FASTER REACHABLE UI development Using JAVASCRIpt ,REACT and NODE

Contents

[VENDOR NAME 2](#_Toc106134176)

[PROGRAM NAME 2](#_Toc106134177)

[Program Summary 2](#_Toc106134178)

[DURATION DAYS 2](#_Toc106134179)

[Cost 2](#_Toc106134180)

[Target Audience 2](#_Toc106134181)

[Prerequistes 2](#_Toc106134182)

[Lab Setup 3](#_Toc106134183)

[Terminal Objectives – what will the learner be able to ‘do’ at the end of the program? 3](#_Toc106134184)

[content outline 4](#_Toc106134185)

[Day 1-2 4](#_Toc106134186)

[How to Write Manageable JavaScript Code 4](#_Toc106134187)

[Introduction to TypeScript and ES6, ES 7 5](#_Toc106134188)

[The Challenges with SPA 6](#_Toc106134189)

[Introduction 7](#_Toc106134190)

[Building and Serving 7](#_Toc106134191)

[Introduction 7](#_Toc106134192)

[Organizing Features and Modules 7](#_Toc106134193)

[Components 8](#_Toc106134194)

[Services and Dependency Injection 8](#_Toc106134195)

[Structuring Components 8](#_Toc106134196)

[Templates 9](#_Toc106134197)

[Directives 10](#_Toc106134198)

[Asynchronous Operations 11](#_Toc106134199)

[Testing Overview 12](#_Toc106134200)

[Assessment approach 12](#_Toc106134201)

[LAB SETUP 12](#_Toc106134202)

# Client – Name

Tredence

# PROGRAM NAME

FASTER REACHABLE UI development Using JAVASCRIpt ,REACT and NODE

# Program Summary

The Full-Stack Developer training program begins with JavaScript and ES 6 Specification, React and Node.js, which provide a solid foundation to understand and build web applications with the help of JavaScript. Next, you will gain a comprehensive understanding of Express js to develop Microservices (Restful services).. We have included AWS Devops Pipeline to setup CI/CD pipeline to deploy full stack application

# DURATION DAYS

30 Days

# Target Audience

* Web Programmers

# Prerequistes

* Web Architecture
* HTTP basics
* CSS
* HTML

# Lab Setup

* Microsoft Visual Studio / Core /WebStrom / Eclipse
* Angular CLI
* Node JS
* Chrome Browser with Angular Inspector and Angular Augury extensions
* Internet Connectivity (Proxy Free)

# Terminal Objectives – what will the learner be able to ‘do’ at the end of the program?

* Thoroughly understand core mechanisms of the JavaScript language, such as scope, closures, and inheritance
* Introduction to ES6 , TypeScript
* Exploring React Js to build Faster Web Views
* Components Basics (props ,state and Lifecycle)
* To create smaller reusable components from complex components
* Understand how virtual Dom works
* Understand Component Communication Techniques
* Content Based Component Composition
* High Ordered Component to Decorate Components
* How to Choose B/w Container Components vs Presentational Components
* Understand Property Drilling v/s React Context Usage
* Usage of Provider Pattern and High Ordered Components
* Explore usage of Pure Components
* Learn unidirectional data flow architecture Flux and Redux to allow components to interactivity
* Learn How to test React Components using Jest
* Use JavaScript for building Web Api Services
* Devops Pipeline setup using GitHub Actions

# content outline

### How to Write Manageable JavaScript Code

Reintroducing JavaScript

* Variable Scope
* Variable Hoisting
* Advanced Variable Hoisting and Execution Context
* The Scope Chain
* Prototype based Object Oriented Programming
  + Prototype Chain
* Inheritance
  + New syntaxes
  + Old Syntax
  + Pseudo classical inheritance
  + Patterns
* Deep into Functions
  + Functions and Anonymous Functions
  + “this” magic
  + Self-Executing Anonymous Functions
  + The Module Pattern
    - Module
  + Closures

### Introduction to TypeScript and ES6, ES 7

* Programming Languages for Use with Angular
* TypeScript Syntax
* The Type System – Defining Variables
* The Type System – Defining Arrays
* The Type System – Classes & Objects
* Class Constructors
* Class Constructors – Alternate Form
* Interfaces v/s Class
* Parameter and Return Value Types
* Working with Modules
* TypeScript Transpilation
* Arrow Functions
* Template Strings
  + Template Strings – Variables and Expressions
  + Template Strings – Multiline
* Generics
  + - Class
  + Methods
  + Restricting Types
  + Built- in Typescript Generic Types
    - Readonly
    - ReadonlyArray
    - ReturnType
    - Partial
    - Required
    - NonNullable
    - Pick
    - Record
    - Extract
    - Exclude
    - Omit

### Node.Js (Server Side JAVASCRIPT Programming)

**Essential Node**

* Understanding the Basics of Node
* How and where Node gets used
* The Node.js framework
* Using Node.js to execute scripts
* Advantages of the event loop
* Install Node and the Node Package Manager (NPM)

**Node Projects**

* The Node Package Manager
* Creating a project
* The package.json configuration file
* Global vs. local package installation

**Modules**

* Modularization
* The CommonJS and RequireJS specifications
* The ES 6 Modules using Export and Imports
* Defining modules with exports
* Modules are singletons
* Creating a package
* Module scope and construction
* Using Built In Modules (fs ,util, os ,events etc)

### API Design Using NODE and KOA | EXPRESS |RESTIFY

* Building REST API’s
* Publishing Get, Post, Put, Delete endpoints
* Enabling CORS
* Integration with Data Layer
* Hosting Services using Node Js

### How to setup JavaScript Development Environment

* webserver setup
* Bundling using webpack
* Babel as a Transpiller
* static code analysis using lint
* setup unit test runner
* Automate tasks using Npm Scripts/Grunt/Gulp

### **Building Fast Views Using React**

**Review of React**

* What is it?
* Key React concepts and components
* Building a Hello World UI

**React and ES2015**

* ES2015 features which impact React
* How to create components with ES2015
* Using Babel for ES2015 and JSX
* Using WebPack with React
* Using Immutable.js

**Parameter zing components**

* Disambiguating component properties and state
* Inserting properties into rendered output with this.props
* Passing property values using JSX { } expressions
* Initializing default values with getDefaultProps()

**Manipulating component state**

* Setting starting state values with getInitialState()
* Reading application data with this.state
* Mutating data with this.setState()
* Harnessing React autobinding and event aggregation

**Integrating into the component life cycle**

* Attaching and detaching logic and event handlers in componentDidMount and componentWillUnmount
* Enhancing performance with componentShouldUpdate
* Responding to changes with componentWillReceiveProps

**Advanced React Components**

* Component Life-Cycle
* Virtual DOM
* Events
* State Machines
* Compositions
* Communication Between Components

**Working with the DOM**

* Referencing DOM Elements
* DOM Manipulation

**Manipulating form input components**

* Working with interactive properties
* value
* defaultValue
* checked
* selected
* Capturing form component updates with onChange
* Comparing controlled and uncontrolled components

**Simulating two-way binding**

* Applying the onChange / setState() convention
* Employing the LinkedStateMixin
* Passing ReactLink objects through the props hierarchy

Component Design Patterns

* JSX spread attributes
* Children types (?)
  + Array as children
  + Function as children
* Render prop
* Proxy component
* Event switch

State hoisting

**Connecting React to REST services**

* Making Ajax calls from React
* Mapping CRUD access to REST actions
* GET
* POST
* PUT
* DELETE

**Dashboard and Data Visualization With React**

* Using Chart.js in React
* DevExpress Data Visualization Components
* React dashboards

**Performance Optimization and Best Practices**

* Profiling Components in Browser
* Benefits of Pure Components
* Designing Components to Subscribe for Partial State
* Usage of Immutable Data Structures
* Reacts Context Api usage

### **Modeling client-side data interactivity Using Flux Architecture**

* Identifying Flux participants
* Actions
* Dispatcher
* Stores
* Views
* Harnessing unidirectional data flow
* Comparing Flux to Model View Controller, Redux and Relay

**Coding a Flux application**

* Extending the Facebook Dispatcher
* Utilizing Node.js EventEmitter with custom stores
* Integrating REST access with Flux action creators

**Integrating routing into Flux**

* Exploiting the react-router for semantic URLs
* Mapping URL paths and parameters to components

**Redux**

* Explore differences between Redux and normal Flux
* Reducer Functions
* Working with one store
* Using Middleware (SAAGA)
* Advantages and Disadvantages of Redux
* Using react-redux
  + Provider Component
  + Connect Component
  + Selector pattern

**Introduction to Testing Using Jest**

* Understanding the Jest Api and Enzyme
* Defining Tests
* Mocking Functions
* Unit Testing React Components
* Snapshot Testing
* Testing with React Testing Library
* Testing components directly
* Testing with Test Utilities
* Testing with Test Renderer