be stored in the server implemented as part of Part 3.
- Part 5 (Performance and Scalability): In the last part of our project, you will explore issues of performance and scalability of Web applications.

Late Submission Policy

To accommodate the emergencies that a student may encounter, each student has a 4-day grace period for late submission to use throughout the quarter. The grace period can be used for any part of the project in the unit of one day. For example, a student may use 1-day grace period for part 1 and 2-day grace period for part 3. **Any single project part may not be more than 2-days late.** Note that the grace period can be used in the unit of one day. Even if a student submits a project 12 hours late, he/she needs to use a full-day grace period to avoid the late penalty.

Once you run out of your four grace days, we will start applying **exponential penalty and take off 5^N% of the assignment's value**, where N is the number of late days of your submission. Again, any single project part may not be more than 2-days late.

Partners

Since the final grade will be assigned purely based on projects, all projects should be done individually. It is OK to discuss with others on how to implement different parts of the projects, but explicit code should never be shared between students.

Academic Integrity

At http://www.deanofstudents.ucla.edu/Academic-Integrity, the Office of the Dean of Students presents University policy on academic integrity, with special attention to cheating, plagiarism, and student discipline. The policy summaries don't specifically address programming assignments in detail, so we state our policy here. In order to earn any points on your coursework, you must turn in this signed agreement.

Each of you is expected to **submit your own original work.** On many occasions it may be useful and have an educational value to ask others (the instructor, the TA, or other students) for hints or help, or to talk generally about programming strategies. Such activity is both acceptable and encouraged, but **you indicate any assistance (human or otherwise) that you received and provide explicit citation to any materials that were not produced by yourself.** Any assistance received that is not given proper citation will be considered plagiarism. In addition, to avoid unintended sharing and copying of your work, **publishing your work on a public repository, such as public GitHub, is strictly prohibited.**

So where do we draw the line? We'll decide each case on its merits, but here are some categorizations:

Acceptable:

- Clarifying what an assignment is requiring
- Discussing algorithms for solving a problem, perhaps accompanied by pictures, without writing any code
- Helping someone find a minor problem with their code, provided that offering such assistance doesn't require examining more than a few lines of code
- Using codes from the course text, from reference materials linked on the project page, or from the instructor or the TAs.

Unacceptable:

- Turning in any portion of someone's work without crediting the author of that work, if they are not from the sources mentioned above.
- Using project solutions from earlier offerings of this or any other class
- Soliciting help from an online source where not all potential respondents are subject to the UCLA Student Conduct Code
- Receiving from another person (other than the instructor or a TA) a code fragment that solves any portion of a programming assignment
- Writing for or with another student (except your partner) a code fragment that solves any portion of a programming assignment

In any event, you are responsible for coding, understanding, and being able to explain on your own or as a team all project work that you submit.

Be especially careful about giving a copy of your work to a friend who "just wants to look at it to get some ideas." Frequently, that friend ends up panicking and simply copies your work, thus betraying you and putting you through the hassle of an academic discipline hearing.

You must abide by this policy in addition to the policies expressed in the UCLA Student Conduct Code.

If a violation of the policies is suspected, in accordance with University procedures, we will have to submit the case to the Dean. A typical penalty for a first plagiarism offense is suspension for one or more quarters. A second offense usually results in dismissal from the University of California.

If you have any questions about this policy or about the degree to which we will pursue academic honesty violations, please discuss your concerns with the course staff immediately.

Suggestions

Since this class is offered online, we are likely to experience a few glitches and hiccups. Please bear with us in addressing challenges that we encounter along the way. If you have any suggestions on how to make the online class better, please do not hesitate to let us know. You are likely to know more online tools that the instructor and/or the TAs. If there is anything that we can learn from you, we would love to learn.