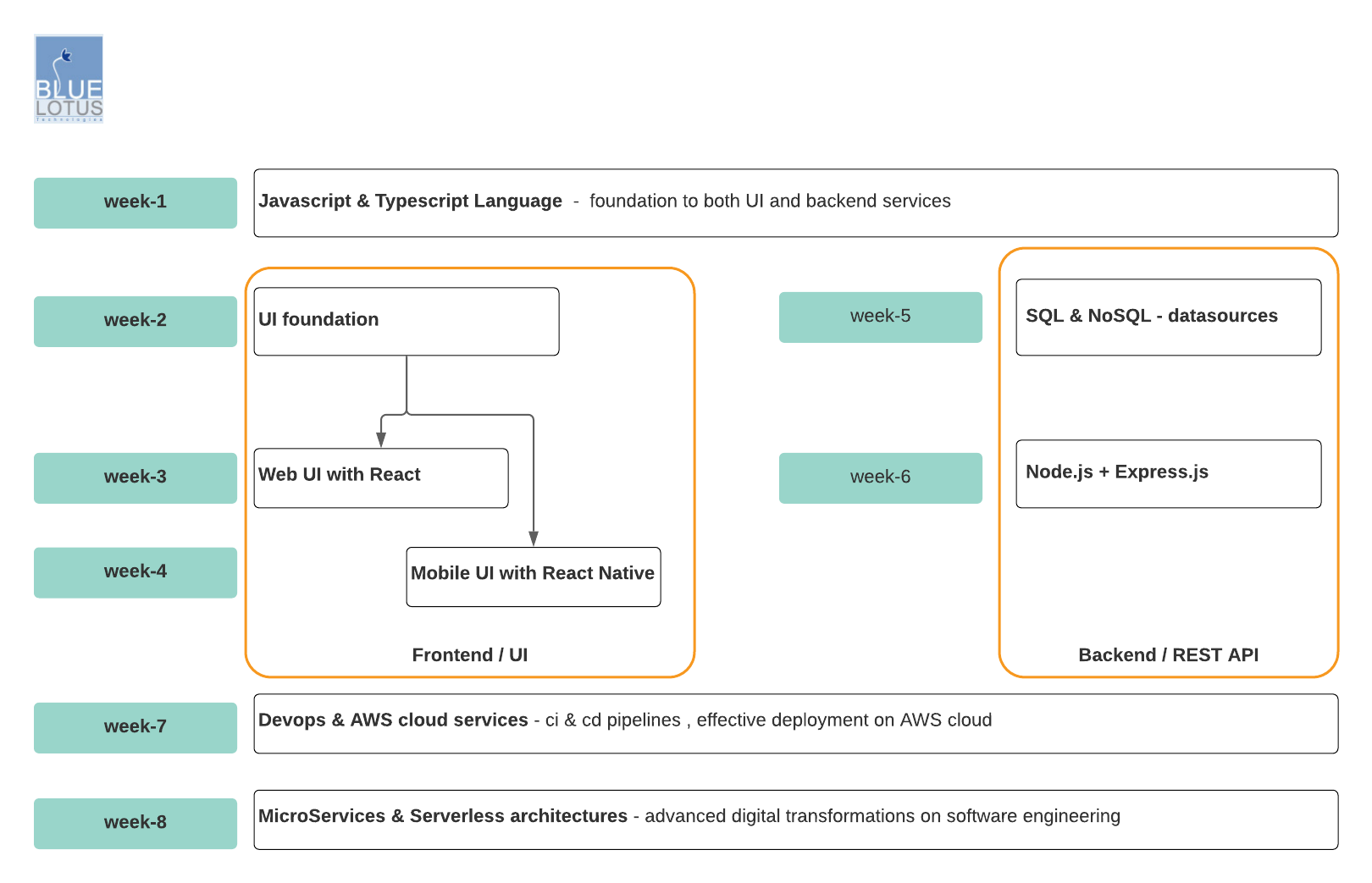
AWS Cloud React and React Native

Duration : 8 weeks

**Plan**



**Week-1 : Javascript & Typescript Language**

**day-1:**

* big picture of javascript language
* data types
  + primitive & reference types
* execution-context / scope
  + variable hoisting
  + let & const keywords
* literal style objects
* operators, conditional & looping statements ( skip )
* error handling
* functional programming
* how to create/define functions
* function parameters
* function with default & rest parameters
* functional programming principles
* higher-order-functions ( HOF )
* closures

**day-2:**

* object oriented programming
* ES5 class / constructor-function
* prototype based inheritance
* ES6 class syntax
* class based inheritance
* class with static variables & methods
* function binding
* mystery of 'this' keyword
* static vs dynamic function-binding
* dynamic function binding with call(),apply and bind() methods
* imperative vs functional style coding
* data structures
* array/list
* set
* map

**day-3:**

* ES6 new-syntax
* arrow Function
* destructuring
* spread operator
* iterables
* for-of-loop
* generator function
* obj-literal enhancements
* modules, packages & tools
  + how to organize javascript code? traditional & modern approach
  + Module standards
    - commonJS modules
    - ES modules
  + package managers
    - NPM
    - Yarn
  + transpiler
    - babel
  + module bundler & loader
    - webpack

**day-4:**

* javascript runtime internal
  + single-thread-model with event-loop
  + stack & heap
  + event-queue with event-loop
  + callbacks
  + non-blocking/asynchronous io
* async programming apis
  + promise api
  + async-await keywords
  + reactive programming with RxJS library
* unit-testing
* why unit-testing important?
* TDD & BDD testing style
* unit-testing framework - JEST
* JEST - sync & async code testing
* JEST - matchers
* JEST - mock functions
* snapshot testing

**day-5:**

* typescript
  + javascript vs typescript
  + basic & advanced types
  + interfaces
  + classes, enum and generics
  + decorators
* utility libraries
* lodash
* moment.js

**Week-2 : UI foundation**

**day-6:**

* what makes UI?
  + elements
  + style
  + behavior
  + native Apis
* HTML5 new elements
* web content accessibility guidelines
* CSS3
  + basic/common style properties
  + box model
  + gradients & shadows
  + transitions & animations
  + Flexbox

**day-7:**

* responsive web design ( RWD )
  + RWD intro
  + RWD Viewport
  + RWD Grid View
  + RWD Media Quaries
  + RWD Images
  + RWD Videos
* bootstrap
* layout
* content
* components
* utilities

**day-8:**

* Making dynamic web-ui
  + Event-Driven UI
  + DOM API
  + timer Api
  + XHR & fetch api

**day-9:**

* HTML5 Apis
* connectivity
  + web sockets
  + server sent events (sse)
* offline & storage apis
  + session & local storage
  + service workers
* multi-media
  + audio & Video
* camera Api
* graphical api
  + canvas
  + SVG
* performance & integration api
  + web workers
  + history api
  + drag & drop
  + geo-location
  + full screen api
  + device access
  + detecting device orientation

**day-10:**

* challenges while developing UI applications
* compatibility issues
* performance & memory issues
* data-binding issues
* UI architectures
  + MVC vs component-based architectures
* component-based UI architecture
  + what is component?
  + principles/characteristics of components
* how to create UI-components?
  + standard web-component apis
  + Template api
  + shadow-DOM api
  + custom elements
  + using third-party's javascript libraries
    - e.g. react

**Week-3 : web-ui with React.js**

**day-11:**

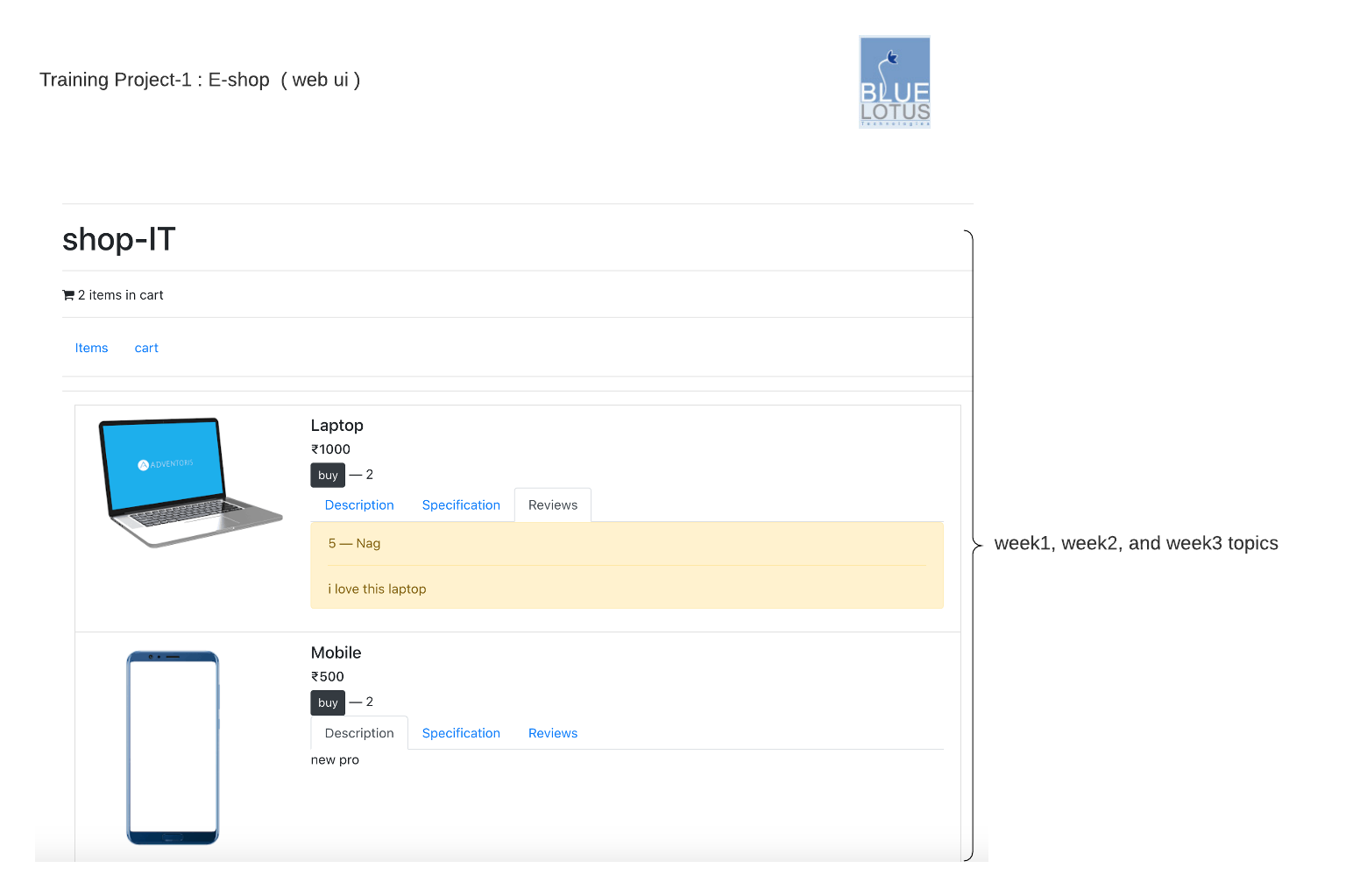
* web-UI: without & with react.js library
* why we need react.js and how it differ from other lib
* creating react.js application with CLI tools
* 101 concepts
  + JSX
  + props
  + state
  + handling events
  + inter-component communication
  + stateful & stateless components
  + component's lifecycle methods
  + class vs function syntax components
* 201 concepts
  + lists & keys
  + context
  + container & presentational components
  + re-use component's common logic patterns
  + higher-order-components ( HOC )
  + render-prop pattern
  + error boundaries
  + fragments

**day-12 :**

* Hooks. ( 16.8 )
* what & why we need?
* useState
* useReducer
* useEffect
* useMemo
* useCallback
* useRef
* useContext
* re-use common logic using custom-hook(s)
* form
  + controlled & uncontrolled inputs fields
* react-routing package
* basic routes
* URL parameters
* redirects
* preventing transitions
* route config
* nested routes
* **unit testing on react components**
* **Testing Overview**
* **Testing Recipes**
* **Testing Environments**
* **JEST with Enzyme**

**day-13:**

* Project 1: **E-shop** web-ui with react.js only



* challenges with UI's state/data with react-components only
* state management solutions with third-party libraries
* Redux
* MobX (overview). - FYI

**day-14:**

* redux
  + what & why we need it
  + redux core principles
  + Action & Action creators
  + reducers
  + store
* middleware to handle async actions
  + thunk
  + saga
  + observables
* react-redux. => to bind react with redux
* use redux with react
  + with hooks - useSelector() & useDispatch() hooks
  + with HOF - connect ()
* use-case: E-commerce web-ui with react.js & redux

**day-15:**

* **GraphQL**
* **queries & mutations**
* **schemas & types**
* **validation & execution**
* **best practices**
* **UI advanced concepts**
* **Authentication & Authorization**
* **Caching**
* **XSS & CSRF**

**Week-4 : native mobile app with react-native**

**day-16:**

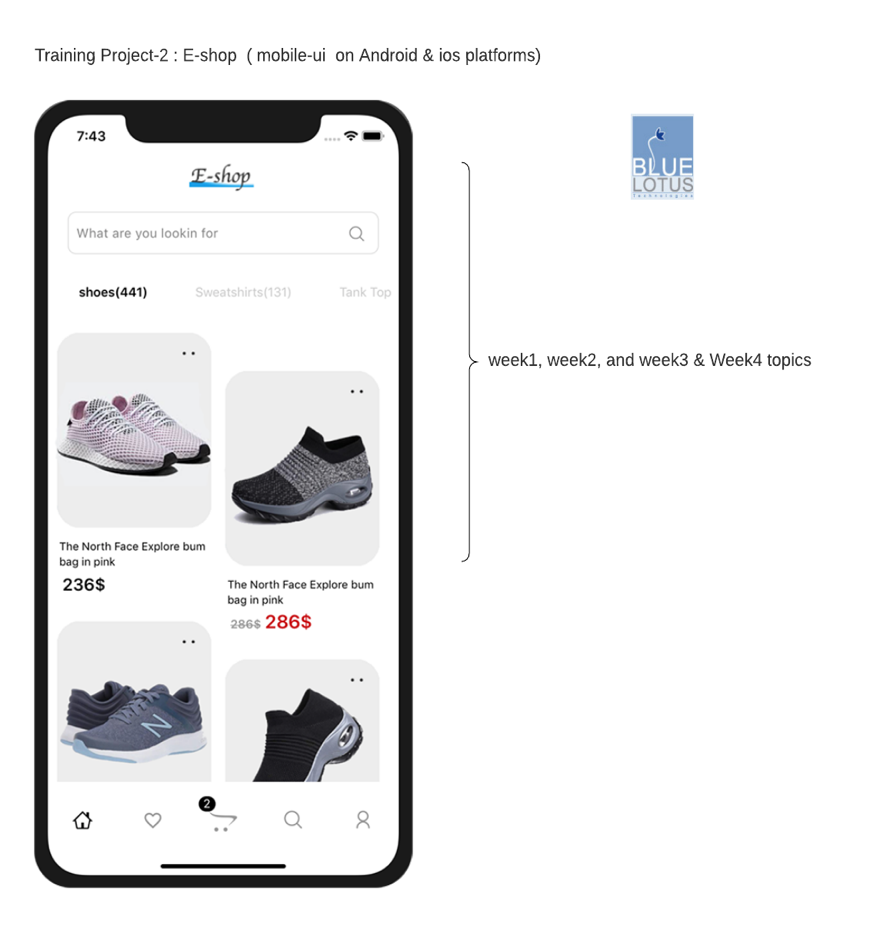
* environment setup
  + https://reactnative.dev/docs/environment-setup
* native core-components
  + View
  + Layout
  + Text
  + Image
  + Button
  + Touchable
  + Touchable Opacity
  + Touchable Highlight
  + ScrollView
  + FlatList
  + ScrollList

**day-17:**

* project structure
* navigation
  + react-navigation
* data management
  + useReducer
  + redux
  + mobX
  + realm
* persistent
  + async-storage
  + redux-persistent
  + realm
* networking
* fetch
* axios
* animation
  + animated
  + reanimated
* gestures
  + pan responder
  + gesture handlers
* component libraries
  + react-native-elements
  + native-base

**day-18:**

* use-case : **E-shop** mobile app



**day-19:**

* Implementing Authentication and Authorization
* Token Management
* Async Storage for persistent Data
* Advanced Navigation strategies in react-native

**day-20:**

* Native Modules & Build
  + Implementing Splash Screen

<https://docs.expo.io/guides/splash-screens>

* + Working with camera plug-in

<https://docs.expo.io/versions/latest/sdk/camera>

* + Working with Infinity scroller
  + Working document Picker

<https://snack.expo.io/@eamaya/4037b8>

* + Working with file uploads
  + **Implementing Finger Print Scanner**

[**https://snack.expo.io/@vkural/fingerprint-example**](https://snack.expo.io/@vkural/fingerprint-example)

* + **Implementing Maps on React-Native**

[**https://docs.expo.io/versions/latest/sdk/map-view**](https://docs.expo.io/versions/latest/sdk/map-view)

* + **Production Build**
  + **Building Apk**
  + **Uploading to Play Store & App Store**

**Week-5 : SQL & NoSQL data sources**

**day-21:**

* SQL vs NoSQL data sources
* choosing right data source
* exploring different data source platforms
* SQL data source: **postgres** - basics
  + SQL data source terminology
  + Introduction to SQL
  + Retrieving Data
  + Updating Data
  + Inserting Data
  + Deleting Data

**day-22:**

* SQL data source: **postgres** advanced
  + Sorting and Filtering Data
  + Advanced Filtering
  + Summarizing Data
  + Grouping Data
  + Using Subqueries
  + Joining Tables
  + Managing Tables
  + **Using Views**
  + **Stored Procedures**
  + **Using Cursors**
  + **Using Transactions**

**day-23:**

* NoSQL data source : **MongoDB** - basics & advanced
  + What is MongoDB?
  + Importing, Exporting, and Querying Data
  + Creating and Manipulating Documents
  + Aggregation Pipeline
  + **Indexing**

**day-24:**

* NoSQL data source : Redis ( REmote Dictionary Server )
  + what & why we need Redis
  + Redis data structures e.g. list, hashmap
  + Redis as cache

**day-25:**

* + **e-shop** application: design data model with SQL & NoSQL data sources | Advanced CRUD Operations

**Week-6 : Node.js & Express.js – backend / REST api**

**day-26:**

* Node.js – core concepts
  + Traditional server-side runtimes vs Node.js runtime
  + Blocking vs Non Blocking IO model in-depth
  + Event Driven Model
  + Streaming IO
  + Understanding Event-Lopp in-depth, process.nextTick

**day-27:**

* Node.js - core-modules
  + FileSystem modules
  + Net
  + HTTP modules
  + Buffers
  + Async Hooks
  + Errors
  + OS
  + Path
  + Process
  + real-time app with **socket.io**

**day-28:**

* Node.js – web applications with Express
  + Web-server
  + Handling HTTP Requests
  + Node.js web frameworks – Express.js
  + Express.js basic & advanced routing
  + Using Middleware
  + Views with template-engine like hbs
  + REST API
* Node.js – case-study implementations
  + Working with SQL & NoSql databases
  + With MongoDB
  + Complete case-study with CRUD functionalities
  + Unit-Testing with mocha / jest

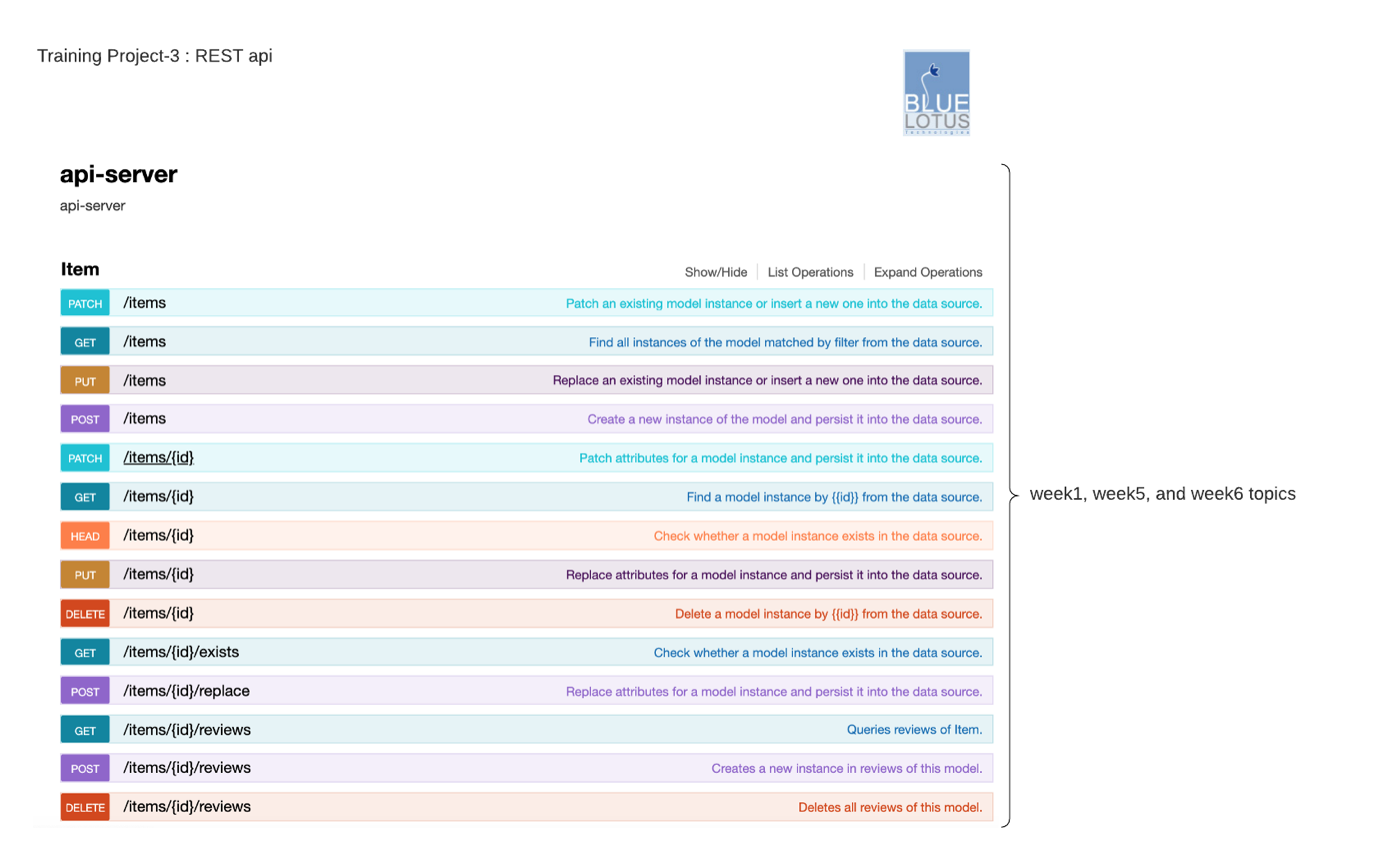
**day-29:**

* Node.js – Advanced concepts
  + Security
  + Logging modules
  + Microservices Architectures
  + Design pattern at right place
  + Performance considerations

**day-30:**

* Node.js – deployments
* Scaling Node.js applications with Cluster module
* PM2 tool
  + Monitoring tools
* **Detecting Memory Leaks**

**REST api for e-shop**



**Week-7 : Dev-Ops & AWS cloud**

**day-31:**

* DevOps Culture
* The Goals of DevOps
* A Story of DevOps vs. Traditional Silos
* DevOps Concepts and Practices
  + Build Automation
  + Continuous Integration
  + Continuous Delivery and Continuous Deployment
  + Infrastructure as Code
  + Configuration Management
  + Orchestration
  + Monitoring
  + Microservices ( introduction )
* DevOps Tools
  + Introduction to DevOps Tools
  + Tools for Build Automation and Continuous Integration
  + Tools for Configuration Management
  + Tools for Virtualization and Containerization
  + Tools for Monitoring
  + Tools for Orchestration
* DevOps and the Cloud
  + DevOps and Cloud providers like AWS

**day-32:**

* Dev-Ops Tools
* scm tool
  + Git
  + Git Branches
  + Git Log, Show and Diff
  + Git Remotes
  + Git Merge
  + Git Merge Conflicts
  + More..
* containerization tool
* docker 101
* docker images & containers
* dockerfile
* docker compose
* docker networking
* ci & cd tool
* Jenkins
* Pipeline script

**day-33:**

* Cloud Computing
  + Fundamentals of Cloud Computing
  + Cloud Computing: Big Picture
  + IASS
  + PASS
  + SAAS
  + Public, Private, Hybrid Cloud
  + AWS, Azure, GCP, IBM cloud and Aliba cloud ( overview )
* AWS Cloud for developer
  + AWS Overview
  + What is Amazon AWS
  + Sign Up for an AWS account
  + AWS IAM
  + Introduction to CLI
  + Overview of AWS services
* Security, Identity & Compliance
  + IAM

**day-34**

* AWS command line tool
* Compute
  + EC2
  + Lambda
  + Elastic Bean Stack
* Storage & Database
* S3
* RDS
* DynamoDB
* Containers
  + ECR, ECS & EKS
  + AWS fargate

**day-35:**

* HA Architecture & Applications
  + Load Balancers
  + SNS, API Gateway
* Developer Tools
  + Code Commit
  + Code Build
  + Code Artifact
  + Code Deploy
  + Code Pipeline
* Management Tool
  + CloudFormation

**Week-8: MicroServices**

**day-36:**

**The Emergence of Microservices Architecture**

* Explore the ideal software development practice
* Learn how a fine-grained Service-Oriented Architecture (SOA) can help to achieve the ideal
* Learn how Microservices attempts to achieve the ideal

**Microservice Design Principles**

* Designing small microservices
* Designing independent microservices
* Designing resilient microservices

**Integrating Microservices**

* Understand design goals when integrating microservices
* Explore effective message formats and lightweight inter-service communication approaches
* Review the pros and cons of various service communication patterns

**Microservice Technologies**

* Learn about popular technologies that enable the development, deployment, and support of microservices

**Decomposing the Monolith**

* Understand monolithic decomposition as an approach towards application modernization
* Review successful decomposition patterns
* Understand helpful practices when decomposing a monolithic application

**Deploying and Maintaining Microservices**

* Explore the intersection of DevOps and microservices
* Learn to leverage virtual, cloud, and containerized environments for microservice deployment
* Discover how to monitor a microservices environment and take appropriate action to enable scaling or react to system faults

**day-37:**

* Overview of serverless platforms
* Synergy between Containers and Serverless
* When (and why) to go Serverless
* DevOps with Serverless
* Debugging, logging and monitoring with Serverless
* Microservices vs Serverless architectures

**day-38:**

* Moleculer framework
* Overview
  + Broker
  + Configuration
  + Services
  + Actions
  + Events
  + Context
  + Lifecycles
  + Logging
  + Middlewares
  + Networking
  + Discovery & Registry
  + Load balancing
  + Fault tolerance
  + Caching
  + Validating
  + Metrics
  + Tracing
* Modules
  + API Gateway

**day-39 :**

* End –To- End - Full stack deployments , with all apps developed so far in training.

**day-40 :**

* **Q & A**

**Post training projects**

