CPNT 262 - Web Client and Server Programming

Course Description:

This course provides an introduction to Web Application development concepts for client facing and server side programming to build multi-tiered database applications accessed through thin-clients. Learners will enhance websites to provide form validation, rollover effects, and cookie handling.

1.5 credits

Time Guidelines:

The standard instructional time for this course is 144 hours.

Effective Year

2019/2020

Course Assessment:

Daily Exercises	30%
Workshops	70%
	15%
	35%
Total:	100%

SAIT Policies and Procedures:

For information on the SAIT Grading Scale, please visit policy AC 3.1.1 Grading Progression Procedure: http://www.sait.ca/Documents/About SAIT/Administration /Policies and Procedures/AC.3.1.1 Grading and Progression Procedure.pdf

For information on SAIT Academic Policies, please visit: www.sait.ca/about-sait/administration/policies-and-procedures/academic-student

Required Course Publication(s):

Young, A. R., Meck, B., Cantelon, M., Oxley, T., Harter M., Holowaychuk, TJ, and Rajlich, N. (2017). *Node.js in Action* (2nd ed.). Manning Publications. ISBN: 9781617292576.

Wilton, P. and McPeak, J. (2015). *Beginning JavaScript* (5th ed.). Wiley. ISBN: 9781118903339.

Optional Reference Publication(s):

Subramanian, V. (2017). *Pro MERN Stack: Full Stack Web App Development with Mongo, Express, React, and Node*. Apress. ISBN: 9781484226520.

Course Learning Outcome(s):

1. Apply the JavaScript language to enhance web pages.

Objectives:

1.1.Explain the history and purpose of JavaScript.

1.2.Describe the structure of a simple JavaScript script.

1.3.Describe how JavaScript is integrated into a web page.

1.4.Identify the JavaScript variable data types.

1.5.Use variables in a JavaScript program.

2. Use complex control structures to modularize scripts and control the flow of the running script.

Objectives:

2.1.Describe the syntax of conditional statements.

2.2.Explain how the different types of loop structures can be used.

2.3.Employ functions to make a script modular with re-usable blocks of code.

2.4.Demonstrate the use of parameters and return statements to customize functions.

3. Use JavaScript objects to make richer web pages.

Objectives:

3.1.Explain what objects are and why they are used in JavaScript.

3.2.Describe the purpose of object methods and properties.

3.3.Discuss the types of objects that are part of the browser’s JavaScript engine.

3.4.Demonstrate how objects are created and used in a script.

3.5.Demonstrate how to control JavaScript objects on a web page.

3.6.Explain the purpose and structure of Array objects.

3.7.Use JavaScript extension libraries such as jQuery to simplify the coding process throughout the remaining JavaScript modules.

4. Write scripts that perform event handling and form validation.

Objectives:

4.1.Explain Form tags and the related form objects that are generated when a page loads.

4.2.Describe the various event handlers that intercept user interaction with the web page.

4.3.Use event handlers to activate JavaScript functions.

4.4.Use JavaScript for controlling form submission.

4.5.Apply JavaScript to validate form data before submitting it to the server.

5. Write scripts that use built-in browser objects.

Objectives:

5.1.Identify various objects that are available in the browser.

5.2.Discuss the various methods and properties available for working with these objects.

5.3.Use these built-in objects within scripts to enhance the web page functionality.

6. Write scripts that store, retrieve, update, and delete data using browser localStorage, and trigger events using timers.

Objectives:

6.1.Explain how localStorage are used to store data in the browser and make it available to the server.

6.2.Use localStorage to persist web page data beyond a single request.

6.3.Demonstrate how to create, modify and delete localStorage.

6.4.Demonstrate the use of timers to control repetitive events in a script.

7. Use advanced JavaScript and DHTML features to add animation and complex functionality to pages.

Objectives:

7.1.Describe advanced features of JavaScript.

7.2.Use JavaScript to detect browser version and brand and provide browser independence.

7.3.Explain how the sizing and positioning features of Cascading Style Sheets can be accessed and modified from JavaScript to create dynamic animation features.

7.4.Use Cascading Style Sheet positioning, the Document Object Model, and JavaScript event handling to build dynamic HTML features into a web page.

8. Use Asynchronous JavaScript and XML techniques to make web pages dynamic.

Objectives:

8.1.Explain how tags such as “div” can be controlled to provide dynamic size, position, and contents changes after the main page is loaded.

8.2.Describe the mechanism used by JavaScript to communicate asynchronously with servers and update page regions.

8.3.Use AJAX techniques in scripts to dynamically update web page regions using asynchronous communications with web servers.

9. Use Object-Oriented patterns to structure web applications.

Objectives:

9.1. Create objects, assign properties and call methods.

9.2. Use inheritance to extend and modify behaviour of an existing class.

9.3. Create objects that perform useful functions in a web application context.

1.

10.Construct simple server-side programming scripts with variables and decision structures.

Objectives:

10.1.Explain the history and purpose of Node.js.

10.2.Demonstrate the ability to set up a Node.js environment.

10.3.Use operators, built-in functions, and control structures to generate a web page.

11. Apply modular design to scripts.

Objectives:

11.1.Explain the purpose of modular design.

11.2.Create Node.js modules and express code modularity in an application.

11.3.Use 3rd-party modules to add additional functionality to scripts.

12. Create scripts that receive and process web form data.

Objectives:

12.1.Explain the web application architecture that enables the construction of web applications.

12.2.Describe the request/response mechanism for transferring data between clients and servers.

12.3.Create web forms in an HTML page and receive the data in a Node.js script on the server.

12.4.Use regular expressions to validate form data in a server script, generating a success page or an error page depending on validity.

13. Construct web applications leveraging open source components to streamline development.

Objectives:

13.1.Install, configure and deploy a framework-based web application.

13.2.Describe the features and components that frameworks provide and when they would be used.

13.3.Use frameworks to handle routing, form validation, authentication and persistence.

13.4.Explain the MVC pattern and why you would use it.

14. Create and maintain databases on a database server.

Objectives:

14.1.Construct a small database with MongoDB.

14.2.Demonstrate features for creating and maintaining databases.

14.3.Insert, update, retrieve and delete data.

15. Create complex applications that interact with HTML forms and databases to dynamically generate web pages and store form data in a database.

Objectives:

15.1.Describe Mongoose database functions.

15.2.Demonstrate database connectivity using a Node.js script.

15.3.Retrieve data and generate a customized web page displaying the data.

15.4.Construct a script that receives form data from a web page and inserts it into a database collection.