SELECTED TOPICS IN REACT (REACT-404)

LEVEL: ADVANCED

OVERVIEW

3 days for seasoned programmers

The training assumes good knowledge of React. For topics involving AWS services, participants must have an AWS account to follow along.

PREREQUISITES

- Good knowledge of JavaScript – functions, objects, functional programming, array iterator methods.
- Good knowledge of React

APPLICATION BUILT DURING TRAINING

Workshops application

Add the end of this bootcamp, participants would have built a workshops application. They shall be provided a backend server. The application will involve communicating with the backend and listing technical workshops, adding, editing and removing topics for workshops. The application shall be deployed on AWS using ECS.

LIST OF SOFTWARE TO BE INSTALLED BEFORE TRAINING BEGINS

1. Git CLI on participant systems and GitHub account should be created for every participant (to be created individually by participant). The **GitHub account should be a personal one** and not one associated with the company's GitHub account (I will not be able to add a company account as collaborator on my repositories, and hence shall not be able to share code).

   Git CLI download: https://git-scm.com/downloads

   GitHub link for account creation: https://github.com/join?source=header-home

   Open a terminal and check installation went on fine by typing

   $> git --version

   You will see the version number of git ($> indicates the command prompt)

   The list of GitHub user names needs to be shared with me.

2. Node.js needs to be installed on all systems – Mac OSX, Linux and Windows is supported. The 20.x.x (LTS version) may be installed. This will also install npm.

   Node.js https://nodejs.org/en/download/

3. Download and install Visual Studio Code (VSCode) from https://code.visualstudio.com/download

   It is available for Windows, Mac OSX and popular Linux distributions.

4. Latest version of Chrome. Internet Explorer is not acceptable.

   Chrome: https://www.google.com/chrome/browser/desktop/index.html

5. For topics involving AWS services, participants must have an AWS account to follow along. Most services are free for the first 12 months from account creation. If older, the services used in this training will cost little, provided resources provisioned are also released after trying them out (instructor will guide participants on this).

   https://aws.amazon.com/free/ (Create a Free Account)

6. Download and install Docker Desktop.

   https://www.docker.com/products/docker-desktop/

7. **Additionally, it would be great if participants have as little restrictions (as permissible) on internet access during the session**

CHAPTERS AND TOPICS

DAY 1

Quick Overview of TypeScript

NOTE: Working knowledge of TypeScript is assumed, and only integration with React shall be covered in a hands-on manner

Why Typescript?

Installation and getting started

Basic types - Primitives, the any type, array, union, intersection, enum

Type alias

Type assertion (type-casting)

Function types, typing callback functions, interfaces to define function signatures, overloaded function types

Class, access modifiers, shortcut syntax for defining properties, and inheritance

Interface, implementing an interface in a class

Interfaces vs. type aliases

Creating and using Generics, type defaults

Using decorators

Integrating TypeScript with a React app

Application development best practices (select topics)

Organizing your project files and folders

Grouping by file types, features, and modules

Using common and consistent file and folder names

Organizing Redux code

Organizing unit test code

ESLint and Prettier

Browser extensions for ESLint and Prettier

Setting up ESLint

Understanding ESLint plugins, rules and configuration options

Using ESLint in practice

Setting up Prettier

ESLint vs Prettier

Configuring Prettier

Having ESLint and Prettier work well together

Setting pre-commit hooks

Select Topics in React and Redux

NOTE: Knowledge of useState(), useEffect() and useRef() is assumed

Design patterns for shared logic in components

Understanding Higher-order components (HOC) for class components

Custom hooks for function components

Maintaining state and state changes using useReducer()

Sharing state using useContext()

Performance optimization – useCallback(), React.memo(), useMemo()

Using the Profiler tab (React Devtools)

Select libraries / packages for React apps

NOTE: Use of these libraries shall be demonstrated through individual examples except for React router, Redux and Mobx

React Hook Form

Validation approaches in general

Validation approaches with React Hook Form – controlled and uncontrolled inputs

useForm() and Controller

Registering fields and applying validation

Handling errors

Custom form validation

Cross-field validation

DAY 2

Select libraries / packages for React apps (continued)

NOTE: Use of these libraries shall be demonstrated through individual examples except for React router, Redux and Mobx

React router

Workshops application with React and React Router

Route configuration – BrowserRouter, Link, NavLink, Route, Switch/Routes

Redirect/Navigate components

Handling path params

Programmatic route changes

Code-splitting and lazy loading

Differences between v5 and v6

Redux

NOTE: Working knowledge of Redux is assumed, and this is only a quick overview using Redux Toolkit

Example: Workshops application with React, React Router and Redux

Redux flow overview

Redux Toolkit

Actions and Stores

Immutability

Reducers and Slices

Creating and using middleware in Redux

Implementing custom middleware

Redux Thunk (redux-thunk)

Connecting React to Redux (react-redux) – Provider, useDispatch(), useSelector()

React devtools for state snapshots

Mobx

NOTE: Knowledge of Mobx is not necessary and Mobx shall be explained in detail

Example: Workshops application with React, React Router and Mobx

Mobx flow overview

State, Derivations and Actions

- Computed values vs Reactions
- Principles of Mobx
- Working with Observables and Decorators
- Flow
- Utility functions
- mobx-react

Other topics and packages

- Virtualized lists using react-virtualized
- Loading data on scroll
- Charting using React Plotly
- Quick introduction to Material UI
    - Setting up Material UI
    - Overview of components
    - Theming
    - Data Tables
- Overview of axios
- The debounce and throttle methods (using lodash)
- Web workers in JavaScript and when to use them

DAY 3

Next.js

Example: Workshop application with Next.js

Introduction to Next.js

Generating a Next.js application

Pages vs App router

Pages and routing

SEO and metadata

Dynamic routing, using path parameters

API routes

Server-side rendering, pre-rendering and Static Site Generation

Some Authentication Strategies

Stateful vs Stateless mechanisms

JWT Authentication

Handling authentication in a React app

Refresh token

The OAuth 2.0 flow – client, resource owner, resource server, authorization server, authorization grant, access token, protected resource

OAuth 2.0 in practice – usage in a React app

Overview of select AWS Services

NOTE: Focus and hands-on shall be only on S3, Cloudfront and ECS

AWS Infrastructure Overview – Region, AZ

S3 – some features and serving static sites

Networking - VPC, Subnets, Security Groups, NACLs, Internet Gateway etc.

EC2, ASG, ALB – running a backend

Cloudfront and Route 53

Elastic Container Service (ECS)

Fargate

Secrets Manager

Deployment

- Configuration management in a create-react-app application
- Creating a production build
- Deploying through S3 and Cloudfront
- Quick Overview of Docker
- Dockerizing a React app
- Dockerizing a Node.js-based backend
- Deploying using ECS
- Setting up a Deployment Pipeline
- Managing Secrets using Secrets Manager