Avsoft Syllabus

Spring & Spring Boot

- **Spring Core**
  - Spring Injections (Field, Constructor, Setter)
  - Spring IOC (Inversion of Control) and Nested IOC
  - Bean Scopes (Singleton, Prototype, etc.)
  - Bean PostProcessor
  - Spring Bean Lifecycle
  - **Spring AOP** (Aspect-Oriented Programming)
  - Spring Transactions (Propagation and Isolations)
  - Spring Auto-configurations (how it works)
  - Spring Configurations (using XML and Annotations)
  - **Spring Boot Actuator** (Monitoring & Metrics)

- **Spring REST**
  - Introduction to RESTful Services
  - Building REST APIs with Spring
  - Consuming REST APIs using RestTemplate or WebClient
  - HATEOAS in Spring REST
  - Exception Handling in Spring REST
  - **Swagger** (API documentation)
- **Embedded Server**
  - Understanding embedded servers in Spring Boot (Tomcat, Jetty, etc.)
- **Spring Kafka Integration**
  - Kafka Producer and Consumer integration with Spring
- **Spring Cache** (with Redis Cache)
- **Spring Security**
  - Session-based Authentication
  - OAuth2 (Google OAuth 2.0 Integration)
  - JWT Token Authentication
  - Spring Security with **Keycloak** (Role-Based Access Control)
- **Spring Scheduling**
  - Task scheduling with @Scheduled
- **Spring MVC**
  - DispatcherServlet (Internal Flow)
  - Spring WebApplicationContext vs ApplicationContext vs BeanFactory

Java (Core & Advanced)

- **Basics**
  - OOP Concepts
  - Arrays
  - Object Class Methods
- **Collections Framework (with Internals)**
  - **List**: ArrayList, LinkedList, Vector, Stack
  - **Set**: HashSet, LinkedHashSet, TreeSet
  - **Queue**: PriorityQueue
  - **Map**: HashMap, LinkedHashMap, TreeMap, Hashtable, IdentityHashMap, WeakHashMap
- **Concurrent Collections and Internals**
  - ConcurrentHashMap, CopyOnWriteArrayList, BlockingQueue
- **Multithreading & Concurrency**
  - Synchronization
  - Locks (ReentrantLock, ReadWriteLock)
  - Executor Framework (Thread Pooling)
  - ThreadLocal, Callable/Future
  - Virtual Threads (Project Loom)
  - Thread Groups
  - Advanced Multithreading Concepts (Fork/Join, CompletableFuture)

- **JVM Internals**
  - JVM Architecture
  - Class Loading Mechanism
  - Garbage Collection (G1, CMS, etc.)
  - Cloning (Shallow and Deep)
  - Serialization/Deserialization
- **Comparable vs Comparator**

Hibernate

- Hibernate Session, SessionFactory
- First-Level Cache vs Second-Level Cache
- Entity Relations (One-to-One, One-to-Many, Many-to-Many)
- Hibernate Query Language (HQL)
- Transactions with Hibernate
- Hibernate Utility Class

Java 8 Features

- Functional Programming (Lambdas, Method References)
- Functional Interfaces (Function, Predicate, Consumer, Supplier)
- Stream API (Processing collections)
- Interview-Oriented Java 8 Programs

Design Patterns

- SOLID/ACID Principles with Real-Time Use Cases
- Design Patterns:
  - Factory
  - Abstract Factory
  - Singleton
  - Decorator
  - Adapter
  - Command
  - Template
  - Facade

Angular Syllabus

1. Introduction to Angular

- What is Angular?
- Difference between AngularJS and Angular
- Angular Features and Architecture
- Installing Angular CLI
- Setting up the Development Environment
- Creating your first Angular Application

2. Angular Components

- Components in Angular
- Creating Components using Angular CLI
- Component Decorators
- Component Templates, Styles, and Metadata
- Component Lifecycle Hooks (ngOnInit, ngOnDestroy, etc.)
- Component Interaction (Input and Output properties)
- ViewChild, ContentChild, and View Encapsulation

3. Angular Directives

- Structural Directives (*ngIf, *ngFor, *ngSwitch)
- Attribute Directives (ngClass, ngStyle)
- Custom Directives (Building your own directives)

4. Angular Data Binding

- Interpolation
- Property Binding
- Event Binding
- Two-way Binding ([(ngModel)])
- Template Reference Variables

5. Angular Services and Dependency Injection (DI)

- What are Services in Angular?
- Creating Services
- Dependency Injection in Angular
- Injecting Services into Components
- Hierarchical Dependency Injection

6. Angular Routing and Navigation

- Setting up Angular Router
- Configuring Routes and Child Routes
- Lazy Loading Modules
- Route Guards (CanActivate, CanDeactivate)
- Route Parameters and Query Parameters
- Router Events and Navigation Guards

7. Angular Forms

- Template-driven Forms
- Reactive Forms
- FormControl, FormGroup, FormBuilder
- Form Validation (Built-in Validators & Custom Validators)
- Dynamic Forms
- Handling Form Submissions
- Error Handling and Displaying Validation Messages

8. Angular Pipes

- Built-in Pipes (e.g., DatePipe, UpperCasePipe, CurrencyPipe)
- Custom Pipes (Creating reusable pipes)
- Pure and Impure Pipes

9. Angular Modules

- Introduction to NgModules
- Creating Feature Modules
- Shared Modules
- Core Module
- Module Lazy Loading

10. HTTP Client and Backend Interaction

- Using Angular HttpClient to make API requests
- GET, POST, PUT, DELETE requests
- Handling HTTP Errors
- Interceptors in Angular
- Consuming REST APIs
- Promises vs. Observables

11. RxJS (Reactive Extensions for JavaScript)

- Introduction to RxJS in Angular
- Observables and Subscriptions

- Common RxJS Operators (e.g., map, filter, switchMap, mergeMap)
- Handling Asynchronous Data Streams
- Subject and BehaviorSubject
- Error Handling in Observables

12. Angular Styling

- Angular Styling with CSS
- Using Angular Material for UI Components
- Adding External Libraries (Bootstrap, FontAwesome, etc.)
- Theming and Customizing Angular Material
- Responsive Design in Angular

14. State Management in Angular

- Introduction to State Management
- Managing State with Services
- Using NgRx for State Management
- Store, Actions, Reducers, and Effects in NgRx
- NgRx DevTools and Debugging
- End-to-End (E2E) Testing with Protractor or Cypress

16. Advanced Angular Concepts

- Angular Universal for Server-Side Rendering (SSR)
- Progressive Web Apps (PWA) with Angular
- Angular Change Detection Strategy (OnPush vs Default)
- Content Projection using ng-content
- Optimizing Angular Applications (Lazy Loading, Tree Shaking)
- Building and Deploying Angular Applications

Real-Time Project Development

- **Mini Projects**: Four mini-projects focusing on different aspects of the Spring ecosystem.
- **Live Real-Time Project**:
  - Tools: Maven, Git, JIRA, etc.
  - Server: Real Production Server
  - Team Size: 10 members
  - **Agile Methodology**:
    - Sprint-wise planning
    - Daily Scrum meetings
    - Backlog Refinement sessions

- **Deploying on Production Servers**
- ----------

Tools

- **Real-Time Tools**:
  - Jira, Git, Bitbucket, Jenkins
  - Maven, Gradle
  - Docker, Kubernetes (Optional)
  - Prod Deployment

Road Map

- **Theory**: 90 mins
- **Practical**: 180 mins
- **System Design Concepts**
- Mini Projects with Real-Time Scenarios
- **Weekly Mock Interviews**
- **Resume Building**: Tailoring the resume for tech interviews
- Uploading Resume on Job Portals
- **Support After Job**
- **Pay After Job**