**FLORIDA VOCATIONAL INSTITUTE**

**PROGRAM SYLLABUS**

­­­­­­­­­­­­­­­­­­­­­­­­­­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**PROGRAM TITLE: Review Date:**

**Web and Application Development Engineer**  **TBD**

**Program Objective/Description:**

As a Web Development Engineer, you will gain a wide array of fundamental and in-depth training on front end web development, as well as fundamentals of back end development. You will learn how to effectively write front-end programs which interact with servers and load asynchronously. You will also learn to design and implement graphical interfaces which follow best practices of UX design and are mobile-friendly. Your back-end experience will include an introduction to common design patterns.

**------------------------------------------------------------------------------------------------------------------------------**

**TRAINING OBJECTIVE**:

To build an understanding of designing, creating, and maintaining websites through different programming languages. ***The way the program is structured, students take WEB1010 first. After that, they must take WEB1000, WEB2000, WEB2010, WEB2020, WEB2040 but it will usually not be in this order. This is what we know as “the front end wheel” of courses. Once those six classes are completed, students move on to “the back end wheel” of courses and take WEB2900, WEB2910, ROR1000, WEB3000, and CAP2000, these will also be taken in scrambled order by most students.***

**WEB1010 Basic Front End Programing (JavaScript, HTML5, CSS)**

(**15 Theory/Clock Hours/45 Lab Hours/0 Externship Hours/2 Credit Hours)**

Students learn how to hack through documentation of CSS frameworks by using Bootstrap and Materialize to create grid-based layouts. They also learn the basics of material design, forms posting, intro to javascript (variables, if statements, loops, simple objects, and attacking event handlers), selecting elements with jquery, modifying the contents of elements with jquery, and sending ajax requests with jquery. The expected level of proficiency by the end of the course is being able to use a css framework to design a responsive grid, knowing enough code to solve simple algorithmic challenges like aggregating the elements of an array, and knowing how the request-response model works and thus being able to do very basic ajax programming using jQuery.

**WEB1000 Problem Solving**

(**15 theory/Clock Hours/45 Lab Hours/0 Externship Hours/2 Credit Hours)**

In this unit, students study techniques for web design and programming which have a root in mathematics. The goal of this unit is to get students to know how to emulate randomness, understand asynchronous programming and callbacks, use the modulus operator to restrict execution flows and avoid out of bounds exceptions, write effective if statements (by recognizing occurrences of DeMorgan’s laws as well as the distributive property of ands and ors), and really understand if-else chains and negation.

**WEB2000 Intermediate Front End Structure**

(**15 Theory/Clock Hours/45 Lab Hours/0 Externship Hours/2 Credit Hours)**

Students will learn about html5 semantic elements, basics of the shadow DOM, HTML 5 templating, work with the canvas object, design html emails using tables, performance optimization of webpages, and basics of UX design.

**WEB2010 Intermediate Front End Programming**

(**15 theory/Clock Hours/45 Lab Hours/0 Externship Hours/2 Credit Hours)**

This unit is devoted to learning advanced jQuery techniques such as event delegation, different types of ajax requests (including jsonp), interacting with a firebase back-end and also with the Wikipedia api. Students will also receive an introduction to node.js and work through developing a template for a site which implements authentication and authorization functions.

**WEB2020 Intermediate Styling Techniques**

(**15 theory/Clock Hours/45 Lab Hours/0 Externship Hours/2 Credit Hours)**

This course teaches students about the principles of web typography, picking font schemes, working with SVG graphics, using pseudo-elements ( ::after and ::before), use greesock animation suite to create css effects, and takes a deeper dive into bootstrap and learning about using it to easily create sliders and other components. Students will also have the opportunity to work on two ajax applications to fine-tune their front end programming skills.

**WEB2040 SQL Databases**

(**15 theory/Clock Hours/45 Lab Hours/0 Externship Hours/2 Credit Hours)**

This course emphasizes what developers need to know about SQL. Students go through ample practice with nested selects and joins, loading pre-existing datasets into MariaDB, locating and fixing errors in a table, understanding foreign keys and relationships between entities, and creating tables of appropriate data types. An intro to MongoDB is also given, with some basic exercises. This course also serves as a Node.JS introduction and AJAX refresher, as students develop an application which creates html visualizations of the outputs of several specific queries.

**WEB2900 Modern MV\* Front End Frameworks**

(**15 theory/Clock Hours/45 Lab Hours/0 Externship Hours/2 Credit Hours)**

In this course, students learn the basics of Angular JS and the MVC design pattern. They will develop two applications of their own using Angular.

**WEB2910 Modern MV\* Back End Frameworks**

(**15 theory/Clock Hours/45 Lab Hours/0 Externship Hours/2 Credit Hours)**

In this course, Students will take a deeper Dive into Node.JS. Students will learn how to handle request parameters, query strings, session data, and cookies. They will also learn how to do OAuth authentication with Facebook, Twitter, and Google, and how to connect to a database. An introduction to Object oriented programming in JS is also given.

**ROR1000 Fundamentals of PHP Development**

(**30 Theory/Clock Hours/90 Lab Hours/0 Externship Hours/3 Credit Hours)**

This course introduces the PHP programming language and also how to do the basic back end tasks (cookies, session data, request parameters, query string handling, forms handling, authentication and authorization) in php. It will also introduce students to developing laravel applications and give a survey of design patterms to solve the object-relational mapping problem.

**WEB3000 Agile and TDD (QA/Test)**

(**15 theory/Clock Hours/45 Lab Hours/0 Externship Hours/2 Credit Hours)**

In this course, students get a robust intro to the scrum implementation of agile, with an option to get certified. They also learn the Javascript implementation of some advanced programming concepts such as function purity, writing testable functions, loose coupling between functions, inheritance/polymorphism, and get plenty of practice with filter, map, and reduce as a gateway into understanding functional programming.

**CAP2000 Capstone Project II**

(**15 theory/Clock Hours/45 Lab Hours/0 Externship Hours/2 Credit Hours)**

The Capstone project is designed for the students to select a key project based on either PHP or Node.JS and through applied learning develop a portfolio project in a structured learning environment. The Capstone Project gives the student an opportunity to apply everything they have learned. The students will form teams, create project plans, conduct risk analyses, create test plans, and write software. Students will practice how to handle project and cost overruns, schedule overruns and inconsistent supporting technology. Students will learn how to account for scope creep, under-productive team members, and angry customers. Each individual on the team will get the opportunity to lead the team and will be in charge of one aspect of the project. All individuals will have to contribute to all aspects of the project, under the direction of that aspect's leader. The Coding phase class time will focus on the different types and styles of progress tracking meetings. True to the real world experience of software development, expect to have to adjust your plans and deliverables. The components the students will learn during this phase are, Project Selection, Team Selection, Role Assignment, and Planning, Technology Research, Feasibility Study, and Prototyping, Platform, Language, and DBMS Selection, Development Methodology, Architecture, and Framework Selection. This will culminate in Software Delivery, Presentation Preparation, and Software Project Presentations.