**SYLLABUS**

**Unit 1: Introduction To HTML And CSS**

**OBJECTIVES**

**HTML**

* Decide between a variety of text editors for writing code
* Identify the parts that make up an HTML
* Determine when to use specific HTML tags
* Correctly structure nested HTML content
* Form creation

**CSS**

* Identify the benefit of separating style from content
* Use CSS to style a website
* Test styles by manipulating CSS properties
* Use CSS references to lookup standard CSS properties and values

**Responsive Design Concepts With Bootstrap**

* Create your own responsive web page that works well on any device: phone, tablet, desktop or anything in between.
* Explore what makes a site responsive and how some common responsive design patterns work across different devices.
* Create your own responsive layout using the `viewport` tag and CSS media queries.
* Experiment with major and minor breakpoints
* Optimize text for reading.
* Build HTML elements for any screen size.
* Use the browser viewport to create consistent user experiences.
* Use media queries and breakpoints to create responsive web page designs
* Bootstrap classes
* Using Bootstrap’s Grid System (Containers, Rows, & Columns)
* Creating Columns & Adding Content
* Adjusting Column Sizes Across Screen Sizes
* Using Some of Bootstrap’s Components & Pre-Made Styles
* Nesting Grids
* Adding a Navbar & Other Components
* Showing & Hiding Elements at Specific Sizes
* Changing the Layout Across Screen Sizes

**UNIT 2: JAVASCRIPT BASIC**

In this part, you’ll understand the basic principles of the most used programming language on the web – JavaScript. At the end of this, you’ll be comfortable writing pure JavaScript codes.

* Storing and Tracking Information with Variables

**Conditional Statements & Functions**

* Introducing Conditional Statements
* Comparison Operators
* Boolean Values
* Combining Multiple Tests Into a Single Condition
* Introducing Functions
* Getting Information From a Function
* Giving Information to Functions
* Variable Scope

**Loops and Arrays**

* Simplify Repetitive Tasks with Loops
* Types of Loops
* What is an Array?
* Accessing, adding, removing Items in an Array
* Using For Loops with Arrays
* Useful Array Methods

**Unit 3: Object-Oriented JavaScript**

Objects in JavaScript encapsulate both data and functionality. You'll create, access, and modify objects to build a solid foundation for object-oriented programming.

**Objects in Depth**

* Access an object's properties
* Create objects using object literal notation
* Add properties to objects
* Write methods to access an object with the this keyword
* Compare an object with another object
* Identify global variables as properties of the window object
* Identify the risks of using global variables
* Extract properties and values from an object

**Classes and Objects**

* Model real-world "things" using object-oriented programming
* Write a constructor function to instantiate objects
* Identify various ways a function can be invoked, including each approach's effect on the value of this
* Leverage call , apply , and bind to manually set the value of this
* Access and add properties to an object's prototype
* Implement prototypal inheritance to base an object on another object

**ES6 Functions**

* With ES6, functions are getting some much-needed improvements. Learn a number of new things including arrow functions and classes.
* The JavaScript environment provides you with a number of features by default. You'll learn about Sets, Maps, Proxies, Generators, how iteration works, and more!

**UNIT 4: The DOM**

* What is the DOM?
* Select a Page Element By Its ID
* Select All Elements of a Particular Type
* Selecting Elements with the Same Class Name
* Using CSS Queries to Select Page Elements

**DOM and DOM Transversal**

* Getting and Setting Text with textContent and innerHTML
* Styling Elements
* Creating New DOM Elements
* Appending and removing Nodes

**Responding to User Interaction**

* What is an Event?
* Functions as Parameters
* Delaying Execution with setTimeout()
* Listening for Events with addEventListener()
* The Event Object

**Unit 5: The REACT Library**

**Why React?**

* Why React was built?
* Use react components to build complex functions from simple ones
* Recognize that React is just JavaScript

**Rendering UI with React**

* Use create-react-app to create a new React application
* Create reusable, focused Class components with composition
* Leverage JSX to describe UI

**State Management**

* Create reusable, focused Class components with composition
* Leverage JSX to describe UI
* Manage state in applications
* Use props to pass data into a component
* Create functional components focused on UI rather than behavior
* Add state to components to represent mutable internal data
* Use the “**this**” keyword to access component data and properties
* Update state with setState()
* Use PropTypes to typecheck and debug components
* Use controlled components to manage input form elements
* Conceptualize the lifecycle of a component
* Use React's componentDidMount lifecycle hook for HTTP requests

**Manage App Location with React Router**

* Use React Router to add different routes to applications
* Use state to dynamically render a different "page"
* Use React Router's Route component
* Use React Router's Link component

**Unit 6: Version Control and Deployment**

**Version Control, Git & GitHub**

* You'll learn about the benefits of version control and install the version control tool Git!
* Create a new repository from scratch
* Cloning an existing repository.
* Review an existing Git repository's history of commits.
* View files that have been modified.
* View changes that have been made.
* Make commits that are saved to the repository.
* Write descriptive commit messages.
* Verify the changes you're about to save to the repository.
* Add special markers called tags to commits.
* Work on isolated development tracks by making use of Git's branches.
* Combine branches together.
* Modify or undo changes that have been saved to a repository.
* Create remote repositories on GitHub.
* Get and send changes to a remote repository.
* Create copies of a project by forking another developer’s repository.
* Collaborate with other developers by contributing to a public project.
* Leverage pull requests to send suggested changes to another developer.
* Move or combine commits with `git rebase`.

**UNIT 9: PYTHON BASIC**

Python is a general purpose, dynamic, high-level, and interpreted programming language. It supports Object Oriented programming approach to develop applications. It is simple and easy to learn and provides lots of high-level data structures.

* Storing and Tracking Information with Variables
* Datatypes (String, Integer, Float, Boolean)
* Operators
* Input
* Conditional Statements
* Comparison Operators
* Introducing Python Functions
* Simplify Repetitive Tasks with Loops
* Types of Loops
* List
* Accessing, adding, removing Items in a List
* List Methods
* Tuple
* Sets
* Dictionaries
* Args and Kwargs
* Classes

**UNIT 8: DJANGO**

Django is a web application framework written in Python programming language. It is based on MVT (Model View Template) design pattern. This course will prepare you to use this framework to build dynamic and database driven websites.

* Client-Side Frameworks for Developing Modular Web Page Components
* Building Scalable Web Apps with Python
* Collecting form data from the front-end
* Add, delete, modify data in your database

**UNIT 9: Deployment**

Deploying to a production environment

* Deploying with github
* Deploying the netlify
* Deploying to Heroku

**WEEK ONE**

• **Introduction To HTML:**

* + Brief history of the Web
  + Elements
  + Tags
  + Attributes
  + Class activity – <https://codepen.io/w3devcampus/pen/pPaPXZ>
  + Class activity – <https://codepen.io/w3devcampus/pen/OmvMba>

• Images

• Hyperlinks

**• Introduction to CSS:**

• Introduction To CSS

• The style and link tags

• Selectors and Declarators

• ID and Classes

• Properties and values

• Common CSS properties –

* font-size
* line-height
* text-align
* text-decoration
* font-weight
* font-style and
* font-family.
* Colors
* Units
* Class Activity - https://codepen.io/w3devcampus/pen/YVeQZN
* Combining selectors
* Descendant selectors
* Precedence
* Class Activity
* The CSS box model
* Margins, Paddings and Borders
* Debugging
* Tables (HTML and Styling)
* Background images and stylings
* Decorative borders and shadows
* Class Activity
* Pseudo classes
* The Position properties

• Forms (HTML and Styling)

**WEEK TWO**

• Introduction to Bootstrap

• Downloading files and using CDN

• Containers

• Jumbotron

• Understanding the bootstrap grid system

• Bootstrap classes

• Activities

• Project

**WEEK THREE**

**• Version Control, Git & GitHub**

* Git (the version control software GitHub is built on)
* Github:
  + Create remote repositories on GitHub.
* Repositories:
  + Creating new Repo
  + Cloning Repo
* Branches:
  + Work on isolated development tracks by making use of Git's branches.
  + Combine branches together.
* Commits:
  + Write descriptive commit messages.
  + Make commits that are saved to the repository.
* Pull Requests:
  + Collaborate with other developers by contributing to a public project.
  + Leverage pull requests to send suggested changes to another developer.

**• Deployment**

* Deploying to a production environment

• **PROJECTS:**

* Creating a frontend project and deploying to netlify

**WEEK FOUR**

**• Introduction to JavaScript**

• **Variables**:

* Creating and naming variables

**• Data-types:**

* Strings
* Numbers
* Booleans
* Null
* Undefined
* Arrays
* Objects

**• String properties and Methods:**

* Length property
* toUpperCase
* toLowerCase
* indexOf
* slice

**• Number data-types:**

* Learn about number operators:

**• Math operators:**

* Addition,
* Subtraction,
* Multiplication,
* Division
* Incremental
* Decremental

• **Comparison Operators**: Determining equality and differences between variables

* Greater than >
* Less than <
* Equal to == ===

**• Logical Operators:** Determining the logic between variables and values

* Logical And &&
* Logical Or ||
* Logical Not !

**• Concatenation:**

* Combining strings and Variables
* Template Literals

**• Conditional Statements:**

* Introducing Conditional Statements
* If else Statement
* Ternary Operators
* Switch Statements

**• LOOPs:**

* Simplify Repetitive Tasks with Loops
* For Loops
* `do ... while` Loops
* While Loop
* Exiting Loops

**WEEK FIVE**

**• Introduction to Functions:**

* Introducing Functions
* Variable Scope
* Types of Functions:
  + Function Declarations
  + Function Expression
  + ES6 Arrow Function
* Getting Information From a Function
* Giving Information to Functions

• **Class Activities**

• **Arrays**:

* What is an Array?
* Accessing, adding, removing Items in an Array
* Using For Loops with Arrays
* Useful Array Methods:
  + Pop and Push
  + Shift and Unshift
  + IndexOf
  + Concat
  + ForEach
  + Map
  + Filter
  + Includes
  + Find
  + Reduce

• **Objects:**

* Access an object's properties
* Create objects using object literal notation
* Add properties to objects
* Remove properties from objects using the delete operator
* Write methods to access an object with the **this** keyword
* Compare an object with another object
* Identify global variables as properties of the window object
* Identify the risks of using global variables
* Extract properties and values from an object

**• JavaScript Classes**

* Model real-world "things" using object-oriented programming
* Write a constructor function to instantiate objects
* Identify various ways a function can be invoked, including each approach's effect on the value of this
* Leverage call , apply , and bind to manually set the value of this
* Access and add properties to an object's prototype
* Implement prototypal inheritance to base an object on another object

**• ES6 Functions:**

* Learn new ES6 syntax including arrow functions and classes.

**WEEK SIX**

• **Welcome to the DOM:**

* What is the DOM?
* Select a Page Element By Its ID
* Select All Elements of a Particular Type
* Selecting Elements with the Same Class Name
* Using CSS Queries to Select Page Elements

• **DOM and DOM Transversal:**

* Getting and Setting Text with textContent and innerHTML
* Styling Elements
* Creating New DOM Elements
* Appending and removing Nodes
* Using parentNode to Traverse Up the DOM
* Using previousElementSibling, nextElementSibling and insertBefore
* Getting All Children of a Node with children
* Getting the First and Last Child

• **Responding to User Interaction:**

* What is an Event?
* Functions as Parameters
* Delaying Execution with setTimeout()
* Listening for Events with addEventListener()
* Event Bubbling and Delegation
* The Event Object

**WEEK SEVEN**

• **PROJECTS:**

* Creating a frontend JavaScript project with all the knowledge that have been gained so far
* Deploy to Github and Netlify

**WEEK EIGHT**

**• Welcome to the DOM**

**• Rendering UI with React:**

* Create reusable, focused Class components with composition
* Leverage JSX to describe UI

**• State Management:**

* Create reusable, focused Class components with composition
* Leverage JSX to describe UI
* Manage state in applications
* Use props to pass data into a component
* Create functional components focused on UI rather than behavior
* Add state to components to represent mutable internal data
* Use the “**this**” keyword to access component data and properties

**WEEK NINE**

* Update state with setState()
* Use PropTypes to typecheck and debug components
* Use controlled components to manage input form elements
* Use create-react-app to create a new React application
* Conceptualize the lifecycle of a component
* Use React's componentDidMount lifecycle hook for HTTP requests

**• Manage App Location with React Router:**

* Use React Router to add different routes to applications
* Use state to dynamically render a different "page"
* Use React Router's Route component
* Use React Router's Link component

**WEEK TEN**

**• Project**

**WEEK ELEVEN**

**• Introduction To Python**

* **Installation**

**• Introduction To Python**

* **Python Syntax:** Python syntax can be executed by writing directly in the Command Line Or by creating a python file on the server, using the .py file extension, and running it in the Command Line.
* **Python Indentation:** Indentation refers to the spaces at the beginning of a code line. Where in other programming languages the indentation in code is for readability only, the indentation in Python is very important.
* **Python Variables**
* **Datatypes:**
  + **Text Type: str**
  + **Numeric Types: int, float, complex**
  + **Sequence Types: list, tuple, range**
  + **Mapping Type: dict**
  + **Set Types: set, frozenset**
  + **Boolean Type: bool**

**• Python Functions:**

* Introducing Functions
* Variable Scope
* Getting Information From a Function
* Giving Information to Functions

• **Class Activities**

**WEEK TWELVE**

• **Dictionary:**

* Access a dictionary property
* Create objects using object literal notation
* Add properties to dictionary
* Remove properties from dictionary using the delete operator
* Extract properties and values from a dictionary

**• Python Classes**

* Model real-world "things" using object-oriented programming
* Write a constructor function to instantiate Classes
* Identify various ways a function can be invoked.
* Access and add properties to a Class prototype
* Implement prototypal inheritance to base a Class on another object

**WEEK THIRTEEN**

* **Virtual Environment**
  + Setting up a Virtual Environment
* **Setup Development and Environment:**
  + Django Installation
* **Django Project:**
  + Starting a project
  + Migrating
  + Running Server
* **Python Packages (PIP):**
  + Downloading and installing packages
  + Tracking packages
  + Version and Dependency management
* **Creating Apps:**
  + Installing Apps
  + settings
* **Model:**
  + Types of Database
  + Schemas
  + DB SQLite
  + Creating Model Classes
* **Accessing Responses:**
  + write
  + send
  + sendFile
  + end

**WEEK FOURTEEN**

* **Views:**
  + Types of Views:
    - Class based Views
    - Function Views
* **Routing & Middleware:**
  + Path
* **Templates:**
  + The site logic
  + Gluing models and views.

**WEEK FIFTEEN**

* **Form and Form Validation:**
  + Exchanging data between the frontend to backend
* **Request Data:**
  + Get
  + Post
  + Delete
* **Static Files:**
  + Linking CSS
  + Adding JS
  + Images
* **Media Files:**
  + Storing media files in Database
  + Storing files on AWS
* **Deployment:**
  + Heroku deployment

**WEEK SIXTEEN**

* **Project**