COS 243
Multi-tier Web Application Development

Department of Computer Science and Engineering
Taylor University

1 Instructor

Dr. Tom Nurkkala
Associate Professor, Computer Science and Engineering
Director, Center for Missions Computing

Office	Euler Science 211	
E-Mail	`tnurkkala@cse.taylor.edu`	
Phone	765/998-5163	
Hours	M	2:00–3:00
	T	10:00-12:00
	W	2:00–3:00
	Or by appointment	

2 Texts

There are two *required* texts for the course covering Hapi [2] and Angular 2 [1]. Refer to page 7 for details.

3 Learning Objectives

Upon successful completion of this course, you will be able to:

1. Leverage modern web application architecture
2. Store application data in a relational database management system
3. Create entity-relationship diagrams of a data model
4. Use the Git distributed revision control system and the GitHub hosting service for Git
5. Employ the massive ecosystem of server-side JavaScript
6. Design and program using asynchronous JavaScript, including callbacks and promises

7. Know how to use a database-independent query builder to create queries that work across database platforms
8. Understand object-relational mapping and apply it in a JavaScript-based web server
9. Leverage the full capability of the HTTP protocol
10. Design and document flexible and powerful RESTful web services
11. Construct a RESTful web server
12. Create tests that validate the behavior of a RESTful server
13. Validate RESTful requests received by a server
14. Use the TypeScript language, an extension of JavaScript
15. Design and build a user interface using a standards-compliant toolkit
16. Employ Angular 2 to create a non-trivial single-page web application
17. Tame client-side asynchronous behavior using observables
18. Implement regression tests for client-side code

4 Course Outline

We will cover the following topics.

1. Introduction
   - (a) Course Overview
   - (b) Modern Web Architecture
     - i. Design Patterns
     - ii. Example Application
2. Data Persistence
   - (a) Relational Database Management
   - (b) Entity-Relationship Diagrams
   - (c) Git and GitHub
   - (d) Server-Side JavaScript
   - (e) Asynchronous JavaScript
   - (f) Query Building

   (g) Object-Relational Mapping

3. RESTful Web Services
   (a) HTTP
   (b) RESTful APIs
      i. Design Guidelines
      ii. Endpoints
      iii. Testing
   (c) Hapi REST Server
      i. Concepts
      ii. Server
      iii. Validation
      iv. Testing
4. Single-Page Applications
   (a) TypeScript
   (b) Angular 2
   (c) UI Toolkits
   (d) Development Tools
   (e) RxJS
   (f) Authentication
   (g) Testing

5 Project

During the course, you will build a modern, full-stack web application using all the technologies we will cover.

6 Evaluation

The grading breakdown for the course is shown in table 1. Refer to my *Periodic Table of the Grades* (on Moodle) for the grading scheme. I reserve the right to award a higher grade than strictly earned; outstanding attendance and class participation figure prominently in such decisions.

Category		Weight
Homework	Written	10%
	Programming	25%
Project		25%
Exams	Midterm 1	10%
	Midterm 2	10%
	Final	20%
	Total	100%

Table 1: Grading details

7 Course Expectations

Following are my expectations regarding the course.

7.1 Attendance

You are required to attend all class sessions. I will be in class each day, and I expect you to be there also.

In general, I am very understanding about students who must miss class due to a sanctioned Taylor activity, medical appointment, job interview, family emergency, and the like. If possible, let me know in advance that you will not be in class; I will work with you to arrange make-up instruction, homework, exams, etc.

7.2 Late Work

All course assignments will include an unambiguous due date. Usually, assignments are due at the beginning of class on the due date. If there are multiple sections of a class, the assignment is due at the beginning of the earliest such section. Barring exceptional circumstances like those mentioned in section 7.1, I expect your work to be submitted *on the due date*. Late work will *not* be accepted.

This policy on late work is intended to prepare you for real-world experience after graduation. In the marketplace, late work is not merely an inconvenience. Missing a deadline may alienate your customer, upset your manager, ruin your project, or terminate your employment! *Now* is the time to learn the self discipline and time management skills required to complete your work when it is due.

7.3 Conduct

I expect you to be prepared, awake, aware, and participatory during class. I will not hesitate to ask you to stand or move if you are distracted or sleepy.

I expect you to join in discussions, respond to questions from me and from your colleagues, and ask questions of me. I expect you to hold my feet to the fire if I am being unclear, unkind, or contradictory.

7.4 Gizmos

You may not use a laptop, tablet, or similar device to check e-mail, engage in social networking, surf the web, or any other activity not directly relevant to current classroom activity. If you use an electronic gizmo during class for legitimate academic purposes (e.g., note taking), be prepared to demonstrate relevant use on demand at any time.

8 Course Management

We use email, Moodle, and Slack to manage the course and for on-line communication.

8.1 Email

Electronic mail is an official channel of communication between all members of the university community. You are responsible to check your email regularly (daily) for information related to the course.

8.2 Moodle

The Computer Science and Engineering department uses Moodle as our Learning Management System. The URL for Moodle is `https://moodle.cse.taylor.edu`. To sign on to the course site for the first time, you will need an enrollment key.

The enrollment key for this course is: `nerds4christ`

You are responsible for checking Moodle regularly to keep up with assignment due dates and other announcements. For due dates, *the Moodle calendar is your friend.*

8.3 Slack

This course will use Slack for informal communication, Q&A, last minute announcements, jokes, and the like. Find the *TU CSE Student* slack team at `tucsestudents.slack.com`. Look there for a *channel* dedicated to the course.

9 Academic Integrity

As a student at an institution whose goal is to honor Christ in all that it does, I expect you to uphold the strictest standards of academic integrity. You must do your own work, cite others when you present their work, and never misrepresent your academic performance in any way. Violation of these standards stains the reputations of you as a student, Taylor as an institution, and Jesus as our Lord.

Every assignment should indicate clearly that it is either:

- An **individual** assignment, to be done *entirely by you*, without any direct participation from other students.
- A **group** assignment, to be done *collectively with a group*

Unless otherwise stated, assignments are **individual** assignments.

You are *always* welcome to get help from the instructor on *any* homework assignment or project, whether an individual or group assignment.

9.1 What Constitutes Academic Dishonesty?

For purposes of this course, the following are *non-exhaustive* examples of violations of academic integrity.

1. Sharing code or other electronic files by copying, retyping, looking at, or supplying a copy of a file from this or a previous semester.
2. Sharing written assignments or exams by looking at, copying, or supplying an assignment or exam.
3. Using another student's code. Using code from this or previous offerings of the class, from courses at other institutions, or from any other source (e.g., software found on the Internet).
4. Looking at another student's code. Although mentioned above, it bears repeating: looking at other students' code or allowing others to look at yours is academic dishonesty. There is no notion of looking "too much," since no looking is allowed at all.

9.2 What Does Not Constitute Academic Dishonesty?

In contrast, the following are *non-exhaustive* examples of activities that *do not* violate academic integrity.

1. Clarifying ambiguities or vague points in class handouts or textbooks.

2. Helping others use the computer systems, networks, compilers, debuggers, profilers, or other system facilities without regard to a particular assignment or project.
3. Helping others with high-level design issues.
4. Helping others with high-level (*not* code-based) debugging.
5. Using code provided by the instructor from the course web site or elsewhere.

9.3 From the Provost

Taylor's Provost[1] defines *plagiarism* as follows:

> In an instructional setting, plagiarism occurs when a person presents or turns in work that includes someone else's ideas, language, or other (not common-knowledge) material without giving appropriate credit to the source. Plagiarism will not be tolerated and may result in failing this course, and may also result in further consequences.

Academic dishonesty constitutes a serious violation of academic integrity and scholarship standards at Taylor that can result in substantial penalties, at the sole discretion of the University, including but not limited to, denial of credit in a course as well as dismissal from the University. In short, a student violates academic integrity when he or she claims credit for any work not his or her own (words, ideas, answers, data, program codes, music, etc.) or when a student misrepresents any academic performance.

For more information, see the Taylor University Undergraduate Catalog.

References

[1] Yakov Fain and Anton Moiseev. *Angular 2 Development with TypeScript*. Manning Publications, 2016. ISBN: 978-1-61-729312-2.

[2] Matt Harrison. *hapi.js in Action*. Manning Publications, 2016. ISBN: 978-1-63-343021-1.

[1] At Taylor, the *Provost* is our Chief Academic Officer.