Index

JAVA FULL STACK STRUCTURE (12 WEEKS)

JEE LoT provides exposure to the entire spectrum of Java technologies starting from Core Java to Spring as well as testing. It focuses on Web Application development using Spring Technology. The following table lists the course structure for JEE LoT.

S.No	Course	Duration (In Days)
1	Discover	2
2	Campus to Corporate	2
3	Core Java 8 + Database & SQL + JDBC	17
4	**Core Java 8 Test**	**1**
5	HTML 5, CSS 3 with bootstrap 4	2
6	Javascript, Typescript & Angular 5	5
7	Servlets 3.0	2
8	**HTML5, CSS + Angular 5 Test**	**0.5**
9	soft skills Part I	0.5
10	JPA with Hibernate	3
11	Spring 5.0	7
12	**Parallel Project Evaluation**	**1**
13	**Spring 4.0 + JPA with Hibernate 3.0 Test**	**0.5**
14	Soft Skills Part II	0.5
15	BDD	5
16	**BDD Test**	**0.5**
17	Soft Skills Part III	0.5
18	**Quality Process Awareness + PLP + PLP Presentation**	**8**
19	**L1 Preparation + Test**	**2**
	Total Training Duration	**60**

JEE Curriculum

Core Java 8 + JDBC

Program Duration: 15 days

Contents:

- **Declarations and Access Control**
  - Identifiers & JavaBeans
  - Legal Identifiers
  - Sun's Java Code Conventions
  - JavaBeans Standards
  - Declare Classes
  - Source File Declaration Rules
  - Class Declarations and Modifiers
  - Concrete Subclass
  - Declaring an Interface
  - Declaring Interface Constants
  - Declare Class Members
  - Access Modifiers
  - Nonaccess Member Modifiers
  - Constructor Declarations
  - Variable Declarations
  - Declaring Enums

- **Object Orientation**
  - Encapsulation
  - Inheritance, Is-A, Has-A
  - Polymorphism
  - Overridden Methods
  - Overloaded Methods
  - Reference Variable Casting
  - Implementing an Interface
  - Legal Return Types
  - Return Type Declarations
  - Returning a Value
  - Constructors and Instantiation
  - Default Constructor
  - Overloaded Constructors
  - Statics
  - Static Variables and Methods
  - Coupling and Cohesion

- **Assignments**
  - Stack and Heap—Quick Review
  - Literals, Assignments, and Variables
  - Literal Values for All Primitive Types
  - Assignment Operators
  - Casting Primitives
  - Using a Variable or Array Element That Is Uninitialized and Unassigned
  - Local (Stack, Automatic) Primitives and Objects
  - Passing Variables into Methods
  - Passing Object Reference Variables
  - Does Java Use Pass-By-Value Semantics?
  - Passing Primitive Variables
  - Array Declaration, Construction, and Initialization

- Declaring an Array
- Constructing an Array
- Initializing an Array
- Initialization Blocks
- Using Wrapper Classes and Boxing
- An Overview of the Wrapper Classes
- Creating Wrapper Objects
- Using Wrapper Conversion Utilities
- Autoboxing
- Overloading
- Garbage Collection
- Overview of Memory Management and Garbage Collection
- Overview of Java's Garbage Collector
- Writing Code That Explicitly Makes Objects Eligible for Garbage Collection

- **Operators**
  - Java Operators
  - Assignment Operators
  - Relational Operators
  - instanceof Comparison
  - Arithmetic Operators
  - Conditional Operator
  - Logical Operators

- **Flow Control, Exceptions**
  - if and switch Statements
  - if-else Branching
  - switch Statements
  - Loops and Iterators
  - Using while Loops
  - Using do Loops
  - Using for Loops
  - Using break and continue
  - Unlabeled Statements
  - Labeled Statements
  - Handling Exceptions
  - Catching an Exception Using try and catch
  - Using finally
  - Propagating Uncaught Exceptions
  - Defining Exceptions
  - Exception Hierarchy
  - Handling an Entire Class Hierarchy of Exceptions
  - Exception Matching
  - Exception Declaration and the Public Interface
  - Rethrowing the Same Exception
  - Common Exceptions and Errors

- **Maven Fundamentals**
  - Introduction
  - Folder Structure
  - The pom.xml
  - Dependencies
  - Goals
  - Scopes
  - The Compiler Plugin
  - Source Plugin
  - Jar Plugin

- **TDD with Junit 5**
  - Types of Tests
  - Why Unit Tests Are Important
  - What's JUnit?
  - JUnit 5 Architecture
  - IDEs and Build Tool Support
  - Setting up JUnit with Maven
  - Lifecycle Methods
  - Test Hierarchies
  - Assertions
  - Disabling Tests
  - Assumptions
  - Test Interfaces and Default Methods
  - Repeating Tests
  - Dynamic Tests
  - Parameterized Tests
  - Argument Sources
  - Argument Conversion
  - What Is TDD?
  - History of TDD
  - Why Practice TDD?
  - Types of Testing
  - Testing Frameworks and Tools
  - Testing Concepts
  - Insights from Testing
  - Mocking Concepts
  - Mockito Overview
  - Mockito Demo
  - Creating Mock Instances
  - Stubbing Method Calls

- **Strings, I/O, Formatting, and Parsing**
  - String, StringBuilder, and StringBuffer
  - The String Class
  - Important Facts About Strings and Memory
  - Important Methods in the String Class
  - The StringBuffer and StringBuilder Classes
  - Important Methods in the StringBuffer and StringBuilder Classes

- File Navigation and I/O
- Types of Streams
- The Byte-stream I/O hierarchy
- Character Stream Hierarchy
- RandomAccessFile class
- The java.io.Console Class
- Serialization
- Dates, Numbers, and Currency
- Working with Dates, Numbers, and Currencies
- Parsing, Tokenizing, and Formatting
- Locating Data via Pattern Matching
- Tokenizing

- **Generics and Collections**
  - Overriding hashCode() and equals()
  - Overriding equals()
  - Overriding hashCode()
  - Collections
  - So What Do You Do with a Collection?
  - List Interface
  - Set Interface
  - Map Interface
  - Queue Interface
  - Using the Collections Framework
  - ArrayList Basics
  - Autoboxing with Collections
  - Sorting Collections and Arrays
  - Navigating (Searching) TreeSets and TreeMaps
  - Other Navigation Methods
  - Backed Collections
  - Generic Types
  - Generics and Legacy Code
  - Mixing Generic and Non-generic Collections
  - Polymorphism and Generics

- **Threads**
  - Defining, Instantiating, and Starting Threads
  - Defining a Thread
  - Instantiating a Thread
  - Starting a Thread
  - Thread States and Transitions
  - Thread States
  - Preventing Thread Execution
  - Sleeping
  - Thread Priorities and yield( )
  - Synchronizing Code
  - Synchronization and Locks
  - Thread Deadlock
  - Thread Interaction
  - Using notifyAll( ) When Many Threads May Be Waiting

- **Concurrent Patterns in Java**
    - Introducing Executors, What Is Wrong with the Runnable Pattern?
    - Defining the Executor Pattern: A New Pattern to Launch Threads
    - Defining the Executor Service Pattern, a First Simple Example
    - Comparing the Runnable and the Executor Service Patterns
    - Understanding the Waiting Queue of the Executor Service
    - Wrapping-up the Executor Service Pattern
    - From Runnable to Callable: What Is Wrong with Runnables?
    - Defining a New Model for Tasks That Return Objects
    - Introducing the Callable Interface to Model Tasks
    - Introducing the Future Object to Transmit Objects Between Threads
    - Wrapping-up Callables and Futures, Handling Exceptions

- **Concurrent Collections**
    - Implementing Concurrency at the API Level
    - Hierarchy of Collection and Map, Concurrent Interfaces
    - What Does It Mean for an Interface to Be Concurrent?
    - Why You Should Avoid Vectors and Stacks
    - Understanding Copy On Write Arrays
    - Introducing Queue and Deque, and Their Implementations
    - Understanding How Queue Works in a Concurrent Environment
    - Adding Elements to a Queue That Is Full: How Can It Fail?
    - Understanding Error Handling in Queue and Deque
    - Introducing Concurrent Maps and Their Implementations
    - Atomic Operations Defined by the ConcurrentMap Interface
    - Understanding Concurrency for a HashMap
    - Understanding the Structure of the ConcurrentHashMap from Java 7
    - Introducing the Java 8 ConcurrentHashMap and Its Parallel Methods
    - Parallel Search on a Java 8 ConcurrentHashMap
    - Parallel Map / Reduce on a Java 8 ConcurrentHashMap
    - Parallel ForEach on a Java 8 ConcurrentHashMap
    - Creating a Concurrent Set on a Java 8 ConcurrentHashMap
    - Introducing Skip Lists to Implement ConcurrentMap
    - Understanding How Linked Lists Can Be Improved by Skip Lists
    - How to Make a Skip List Concurrent Without Synchronization

Database & SQL

Program Duration: 2 days

Contents:

- DROP TABLE

JDBC

- **Introduction to JDBC**
  - Introduction to JDBC
  - Architecture of JDBC
  - Role of Driver Manager
  - Understanding JDBC Driver Types

- **Getting Started with JDBC**
  - Connecting to Database using JDBC
  - Managing Database resources
  - Handling JDBC Exceptions

- **Performing Basic CRUD operations using JDBC**
  - Executing Static SQL Statements
  - Iterating Through ResultSets
  - Understanding Scrollable ResultSets
  - Understanding Updatable ResultSets
  - Understanding PreparedStatement
  - Retrieving Data Using PreparedStatement
  - Inserting the Record
  - Updating the Record
  - Removing the Record

HTML 5, CSS 3 with Bootstrap

Program Duration: 2 days

Contents:

HTML 5:

- HTML Basics
  - Understand the structure of an HTML page.
  - New Semantic Elements in HTML 5
  - Learn to apply physical/logical character effects.
  - Learn to manage document spacing.
- Tables
  - Understand the structure of an HTML table.

    - Learn to control table format like cell spanning, cell spacing, border
- List
    - Numbered List
    - Bulleted List
- Working with Links
    - Understand the working of hyperlinks in web pages.
    - Learn to create hyperlinks in web pages.
    - Add hyperlinks to list items and table contents.
- Image Handling
    - Understand the role of images in web pages
    - Learn to add images to web pages
    - Learn to use images as hyperlinks
- Frames
    - Understand the need for frames in web pages.
    - Learn to create and work with frames.
- HTML Forms for User Input
    - Understand the role of forms in web pages
    - Understand various HTML elements used in forms.
    - Single line text field
    - Text area
    - Check box
    - Radio buttons
    - Password fields
    - Pull-down menus
    - File selector dialog box
- New Form Elements
    - Understand the new HTML form elements such as date, number, range, email, search and datalist
    - Understand audio, video, article tags

CSS 3

- Introduction to Cascading Style Sheets 3.0
    - What CSS can do
    - CSS Syntax
    - Types of CSS
- Working with Text and Fonts
    - Text Formatting
    - Text Effects
    - Fonts
- CSS Selectors
    - Type Selector
    - Universal Selector
    - ID Selector

    - Class selector
- Colors and Borders
    - Background
    - Multiple Background
    - Colors RGB and RGBA
    - HSL and HSLA
    - Borders
    - Rounded Corners
    - Applying Shadows in border

BootStrap

- Introduction to Bootstrap
    - Introduction
    - Getting Started with Bootstrap
- Bootstrap Basics
    - Bootstrap grid system
    - Bootstrap Basic Components
- Bootstrap Components
    - Page Header
    - Breadcrumb
    - Button Groups
    - Dropdown
    - Nav & Navbars

JavaScript, Typescript & Angular 5

Program Duration: 5 days

Contents:

Javascript

- JavaScript Language
    - Data Types and Variables
    - JavaScript Operators
    - Control Structures and Loops
    - JavaScript Functions
- Working with Predefined Core Objects
    - Data Types in JavaScript
    - String Objects
    - URL String Encoding and Decoding
    - Math Properties
    - Math Objects
    - Date Objects
    - Date and Time Arithmetic

- Working with arrays
    - Arrays object, its properties and methods
- Document Object Model
    - Understand the JavaScript Object Model
    - Understand the Window object
- Working With Document Object
    - Document Object and its properties, methods and events
- Working with Form Object
    - Form Object Properties, Methods & Event Handlers
    - Text-Related Objects
    - Button Objects
    - Check Box and Radio Objects
    - Select Objects
    - Validate Data and Form Submission

- Introduction to Angular 5
    - What is Angular 5?
    - Why Angular 5?
    - What is nodeJS?
    - Scope and Goal of Angular 5
    - Installing and using Angular 5
    - Building Blocks of an Angular 4 Application
    - Difference between Angular2 & 5
    - A Basic Angular 4 Application
    - Working with Angular 5 with Eclipse

- Components
    - What is a component?
    - Developing a simple component.
    - Templates for a component.
    - Component lifecycle

- Data Binding
    - What is data binding
    - One way data binding
    - Two way data binding
    - Nested component

- Event binding

- Directives
  - What are directives?
  - Types of directives - component, structural and attribute
  - Creating a basic directive
  - Handling event & Binding input in attribute directive
  - Creating your own structural directive
  - Using the structural directive
  - Binding input to a structural directive

- Working with Forms
  - Forms in Angular 5
  - Template Driven Form
    - Create the component that controls the form
    - Create a template with the initial form layout
    - Bind data properties to each form input control with the ngModel two-way data binding syntax
    - Add the name attribute to each form input control
    - Add custom CSS to provide visual feedback
    - Show and hide validation error messages
    - Handle form submission with ngSubmit
    - Disable the form's submit button until the form is valid
    - Resetting the form
  - Model Driven Forms
    - Reactive Forms Introduction
    - More Form Controls
    - Form Control Properties
    - setValue and patchValue
    - Validating Form Elements
  - A Basic Angular Form
  - Binding Input Fields
  - Displaying Form Validation State & Field Validation State
  - Displaying Validation State Using Classes
  - Disabling Submit when Form is Invalid

- Service and Dependency Injection
  - What is a service?

    - Injecting a service to a component
    - Application wide dependency injection
    - @Injectable classes
    - Multiple service instances
    - @Optional and @Host

- HTTP Client
    - The new HTTP providers
    - Injecting the providers
    - GET call
    - Handling error
    - About Observables
    - POST request
    - Working with headers
    - Sequential calls & parallel calls

- Pipe
    - What is a pipe?
    - Passing parameters to a pipe
    - Chaining pipes
    - Developing a custom pipe

- Routing
    - Why use routing?
    - Defining a route table
    - New Routing architecture Angular 5
    - Navigation using hyperlink & code
    - Supplying parameters to a route URL
- Animation using Angular 5

Servlets 3.0

Program Duration: 2 days.

Contents:

- Java Web Applications
    - Web Applications – An Overview
    - Web Components
    - JEE Containers

- Introduction to Servlets API and Ease of Development through Annotations
    - Introduction to Servlet
    - Role of Servlets in Web Application Design
    - Advantages of Servlets
    - Basic Servlet Architecture: Servlet Container
    - Servlet Lifecycle
    - Ease of Developing Servlets through Annotations
    - Retrieving Information from HTML Page
- Request Object
    - Processing Get and Post Requests from Web Clients
    - Request Dispatcher
- Listener
- Session Tracking
    - Introduction and Need for Session Tracking
    - Different Techniques for Session Management
    - Examples and Best Practices

JPA with Hibernate 3.0

Program Duration: 3 days

Contents:

- Introduction to ORM and its need
- The Persistence Life Cycle
- Java persistence API (JPA)
- JPQL
- Association and Mapping

Spring 5.0

Program Duration: 7 days

Contents:

- Introduction to Spring Platform and environment and Spring Boot Features

- Introduction to Spring Framework, IoC
    - What is Spring Framework,Benefits of Spring
    - The Spring architecture
    - IOC – Inversion of control, wiring beans
    - Bean containers, lifecycle of beans in containers
    - Customizing beans with BeanPostProcessors & BeanFactoryPostProcessors
    - XML and Annotation-based, mixed configurations
    - Java configuration
- Spring MVC framework
    - Introduction: DispatcherServlet, Handler mappings, Resolving views
    - Annotation-based controller configuration
- Spring JPA Integration
    - Spring support for JPA
    - Implementing Spring JPA integration
    - Spring Data
    - Spring Boot(Annotation based and Java configuration)
    - Spring ReST
    - Spring DATA ReST

Parallel Project Evaluation

Program Duration: 1 day

Contents:

BDD

Program Duration: 5 days

Contents:

- Introduction to BDD; TDD vs BDD; How BDD supports Agile
- Gherkins Language- Basics
- Gherkins Language- Feature, scenarios, scenario outlines; Data driven scenario outlines, Backgrounds

- Cucumber - Step definition binding; Running and debugging scenarios
- Cucumber - Creating a step definition with data table
- Automated UI vs Manual UI testing; Introduction to selenium
- Selenium - Webdriver; Creating first test HTML element selection; working with forms
- Selenium - Business readbale UI automation with cucumber Adding a cucumber scenario
- Selenium - Introducing Page Object Models

- Testing a REST api with cucumber and gherkins (Case study)

Quality Process Awareness

Program Duration: 0.5 day.

Contents:

- Understand the following :
  - Quality – What and Why
  - Introduction to Quality Management System
  - QMS support to Software Methodology
  - Metrics
  - Defect Prevention

Pseudo Live Project (PLP)

Program Duration: 7.5 days.

Contents:

Pseudo Live Project (PLP) program is primarily to handhold participants who are fresh into the IT stream & newly recruited from college. PLP project is

executed to orient the trainees towards Quality processes followed in the organization. In PLP, more importance given to “Process Adherence” . Maven, Jenkins, GIT, Sonar are the Devops concepts to be introduced during PLP.

- Agile methodology should be carried out during PLP