Module Title	:	**Java with Spring Framework**
Duration	:	**5 days**

Course Description

Intensive and hands-on, the course emphasizes becoming productive quickly as a Java application developer. This course quickly covers the Java language syntax and then moves into the object-oriented features of the language. As a participant you will be able to create basic applications using the Java 8 programming language.

The participants will also be able to describe object-oriented concepts, object-oriented programming (OOP) concepts, and Java platforms and technologies. This course will use Eclipse as an IDE.

A **framework** provides a standard way to build and deploy applications. Framework provides generic functionality, which can be selectively changed by additional user-written code, thus providing application-specific software. Frameworks aim to facilitate software development by allowing programmers to devote their time to meeting software requirements rather than dealing with more standard low-level details of providing a working system, thereby reducing overall development time. For instance, a team developing a banking website can focus on writing code particular to banking rather than the mechanics of request handling, user role management and state management.

Web applications can be developed using existing JEE technologies such as servlets, JDBC, JSP. When application grows without a framework the hard-wired page navigations, control logic, business logic will start causing headaches and the code gets messy. **Spring** is an open source powerful Java application framework, used in a wide range of Java applications. Spring provides enterprise services to Plain Old Java Objects (POJOs) and uses dependency injection to achieve simplification and increase testability. The extensions available in Spring framework assist in building web applications on top of Java EE platform. Spring framework targets to make J2EE development easier to use and promotes good programming practices by enabling a POJO-based programming model.

This course provides an introduction to Spring framework. It discusses the features of Spring and how to utilize Spring to rapidly develop web applications using Spring.

Target Participants

This course is meant for programmers who wish to move to object-oriented programming using Java. This course is designed for participants who are looking for an entry into an application development or a software project management career using Java technologies. The primary target audience for this course is software developers / programmers. Students new & beginner to the Spring Framework.

Program Prerequisites

The course assumes knowledge of programming.

Course Objectives

Participants will learn how to

- Write, compile and execute Java programs
- Build robust applications using Java's object-oriented features
- Create robust applications using Java class libraries
- Understand the basics of Web Application - POST, GET, HTTP, MVC Pattern
- Create Spring MVC Applications using various options
- Run the applications using the embedded servlet containers
- Styling the web project using Bootstrap framework
- Connect to databases using Spring Data
- Build REST Services in Spring Boot
- Secure your applications using Spring Security
- Create an executable JAR of your application
- Understand Spring Boot Actuator and Spring Boot Developer Tools

Mode of Delivery

Every participant will be provided with course notes and are required to bring in their own laptops for hands-on exercise and practice sessions.

Course Outcome

By end of this program every participant will complete "creating a web-based customer profile solution". The solution will allow the administrator to login and create customers. Every customer record will have multiple tabs consisting of address details, order details, payment details and shipping details.
The customer will be able to login to the portal and view their details, modify their address details, order details, and update their payment details.

Course Content

Day 1

1. Writing Java Programs
   - Programming Paradigms
   - What is Java
   - The Object-Oriented Paradigm and Java
   - What is a Java Program
   - Writing the Source Code
   - Compiling the Source Code
   - Running the Compiled Code
   - Installing and configuring Eclipse IDE
   - Downloading and Installing Maven
2. Data Types and Operators
   - Data Types
   - Identifier
   - Strings
   - Arrays
   - Operator
   - Assignment Operator
   - Arithmetic Operators
   - String Concatenation Operator
   - Relational Operators
   - Boolean Logical Operators
3. Statements
   - A Block Statement
   - The if-else Statement
   - The switch Statement
   - The for Statement
   - The while Statement
   - The do-while Statement
   - The break Statement
   - The continue Statement
4. Classes and Objects

- What is a Class
- Declaring Fields in a Class
- Creating Instances of a Class
- The null Reference Type
- Using Dot Notation to Access Fields of a Class
- Default Initialization of Fields
- Access Level Modifiers for a Class
- Import Declaration
- Declaring Methods of a Class
- Local Variables
- Instance Method
- Class Method
- Invoking a Method
- Access Level for Class Members

5. The Object and Objects Classes
   - The Object Class
   - What is the Class of an Object
   - Computing Hash Code of an Object
   - Comparing Objects for Equality
   - String Representation of an Object
   - Cloning Object
   - Finalizing an Object
   - Immutable Objects
   - The Objects Class

Day 2

6. Java Package
   - Package declaration
   - Sub-Packages in Java
   - Naming Convention
   - Importing Java Package
   - Access Modifier
   - Java Archive (JAR) Files
7. Exception Handling

- What is an Exception
- Using a try-catch Block
- Exception Class Hierarchy
- Arranging Multiple catch Blocks
- Throwing an Exception
- Creating an Exception Class
- The finally Block
- The try-with-resources Block
- Multi-Catch Exceptions
- Re-throwing Exceptions
- Try with Resources
- AutoCloseable Interface

8. Wrapper Classes
   - Wrapper Classes
   - Dates and Times
   - Collections
   - Autoboxing and Unboxing
   - Beware of Null Values
   - Overloaded Methods
   - Comparison Operators
9. Inheritance & Interface
   - What is Inheritance
   - Method Overriding and Method Overloading
   - Method Hiding and Field Hiding
   - Disabling Inheritance
   - Implementing an Interface
   - Implementing Multiple Interfaces
   - Interface Inheritance
   - The instanceof Operator
   - Polymorphism – One Object, May Views
10. Pluggable Annotation Processing API
    - Common Annotations and Role Based Annotations
    - @Generated

- @Resource
- @Resources
- @PostConstruct
- @PreDestroy
- @DeclareRoles
- @RolesAllowed
- @PermitAll
- @DenyAll

Day 3

11. Introduction to Spring
    - Overview
    - Spring Framework
    - Why Spring
    - What is Spring
    - Why use Spring
    - Spring Philosophy
    - Spring Architecture
    - Spring Modules
    - Setup Spring
    - A First Spring Application
12. Spring Java Configuration and Annotation based Configuration
    - Shortcomings of Java EE and the Need for Loose Coupling
    - Managing Beans, The Spring Container, Inversion of Control
    - @Configuration and @Bean annotations
    - Defining bean scopes
    - @Import: working with multiple configuration files
    - The Factory Pattern
    - Configuration Metadata - XML, @Component, Auto-Detecting Beans
    - Dependencies and Dependency Injection (DI) with the BeanFactory
    - Setter Injection
    - Launching a Spring Application and obtaining Beans
13. Dependency Injection
    - Using the Application Context

- Constructor Injection
- Property Editors and Factory Methods
- Crucial Namespaces 'p' and 'c'
- Configuring Collections
- Bean Definition Inheritance and Collection Merging
- Expression Languages, SpEL
- Autowiring and component scanning
- Lifecycle annotations: @PostConstruct and @PreDestroy

14. The Spring Container and API
    - The Spring Managed Bean Lifecycle
    - Key interfaces, Annotations and BeanPostProcessors
    - Event Handling and Listeners
    - Message Sources and Internationalization
    - Autowiring Dependencies
15. Other Metadata Configurations
    - Annotation Configuration @Autowired, @Required, @Resource
    - @Component, Component Scans. Component Filters
    - @Value and @Qualifier
    - Life Cycle Annotations
    - Java Configuration, @Configuration, XML free configuration (Optional)
    - The Annotation Config Application Context
16. Spring AOP
    - PointCuts, JoinPoints,
    - Aspects, Advices
    - @Before, @After, @AfterReturning, @AfterThrowing, @Around
    - Annotation Configuration
    - XML Configuration
17. Spring Specifics
    - Groovy Bean Definition DSL
    - Generic Qualifiers for Bean Injection
    - Using CGLib Proxy classes with the objenesis library
    - Using @Conditional and @Lazy
    - Java 8 feature support

Day 4

18. Spring and Persistence
    - Spring and JDBC
    - JdbcTemplate / JdbcDaoSupport
    - Spring and Hibernate
    - HibernateTemplate /HibernateDaoSupport
    - Spring and JPA
    - JpaTemplate / JpaDaoSupport
    - Direct access to JPA via @PersistenceUnit and @PersistenceContext
    - CRUD methods
19. Data Access Support
    - Data Access
    - Exception Handling
    - Data source configuration
    - Templating
    - JDBC Templates
    - Callbacks
    - NamedParameterJdbcTemplate
    - SimpleJdbcInsert / SimpleJdbcCall
20. Transaction Management
    - Transaction Propagation
    - Declarative Transaction Management: @Transactional
    - Annotation Configuration
    - Rollback Rules, Isolation
    - Advisors
    - XML Configuration
21. ORM Integration
    - ORM Data Access
    - Exception Translation
    - Hibernate Support
    - Hibernate Integration
    - Java Persistence API
    - JPA Support
    - JPA Integration

Day 5

22. Develop Web Applications using the Spring Framework – Spring MVC
    - The WebApplicationContext and the ContextLoaderListener
    - Model View Controller
    - Front Controller Pattern
    - DispatcherServlet Configuration
    - Controllers, RequestMapping
    - Working with Forms
    - Getting at the Request @RequestParam
    - Getting at the @RequestHeader
    - Getting at the @CookieValue
    - ModelAndView
    - Using the POST Redirect GET pattern with FlashAttributes
23. Thymeleaf
    - Spring Boot Web Module
    - Thymeleaf Introduction
    - Setup Thymeleaf and Initial Template
    - Spring Boot Developer Tools
    - Game Service Challenge
    - Play Thymeleaf Template
    - Thymeleaf Pre-processing
    - Thymeleaf Template Challenge
    - Thymeleaf Fragments
    - Thymeleaf Fragment Challenge
    - Thymeleaf Decoupled Template Logic
    - Thymeleaf Decoupled Template Logic Challenge
    - Spring Internationalization
    - Thymeleaf Internationalization Challenge
    - Message Generator Internationalization Main Message
    - Message Generator Internationalization Result Message
    - Request Interception
    - Locale Change Interceptor
    - Thymeleaf Recap
24. Advanced techniques

- Spring form tags and Model Binding, @ModelAttribute
- Chain multiple View Resolvers
- Handler Interceptors
- Data Validation JSR303

25. Spring Controllers and Ajax
    - JavaScript (JQuery) access to Controllers
    - URI Templates
    - Using @ResponseBody
    - JSON and XML data exchange
    - HTTP GETS and POSTS
26. RESTful Web Services
    - Core REST concepts
    - REST support in Spring 4.x
    - Use Spring MVC to create RESTful Web services
    - REST specific Annotations in Spring
    - Introduction to spring Microservices
    - Working with RestTemplate
    - URITemplates, @PathVariable, @RequestParam
    - JSON and XML data exchange
    - @RequestMapping
27. Spring Security
    - Securing Web Applications with Spring Security 3.0
    - Spring Security 3.0
    - Authentication and Authorization
    - Programmatic v Declarative Security
    - Getting Spring Security from Maven
    - Spring Security Configuration
    - Spring Security Configuration Example
    - Authentication Manager
    - Using Database User Authentication
    - LDAP Authentication
    - Security Basics
    - Storing Passwords

- Hashing Passwords (MD5 and SHA256)
- Spring Security & JPA
- Auto-configuration/simple security
- JDBC security
- OAuth2 Login Java Config
- OAuth2 Client support
- OAuth2 Security in a microservice architecture