# Full Stack Training Including Nest JS and Next JS along with PostgreSQL

## Duration: 15 Days (6 hours/day)

High Level View of the Topics:

* HTML, CSS
* JavaScript and TypeScript
* Node JS and Express JS
* Nest JS and Service API development using Hexagonal Pattern
* PostgreSQL
* TypeORM Integartion with Nest JS
* React JS
* Next JS

**Detailed Curriculum:**

### Week-1

Day-1: JavaScript

Module1: JavaScript Fundamentals

* Introduction to JavaScript
* Understanding script tag
* Variables and data types
* Operators and control flow statements i.e. loops, if-else etc
* Strings and Arrays
* Function declaration
* Function parameters and arguments
* Scopes- local and global
* Function expression and anonymous functions
* Object literals
* Object constructors
* Behavior of ‘this’ keyword in constructors
* Best practices
* Expressive JavaScript
* The Flexibility of JavaScript
* JavaScript as a Loosely Typed Language
* Functions as First-Class Objects
* Functions In Depth
* Function Objects
* Function Literal
* Function Invocation Patterns
* Recursion & Closures
* Functions as Callbacks
* Events
  + Different ways of event handling
  + Understanding the event object
* Ajax
  + Characteristics & issues of conventional web applications
  + Ajax – need & essential features
  + Basic Ajax process - states, methods and properties of XMLHttpRequest

Module 2: Object Oriented JavaScript

* [Introduct](about:blank)ion
* [Introduction to Object Oriented JavaScript](about:blank)
* [Object-oriented programming](about:blank#Object-oriented_programming)
* [Prototype-based programming](about:blank#Prototype-based_programming)
* [The object (class instance)](about:blank#The_object_(class_instance))
* [The constructor](about:blank#The_constructor) function
* [The property (object attribute)](about:blank#The_property_(object_attribute))
* [The methods](about:blank#The_methods)
* [Inheritance and the prototype chain](about:blank)
* [Encapsulation](about:blank#Encapsulation)
* [Abstraction](about:blank#Abstraction)
* [Polymorphism](about:blank#Polymorphism)
* [Closures](about:blank)
* [Strict mode](about:blank)
* This keywod

Day-2: JS New Features and TypeScript

Module-3: ES6 Features

* Arrow Operator
* Classes
* Enhanced object literals
* Template strings
* De-structuring
* Default, rest and spread
* Let and const
* Iterators and for..of
* Generators
* Modules
* Module loaders
* Map + set + weakmap + weakset
* Proxies
* Symbols
* Promises

Module-4: INTRODUCTION TO TYPESCRIPT

* Why use TypeScript?
* TypeScript Features
* TypeScript Syntax, Keywords, and Code Hierarchy
* Tooling and Framework Options
* Tooling and Framework Options - Visual Studio Code
* Tooling and Framework Options - TypeScript Compiler
* Hello World Example

Module-5: TYPING, VARIABLES, AND FUNCTIONS

* Overview
* Grammar, Declarations, and Annotations
* Type Inference
* Grammar
* Static and Dynamic Typing
* Compile Time or Run Time
* Ambient Declarations and Type Definition Files
* The Any Type and Primatives
* Applying Types
* Objects
* Functions
* Arrow Functions and Debugging
* Functions and Interfaces
* Static Typing Recap
* Optional and Default Parameters
* Rest Parameters
* Demo: Defining Parameters
* Function Overloads
* Demo: Using Function Overloads

Module-6: CLASSES AND INTERFACES

* Introduction
* Defining Classes
* Demo: Defining Classes
* Demo: Property Limitations
* Casting and Type Definition Files
* Demo: Casting and Type Definition Files
* Extending Types
* Demo: Extending Types
* Using Interfaces
* Demo: Using Interfaces
* Extending an Interface
* Demo: Extending an Interface
* Static Members
* Method overriding
* Why no static classes?
* Protected members
* Class Expressions
* Abstract class and members
* Relationship between classes

Module-7: GENERICS

* Introduction and Overview
* What are Generics and Type Parameters?
* Using Array <T>
* Generic Functions
* Demo: Creating and Using Generic Functions
* Generic Interfaces and Classes
* Demo: Creating and Using a Generic Class
* Generic Constraints
* Demo: Applying Constraints to Generics

Module-8: COMPILER OPTIONS AND PROJECT CONFIGURATION

* Introduction and Overview
* Specifying Compiler Options
* Demo: Using Compiler Options
* Role and Structure of tsconfig.json
* Demo: Managing a Project with tsconfig.json

Day-3: Node JS and Express JS

Module-9: Node JS

* What is NodeJS?
* Why Should I learn or use NodeJS?
* Setting up NodeJS
  + Installing NodeJS
  + Understanding NodeJS console
* Understanding how JS differs on the front-end and back-end
* NPM and Various NPM commands
* First Node App
* V8 JavaScript Engine and Node Core
* Modules, Exports, and Require
  + Creating our own Module
  + Module - export and require
  + Module Patterns
  + Exports vs module.exports
  + Modules in ES6
* Events and Streams
  + Non-Blocking IO/ Asynchronous Execution
  + Blocking IO / Synchronous Execution
* Events
  + Event Emitter
  + Event Loop
* Accessing File System
  + Process Object
  + Interacting with the file system
* OS Module
* Interacting with the Web
  + Making web requests in Node
  + Web server in Node

Module-10: Express JS - Introduction

* Using Express Framework
* Starting Express App
* Understanding Middleware
* Router
* Handling HTTP Requests
* Sending HTML Responses using template engine
* Making REST API

Day-4: Nest JS

Module-11: NestJS Basics

* Introduction to NestJS
* Installing the NestJS CLI (command-line interface)
* Generating our first NestJS Application
* What’s inside a NestJS Application
* Project Set-up
* Typescript Configuration
* Creating a Controller
* Starting Up a Nest App
* File Naming Conventions
* Routing Decorators

Module-12: Generating Project with Nest CLI

* App Setup
* Using the Nest CLI to Generate Files
* More on Generating Files
* Adding Routing Logic

Module-13: Validating Request Data with Pipes

* Accessing Request Data with Decorators
* Using Pipes for Validation
* Adding Validation Rules
* Behind the Scenes of Validation
* How Type Info is Preserved

Day-5: NestJS (continued)

Module-14: Nest Architecture: Services and Repositories

* Services and Repositories
* Implementing a Repository
* Reading and Writing to a Storage File
* Implementing a Service
* Manual Testing of the Controller
* Reporting Errors with Exceptions
* Understanding Inversion of Control
* Introduction to Dependency Injection
* Refactoring to Use Dependency Injection
* Few More Notes on DI

Module-15: Organizing Code with Modules

* Project Overview
* Generating a Few Files
* Setting Up DI Between Modules
* More on DI Between Modules
* Consuming Multiple Modules
* Modules Wrap-up

Module-16: Add PostgreSQL with TypeORM

* Before we Get Started
* Running PostgreSQL
* Introducing the TypeORM Module
* Creating a TypeORM Entity
* Use Repository to Access Database
* Create a Relation between two Entities
* Retrieve Entities with their Relations
* Using Cascading Inserts and Updates
* Creating and Saving data
* Querying for Data
* Updating Data
* Finding and Filtering Records
* Removing Records
* Updating Records
* A Few Notes on Exception

### Week-2

Day-6: NestJS (continued)

Module-17: Custom Data Serialization

* Excluding Response Properties
* Solution to Serialization
* How to Build Interceptors
* Serialization in the Interceptor
* Customizing the Interceptor's DTO
* Wrapping the Interceptor in a Decorator
* Controller-Wide Serialization
* A Bit of Type Safety Around Serialize

Module-18: Managing App configuration:

* Introducing the Config Module
* Custom Environment File Paths
* Schema Validation
* Using the Config Service
* Custom Configuration File
* Configuration Namespaces and Partial Registration
* Asynchronously Configure Dynamic Modules

Day-7: NestJS and Hexagonal Pattern

Module-19: Hexagonal architecture

* Principles of Hexagonal Architecture
* How is the code organized inside and outside?
* At the Runtime
* Dependencies inversion on the right
* Why an Interface on the left?
* Testing in Hexagonal Architecture
* To go further ahead
* Creating NestJS service APIs using Hexagonal Pattern

Module-20: Testing

* Unit Testing
  + Testing Overview
  + Testing Setup
  + Getting TypeScript to Help with Mocks
  + Quick Note to Help Speed Up Your Tests
  + Improving File Layout
  + Ensuring Password Gets Hashed
  + Changing Mock Implementations
  + Testing the Sign-in Flow
  + Checking Password Comparison
  + More Intelligent Mocks
  + Refactoring to Use Intelligent Mocks
  + Unit Testing a Controller
  + More Mock Implementations
  + Not Super Effective Tests
  + Testing the Sign-in Method
* Integration Testing
  + Getting Started with End to End Testing
  + Creating an End to End Test
  + App Setup Issues in Spec Files
  + Applying a Globally Scoped Pipe
  + Applying a Globally Scoped Middleware
  + Solving Failures Around Repeat Test Runs

Day-8: React JS

Module-21: Introducing React

* Introduction
* Advantages
* Architecture
* Why React JS?
* Difference between React JS and Other JavaScript frameworks and libraries
* Virtual DOM
* Benefits of using Virtual DOM
* Summary

Module-22: Creating a simple React App without any tool and configurations

* Packages required for creation
* Installing packages using NPM
* Choosing transpiler: Babel
* Different loaders: babel-loader, css-loader, style-loader, file-loader etc.
* Loading JSON using WebPack
* Adding CSS to webpack Build
* Bundling, minification: Webpack
* Webpack configuration for development
* Using a development server: Webpack-dev-server
* Different Plugins: HtmlWebpackPlugin, CleanWebpackPlugin etc. and their purpose
* Using webpack-dev-middleware along with express server to host and run application

Module-23: Creating a simple React App with the help of create-react-app tool

* Generating a React Project
* Why Create React App
* Exploring a Create-React-App Project
* Starting and Stopping a React App
* JavaScript module system
* Displaying Content with Functional Components

Module-24: JSX

* Introduction
* What is JSX?
* Not using JSX
* JSX for Components
* Just-in-time JSX Transformer
* Pre-process Transformer
* Converting HTML to JSX
* Inline Styling with JSX
* Converting Styling to JSX Format
* Class vs ClassName
* Referencing JS Variables in JSX
* Values JSX Can't Show
* Finding Forbidden Property Names
* Attribute Expressions
* Child Expressions
* Whitespace
* Un-escaping Content
* Returning adjacent elements from a component

Module-25: Communication Using Props

* Three Tenets of Components
* Semantic UI CDN Link
* Some Free Styling
* Naive Component Approach
* Specifying Images in JSX
* Duplicating a Single Component
* Extracting JSX to New Components
* Component Nesting
* React's Props System
* Passing and Receiving Props
* Passing Multiple Props
* Component Reuse
* Showing Custom Children

Day-9: React JS (continued)

Module-26: Class Based Components in React JS

* Class-Based Components
* Scaffolding the App
* Handling Asynchronous Operations with Functional Components
* Refactoring from Functional to Class Components

Module-27: State in React

* The Rules of State
* Initializing State through Constructors
* Updating State Properties
* App Lifecycle Walkthrough
* Handling Errors Gracefully
* Conditionally Rendering Content

Module-28: Building List of Records – Lists and Keys

* Rendering Lists
* Review of Map Statements
* Rendering Lists of Components
* The Purpose of Keys in Lists
* Implementing Keys in Lists

Module-29: Lifecycle Methods

* Introducing Lifecycle Methods
* Why Lifecycle Methods?
* Refactoring Data Loading to Lifecycle Methods
* Different lifecycle hooks
* When to use which life cycle methods
* Advantages and disadvantages
* Stop unnecessary rendering of functional component using memo()
* Stop unnecessary rendering of class component

Day-10: React JS (continued)

Module-30: Forms and Events

* Application Overview for this Module
* Component Design
* Adding Some Project Structure
* Showing Forms to the User
* Adding a Touch of Style
* Creating Event Handlers
* Alternate Event Handler Syntax
* Uncontrolled vs Controlled Elements
* More on Controlled Elements
* Handling Form Submittal
* Solving Context Issues
* Communicating Child to Parent
* Invoking Callbacks in Children

Module-31: Making API Requests with React

* Fetching Data
* Axios vs Fetch
* Viewing Request Results
* Handling Requests with Async Await
* Setting State After Async Requests

Module-32: Using Refs for DOM Access

* ‘ref’ prop of a React Element
* Callback Ref technique
* createRef() method
* reference to a React element
* reference to a class component
* forwardRef()

Module-33: Advanced Concepts: HOC and Render Props

* Mixins in React
* What is HOC?
* How to create HOC
* Built-in HOCs (memo, forwardRef etc.)
* Advantages and Disadvantages of HOC
* Render props
* What is the difference between Render prop and HOC?
* Example
* Advantages and disadvantages

Module-34: Handling errors in React

* Errors in React
* Using try…catch
* Error boundary concept
* Using HOC for error handling
* Best practices

### Week-3

Day-11: React JS (continued)

Module-35: React Routing

* Introducing React Router
* How React Router Works
* How Paths Get Matched
* Navigating with React Router
* [Optional] - Different Router Types
* Wiring Up Routes
* Link and ActiveLink component
* Using Route parameters
* Nested Routes

Module-36: Introduction to Redux

* Introduction to Redux
* Why Redux?
* Modeling with Redux
* Introducing Actions and Action Creators
* Providing a Store and Introducing Reducers
* Rules of Reducers
* Combining reducers

Module-37: Integrating React with Redux

* React Cooperating with Redux
* React, Redux, and... React-Redux?
* Design of the Redux App
* How React-Redux Works
* Redux Project Structure
* Wiring Up the Provider
* The Connect HOC Function
* Configuring Connect with Mapping State to Props
* Mapping Dispatch to Props
* Building a List with Redux Data
* Calling Action Creators from Components
* Functional Components with Connect
* Conditional Rendering
* Connecting Components to Redux

Day-12: React JS (continued)

Module-39: Async Actions with redux-thunk and Logging Redux store

* How to Fetch Data in a Redux App
* Wiring Up an Action Creator
* Making a Request from an Action Creator
* Understanding Async Action Creators
* More on Async Action Creators
* Middleware in Redux
* Behind the Scenes of redux-thunk
* Shortened Syntax with redux-thunk
* Logging Redux store using Redux dev-tools extension

Module-40: Hooks in React

* Introducing Hooks
* Building Stateful Class Components
* The UseState Hook
* UseState in Detail
* Lifecycles with UseEffect
* Actual Code Reuse
* Code Reuse with Hooks
* Redux hooks: useDispatcher, useSelector
* Routing Hooks: useHistory, useParams, useLocation

Module-41: Testing React Application

* Jest
* Enzyme
* Cypress

Day-13: Nest JS

Module-42: Introduction

* Overview
* Environment Setup
* Features
  + Pages
  + Static File Serving
  + Meta Data
  + CSS Support
  + Global CSS Support
  + Pre-Rendering
* Project Structure
* Simple Hello world example

Module-43: Routing

* Routing with Pages
* Nested Routes
* Dynamic Routes
* Nested Dynamic Routes
* Catch All Routes
* Link Component Navigation
* Custom 404 Page

Day-14:

Module-44: Pre-rendering and Data Fetching

* Pre-rendering
* Static Generation
* Static Generation with getStaticProps
* Pages vs. components
* More on getStaticProps
* Inspecting Static Generation Builds
* SSG with Dynamic Parameters
* SSG with getStaticPaths
* Inspecting getStaticPaths Builds
* Fetching Paths for getStaticPaths
* getStaticPaths fallback false
* getStaticPaths fallback true
* getStaticPaths fallback blocking
* Incremental Static Regeneration
* Inspecting ISR Builds
* Server-side Rendering
* SSR with getServerSideProps
* SSR with Dynamic Parameters
* getServerSideProps context
* Inspecting SSR Builds
* Client-side Data Fetching
* SWR for Client-side Data Fetching
* Pre-rendering + Client-side Data Fetching

Module-45: API Routes

* Introduction to API Routes
* API GET Request and POST Request
* Dynamic API Routes
* API DELETE Request
* Catch All API Routes
* APIs and Pre-rendering

Module-46: Styling

* Global Styles
* Component Level Styles
* SASS Support
* CSS in JS

Day-15: Next JS (continued)

Module-47: Miscellaneous

* App Layout
* Head Component
* Image Component
* Absolute Imports & Module Paths
* Static HTML Export
* TypeScript Support
* Preview Mode
* Redirects
* Environment Variables

Module-48: Authentication

* Authentication in Next JS
* NextAuth Set-Up
* Sign-in and Sign-out
* Client-side Authentication
* Securing Pages Client-side
* NextAuth Provider
* Server-side Authentication
* Securing Pages Server-side
* Securing API Routes
* Connecting to a Database
* Callbacks

Module-49: Deployment

* Deploying Next.js Apps to Vercel

*Note: Additionally, Service worker with IndexedDB, PWA also to be covered with React JS*