TOPSTECHNOLOGIES
Training | Outsourcing | Placement | Study Abroad

Python Lectureflow

Module-1) SE - Overview of IT Industry	**5**
• Introduction of students • Career in IT • Understanding Student Login of TOPS ERP • Using Lab • What is Program • What is programming? • Types of Programming Language • World Wide Web • How Internet Works • Network Layers on Client and Server • Client And Servers • Types of Internet Connections • Protocols • Application Security • Software Applications and its types • Software Architecture • Layers in Software Architecture • Software Environments • Types of Programming Languages • Source Code • Github and introductions • Student Account in Github • Types of Software • Introduction of Software • Application software • Software development process • Software Requirement • Software Analysis • System Design • Software Testing • Maintenance • Development • Web Application • Designing • moble application • DFD • Desktop Application • Flow Chart	

Module-3) SE - Fundamentals of Programming	15

- Basic Syntax
- Data Structures
- Variables
- Operators
- Control and looping Structures
- functions
- Arrays and strings
- Introduction to C
- What is Language?
- What is programming and program?
- Fundamental of Algorithms and Flowchart
- Real world problems - get solution via programs
- Practical Example: 1. Write a Flow chart of real problems - Days to month conversion system.
- Data Types and Variables - Data Types, Void Data Types,
- History of C
- Compiler and interpreter
- environment setup
- Type Modifiers,
- Basic Structure of C Programs
- Importance of C
- Fundamentals of C
- Difference between turbo C and Dev C/C++
- Practical Example : 1. Write a program of scanf 2. Write a program to demonstrate escape sequence 3. Write a program to demonstrate comments
- Comments
- Keywords
- Escape Sequence
- Practical Example: 1. Write a program to print (Hello World). 2. Write a program to print the sum of two numbers. 3. Write a program to exchange values of two variables using the 3rd variable. 4. Write a program to convert days into years and years into days.

Module-4) OOP Concept	8

- Procedure Oriented And object Oriented Programming
- Basic Concepts of OOP
- OOP - Objects and Classes
- Constructors and Destructors
- Data Abstraction and Encapsulation
- inheritance
- Encapsulation
- Types of polymorphism
- Dynamic Binding

- Array
- Types of constructors
- Compile time
- Types of Array
- Class and arrays : 1) Array within class 2) Array of objects
- Run time
- String
- Practical Example: 1. Write a program to print the score card of two students using an array of objects.
- Difference between constructor and destructor
- Practical Example: 1. Write a program to demonstrate difference between constructor and destructor 2. Write a program to demonstrate copy constructor
- Abstract class
- Practical Examples: 1. Write a program to check whether entered number is even or not using if..else statement in C++ 2. Write a menu - driven program to calculate the area of the circle,rectangle and triangle. 3. Write a program to calculate factorial of given number using for loop 4. Write a program to print the fibonacci series using while loop 5. Write a program to check whether the given number is palindrome using do..while loop. 6. Write a program to demonstrate jumping statements
- Practical Example: 4. Write a program to demonstrate pass object to a function 5. Write a program to demonstrate return object from function
- Class and pointer
- Aggregation
- Class and objects
- Practical Example: 1. Write a program to demonstrate pointer with class 2. Write a program to demonstrate dynamic object using new keyword
- Access modifiers
- Practical Example: 1. Write a program to demonstrate function overloading with different types of arguments 2. Write a program to demonstrate function overloading with default arguments 3. Write a program to show the constructor function overloading
- Member Function
- Types of inheritance 1 - Single level 2 - Multi-level 3 - Multiple 4- Hierarchical 5- Hybrid
- Comparisons of class and object
- Practical Example : Write a program to implement single level inheritance 2. Write a program to demonstrate single level inheritance in private mode 3. Write a program to demonstrate the ambiguity in single level inheritance 4. Write a program to demonstrate multilevel inheritance 5. Write a program to demonstrate multiple inheritance 6. Write a program to demonstrate the hierarchical inheritance 7. Write a program to demonstrate the hybrid inheritance
- Namespace
- Static Keyword
- Practical Example: 1) Write a program to demonstrate constructor invocation in inheritance
- Scope resolution operator

Module-5) SE - Database - Full stack and Back end	**15**

- What is Database
- DBMS and RDBMS
- Types of Database
- Normalization
- algebra
- Primary key
- foreign key
- unique key
- Database Programming Language SQL
- SQL Statements Types
- DDL
- DML
- TCL
- TQL
- Database backup and Restore
- What are Joins
- Types of Joins
- Function
- Procedure
- Trigger
- Curser
- Transaction concepts
- properties of transactions
- rollback and commit savepoint
- ER database schema

Module 6) WD - HTML - Full stack and Back end	**10**

- Student Intro , Career Center Login ,What is Internet, HTTP/HTTPS, WWW, Domain name and Top Domain name
- SEO, What is HTML, What is Text Editor, Web Browser, Downloading Text Editor , HTML Structure, First Program in HTML
- 1) HTML Introduction 2) HTML Getting Started 3) HTML Elements 4) HTML Attributes 5) HTML Basic Tags
- 1) HTML Doctypes 2) HTML Layout 3) HTML Head 4) HTML Meta 5) HTML Scripts
- Practical Examples: 1) Create any simple web page to display your name. 2) Importance of meta tag and Doctypes
- Tags and self Closing Tags, Basic Tag , Attribute and Events, Marquee Tag
- HTML - Meta Tags, HTML - Comments, HTML - Images, HTML - Tables, HTML - Lists, HTML - Text Links, HTML - Image Links
- HTML Headings HTML Paragraphs HTML Links HTML Text Formatting HTML Styles HTML Images
- HTML - Frames, HTML - Iframes, HTML - Blocks, HTML - Backgrounds, HTML - Colors, HTML - Fonts

- Anchor Tag, Img Tag, Image Mapping
- HTML - Fonts, HTML - Forms, HTML - Embed Multimedia ,HTML - Marquees, HTML - Header, HTML - Style Sheet, HTML - Javascript ,HTML - Layouts
- List Tag, Tables, Forms
- HTML - Tags Reference, HTML - Attributes Reference, HTML - Events Reference, HTML - Fonts Reference, HTML - ASCII Codes, ASCII Table, Lookup, HTML - Color Names, HTML - Entities, HTML - Fonts, Ref HTML - Events, Ref MIME Media Types, HTML - URL Encoding Language, ISO Codes HTML - Character Encodings, HTML - Deprecated Tags
- PRactical Examples: 1) Create simple Doc and display your name using different heading tag 2) Create link for open google. 3) Create document using all text formatting tags
- HTML online editor
- HTML Tables HTML Lists HTML Forms HTML Iframes
- Practical Examples: 1) Create simple table 2) Create time table for your school 3) Create table with colspanrowspan example 4) Create invoice using table 5) Create hotel menu. 6) Create index page for your book. 7) Create list with different categories.
- PRactical Examples: Create registration form with all fields and validation

Module 7) WD - CSS and CSS 3 - Full stack and Back end	**20**

- 1) CSS 2) In-line CSS Internal Style External Style Sheet @import Style Sheet 3) CSS Class CSS ID
- What is CSS How to Implement CSS Class and ID Width and Height Css Unit Box Model (Margin,padding,Border) and create basic template design
- Practical example : Create page with difference color text
- CSS Selectors , Pseudo Classes and Elements , Float and Clear and Alignment , Font Styling , Opacity and Visibility , Line Height
- 1) CSS Text 2)CSS Font 3) CSS Background 4) CSS Links 5) CSS Lists 6) CSS Display 7) CSS Visibility
- Creating Header of Website , Outline , Background , Counter increment , Counter reset ,Cursor , Overflow
- PRactical Example : Create layout for your project
- Position , Creating Submenu , Border Radius, Transform , Animation , Font Awesome Icons
- 1) CSS Layout Model 2) CSS Border 3) CSS Margin 4) CSS Padding 5) CSS Outline
- Font Family Through Google Font , import fontface rule ,FlexBox
- 1) CSS Float 2) CSS Align 3) CSS Position 4) CSS Element Size 5) CSS Layer
- Practical Example : Create image gallery
- 1) CSS Pseudo Class Selector 2) CSS Pseudo Element Selector
- CSS Properties 1) Background, 2) border 3) bottom 4) caption-side 5) clear 6) clip 7) color 8) content
- Practical Example: Create Menu with logo at left side and contact info at right side using clear effect
- 1) counter-increment 2) counter-reset 3) cursor 4) direction 5) display 6) empty-cells
- Practical Example: 1) Create submenu list using counter
- 1) float 2) font 3) height 4) left 5) letter-spacing 6) line [height, style, style-7) image, style-position, 8) style-type] 9) margin 10) outline 11) overflow 12) padding
- 1) page-break 2) position 3) quotes 4) right 5) table-layout 6) text 7) top 8) vertical-align 9) visibility 10) white-space 11) width 12) word-spacing 13) z-index

- Practical Example: wireframe layout for your template using div
- Media Query (For Responsive Website) , Creating a Responsive Website
- Validate a Website, Hosting a website with free domain name, Column , Clippath , Gradient Color , Filter, Border Image

Module 8) Website Designing - HTML5 - Full stack and Back end	**5**

- HTML5 Tags, HTML5 Input and Attribute
- Audio and Video, Semantic Element in HTML5
- Canvas, Svg
- Display Grid

Module 9) WD JavaScript Essentials And React -Advanced	**10**

- Basic JavaScript, Js comment, Js variables , Understanding var, let and Const, JS switch, if, else,JS loop , Js global variables, Js data types, Js operators, Js Functions
- Functions - Function Declaration in JS - Arrow Functions - Higher Order Functions - Map, Reduce and Filter
- Javascript Objects, Js object , Js Array , Js string, Js Date, Js Math, Js number, Js Boolean
- Javascript BOM ,Broswer Objects , Window object, History object, navigator object, Screen object
- Javascript DOM, Document object, getElementById, getElementByName, getElementByTagName, JS innerHTML property, JS innerTEXT property
- Javascript OOPS, JS class, JS object, JS prototype, JS constructor method, JS static method, JS encapsulation, JS inheritance, JS polymorphism, JS abstractions
- Javascript Exception Handling, JS exception handling , Javascript try-catch
- Javascript MISC, JS this keyword , JS Debugging , JS Hoisting , JS Strict Mode, JS promises, JS typeof , JS ternary operator, JS reload() method, JS setAttributes () method, JS setInterval() method, JS setTimeout() method.
- Javascript Events, Javascript Events, Javascript AddEventListener,(), jsOnclick event, jsdbclick event, JS onload event, JS onresize event.
- Array in JS, Creating Array, Array methods, The Spread &amp; Rest operators, Destructuring
- JS Async, Callbacks, Promises, Async/Await
- ES6 Basics and Babel, New features in ES 6, Arrow functions, The . Operator, For/of , Map Objects, Set Objects, Promises, Functions Rest parameter, String.includes(),String.starts.With(), String.endWith(), Array.form(), Array.keys(), Array find(), Array findIndex(), javascript Modules
- Small Project using ES6

Module 10) WD - JQuery Basic, Effects &amp; Advanced	**6**

- What is JQuery , Downloading JQuery File , First Program in JQuery
- Jquery Selectors
- Jquery Display Effeects
- Jquery Fadding Effects

- Jquery Sliding Effects
- Add Elements using jquery
- Get and Set Content and Attributes
- Remove Elements
- Other Jquery HTML Methods - clone, scroll TOP, attr(), prop(),
- Wrap Element using Jquery
- Jquery CSS methods
- JQuery manipulating CSS
- Jquery Dimension Methods
- Jquery Form
- Jquery Mouse events
- Jquery Keyboard Events
- Jquery Form Events
- Jquery Document/Window Events
- Jquery Traversing - Ancestor
- Jquery Traversing - Descendents
- Jquery Traversing - Siblings
- Jquery Traversing - Filtering
- Practicals and assignments
- ajax
- ajax get and post

Module 11) WD - Bootstrap Basic &amp; Advanced	9

- Bootstrap Basic 1) Bootstrap Introduction 2) Bootstrap Getting Started 3) Bootstrap Grid System 4) Bootstrap Fixed Layout 5) Bootstrap Fluid Layout 6) Bootstrap Responsive Layout
- Bootstrap Utilities
- 1) Bootstrap Typography 2) Bootstrap Tables 3) Bootstrap Lists 4) Bootstrap List Groups 5) Bootstrap Forms 6) Bootstrap Custom Forms 7) Bootstrap Input Groups 8) Bootstrap Buttons 9) Bootstrap Button Groups
- Bootstrap with CSS: Grid System
- 1) Bootstrap Images 2) Bootstrap Cards 3) Bootstrap Media Objects 4) Bootstrap Icons 5) Bootstrap Navs 6) Bootstrap Navbar 7) Bootstrap Breadcrumbs 8) Bootstrap Pagination 9) Bootstrap Badges 10) Bootstrap Progress Bars 11) Bootstrap Spinners 12) Bootstrap Jumbotron 13) Bootstrap Helper Classes
- Bootstrap with CSS - typography
- Bootstrap Advanced 1) Bootstrap Modals 2) Bootstrap Dropdowns 3) Bootstrap Tabs 4) Bootstrap Tooltips 5) Bootstrap Popovers 6) Bootstrap Alerts 7) Bootstrap Stateful Buttons 8) Bootstrap Accordion 9) Bootstrap Carousel 10) Bootstrap Typeahead 11) Bootstrap ScrollSpy 12) Bootstrap Toasts
- Bootstrap with CSS: Tables
- Bootstrap Striped Row Table
- Bootstrap Bordered Table
- Bootstrap Hover Row Table

- Bootstrap Condensed Table
- Bootstrap Contextual Classes
- Bootstrap Responsive Table
- Bootstrap with CSS - Forms
- bootstrap 4 Form
- Bootstrap with CSS - Buttons
- Bootstrap with CSS - Images
- Bootstrap helper Classes
- Border Classes
- Bootstrap with CSS - Responsive Utilities
- Bootstrap Layout - Glyphicon
- bootstrap Layout - Dropdowns
- Bootstrap Dropup
- Bootstrap - Button Group
- Bootstrap Layout - Navigation Elements
- Navbar
- Breadcrumb
- pagination
- Input Group
- Labels
- Badges
- Jumbotron
- Page Headers
- Alerts
- Progress Bar
- List Group
- Panels
- Wells

Module 13) Python - Fundamentals of python language	**6**

- Introduction of students
- Understanding Student Login of TOPSERP
- Career in IT
- Using Lab
- Introduction of Python
- Programming Style
- Core python concepts
- Conditional Statements
- If- else Nested if-else
- Practical Examples: 1) How to the python code Structure work? 2) How to create variable in python? 3) How to take user input? 4) How to check the type of variable dynamically. 5) W.A.P to find greater and less than number using If_else 6) W.A.P to find prime number using if_else 7) W.A.P to find the grade according to percentage using if_else ladder. 8) W.A.P to find that who can donate the

blood using Nested if.

- Looping For , While
- Generators and Iterators
- Nested loops
- map(), reduce(), filter() and Closures and Decorators
- Control Statements
- 1) WAP to print each fruit in list using simple for loop. List1 (apple,banana,mango) 2) WAP to find the length of string using simple for loop List1 (apple,banana,mango) 3) WAP to find particular string using simple for loop and simple if condition. 4) Print this pattern using nested for Loop.
- Break
- Continue
- Pass
- Practical Example: 1) W.A.P to skip the (Banana) from the list using Continue Statement List1 -(apple,banana,mango) 2) W.A.P to break the for loop when (Banana) get in if Condition.
- String Manipulation
- Accessing Strings
- Basic Operations
- String slices
- Function and Methods
- 1) W.A.P to print (Hello) using string 2) W.A.P to allocate the string to a variable. 3) W.A.P to print String using three quotes 4) W.A.P to access the 1st position character using index value. 5) W.A.P to Access the string after the index value 1. 6) W.A.P to Access the string before the index value 5. 7) W.A.P to Access the String between the index value 1 to 4 8) W.A.P to print the string from the last index value. 9) W.A.P to print the String alternate character after the index value 1. 10) W.

Module 14) Python - Collections, functions and Modules in Python	**8**

- Accessing list
- Operations
- Working with List
- Function and Method
- Practical Example: 1) W.A.P create the list of multiple datatype element. 2) W.A.P to find the length of the list. 3) W.A.P to update the list using the insert() and append() 4) W.A.P to remove the element using the pop() and remove()
- Tuple
- Accessing Tuples
- Operations Working
- Functions and Method
- Dictionaries
- Accessing value in dictionaries
- Working with dictionaries
- Property
- Practical Example: 1) W.A.P to access value on index value in the list 2) W.A.P to access the value after the index value 1. 3) W.A.P to access the value between 1 to 5 4) W.A.P to access the value till

index 5. 5) W.A.P to update the list using the index value. 6) W.A.P to irate the list using for loop. 7) W.A.P to insert the value in empty list using for loop and append(). 8) W.A.P to delete the element using del() 9) W.A.P to sort the list using sort() and sorted()

- 10) W.A.P to round the value in list using round() and for loop. 11) W.A.P to convert the list into tuple. 12) W.A.P to create tuple with multiple data type. 13) W.A.P to concate the two tuple into one tuple. 14) W.A.P to access the value of index value 1st in tuple. 15) W.A.P to access the value from last in tuple. 16) W.A.P to access the value between index 1st to 5th from the tuple. 17) W.A.P to access the alternate value between index 1st to 5th.
- 18) W.A.P to create the dictionary of having 6 key and value pair. 19) W.A.P to access the value using the key from dictionary. 20) W.A.P to update the value on particular key. 21) W.A.P to separate the key and value from dictionary using keys() and values() of dictionary. 22) W.A.P to convert the two list into one dictionary using for loop. 23) W.A.P to convert the list using zip() of dictionary. 24) W.A.P to count the character repeat in string.
- Function
- Types of Function
- Function Argument
- anonymous function
- Practical Example: 1) W.A.P to print the String using the function. 2) W.A.P to create the parameterized function. 3) W.A.P to print multiple string using function. 4) W.A.P to create calculator using function. 5) W.A.P to create lamba function using one expression. 6) W.A.P to create lamba function using two expression. 7) W.A.P to create lamba function using three expression. 8) W.A.P to create a return type function using lamda function.
- Modules
- Importing Module
- Math Module
- Random module
- Packages
- Practical Example: 1) W.A.P to import another module into one module. 2) W.A.P to use all the Math module function.

Module 15) Python - Advance python programming	**10**

- Printing on screen
- Reading data from keyboard
- opeaning and closing file
- reading and writing file
- Practical Example : 1) W.A.P to create the file using the python. 2) W.A.P to create a file and print the string into the file. 3) W.A.P to read a file and print the data on console. 4) W.A.P to write the multiple String into file 5) W.A.P to read multiple String from the file. 6) W.A.P to check where is the cursor in the file.
- Exception Handling
- Handling Exception
- Finally Clause

- PRactical Example: a) W.A.P to handle exception in calculator. b) W.A.P to handle multiple exception at time in one program. c) W.A.P to handle File Exception and use finally block for closing the file. d) W.A.P to print multiple exception using if else. e) W.A.P to print user define exception.
- class and object
- Attribute
- Inheritance
- Overloading
- Overriding
- Practical Example: 1) W.A.P to create a class and access the property of class using object. 2) W.A.P to create local variable and global variable. 3) W.A.P to show single inheritance. 4) W.A.P to show Multilevel inheritance. 5) W.A.P to show Multiple inheritance. 6) W.A.P to show Hierarchical inheritance. 7) W.A.P to show Hybrid inheritance. 8) W.A.P to using super() in inheritance. 9) W.A.P to show method overloading. 10) W.A.P to show Method overriding.
- sqlite3 and pymysql modules (database connectors)
- Search Function
- Match Function
- Matching Vs Searching
- Modifiers
- Practical Examples: 1) W.A.P to search a word from the string using Search() 2) W.A.P to match the word in string using Match().
- GUI Programming Introduction Tkinter programming
- Tkinter widgets
- Practical Example: 1) W.A.P to create GUI Frame. 2) W.A.P to create all the widgets using Tkinter.

Module 16) Python - DB and Python Framework - Industry Program	**18**

- HTML
- CSS
- javascript
- Django Introduction Advantages of django Django vs Flask
- Virtual Environment
- Project and app creation
- MVT pattern architecture
- Practical Example 1 Create Django Admin Panel 2 Creating the Doctor Finder Project. Project Practical Registration login , forgot password session management , email template , profile,updation , working with media , CRUD operations
- Practical Example: 1) Create Django Admin Panel 2) Creating the Doctor Finder Project.
- Django Admin
- URL pattern
- Template integration
- form validation using javascript
- Django Database connectivity mysql or sqllite
- ORM, Query set
- Django forms,Django authentication

- CRUD operationgs using AJAX
- Customization django admin panel
- Payment integration using paytm
- Github project deployment
- Live project deployment Python anywhere

Module 17) Python - Rest Framework - Industry Program	**3**

- Introduction
- Requirements
- Serialization
- Requests and Responses
- Views
- URLs
- Pagination
- Settings
- Project setup
- Social Auth, Email and OTP Sending API, Third Party Integration
- RESTful API: Representational State Transfer (REST) is a widely used architectural style for building web services. Understanding REST principles and being able to create RESTful APIs is essential. CRUD API: CRUD stands for Create, Read, Update, and Delete, which are the basic operations performed on data. Creating APIs that allow these operations is fundamental to backend development. Authentication and Authorization API: Knowing how to implement user authentication and authorization mechanisms is crucia
- OpenWeatherMap API: This API provides weather data for various locations worldwide. You can retrieve current weather conditions, forecasts, and historical weather data.
- Google Maps Geocoding API: This API allows you to convert addresses into geographic coordinates (latitude and longitude) and vice versa. You can use it to retrieve location data, calculate distances between points, and display maps.
- GitHub API: GitHub provides an API that enables you to interact with repositories, issues, pull requests, and more. You can perform actions like retrieving repository information, creating issues, and accessing user data.
- Twitter API: Twitter offers an API that allows you to integrate Twitter functionality into your applications. You can fetch tweets, post tweets, retrieve user information, and perform searches.
- REST Countries API: This API provides information about countries, including details like population, languages spoken, currencies, and more. You can retrieve country-specific data and use it for various applications.
- endGrid provides an API for sending transactional and marketing emails. You can integrate it into your applications to send emails, manage templates, and track delivery statistics.
- Social authentication (For eg; Login with Google, Login with Facebook...etc)
- Email sending APIs (For eg; Mailchimp, Mailgun...etc)
- SMS sending APIs (For eg; Twilio)
- Normal payments (For eg; Paypal, Stripe)
- Google Maps API

Module-26) React - Components, State, Props	**8**

- Installation - Add React to a HTML Website - Create New React App - Hello World
- Getting started in React
- JSX
- Components
- Component Composition
- JSX - Why JSX? - Embedding Expressions in JSX - Attributes with JSX - Children with JSX
- Props &amp; Prop Types
- Event Handlers
- State
- React Web App
- Components, State, Props - Function Component - Class Component - Props - State - Class Component Lifecycle

Module 4) Lists and Hooks	**6**

- Conditional Rendering - Lists and Keys - Forms - Handling Events - Lifting State up
- Hooks - Introduction - Using the State hook - Using the Effect hook - Rules of Hook - Custom Hook
- Rendering Lists inside components
- React Keys
- Using keys wit component
- Uniqueness of keys among siblings
- React refs
- Uses of react Refs
- How to access of Refs
- Refs current properties
- Add Refs to DOM elements
- Add refs to class components
- Callback refs
- Forwarding Ref from one component to another component
- React with useRef
- React conditional rendering
- React if, logical &amp; operator, Ternary operator, switch case operator, Conditional Rendering with Enum, Preventing components from rendering

Module-28) React - Styling &amp; Advance React	**5**

- Creating the first App
- Understanding the App
- Styling the App
- Inspecting &amp; Debugging styles
- Built-in components
- Working with Images

- ListViews
- TextInput
- Styling React Components - CSS stylesheet - Inline Styling - CSS Modules - CSS in JS Libraries (styled components)
- Creating Views (Scenes)
- Conditional Rendering - Lists and Keys - Forms - Handling Events - Lifting State up
- Hooks - Introduction - Using the State hook - Using the Effect hook - Rules of Hook - Custom Hook
- Advance Concepts - Context, useContext() - Working with Refs and useRefs() - Fragments - Performance optimization with useMemo() - Styling React Components - CSS stylesheet - Inline Styling - CSS Modules - CSS in JS Libraries (styled components)
- Bootstrap with React
- React Router - Browser - Router - Link - Route - Template integration - Http Request in React - Get and Post data

Module 6) React Router	**8**

- React Router
- B r o w s e r - R o u t e r - Lin k - R o u t e
- Need of react router
- T e m pla t e in t e g r a tio n - H t t p R e q u e s t in R e a c t - G e t a n d P o s t d a t a
- React router installation
- React router, react-router-native, react-router-Dom
- Component in react router , Browser Router , HashRouter
- What is Route
- What is Link component , Adding navigation using Link component
- Link vs NavLink
- React Router Switch , React Router redirect
- Nested Routing in React
- Template integrations Using Browser Router , Routes , Route , Link and Hash Router
- Advantages of react Router

Module-30) React - Applying Redux	**8**

- State
- State storage problem
- Redux Basics
- Redux Principles
- Implementing Redux
- React-Redux
- Middleware
- Counter App Demo
- Redux - Complexity of Managing state - Understand the Redux Flow - Setting up Reducer and store - Dispatching Actions - Passing and Retrieving Data with Action - Combining Multiple Reducers - Adding Middleware - Redux Dev tools