MERN STACK SYLLABUS

MERN Stack Development - JavaScript

- □ **Introduction:**
  - □ Introduction to MERN stack
  - □ Introduction to JavaScript programming language
  - □ Technologies around JavaScript.
- □ **Environment Setup:**
  - □ Introduction to development tools
  - □ Text Editor, IDE
  - □ Command Prompt(command line)
  - □ Node setup
  - □ Postman

Javascript

- □ **Overview of JavaScript**
  - □ Brief History of Javascript
  - □ Variables
  - □ Operator and Operands
  - □ Conditionals and Iterations
  - □ Data types and coercion
  - □ A brief introduction to data structures like stack, queues, array, and objects Debugging a JS code
  - □ JS naming conventions
  - □ A brief introduction to javascript functions and classes
- □ **EcmaScripts and Modern JavaScript**

- □ Need of ES language specification Understanding of transpilers and polyfills
- □ CommonJs vs ES modules
- □ Deep guide to modern ES6+ features
- □ Brief introduction to Typescript
- □ TypeScript Vs JavaScript

□ Js- Under the Hood

- □ Hoisting
- □ Block/Global/Local Scopes and Temporal Dead Zones Scopes and Closures
- □ Prototypes
- □ Asynchronous javascript and WebAPIs usages
- □ Global execution context
- □ Event loops

□ JavaScript Functions

- □ Arguments and Parameters
- □ Functions declaration and execution
- □ ES6 arrow functions
- □ Callback functions
- □ Higher Order Functions

Git and Versioning

□ Github And Version Controlling

- □ Introduction to version control and usage Github, Bitbucket and Gitlab
- □ Initializing and cloning git projects
- □ Git commit/push
- □ Branches and Pull Requests

    - □ Standard practices for naming GitHub branches
    - □ The .gitignore file
    - □ Github exercises: Collaborating in the same repo and handling merge conflicts live.

Web Technology

- □ **Introduction to web flow and Architecture**
    - □ Introduction to web and how it works
    - □ Client-server architecture definition
    - □ Introduction to API and it’s need
    - □ Introduction to REST API
    - □ The software development/design pattern
    - □ The famous - MVC pattern

Node js

- □ **Nodejs**
    - □ Introduction to nodejs
    - □ Requirements for nodejs
    - □ Nodejs and it’s usages
    - □ Setup nodejs
- □ **NPM (Node package Manager)**
    - □ Introduction to NPM
    - □ Global Vs Local Installation
    - □ The package.json file
    - □ Attributes of Package.json
    - □ Installing and uninstalling packages using npm
    - □ Introduction to dependency and dev-dependency

Introduction to Node builtin modules

Express JS

- □ **Introduction to Express JS**
  - □ Introduction to framework and Express
  - □ Setting up an express application
  - □ Express exercise: Creating a hello world application using express
- □ **Architecturing the Express js application**
  - □ Express flow and how it works
  - □ Introduction to express application and usages
  - □ The HTTP protocol
  - □ Introduction to express routing
  - □ Express request, response object
- □ **Express Middlewares**
  - □ Introduction to middlewares
  - □ Types of middlewares
  - □ Usages of middlewares
  - □ The next scope
  - □ Error handling using middlewares
  - □ Serving the static files using static
  - □ File handling using middlewares
- □ **Authorization and Authentication in Express**
  - □ Introduction to auth and authorization
  - □ Introduction to json web token
  - □ Authorization using jwt
  - □ API protection

MongoDB

- □ **Detail of MongoDB**
  - □ Introduction to database and existing database around
  - □ Understanding NoSQL (schema less database)
  - □ Introduction to MongoDB (collection and documents)
  - □ CRUD operations using shell command
  - □ Using Mongodb Native driver with Node
  - □ Database modeling using ODM
  - □ Using mongodb-compass (ui based database system)
  - □ Update, insert delete and upsert documents
  - □ Aggregation in mongodb
  - □ Map reduce
- □ **Express with MongoDB**
  - □ Building APIs with express js and mongodb
  - □ Using MongoDB as data store in express app with Mongoose

    Backend Project: API development using node js, express js and mongodb

Frontend Technology

- □ **Basic Web Overview**
  - □ Introduction to HTML
  - □ Introduction to html tags, attributes and properties
  - □ Introduction to css
  - □ Importance and uses of css in a web design
  - □ HTML/CSS Practice: Creating and designing basic web designs
- □ **Responsive Web design tools**
  - □ Introduction to responsive vs non-responsive design
  - □ Bootstrap, a css based framework

    - □ Alternatives of bootstrap

The React js

- □ **Introduction**
    - □ The objective of this training
    - □ Present scenario/Scopes of this technology.
    - □ Environment setup
    - □ Node and npm
    - □ Vscode
- □ **Overview of Javascript**
    - □ Brief History of Javascript
    - □ Variables
    - □ Operator and Operands
    - □ Conditionals and Iterations
    - □ Data types and coercion
    - □ A brief introduction to data structures like stack, queues, array, and objects
    - □ Debugging a JS code
    - □ JS naming conventions
    - □ A brief introduction to javascript functions and classes
- □ **EcmaScript and modern javascript**
    - □ Need of ES language specification
    - □ Understanding of transpilers and polyfills
    - □ CommonJs vs ES modules
    - □ Deep guide to modern ES6+ features
- □ **JS- under the hood**
    - □ Hoisting
    - □ Block/Global/Local Scopes and Temporal Dead Zones

    - □ Scopes and Closures
    - □ Prototypes
    - □ Asynchronous javascript and WebAPIs usages
    - □ Global execution context
    - □ Event loops
- □ **Functions**
    - □ Arguments and Parameters
    - □ Functions declaration and execution
    - □ Pure and Impure Functions
    - □ ES6 arrow functions
    - □ IIFE and anonymous functions
    - □ Callback functions
    - □ Higher Order Functions
- □ **Github**
    - □ Introduction to version control and usage
    - □ Github, Bitbucket and Gitlab
    - □ Initializing and cloning git projects
    - □ Git commit/push
    - □ Branches and Pull Requests
    - □ Standard practices for naming GitHub branches
    - □ Github exercises: Collaborating in the same repo and handling merge conflicts live.
- □ **Overview of React**
    - □ Understanding react as a library and its ecosystem
    - □ Difference between libraries and frameworks
    - □ Setting up react with CDN without using any frameworks

- □ A brief overview of how react library can be used in web frameworks such as create- react-app, Gatsby, next.js, etc and mobile frameworks like react-native
- □ Introduction to react components, folder structure, and architecture
- □ Functional and Class based Components in react
- □ React Component Lifecycle Methods
- □ Styling a react component
- □ Introduction to SPAs
- □ Virtual DOM in react

□ **Getting started with react fundamentals**

- □ State
- □ State exercise: Build a like button, counter apps, etc using react state and useState hooks
- □ Props
- □ Props exercise: pass data from parents to child, child to parent, and from one child to its siblings
- □ Refs
- □ Refs exercise: Targeting a particular DOM, scrolling to a particular DOM, auto focusing an input field programmatically, etc using useRef hooks
- □ Higher-Order Components
- □ HOC exercise: Building reusable components using Recompose

□ **React Router**

- □ Introduction
- □ Setup react-router to handle routes
- □ Protected and nested routes

□ **React Forms**

- □ Basics of react forms and handling various types of form fields and events
- □ Client-side form validation

    - □ Advanced form handling with Formik and schema validators like Yup or regex

- □ **REST API**
    - □ Understanding of REST API
    - □ HTTP methods and protocols
    - □ Understanding of Fetch API, axios, and interceptors
- □ **NEXT JS**
    - □ Introduction to Next.js and its benefits.
    - □ Intro to nextjs page router and app router
    - □ Creating API endpoints with Next.js.
    - □ Understanding of Server-Side Rendering (SSR) and Static Site Generation (SSG) with next js
    - □ Fetching data from API routes.
    - □ Integrating API routes into a Next.js app.
    - □ Implementing user authentication and handling cookies storages.
    - □ Handling user sessions and access control.
    - □ Introduction to serverless functions in Next.js.
    - □ Implementing OAuth-based authentication (e.g., using OAuth providers like Google, Facebook, or GitHub).
- □ **Redux**
    - □ Flux Architecture
    - □ State management using Redux
    - □ Reducers
    - □ Action Creators and action types
    - □ Using redux middleware libraries such as sagas, thunks, and Rxjs to handle asynchronous side effects
    - □ Using redux logger middlewares and redux dev tools to track state changes

    - □ Using redux persist to handle storage in the react application
- □ **GRAPHQL**
    - □ Understanding of GraphQL
    - □ Understanding of data under-fetching and over-fetching
    - □ Query, arguments, and mutation
    - □ Variables in graphql
    - □ Graphql fragments
    - □ Setting up client side graphql in react with Apollo
- □ **WebSockets**
    - □ Introduction to real-time web socket communication
    - □ Socket io client setup in react
    - □ Getting connected to sockets and disconnecting a web socket
    - □ Emitting data and retrieving data to and from a connected socket
- □ **Test Driven Development with React**
    - □ Overview of various types of software testing
    - □ Introduction to Unit testing a web application
    - □ Setting up Jest
    - □ UI testing, writing tests for mocked APIs
    - □ Code coverage with jest
- □ **React Performance**
    - □ Understanding of how a react component renders and re-renders
    - □ An in-depth guide to memoization in react and using performance optimization hooks such as useMemo and useCallback
    - □ Guide to using dev tools and react Profiler
    - □ Lazy loading and code splitting
    - □ Setting up webpack configs, compressing the react app build size

Projects

- □ Online Reservation System
- □ E-commerce Website
- □ Project Management System
- □ Online Inventory
- □ Ride Sharing Application