# Java / Design patterns + TDD / QA – 4 days

## Learning Objectives

* Understand basics of Java coding, debugging and code optimization.
* Understand the Java coding best practices.
* Understand SOLID principles
* Understand some of the key creational, structural and behavioral patterns as defined by GoF book.
* Understand how the patterns are implemented in core Java libraries.
* Expanding from Microservices to APIs
* Gain an understanding of the Microservices architecture and pros/cons of this architecture approach
* Gain an understanding of Monolith versus Microservices versus Serverless architecture styles
* Understand common design patterns used in Microservices development
* Understand some of the popular tools and frameworks in use for Microservices development
* High level approach to Testing and Software Quality
* Rationale of developer writing test cases
* Basic understanding of other practices like BDD and how it differed from TDD
* TDD: Process and its pitfall
* Testing Framework (Mockito, Wiremock, etc.)
* What is QA ?
* What is Software Testing? Definition, Basics & Types
* Basic of MTS (Minimum Testing Standards)

**Topics**

**Day 1 (1st half)**

* Java evolution
  + Understanding evolution of java quick understanding
  + Java Versions and Roadmap
* where is Java good to use
  + Platform Compatibility
* where not to use Java
  + Interpreted with Object oriented
  + Writing your first Java program
* Core libraries (io, database, collections, streams, logging, functions, annotations, lambdas)
  + Introduction to package
  + Custom and built-in Package
  + Understand java rich functionality
* Data structures

**Day 1 (2nd half)**

* Functional programming
* **Spring frameworks**

**Day 2 (1st half)**

* GoF patterns - Creational, Structural, Behavioral
  + Singleton
  + Factory
  + Proxy
  + Façade
  + Command

**Day 2 (2nd half)**

* Java Debugging
* Application Profiling
* Program Documentation
* **Software quality and basic of Unit testing (1.5 hours)**
* Definition of Unit and technique to test Unit in isolation
* Example of testing simple math like units
* Introduce JUnit framework and basic hands on examples

**Day 3 (1st half)**

* How to test Unit with dependencies (3.5 hours)
* Introduce the concept of Mocks and Stubs (hand built)
* Dummy - just fillers
* Fake - an in memory database
* Stubs - provide canned answers to calls made during the test, usually not responding at all to anything outside what's programmed in
* for the test
* Mocks - are what we are talking about here: objects pre-programmed with expectations which form a specification of the calls they
* are expected to receive
* Mocking Database operations
* Introduce a framework which can help with the above - Mockito
* Dependency injection helping to isolate dependencies

**Day 3 (2nd half)**

* Introduction to Microservices
* Understanding the Monolith application and its challenges
* What are Microservices
* Basic building blocks of Microservices
* How do Microservices solve the challenges with Monolith?
* Microservices in action
* How to build, deploy and test Microservices
* Common design patterns for creating Microservices
* REST and JSON
* Popular frameworks in use
* Key benefits
* Key challenges
* Monitoring/Alerting

**Day 4 (1st half)**

* Strategies of testing (1/2 hour)
* State based testing and how the assertion looks
* Interaction based testing and how the assertion looks
* Mockito deep dive (1.5 hours)
* Argument capture
* Answers for complex mocks
* Do return when and do throw when
* Using fake (1 hour)
* Using an in-memory database H2 or HSQL database to do end to end testing

**Day 4 (2nd half)**

* Microservices and APIs
* How to expand Microservices to APIs?
* API infrastructure – API Gateway, Developer Portal
* Using Wiremock (REST API Mock) (1 hours)
* Developing a mocked REST API service
* Beyond Microservices
* Serverless architecture is on the horizon
* Microservices are not a panacea for all - choose wisely
* MTS (Minimum Testing Standards) and Test Coverage

**Design principles + UI - 3 days**

**Learning Objectives**

Upon completion of this module, the participant will be able to:

Understand web development using HTML, CSS, JavaScript and React

Understand some of the key tags, attributes, aesthetics, operators and functions to build web applications

Understand how to write clean, responsive code with good test coverage

Topics

**Solid Design Principles** – 1.5 hrs on Day – 1

1 **Introduction to UI Programming** - 0.5 hrs on Day - 1

* Role of UI Developer Subtopic
* How it's different from UX Design? Subtopic
* Course content - UI Technologies Subtopic
* Expectations from the course Subtopic
* Building blocks Subtopic

2 **Basics of HTML5 -**  1 hr on Day - 1

* What is HTML5? Subtopic
* Mandatory HTML Tags (html, head, body etc.) Subtopic
* Commonly used tags (H1-H6, hr, br, p, ul etc) Subtopic
* Understanding Document Object Model (DOM) Subtopic
* Commonly used attributes (class, id, style etc.) Subtopic
* Layouts, Forms, Buttons and Input Fields Subtopic
* Hands-on programming assignments Subtopic

3 **Basics of CSS3 -**  2 hrs on Day - 1

* What is CSS ? Subtopic
* Applying CSS to elements and priority chain Subtopic
* Adding Borders, Fonts, Backgrounds, Pseudo classes Subtopic
* Adding padding and margins, CSS Box Model Subtopic
* Hands-on programming assignments Subtopic

4 **Web Responsive designs** - 3 hrs on Day - 1

* Overview of various types of devices Subtopic
* CSS media queries Subtopic
* Bootstrap CSS APIs, Explain grid and columns with hands-on coding in details Subtopic
* Hands-on programming assignments Subtopic

5 **Programming with JavaScript -**  4 hrs on Day - 2

* What is JavaScript Language? Subtopic
* Defining and using functions - Default & Rest params, IIFE Subtopic
* Using Variables, Types( var, let, const etc.), Explain scope, hoisting and template literals Subtopic
* Using primitive types and operators Subtopic
* Using conditional statements and loops
* (if-else, for, while, switch etc.), Ternary Operators
* Subtopic
* Understanding data structures -
* (Arrays, Strings, Objects, JSON etc.), Explain es6 methods for updating data structures - map, filter etc…
* Subtopic
* Functional programming, Closures, Prototypes Subtopic
* Asynchronous Javascript, Event Loop, Explain callbacks, promises, async and await Subtopic
* Hands-on programming assignments Subtopic

6 **JavaScript Frameworks - React TSx** - 4 hrs on Day – 2 + 4 hrs on Day - 3

* Introduction to React Subtopic
* Creating React Project Subtopic
* Understanding React Components and Properties, Explain component lifecycle Subtopic
* Stateful and Stateless components, Explain class vs functional components Subtopic
* Working with Events Subtopic
* Forms and Validation Subtopic
* Consuming RESTful web service Subtopic
* React Hooks Subtopic
* Hands-on programming assignments Subtopic

**8.State Management** - 2 hrs on Day - 3

* Redux Subtopic
* Context API in React Subtopic
* Hands-on programming assignments Subtopic

**9. Unit Testing and Code coverage - 2 hrs on Day - 3**

* **Importance of Unit testing**
* **Unit testing and coverage with React Testing Library, Explain with Jest, incl. snapshot testing as well Subtopic**
* **Hands-on programming assignments Subtopic**

**10. Micro Frontends -**  1 hrs on Day - 3

* Introduction to Micro Frontends Subtopic

**CI/CD – 0.5 days (Demo)**

**Learning Objectives**

Upon completion of this module, the participant will be able to:

Comprehension

Understand the flow of code from local machine till production

Understand DevOps as a culture

Understand CI/CD as concept and governance around it

Understand SDLC tools and governance

Understand agile release template and change management

Relate all the above concepts as part of Jenkins pipeline as code

Understanding usage of CI/CD tools using antifactory

Application

Execution of builds and commands for module

**Introduction to CI/CD** (40 mins)

* Why devops and what is devops?
* Introduction to CI/CD w.r.t agile principles
* How devops - Show a pipeline taking code from source to production adhering to governance
* Start with a dummy pipeline (empty stages) which will get filled over the course.
* Checkout build create docker image deploy promote to UAT change management deploy to production

**Bitbucket** (30 mins)

* Bitbucket, Github Enterprise
* Hands-on code review concept
* Using Git(fork, clone springboot project)
* Push the code to Release branch
* Pull request concept

**Introduction of Continuous integration/orchestrator** (40 mins)

* Introduction to Jenkins
* Setup jenkins on fabric (Will keep it ready in interest of time)
* Introduction to jenkins pipeline as a code
* Start with a dummy pipeline (empty stages) which will get filled over the course.
* Checkout build create docker image deploy promote to UAT change management deploy to production
* Static code analysis using sonar

**Artifcatory** (40 mins)

* What is Binary Version Control tool and why we need it?
* Introduction on Artifactory and comparison with any market tool
* Different types of repository
* SDLC binary lifecycle tracking

**Type of environments** (30 mins)

* Introduction to number of environments
* doing configuration management - show helm, saving configurations
* Concept of promotion (image vs configurations)
* Promotion using DAP actions and APIs
* Showing promotion for UAT

**Promotion to Production** (25 mins)

* Service Now change management process
* Basics of change management
* Agile release template
* Promotion using DAP actions and APIs
* Show production promotion

**Wrap Up** (30 mins)

End to end running pipeline from source to production

Q/A

**NFR secure coding – 0.5 days**

**Learning Objectives**

Upon completion of this module, the participant will be able to:

Understand the fundamental techniques of information security

What is NFR

Protocol security

Topics

**Introduction to Information**

* Information governance - The business impacts of cyber breaches and real-world cyber-attacks
* Principle of Information security (CIA)
* Legal, data privacy and regulations

**Threat, Vulnerability and Risk**

* Understanding Threat, Vulnerability, Risk and their types
* identifying threats and Vulnerabilities in an IT infrastructure
* Top Ten Threats of 2021

**Safe and secure – End user awareness**

* Understanding Authorization and Authentication
* Principle of Authorization and Authentication
* Type of Authorization and Authentication

**Application security**

* Importance of Application Security
* Phases of Application Security
* Top 10 OWASP
* Injection
* Broken Authentication:
* Sensitive Data Exposure:
* XML External Entities (XXE)
* Broken Access Control
* Security Misconfiguration
* Cross-Site Scripting XSS
* Insecure Deserialization
* Using Components with Known Vulnerabilities
* Insufficient Logging & Monitoring

**Cryptography**

* Introduction to Cryptography
* Role of Cryptography in information security
* Type of Cryptography

Pre-reading, Resources, Hands-on sessions / exercises

**Database (RDBMS + SQL) – 1.5 days**

**Learning Objectives**

Upon completion of this module, the participant will be able to:

Understand the importance of RDBMS

Be able to model and store data in RDBMS tables

Be able to write SQL commands to create tables and indexes, keys, insert/update/delete data and query data in a relational DBMS

Be able to execute queries using tools such as GUI Oracle PL/SQL Developer

Know the challenges and limitations of RDBMS

Be able to choose between RDBMS vs NoSQL databases

Know the different types of NoSQL options

Know the challenges and limitations of NoSQL

**Topics**

**RDBMS (1 day)**

* Walkthrough on the Service Operations lifecycle
* Introduction to key production process focus areas
* RDBMS overview covering the need for 2/3 tier architectures and databases as a central store for data
* Tables, relationship, keys and normalization
* SQL inserts, updates, select, delete and merge
* Other SQL concepts such as views, indexes, partitions
* Exposure to tools such as SQL Developer
* Database transactions and overview of ACID properties
* Java exercise to cover basic CRUD operations using JDBC which demonstrate connectivity and basic transactions

**NoSQL (0.5 day)**

* Need for NoSQL and brief overview of the different types
* Limitations of RDBMS and advantages of NoSQL (covering challenges around unstructured data, schemas, relationships, object relational mapping, horizontal scalability)
* Basic of document database (MongoDB), covering purpose, use cases, benefits and drawbacks

**Google Cloud (GCP) – 1 day**

**Learning Objectives**

Upon completion of this module, the participant will be able to:

Gain an understanding on the Google Cloud Platform

Participants will be able to leverage GCP for cloud deployments

Participants will be able to understand the best practices for cloud deployment, do's, dont's

**Topics**

**Cloud Compute** –

Explain each of below items and differences between them at high level.- **2Hrs**

* App Engine
* Cloud Run
* Compute Engine

**Containers – 2 Hrs**

* Container Registry
* Container Security
* Google Kubernetes Engine GKE
* Cloud Build

**Databases – 1 Hr**

* Cloud SQL

**API Management** - **1 Hr**

* API Gateway general concepts
* APIGEE API Platform

**Networking – 30 min**

* Cloud DNS, Telemetry, VPC, Network Center

**Storage – 30 min**

* Cloud Storage

**GCP Security – 1 hr**

* Key Management, Secret Manager, Certificates
* Resource Access Management