**COURSE HANDOUT**

|  |  |
| --- | --- |
| **Course Title:** | Java Spring Framework |
| **Course Code:** |  |
| **Trainer:** | Mr. Akash Kale |
| **Commencement Date:** | 11-11-2020 – 07:00 AM (IST) |
| **End Date:** |  |
| **Total Session Planned:** | 35 |
| **Weekly Frequency:** |  |
| **Moderator Name:** |  |
| **CoE Head:** |  |
| **Faculty Consultation Day** | For every five session, there would be a consultation session |

Course Type: Theory

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Date** | **Session** | **Topics** | **Theory / Lab** | **Cases Description** | **Assignment** | **Evaluation (Type and Weightage)** |
|  | **1** | **Basics OOPs Concepts Java**   * Introduction to course, expected outcome * Procedural programming: looping array examples, Dry run, 2 dimension array, * Fundamental OOPs concepts: Abstraction, encapsulation, modularity, hierarchy * Classes and method, static variable, static methods |  |  |  |  |
|  | **2** | **Inheritance, Interface, String, Immutability**   * Packages and visibility modifier: public, protected, private and default * Inheritance instance of operator * Abstract classes, interface example and applications, Field modifier |  |  |  |  |

Course Type: Lab

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Date** | **Session** | **Topics** | **Theory / Lab** | **Cases Description** | **Assignment** | **Evaluation (Type and Weightage)** |
|  | **3** | **IO, Java 5 feature, Exception Handling**   * IO, Serialization, Java 5 enhancements * AutoBoxing and Unboxing, Enhanced For loop, Varargs, Static Import * Enums, examples Inner classes, nested classes, examples |  |  |  |  |
|  | **4** | **Exception Handling**   * Exception Handling basics * Try, catch, throws, throw, finally * Exception handling good programming practice |  |  |  |  |
|  | **5** | **Java Collection**   * Important methods of Object class * Collections Framework * List, Map, Set usages introduction * Comparable, Comparator, examples * Usages ArrayList, LinkedList, HashMap, TreeMap, HashSet |  |  |  |  |
|  | **6** | **Advance Java collection, Generics**   * HashMap internal * Comparable, Comparator interface implementation * User define key in HashMap , Collection gotcha * Generics Introduction * Wildcards, PCES, extends, Super, bounded type |  |  |  |  |
|  | **7** | **Java 8 stream processing**   * Introduction to functional programming in Java 8, Why Java 8 * functional interface, Interface evaluation * Lambda expressions * Difference between anonymous inner class vs lambda expression, performance * Passing code with behavior parameterization * Introduction to stream processing |  |  |  |  |
|  | **8** | **Java 8 stream case study, Date and time api**   * Streams Case study and examples * Optional * Java 8 date and time API |  |  |  |  |
|  | **9** | **Java Multi-Threading**   * Multi-Threading * Creating threads: using Thread class and Runnable interface * Thread life cycle * Synchronization * Deadlock introduction, Thread dump |  |  |  |  |
|  | **10** | **JDBC**   * Java Database programming using JDBC * Using statement, Prepared Statement, CRUD application * DAO, DTO pattern, 3 tier application design |  |  |  |  |
|  | **11** | **GIT**   * GIT architecture * common git bash commands: * init, clone, add, status, commit, push, pull, checkout. reset, log,rm, branch, merge, rebase,stash * Branches in GIT * Creating Branches in GIT * Working with Branches in GitHub * GIT branches workflow * Pull Request in GitHub * Resolve conflicts in code   Resolve conflicts in code |  |  |  |  |
|  | **12** | **Maven & Logging**   * Maven Introduction, POM * Maven Build Life Cycle * Maven Build Profiles, Repositories, Plug-ins * Maven Creating Project, Build & Test Project, Dependencies management * Logging Overview, Log4j, SLF4j * Log4J components, Logger, Appenders and Layouts |  |  |  |  |
|  | **13** | **Junit**   * Using JUnit with Eclipse * Basic fundamental of Test driven development * Test methods: testMethod or @Test method * Assertions in TestCase * Testing for exceptions |  |  |  |  |
|  | **14** | **Java Web development: Introducing JEE, Tomcat 9**   * Introducing JEE * Introduction to web components: Servlet/JSP, where they fit * Web Server vs Web Container vs EJB Container * Servlet Introduction, Servlet Life cycle * Handling Form Requests in Servlets |  |  |  |  |
|  | **15** | **Java Web development: JSP**   * Introduction to JSP * EL & JSTL * Expression Language * Introduction to JSTL Tags * Book store Manage Application |  |  |  |  |
|  | **16** | **Spring framework fundamentals**   * Spring vs Spring boot * Introduction to DI * Implementing DI application using xml, annotation and Java code * Setter/Constructor injection, Scopes, c and p namespace * xml configuration |  |  |  |  |
|  | **17** | **Spring framework , Dependency Injection in detail**   * Spring bean life cycle * Spring EL basics,Spring annotation in details * Using Environment to retrieve properties * What are Profiles? ,Activating profiles |  |  |  |  |
|  | **18** | **Aspect Oriented Programming (AOP)**   * What is AOP and cross cutting concern * Applying AOP to existing bank application * @Aspect, @Before, @After, @AfterReturning, @AfterThrowing, @Around, @PointCut |  |  |  |  |
|  | **19** | **Introduction to Spring JDBC**   * How Spring reduce boilerplate code using Jdbc Template * Applying AOP to existing bank application |  |  |  |  |
|  | **20** | **Hibernate framework basics**   * Introduction to Object Relational Mapping (ORM) framework * Hibernate vs JPA * CRUD application with Hibernate |  |  |  |  |
|  | **21** | **Hibernate framework Joins**   * Hibernate Mapping between relationship * One to One, one to many and many to many, Inheritance Mapping |  |  |  |  |
|  | **22** | **Hibernate Performance, Queries**   * Hibernate N+1 problem and solution * HQL Performance issues and solutions |  |  |  |  |
|  | **23** | **REST with Spring Boot**   * How Spring boot works, Spring boot internals * Getting started with REST web service * CRUD REST operation with Spring boot |  |  |  |  |
|  | **24** | **Exception handling, HttpStatus, JSR validation**   * Handling Exceptions with Spring REST using @ControllerAdvice * Using ResponseEntity & http status code * JSR 303 validation |  |  |  |  |
|  | **25** | **HATEOAS, Swagger, Actuator, Caching**   * Spring rest **HATEOAS** * Swagger Documentation * Monitoring APIs with Spring Boot Actuator * Enable caching |  |  |  |  |
|  | **26** | **Spring Security**   * Configuration Spring security with Spring boot * Spring security Internals, how it work * Spring security JPA configuration |  |  |  |  |
|  | **27** | **Microservice with Spring boot**   * Spring boot microservice architecture, 12 factor application * Developing microservice with netflix oss * Service Discovery with Eureka |  |  |  |  |
|  | **28** | **Load balancing, Fault tolerance , config server**   * Load-Balancing with Netflix Ribbon * Fault Tolerance with Netflix Hystrix * Config server and Config Client |  |  |  |  |
|  | **29** | **Zuul proxy,API Gateway N, Distributed tracing**   * Create Zuul Proxy * Configure Zuul * zuul filters Create a filter * Distributed Tracing in Action   Install Zipkin, Zipkin configuration |  |  |  |  |
|  | **30** | **Config server and client, accessing resources in microservice**   * Create Config Server * Create Local Configuration * Access Default Configuration * Configure Dev Configuration * Configure Config Client |  |  |  |  |
|  | **31** | **Introduction to docker**   * Architecture and Workflow * DOcker basics commands * Docker Run,Docker Commit * Docker Layers and Overlay Storage * Launch MySql Container * Volumes and Bind Mounts,Use Volumes, Use Bind Mounts |  |  |  |  |
|  | **32** | **Dockerfile, Docker Compose**   * Creating Dockerfile,Dockerize Micro Services * Push to Docker Hub * Docker Prune Command * Docker Compose * YAML Syntax * Compose in action,Compose Syntax |  |  |  |  |
|  | **33** | **Cloud foundry microservice deployment**  Setting up cloud foundry account  PCF Architecture  PCF Orgs and Spaces  Deploying Spring Boot + MySQL Application to PCF |  |  |  |  |
|  | **34** | **RabbitMQ**  AMQP vs JMS  Asynchronous, message-driven systems with RabbitMQ  Develop message listeners for specific queues and routings  Implement Topics, Queues, Exchanges and Bindings in RabbitMQ |  |  |  |  |
|  | **35** | **Apache Kafka**  What is Apache Kafka  rabbitmq vs kafka, where fit what  Streaming process  Kafka basics components  Kafka Cluster,Workflow  Basic Operations  Simple Producer Example  Consumer Group Example |  |  |  |  |