SENG 365 – Web Computing Architectures:

Course Outline, Semester 1 2021

Ben Adams
Version 1.0, 28 January 2021

1 Course aims

Welcome to SENG365, Web Computing Architectures course. This course explores web applications and the underlying distributed application architectures and technologies. Topics will include web fundamentals (HTML, HTTP, CSS, JavaScript), asynchronous events, data persistence, web application architectures, AJAX, web APIs, and web computing.

The course involves a significant amount of practical programming; good programming skills and the ability to pick up new languages quickly are helpful.

2 Course goals

- Knowledge of the fundamentals of web applications (e.g., HTTP, HTML, CSS).
- Competence in a server-side runtime environment, e.g. Node.js.
- Competence in a client-side language, e.g., JavaScript.
- Have some understanding of appropriate architectures for web applications, such as 3-tier and MV*.
- Have some understanding of web services and protocols, such as RESTful and GraphQL.
- Have some understanding of HTML5 and its significance for the future.
- Gain basic knowledge of emerging technologies in web applications.

3 Learning outcomes

3.1 Overall

- Demonstrate the ability to develop simple web applications.
- Appreciate the challenges encountered when designing web applications.
- Develop an understanding about the underlying technologies of current web technologies.
- Appreciate the value of emerging web technologies.

3.2 Specific

- Understand the architecture of web applications including the limitations imposed by the underlying stateless HTTP protocol.
- Understand architectures for web applications, e.g., 3-tier, MVC, MVP, and MVVM.

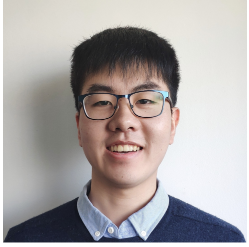
- Appreciate some of the issues of web application security including an awareness of such attacks as SQL injection, cross-site scripting, and identity theft.
- Be able to write a simple web application in appropriate well-structured technologies (e.g., JavaScript, Node.JS, Vue.JS), using a data source for persistent storage (e.g., MySQL or MongoDB), with data queries and updates.
- Understand the principles behind AJAX and be able to make simple use of it in a web application.
- Understand some of the extended capabilities provided by HTML5.
- Understand how web services work (e.g., SOAP-based, RESTful, GraphQL) and be able to design and implement a simple web service.

4 Course team

The course team comprises:

Ben Adams
Course coordinator and lecturer
Room: Erskine 310
Email: benjamin.adams@canterbury.ac.nz
Phone: 03 36 95710

Morgan English
Tutor
Email: mme88@uclive.ac.nz



Yu Duan
Tutor
Email: ydu34@uclive.ac.nz

5 Teaching schedule

The intention is for the lectures and the labs to align and complement each other. The lectures will provide conceptual material together with examples (e.g., code) to demonstrate or illustrate this material. Inevitably, lectures can only provide an overview or introduction to a topic. The labs will provide the opportunity for practical work relating to the material introduced in the lectures. Table 1 presents the provisional teaching schedule for the course. Students should see their personal timetable for lecture and lab times.

Table 1: Lecture and lab schedule

Week	Lectures	Lab
1	Introduction to the course and JavaScript	Pre-labs on your own time
2	JavaScript cont. and asynchronous behaviour	Introduction to JavaScript, Node.js, and APIs
3	Data persistence: SQL, NoSQL, in memory data stores	Persisting data in Node.js
4	HTTP servers and APIs, REST and GraphQL	Structuring applications in Node
5	Security considerations	Assignment 1 support
6	Event-handling, reactive, web sockets	Assignment 1 support
	MID-TERM BREAK	
7	Introduction to client-side programming / Vue.js	Introduction for client-side apps and Vue.js
8	AJAX, XHR Requests, CORS	Interacting with APIs using Vue.js
9	Design patterns	Structuring applications in Vue.js
10	Performance / Virtual DOM	Assignment 2 support
11	Testing: automated and manual	Assignment 2 support
12	Review, feedback, and exam preparation	Assignment in-lab testing (compulsory attendance)

6 Labs

6.1 Self-study preparatory labs

To take account of the different backgrounds of students in this course we have created self-study labs for students to undertake at the beginning of the course. These labs are identified on Learn. They cover topics such as version control and Git, and databases.

6.2 Extra-curricular labs

For those who would like to learn more than the course formally covers we have created additional self-study labs. These will be made available on Learn as the course progresses.

6.3 Scheduled labs

We have developed a series of lab sessions for the scheduled timetable sessions. These are available on Learn.

6.4 Assignment support labs

We have scheduled the labs to provide at least one week in each term for students to explicitly work on their assignments in lab times.

7 Assessment and assessment policy

The assessment will be based on the two assignments, a mid-semester test, and the final exam. Please note the mid-semester test will not be in the normal lecture room, but rather in the K1 Lecture Theatre and will be held on Tuesday 31 March at 7pm.

7.1 Schedule for assessment

Type	**Worth**	**Important dates**
Assignment 1	25%	Set: Beginning of term 1 Hand-in & drop-dead date: Thursday 1 April, 5pm
Mid-semester test	20%	Week 5 **Wednesday 24 March, 7-9pm** **(Location: E8 Lecture Theatre)**
Assignment 2	25%	Set: Beginning of term 1 & week 6 Hand-in & drop-dead date: Friday 28 May, 5pm Compulsory lab attendance: w/c Monday 31 May
Final examination	30%	TBD, during exam period (~2 hrs)

7.2 NO extended 'drop-dead' date on assignments

Due to the way that we test the assignments, the constraints of other assessments on the course, and that we want to give students as much time to do the assignments as possible, there are no one-week extensions for the assignments. If we introduced a one-week 'drop-dead' date we would need to pull the hand-in forward by one week to accommodate the 'drop-dead' date. We would rather just give all students the extra week.

7.3 Assessment policy

The Computer Science department has the following grading policy.

1. In order to pass a course, you must meet the absolute passing mark for the class; i.e. the weighted sum of all your scored items must meet the absolute passing mark set for the class. Marks are sometimes scaled to achieve consistency between courses from year to year. Specific passing marks are therefore set on a class-by-class basis. For example, a total mark of 50% would typically be required to achieve a C pass; although this may vary. Passing a course with a C indicates that we believe that you have just mastered the relevant material. Your ranking in the class may be a more useful indicator of your progress; students ranked above you will receive higher grades, and vice versa.

2. You must average at least 45% on the invigilated assessment items. (Invigilated assessment items are those completed under exam conditions, such as oral and written exams and lab tests.) If you achieve less than 45% over these items, you will receive a D, or E, depending on your marks.

8 Reference material

Information relating to this course can be found on the SENG365 Learn site. Class notices and course notes will also be distributed via Learn. References to recommended reading (mostly web-based) will be given in lectures and labs.