| Roadmap Day | Module | Title | Content | Session Activities | Duration | Reference Notes |
|---|---|---|---|---|---|---|
| 1 | Programming Foundation | Introduction to Java & Setup | **Intro to Java:**<br>Overview of Java Platform (JVM, JRE, JDK).<br>Key Features: Platform Independence, Object-Oriented, Robust.<br>Setting up the Development Environment (JDK 21, IDE - IntelliJ IDEA).<br>**Java Syntax & First Program:**<br>Structure of a Java Program.<br>The main method.<br>Keywords, Identifiers, and Coding Conventions.<br>Comments.<br>**Variables & Primitive Data Types:**<br>Variable Declaration & Initialization.<br>Primitive Types: byte, short, int, long, float, double, char, boolean. | | 3 | |
| 2 | Programming Foundation | Control Flow, Loops & Methods In Java | **Control Flow:**<br>Conditional Statements: if, if-else, if-else-if ladder.<br>switch statement (traditional and with arrows in Java 14+).<br>**Loops:**<br>for loop, while loop, do-while loop.<br>Loop Control Statements: break, continue, labels.<br>**Methods in Java**:<br>Defining Methods, Parameters, and Return Types.<br>Method Signature.<br>Method Overloading (Compile-time Polymorphism). | Build a simple grade calculator using if-else.<br>Create a number guessing game using loops.<br>Print patterns (e.g., pyramids) using nested loops | 3 | |
| 3 | Programming Foundation | Arrays ,String API & Wrapper Classes | **Arrays:**<br>Single-dimensional Arrays.<br>Accessing, iterating (for, for-each).<br>Multi-dimensional Arrays (Introduction)<br>**String Class:**<br>String Immutability.<br>Important Methods: charAt(), length(), substring(), equals(), indexOf() etc.<br>StringBuilder vs StringBuffer .<br>**Wrapper Classes:**<br>Purpose of Wrapper Classes (Integer, Double, Character, etc.).<br>Autoboxing and Unboxing. | Write methods to perform operations on arrays: find max/min, average, search for an element | 3 | |
| 4 | Programming Foundation | Intro to OOP: Classes & Objects and OOps Features | **Intro to Classes and Objects:**<br>Principles of Object-Oriented Programming.<br>Defining a Class, Creating Objects.<br>Constructors (Default and Parameterized).<br>The this keyword.<br>**Encapsulation:**<br>Access Modifiers (public, private, protected).<br>Implementing Getters and Setters.<br>**Inheritance**:<br>extends keyword.<br>super keyword (for accessing parent class members and constructors). | Model a real-world entity like a BankAccount or Employee class with fields and behaviors.<br>Ensure all fields are private and accessed via getters/setters. | 3 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| **5** | Programming Foundation | Core OOP: Polymorphism,Abstraction,Interfaces& Packag | **Polymorphism:**<br>Method Overriding (Runtime Polymorphism).<br>Rules for overriding.<br>@Override annotation<br>**Abstraction**:<br>Abstract Classes.<br>Abstract Methods.<br>**Interfaces**:<br>Defining and Implementing Interfaces.<br>Default and Static Methods (Java 8+).<br>Multiple Inheritance using Interfaces.<br>**Packages:**<br>Organizing classes into packages.<br>import statements. | Create an inheritance hierarchy (e.g., Vehicle -> Car, Bike).<br>Demonstrate polymorphism by storing different subclass objects in a parent class reference. | 3 | |
| **6** | Programming Foundation | Exception Handling | **Exception Handling**:<br>Checked vs. Unchecked Exceptions.<br>try, catch, finally blocks.<br>throw and throws keywords.<br>Creating Custom Exception Classes | Write code that handles FileNotFoundException, ArithmeticException, etc.<br>Create a custom InsufficientFundsException for the BankAccount class. | 3 | |
| **7** | Programming Foundation | Collections Framework | **Collection Framework Overview:**<br>Core Interfaces: Collection, List, Set, Map, Queue.<br>**List & Set Interface:**<br>ArrayList (resizable array).<br>LinkedList (doubly-linked list).<br>Iterating with Iterator and for-each.<br>HashSet (unordered, unique elements).<br>LinkedHashSet (ordered iteration).<br>**Map Interface, Queue Interface &PriorityQueue (intro).**<br>HashMap (key-value pairs).<br>LinkedHashMap (maintains insertion order). | Build a simple student management system using an ArrayList to add, remove, and list students.<br>Compare performance of ArrayList vs. LinkedList for adding elements at the beginning. | 3 | |
| **8** | Programming Foundation | Generics & Lambda Expressions | **Generics:**<br>The need for Generics (Type Safety, eliminating casts).<br>Generic Classes and Methods.<br>Bounded Type Parameters (<T extends Number>).<br>Wildcards (?, ? extends T, ? super T).<br>**Lambda Expressions:**<br>Functional Interfaces (Runnable, Comparator, custom).<br>Syntax of Lambda Expressions.<br>Method References (Class::method). | Create a generic Box<T> class.<br>Write a generic method to print all elements of any List. | 3 | |
| **9** | Programming Foundation | File I/O & Concurrency | **File I/O :**<br>Files utility class for reading/writing.<br>**Multi-threading (Basics):**<br>Creating threads: extending Thread vs. implementing Runnable.<br>Thread lifecycle. | Write a program to read a configuration file and write results to an output file.<br>Create two threads: one to print even numbers, another to print odd numbers. | 3 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| **10** | Programming Foundation | Java Advanced Topics & Features | **Memory Management (Overview):**<br>Stack vs. Heap memory.<br>Introduction to Garbage Collection.<br>**Java Features (8 to 21):**<br>Highlights: var (Local Variable Type Inference), record, sealed classes, Text Blocks.<br>**Annotations & Enums:**<br>Built-in annotations (@Override).<br>Defining and using enum types.<br>**Internalization (I18N) - Overview:**<br>Locale class.<br>ResourceBundle for externalizing strings. | | 3 | |
| **11** | DSA | Introduction to DSA | **Introduction to DSA:**<br>What is DSA?<br>Why DSA for backend engineers?<br>Time and Space Complexity Analysis (Big O Notation) | Session Actitvity:<br>Analyze the time and space complexity of a function that checks whether a given array contains a duplicate element. | 3 | |
| **12** | DSA | Arrays and Strings | **Arrays and Strings:**<br>1D & 2D Arrays<br>Common Array Problems<br>String Manipulation Techniques<br>StringBuilder & StringBuffer<br>Sliding Window & Two Pointer Techniques | | 3 | |
| **13** | DSA | Recursion and Backtracking | **Recursion and Backtracking:**<br>Introduction to Recursion<br>Recursive Tree Patterns<br>Backtracking Problems (e.g., N-Queens, Sudoku Solver)<br>Memoization Basics | | 3 | |
| **14** | DSA | Practice Session | Codekata & Leetcode Problems:Arrays,Strings,Recursion & B | Session Activity:<br>LeetCode Problems:<br>Two Sum<br>Best Time to Buy and Sell Stock<br>Move Zeroes<br>Subsets<br>Permutations<br>N-Queens | 3 | |
| **15** | DSA | Searching and Sorting | **Searching and Sorting:**<br>Linear & Binary Search<br>Binary Search on Answers<br>Bubble, Selection, Insertion Sort | | 3 | |
| **16** | DSA | Searching and Sorting | **Searching and Sorting**<br>Merge Sort & Quick Sort<br>Counting Sort, Radix Sort (Basics)<br>Comparator and Comparable Interfaces | | 3 | |
| **17** | DSA | Practice Session | Codekata & Leetcode Problems:Searching & Sorting | Session Activity:<br>LeetCode:<br>Binary Search<br>Search a 2D Matrix<br>Top K Frequent Elements<br>Median of Two Sorted Arrays | 3 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| **18** | DSA | Linked Lists | **Linked Lists**<br>Singly and Doubly Linked List<br>Fast & Slow Pointers<br>Reversal Techniques<br>Detecting and Removing Cycles<br>Merge Two Sorted Lists, Middle Node | | 3 | |
| **19** | DSA | Stacks and Queues | **Stacks and Queues**<br>Stack using Arrays and LinkedList<br>Queue and Deque Implementations<br>Infix, Prefix, Postfix Expressions Cont.<br>Monotonic Stack/Queue Problems<br>LRU Cache (LinkedHashMap & Deque) | | 3 | |
| **20** | DSA | Practice Session | Codekata & Leetcode Problems:Linked Lists,Stacks & Queues | LeetCode:<br>Reverse Linked List<br>Merge Two Sorted Lists<br>Linked List Cycle<br>Valid Parentheses<br>Min Stack | 3 | |
| **21** | DSA | Hashing | **Hashing**<br>HashMap, HashSet, Hashtable<br>Frequency Counting Problems<br>Grouping Anagrams, Two Sum, Subarray Sums<br>Custom Hashing Techniques | | 3 | |
| **22** | DSA | Trees and Binary Trees | **Trees and Binary Trees**<br>Binary Tree Basics<br>Tree Traversals (Inorder, Preorder, Postorder)<br>Level Order & Zigzag Traversals<br>Height, Diameter, Balanced Trees<br>Constructing Trees from Traversals | | 3 | |
| **23** | DSA | Binary Search Trees (BST) | **Binary Search Trees (BST)**<br>BST Operations (Insert, Delete, Search)<br>Inorder Successor & Predecessor<br>BST Validation<br>Kth Smallest Element<br>Convert Sorted Array/List to BST | | 3 | |
| **24** | DSA | Heaps and Priority Queues | **Heaps and Priority Queues**<br>Min Heap and Max Heap Implementation<br>PriorityQueue Class in Java<br>Heap Sort<br>Top K Elements Problems<br>Median of a Stream | | 3 | |
| **25** | DSA | Practice Session | Codekata & Leetcode Problems:Trees and Binary Trees,BST, Heaps and Priority Queues | Session Activity:<br>Leetcode:<br>Maximum Depth of Binary Tree<br>Symmetric Tree<br>Lowest Common Ancestor<br>Kth Largest Element<br>Merge K Sorted Lists | 3 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| **26** | DSA | Graphs | **Graphs**<br>Graph Representations (Adjacency List/Matrix)<br>BFS and DFS Traversals<br>Detect Cycle in Undirected/Directed Graph<br>Topological Sorting<br>Shortest Path: Dijkstra & BFS-based<br>Connected Components<br>Union-Find (Disjoint Set Union) | | 3 | |
| **27** | DSA | Greedy Algorithms | **Greedy Algorithms**<br>Greedy Strategy Design<br>Activity Selection<br>Job Scheduling<br>Fractional Knapsack<br>Huffman Encoding (Intro) | | 3 | |
| **28** | DSA | Practice Session | Codekata & Leetcode Problems:Graphs,Greedy Algorithms | Session Activity:<br>LeetCode:<br>Number of Islands<br>Clone Graph<br>Jump Game<br>Gas Station | 3 | |
| **29** | Database | Database Introduction | What is a Database?<br>Importance of Databases in Applications<br>Relational vs. Non-Relational Databases<br>Key Differences Between SQL and NoSQL<br>When to Use SQL vs. NoSQL | Session Activity:<br>Sqlkata Problems | 3 | |
| **30** | Database | MYSQL | Introduction to MySQL<br>Installing and Setting Up MySQL<br>Database Basics (Databases, Tables)<br>Data Types in MySQL<br>Basic SQL Commands (SELECT, INSERT, UPDATE, DELETE)<br>WHERE Clause and Filtering Data | Session Activity – MySQL Practical Tasks<br>Setup & Basics<br><br>Install MySQL and create a database named training_db .<br>Create a table students with fields:<br>id (INT, PK, AUTO_INCREMENT)<br>name (VARCHAR 50)<br>course (VARCHAR 30)<br>marks (INT)<br><br>2 CRUD Operations<br><br>Insert 5 sample student records.<br>Update marks of any one student to 85.<br>Delete a record where marks < 50.<br>Fetch all students who scored more than 70 using a WHERE clause. | 3 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| **31** | Database | MongoDB | Introduction to MongoDB<br>Installing and Setting Up MongoDB<br>Documents and Collections<br>Connecting to a MongoDB Database Using the MongoDB Sh<br>MongoDB Compass<br>CRUD Operations (Create, Read, Update, Delete)<br>Querying Documents (Filters, Projection)<br>Aggregation Framework<br>Indexing Basics | Session Activity – MongoDB<br>Practical Tasks<br>Setup & Collections<br>Install MongoDB and connect using Mongo Shell or Compass.<br>Create a database company_db and a collection employees.<br>CRUD Tasks<br><br>Insert 5 employee documents with fields:<br>name<br>age<br>department<br>skills (array)<br>Update one document to add a new skill "Docker".<br>Delete all documents where age < 25.<br>Fetch only the name and department for all employees using projection. | 3 | |
| **32** | Database | MongoDB | Data Modeling and Schema Design<br>MongoDB Transactions<br>Working with Embedded Documents and Arrays<br>Replication and Sharding (Overview)<br>Backup and Restore<br>User Roles and Security | | 3 | |
| **33** | Database | SQLKATA | Practice Session :SQLKata | https://www.guvi.in/sqlkata/sql/ | 3 | |
| **34** | Module 4-Backend Server | Getting Started with Spring & Spring Boot | Introduction to Spring and Spring Boot<br>What is Spring Framework<br>Spring Core concepts (IoC, DI – overview)<br>Why Spring Boot<br>Spring Boot architecture | Session Activity:<br>Identify how Spring Boot manages object creation and dependency injection in a simple Student application | 3 | |
| **35** | Module 4-Backend Server | Spring Boot Project Setup & Configuration | Setting Up Spring Boot Projects<br>Creating Spring Boot project (Spring Initializer)<br>Project structure explanation<br>application.properties / application.yml<br>Running Spring Boot applications<br>Embedded servers (Tomcat) | Sesion Activity:<br>Create a Spring Boot project and explain the role of each major folder and configuration file | 3 | |
| **36** | Module 4-Backend Server | Developing RESTful APIs with Spring Boot | Building RESTful APIs using Java<br>REST principles<br>HTTP methods (GET, POST, PUT, DELETE)<br>@RestController & @RequestMapping<br>Request & Response flow<br>Designing REST APIs | Session Activity:<br>Building RESTful APIs<br>Design REST endpoints for managing students using correct HTTP methods. | 3 | |
| **37** | Module 4-Backend Server | Module 4 - Spring Boot Fundamentals | Handling User Input and Validations<br>@RequestBody, @PathVariable, @RequestParam<br>Bean Validation annotations<br>Custom validation messages<br>Exception handling basics<br>Global exception handling<br>Error Handling and Response Structuring<br>HTTP status codes | Session Activity:(Use Same Above Student App)<br>Validate a User API request and decide the correct error response and status code. | 3 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| **38** | Module 4-Backend Server | Module 4 - Spring Boot Fundamentals | Implementing Search and Filters in APIs<br>Query-based search<br>Filtering REST responses<br>Pagination and sorting | Session Activity:<br>Design a search API that filters products and supports pagination and sorting. | 3 | |
| **39** | Module 4-Backend Server | Module 4 A - Database Integrations using MongoDB | Designing MongoDB Schemas for Real-World Use Cases<br>NoSQL concepts<br>MongoDB document model<br>Schema design principles<br>Embedded documents vs references<br>Real-world schema examples | | 3 | |
| **40** | Module 4-Backend Server | Module 4 A - Database Integrations using MongoDB | Connecting to MongoDB in Java Spring Boot Applications<br>MongoDB installation<br>MongoDB Atlas overview<br>MongoDB Compass<br>Spring Data MongoDB configuration<br>Connecting Spring Boot with MongoDB | Session Activity:<br>MongoDB CRUD Operations | 3 | |
| **41** | Module 4-Backend Server | Module 4 A - Database Integrations using MongoDB | SpringBoot+MongoDB CRUD Operations:<br>Insert operations<br>Find & query documents<br>Repository methods<br>Custom query methods<br>Delete operations<br>Handling missing documents | Session Activity:<br>MongoDB CRUD Operations | 3 | |
| **42** | Module 4-Backend Server | Module 4 A - Database Integrations using MongoDB | MongoDB CRUD Operations: Modifying Query Results<br>Update operations<br>MongoDB Indexes<br>What are indexes<br>Types of MongoDB indexes<br>Index creation & usage<br>Working with Embedded Documents and References | | 3 | |
| **43** | Module 4-Backend Server | Module 4 A - Database Integrations using MongoDB | Using MongoDB Atlas and MongoDB Compass<br>Cloud MongoDB setup<br>Managing collections<br>Querying via Compass<br>Monitoring & performance overview | Session Activity:<br>use MongoDB Compass to inspect and query data stored in MongoDB Atlas. | 3 | |
| **44** | Module 4-Backend Server | Module 4B – Authentication and Security | Introduction to Authentication and Authorization<br>Intro to Spring Security<br>Basic Authentication using Spring Security<br>Securing REST APIs basics<br>Implementing User Signup and Login | | 3 | |
| **45** | Module 4-Backend Server | Module 4B – Authentication and Security | Implementing User Signup and LoginUser registration flow<br>Login mechanism<br>Password handling concepts<br>Role-Based Access Control in Applications<br>Securing API Endpoints in Spring Boot<br>Endpoint security configuration,Authorization filters | Session Activity:<br>User Signup, Login & Role Management | 3 | |
| **46** | Module 4-Backend Server | Module 4C – Testing, Documentation | API Documentation using Swagger or Similar Tools<br>Testing APIs with Postman and Automated Tests<br>Improving Performance and Reducing Redundancy | Session Activity:<br>Document APIs using Swagger | 3 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| **47** | Module 4-Backend Server | Module 4 D - Asynchronous Communication ( Apache Kaf | Introduction to Event-Driven Architecture<br>Synchronous vs asynchronous communication<br>Use cases in microservices<br>Basics of Kafka and Zookeeper<br>Setting Up Kafka and Zookeeper Locally<br>Understanding Topics, Producers, and Consumers<br>Creating Kafka Producers and Consumers in Java | | 3 | |
| **48** | Module 4-Backend Server | Module 4 D - Asynchronous Communication ( Apache Kaf | Creating Kafka Producers and Consumers in Java<br>Kafka Integration with Spring Boot<br>Real-World Use Cases of Kafka in Microservices<br>Monitoring and Managing Kafka Topics | | 3 | |
| **49** | Module 4-Backend Server | Backend Project Implementation | **Order Management System(Backend)**Use Case:<br>APIs to manage users, products, and orders for an e-commerce platform.<br>Key Backend Concepts Covered:<br>User signup & role-based access (Admin/User)<br>Product CRUD with validation<br>Order placement & order status tracking<br>Search products by name/category<br>Centralized exception handling | | 3 | |
| **50** | Module 4-Backend Server | Backend Project Implementation | **Event Registration & Ticketing (Backend)**<br><br>Use Case:<br>Backend system for managing events, users, and ticket bookings.<br>Key Backend Concepts Covered:<br>User registration & login<br>Event creation & seat availability<br>Ticket booking validation<br>Error handling for overbooking<br>Search events by date/location | | 3 | |
| **51** | Module 5 : System Design | System Design Basics & Requirements | What is System Design (Product vs Project view)<br>System Design in real companies & interviews<br>Understanding business problems<br>Functional Requirements (features, workflows)<br>Non-Functional Requirements (scalability, latency, security, av<br>Identifying constraints & assumptions | Convert a problem statement into FR & NFR<br>Requirement analysis for URL Shortener<br>Whiteboard requirement discussion | 3 | |
| **52** | Module 5 : System Design | API Design & Architecture Styles | API Design principles (REST, resource modeling)<br>Request–response lifecycle<br>Sync vs Async communication<br>Error handling & versioning<br>Monolithic vs Microservice Architecture<br>When to choose which architecture | Design REST APIs for User & Order Service<br>API contract writing exercise<br>Monolith vs Microservices decision case | 3 | |
| **53** | Module 5 : System Design | Scalability & Caching Design | Scaling challenges in real systems<br>Load Balancers (working & types)<br>Horizontal vs Vertical Scaling<br>Stateless vs Stateful services<br>Caching strategies (Client, Server, CDN)<br>Cache eviction & invalidation | Design scalable architecture for E-commerce App or similar Usecase<br>Cache placement decision exercise<br>Identify bottlenecks in given system | 3 | |
| **54** | Module 5 : System Design | Async Systems & Database Scaling | Asynchronous communication basics<br>Message Queues & Event-driven systems<br>Use cases for Kafka / RabbitMQ<br>Database scaling challenges<br>Replication (read replicas)<br>Sharding strategies<br>CAP Theorem with real examples | Design async Order Processing System<br>DB scaling decision case<br>CAP theorem scenario discussion | 3 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| **55** | Module 5 : System Design | Reliability, Availability & Traffic Control | High Availability concepts<br>Fault Tolerance strategies<br>Failover & redundancy<br>Rate Limiting techniques<br>Retry logic & idempotency<br>Throttling strategies<br>API Gateway responsibilities<br>Service Discovery & Orchestration | Design highly available Payment System<br>Apply rate limiting to APIs<br>End-to-end system design walkthrough | 3 | |
| **56** | Module 5 : System Design | LLD, SOLID & Design Patterns | Difference between HLD & LLD<br>Breaking HLD into classes<br>SOLID principles (with violations)<br>Factory Pattern<br>Strategy Pattern<br>Observer Pattern<br>When & why to use patterns | Convert HLD into class design<br>Pattern-based solution design | 3 | |
| **57** | Module 5 : System Design | Clean Code, UML & Testable Design | Designing classes & relationships<br>Composition vs Inheritance<br>Writing clean & maintainable code<br>Designing for testability<br>UML Class Diagrams<br>UML Sequence Diagrams<br>Mapping UML to code | Create UML for Booking System<br>Sequence diagram for API flow<br>LLD review & improvement session | 3 | |
| **58** | Module-6A – HTML | HTML Foundations & Structure | What is HTML & how browsers render it<br>HTML basics: Tags, Elements, Attributes<br>HTML Document Structure (doctype, html, head, body)<br>Text formatting tags<br>Semantic tags (header, nav, section, article, footer)<br>Images(img, attributes, accessibility)<br>Links & navigation (a, target, anchor links)<br>Lists(Ordered & Unordered Lists),<br>Tables(thead, tbody, tr, td, th) | Build a semantic webpage from scratch<br>Create navigation menu<br>Design content layout using lists & tables | 3 | |
| **59** | Module-6A – HTML | Forms & User Input | HTML Forms & form structure<br>Input types (text, email, password, radio, checkbox, file, date<br>Labels & Fieldsets<br>Buttons & form submission<br>Basic form validation (required, pattern) | Create registration & login forms<br>Apply form validation rules<br>Simulate form submission | 3 | |
| **60** | Module-6B – CSS | CSS Basics, Box Model & Display | Introduction to CSS<br>CSS Basic: selectors, properties, values<br>Colors, fonts & text styling<br>Box model (margin, padding, border)<br>Display properties (block, inline, inline-block) | Style HTML pages & forms<br>Build card UI using box model<br>Fix spacing & alignment issues | 3 | |
| **61** | Module-6B – CSS | Layouts & Responsive Design | CSS layouts overview<br>Flexbox (container & item properties)<br>CSS Grid (rows, columns, gaps)<br>Responsive design principles<br>Media queries<br>Mobile-first approach<br>Units: %, em, rem, vw, vh | Build responsive layout using Flexbox<br>Create Grid-based dashboard<br>Apply mobile-first media queries | 3 | |
| **62** | Module-6C – Tailwind CSS | Tailwind Utilities & Components | Introduction to Utility-First CSS<br>Setting Up Tailwind in Projects<br>Common Utility Classes (Spacing, Colors, Fonts)<br>Customizing Tailwind Config<br>Responsive Design with Tailwind<br>Creating Reusable Components<br>Animation & Transitions with Tailwind | Build responsive navbar & cards using TailwindCss<br>Create mini landing page with animations | 3 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| **63** | Module 7 – JS | JavaScript Fundamentals | What is JavaScript & where it runs<br>Variables (var, let, const)<br>Data Types (primitive & non-primitive)<br>Operators (arithmetic, comparison, logical,relational etc.) | Write basic JS programs<br>Variable & datatype experiments in browser console<br>Operator-based logic tasks | 3 | |
| **64** | Module 7 – JS | Control Structures & Functions | Conditional statements (if, else, switch)<br>Loops (for, while, do-while)<br>Functions (declaration, expression)<br>Function parameters & return values | Solve logic problems using loops<br>Build reusable utility functions<br>Debug JS flow using console logs | 3 | |
| **65** | Module 7 – JS | Scope, Hoisting & Advanced Functions | Global vs local scope<br>Block scope (let, const)<br>Hoisting (variables & functions)<br>Function types (arrow, anonymous, IIFE) | Predict output exercises<br>Convert functions to arrow functions | 3 | |
| **66** | Module 7 – JS | DOM Basics & Manipulation | DOM introduction & tree structure<br>Selecting elements (getElementById, querySelector)<br>Modifying content & styles<br>Creating & removing DOM nodes | Dynamic content update on webpage<br>Create elements dynamically<br>Remove/update nodes on user action | 3 | |
| **67** | Module 7 – JS | Events & Form Handling | Event handling basics<br>Event listeners<br>Event bubbling & delegation<br>Form events (submit, input, change)<br>Basic form input handling | Add interactive buttons<br>Event delegation practice<br>Client-side form validation logic | 3 | |
| **68** | Module 7 – JS | Objects & Prototypes | JavaScript objects & properties<br>Methods & this keyword<br>Constructor functions<br>Prototypes & prototype chain | Build object-based programs<br>Create reusable object templates<br>Prototype behavior demonstration | 3 | |
| **69** | Module 7 – JS | Async JavaScript Concepts | Synchronous vs Asynchronous JS<br>Callbacks & callback issues<br>Promises (then, catch, finally)<br>Promise chaining | Simulate async operations<br>Convert callbacks to promises<br>Handle async data flows | 3 | |
| **70** | Module 7 – JS | Async/Await & Fetch API | Async/Await syntax<br>Error handling with try/catch<br>Fetch API basics<br>Handling API responses (JSON)<br>Real-world API usage | Fetch data from public API<br>Display API data on UI<br>Mini async JS project | 3 | |
| **71** | Module 8 – React | Introduction to React & JSX | What is React & why React is used<br>SPA vs MPA<br>React vs traditional JS<br>React project structure<br>JSX basics<br>JSX expressions & rules | Create first React app<br>Explore folder structure<br>Write JSX with dynamic data | 3 | |
| **72** | Module 8 – React | React Components | What are components<br>Functional components<br>Class components (intro)<br>Component reuse & composition<br>Export & import components | Create multiple components<br>Convert UI into components<br>Compare functional vs class components | 3 | |
| **73** | Module 8 – React | Props & State | Props concept & data flow<br>Passing props to components<br>State basics<br>Updating state<br>Props vs State comparison | Pass data between components<br>Build counter app using state<br>Debug props & state updates | 3 | |
| **74** | Module 8 – React | Lifecycle & Hooks Basics | Component lifecycle (intro)<br>Why hooks were introduced<br>useState hook<br>useEffect hook<br>Dependency array | Replace lifecycle with hooks<br>Side-effect handling using useEffect<br>Build data-update components | 3 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| 75 | Module 8 – React | Advanced Hooks & State Management | Custom hooks<br>Reusing logic with hooks<br>useContext for state management<br>Context provider & consumer<br>Avoiding prop drilling | <br><br>Create custom hook<br>Global state using useContext<br>Refactor component tree | 3 | |
| 76 | Module 8 – React | React Router | SPA routing concept<br>React Router setup<br>Routes & Route component<br>Nested routes<br>Dynamic routing | <br><br>Add routing to React app<br>Create nested pages<br>Build navigation menu | 3 | |
| 77 | Module 8 – React | Forms in React | Handling forms in React<br>Controlled components<br>Uncontrolled components<br>Form events<br>Input handling patterns | <br><br>Build login & signup forms<br>Handle inputs using state<br>Controlled vs uncontrolled demo | 3 | |
| 78 | Module 8 – React | Form Validation | Form validation strategies<br>Client-side validations<br>Error handling in forms<br>UX best practices | <br><br>Add validations to forms<br>Display error messages | 3 | |
| 79 | Module 8 – React | API Integration – Basics | What is API<br>Fetch API basics<br>Axios introduction<br>GET & POST requests<br>Handling API responses | <br><br>Fetch data from public API<br>Display API data in UI<br>Compare Fetch vs Axios | 3 | |
| 80 | Module 8 – React | API Integration – Advanced & Mini Project | Loading & error states<br>API data lifecycle<br>Best practices for API calls<br>Component-based API design | <br><br>Build Complete React project<br>Integrate API end-to-end | 3 | |
| 81 | Module 9 – DevOps | Git & Version Control Foundations | What is Version Control & why it matters<br>Git basics (init, clone, add, commit, push, pull)<br>Branching strategies (feature, release, hotfix)<br>Using GitHub/GitLab in real projects<br>Pull requests & code reviews | <br><br>Create Git repo<br>Implement branching workflow<br>PR creation & merge simulation | 3 | |
| 82 | Module 9 – DevOps | CI/CD Concepts & Pipelines | Introduction to CI/CD<br>CI vs CD<br>Pipeline stages<br>GitHub Actions / GitLab CI overview<br>Writing basic pipeline config | <br><br>Setup CI pipeline<br>Run automated build<br>Trigger pipeline on push | 3 | |
| 83 | Module 9 – DevOps | Automated Testing & Deployment | Running tests in pipelines<br>Build artifacts<br>Deployment strategies<br>Rollback basics<br>Environment variables & secrets | <br><br>Add test stage to pipeline<br>Deploy app via CI<br>Manage secrets securely | 3 | |
| 84 | Module 9 – DevOps | AWS Foundations for Developers | Setting up AWS account<br>EC2 concepts<br>Launching EC2 instances<br>Installing Node.js & Nginx<br>S3 static file hosting<br>IAM roles & permissions<br>Route 53 basics<br>SSL with ACM / Let's Encrypt | <br><br><br><br>Launch EC2 instance<br>Deploy app on EC2<br>Setup S3 static site<br>Configure domain & SSL | 3 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| 85 | Module 9 – DevOps | Docker & Containerized Deployment | What are containers & Docker<br>Docker architecture<br>Creating & running containers<br>Writing Dockerfiles<br>Managing images<br>Docker Compose<br>Deploying Dockerized apps to AWS | Containerize application<br>Create Dockerfile & Compose file<br>Deploy Docker app on EC2 | 3 | |
| 86 | Module 10-Spring Boot + Gen | Foundations of AI-Powered Applications | What are AI-powered applications<br>Evolution from rule-based to AI systems<br>Role of AI in modern Java & Spring Boot apps<br>Overview of ML, DL & GenAI (high level) | Analyze real AI products<br>Identify AI opportunities in existing Java apps<br>AI vs traditional API comparison | 3 | |
| 87 | Module 10-Spring Boot + Gen | AI Use Cases & Intelligent API Design | Common AI use cases (chatbots, search, fraud detection)<br>Designing intelligent APIs<br>AI request/response patterns<br>Latency & scalability considerations | Convert business use case into AI API<br>Design API contracts for AI services | 3 | |
| 88 | Module 10-Spring Boot + Gen | AI-Driven REST APIs with Spring Boot | Creating REST APIs using Spring Boot<br>Handling JSON payloads<br>AI request orchestration<br>Integrating APIs with frontend apps | Build AI-enabled REST API<br>Test API using Postman/Swagger<br>Frontend integration flow | 3 | |
| 89 | Module 10-Spring Boot + Gen | Working with Pretrained AI Models in Java | What are pretrained models<br>Model formats overview<br>Using AI models in Java apps<br>Performance & memory considerations | Load pretrained model in Spring Boot<br>Run sample inference workflows | 3 | |
| 90 | Module 10-Spring Boot + Gen | ONNX Models & Inference | Introduction to ONNX<br>ONNX Runtime for Java<br>Running inference on text data<br>Image & tabular data inference | Integrate ONNX model<br>Text classification demo<br>Image inference walkthrough | 3 | |
| 91 | Module 10-Spring Boot + Gen | AI Microservices Architecture | AI microservices patterns<br>Separating inference & orchestration layers<br>Scalability challenges in AI systems<br>Stateless vs stateful AI services | Design AI microservice architecture<br>Split monolithic AI API into services | 3 | |
| 92 | Module 10-Spring Boot + Gen | Distributed AI with Spring Cloud | Spring Cloud overview<br>Service discovery for AI services<br>Load balancing AI workloads<br>Fault tolerance & retries | Integrate Spring Cloud<br>Deploy multiple AI services<br>Handle failures gracefully | 3 | |
| 93 | Module 10-Spring Boot + Gen | Conversational AI & Chatbots | Conversational AI concepts<br>Chatbot architecture<br>Building chatbot backend using Spring Boot<br>Session & conversation management | Build chatbot backend API | 3 | |
| 94 | Module 10-Spring Boot + Gen | LLM Integration & Knowledge Search (RAG) | Integrating LLMs with Spring Boot<br>RAG architecture<br>Vector embeddings fundamentals<br>Vector storage (Pinecone, FAISS, Elasticsearch with Spring) | Generate embeddings<br>Store & retrieve vectors<br>Implement RAG flow | 3 | |
| 95 | Module 10-Spring Boot + Gen | Monitoring, Metrics & Production Readiness | Logging AI predictions<br>Model performance metrics<br>Drift detection concepts<br>Accuracy tracking over time<br>Production best practices | Add logging to AI APIs<br>Track prediction metrics<br>Mini end-to-end AI service review | 3 | |