Ref. No. IKGPTU/CRA/..89..... Dated. June/19/2024

Director/Principals
All University Campuses and it's Affiliated Colleges
I K Gujral Punjab Technical University

Sub: ANUDIP FOUNDATION - ACCENTURE CSR PROGRAM with IKG PTU.

Dear Sir/Madam

ANUDIP FOUNDATION - ACCENTURE CSR PROGRAM is launched with IKG PTU for the technical training and placement support of the students. Anudip foundation is a platform, where ACCENTURE provides the technical & non technical training to the final year technical and non technical graduates. Anudip Foundation, a reputable organization committed to empowering youth through skill development and employment opportunities, has established a remarkable decade-long 10 years partnership with ACCENTURE.

This program can significantly enhance the professional development and employability of students. With access to the expertise and resources of both Anudip Foundation and ACCENTURE, students will be equipped with the skills and knowledge necessary to excel in today's competitive job market. Trainers are highly experienced professionals, who are Accenture certified trainers, will deliver the training sessions. These trainers have undergone rigorous certification processes by ACCENTURE ensuring the highest quality of training delivery. The training program will cover the latest industry trends and best practices, providing students with a competitive edge in their career pursuits.

Offered courses along with the curriculum are attached for information. **Students of B.Tech (All streams), BCA & MCA passing out in 2024 are eligible for this program.**

Under this program, placement support is providing in more than 100 companies such as Capgemini, Accenture, TCS, Amazon, City Bank, Netyod, Teleperformance, HCL, Bank of America, iMerit, Concentrix etc. Offering package from 3.50 to 8.00 LPA.

Interested students may register at the below mentioned link:-

https://docs.google.com/forms/d/e/1FAIpQLSf_pvWz-yGXlekzUMKDdhYJq-OuulLuDOORjLsgvt_w6ZhgLg/viewform

ਆਈ. ਕੇ. ਗੁਜਰਾਲ ਪੰਜਾਬ ਟੈਕਨੀਕਲ ਯੂਨੀਵਰਸਿਟੀ ਜਲੰਧਰ, ਕਪੂਰਥਲਾ
I.K. GUJRAL PUNJAB TECHNICAL UNIVERSITY JALANDHAR, KAPURTHALA
Office of Corporate Relations & Alumni

Ref. No. IKGPTU/CRA/.89..... Dated. June/19/Jun24

You are requested to kindly direct the Training & Placement Officer/ Faculty Coordinator of your college/campus to share the information with the concerned students. Last date of online registration is June 25, 2024 before 1700 hrs.

With profound regards,

19/VI/24.

Er. Navdeepak Sandhu
Deputy Director (T&P)

CC:

- **Hon'ble Vice Chancellor, IKG PTU,** for his kind information.
- **Head, CT&P, IKG PTU,** for his kind information.

Advance Java Programming
(100 hours)

Index

- **Core Java - crash course**
- **Collections**
- **MySQL, JDBC**
- **JPA with Hibernate 3.0**
- **SpringBoot**
- **Project**

Company Confidential

Advanced Programming in Java course provides exposure to core and advanced technologies in Java. The following table lists the course structure.

Sr. No.	Course	Duration (hrs)	Remarks
1	Core Java - Crash course on - OOP concepts, flow control, exception handling Collections, Threads	20	
2	DBMS - MySQL, JDBC	20	
3	JPA+Hibernate	10	
4	SpringBoot	36	
5	Project	14	JPA+Hibernate Or SpringBoot based project
	Total	**100**	

Curriculum

Core Java

Introduction and Pretest **Videos on IT - self study**
OOP concepts - intro only **Embed: Procedural vs OOPs - video** **Object, Encapsulation, Abstraction, Polymorphism** **Class & class members** **Operators** **Unary Op, Arithmetic Op** **Shift Op, Relational Op** **Bitwise Op, Logical Op** **Ternary Op, Assignment Op**
Flow Control If Else Loop Switch case Break Continue Inheritance - explanation with example
Polymorphism - Overloading, Overriding with example Abstraction - Abstract classes Interfaces
Exception Checked Exception UnChecked Exception Try Catch Finally Throw Throws
Collections Generics List

Stack Queue Set Interface introduction Implementation classes (HastSet)
Set interface implementation classes - LinkedHashSet, TreeSet Map interface introduction + implementation classes - HashMap, LinkedHashMap
TreeMap Comparator Comparable Override hashCode() & equals() method
Thread Synchronization Inter thread communication JVM

DBMS

Introduction to MySQL & SQL Creating a MySql database & Table Basic SQL Queries Primary Key
MySQL More advanced SQL queries Foreign key Joining Tables Filtering Data
Introduction to JDBC Connecting to MySql from Java Executing SQL Queries from Java
Executing insert,update and delete queries from MySql Handling Errors in MySql
Review of Topics covered Q&A Discussion MCQ Interaction with Dean
Introduction to PlSql Stored Procedures Triggers MySql Administration

Advanced JDBC Topics Prepared Statement Callable Using JDBC with other databases
Performance Tunning Security
Interview Q&A Analyze the performance of a SQL Queries
Interview Q&A Research and learn about Security

JPA-Hibernate

Introduction to JPA Introduction to ORM Tool Introduction to Hibernate and it's state Hibernate Architecture Hibernate Using Annotation Difference between JPA and Hibernate
JPA Annotations Hibernate configuration Introduction to Sessionfactory and session Transaction Management fetching data using get() & load() method
@Embeddable annotation Hibernate Mapping OneToOne,OneToMany,ManyToMany Annotation in Bi directional way and uniDirectional way
Introduction to HQL queries Executing some complex queries using HQL named Queries Inheritance in hibernate
Hibernate caching Second level cache Q&A Discussion MCQ

SpringBoot

SpringBoot Spring framework walkthrough Dependency Injection IOC contrainer Setter and Constructor Injection Introduction to Spting Tool Suite
Spring vs SB Dependencies Bean scope Bean lifecycle injecting Bean Auto wiring beans using Annotation
Introduction to Spring data JPA Advantages of Spring data japa(No more DAO implementation) Create Repositories Custom Access method and Queries Automatic Custom Queries using custom finder method Manual query using @Query annotation Native Query & Named Query
Introduc to SpringBoot What is new in Spring Boot 3? How many way we can create spring boot projects like using spring initializar,using Spring Tool suite
Explain Layered Architecture Why spring boot over spring framework Creating Spring boot application using Spring initializar Creating Spring boot application from Spring Tool Suite
Learn to build Spring boot REST ApI What is Rest API Important HTTP methods(Get,Post,Put,Delete)
Create simple operation using REST API HTTP Standard Status code Explain DTO concept
Introduction to Lombok and show usage in project Global Exception handler All important annotations
Start Building CRUD application using REST API and collection Framework Test Spring Boot application usning Postman client Integrate the Swagger API and implement it in projects
Start Building Application using Real time scenario Let's first create a Student java bean class that the REST API wants to return to the client: Let's create the StudentController class using @RestController Annotation Implement GetMapping() and PostMapping() annotation over here to fetch and Inter the data

Implement PutMapping() to update the students details Build Spring Boot REST API returns List Create Spring Boot REST API with Path Variable
Build Spring Boot REST API with Request Param Develop Spring Boot CRUD REST APIs with Spring Boot 3, Spring Data JPA (Hibernate), and MySQL Database Introduction of application.properties file Create ERD for databse design Configure MySQL Database Create Spring Data JPA Repository
Create Service Layer Create ServiceImpl class Explain RespontEntity Create Controller class to perform CRUD operation
Test the project in postman client Implement some real time scenarion like login Manage Student Manage Attendance Manage Faculty Report ETC
Logger implementation Understanding on Different types of Exceptions and Errors occure at runtime Mysql and backend validations Jwt Token for Authentication Assign projects to students individually or in a group

Discuss Coding Guidelines Importance of DevOps and it's usag Github account creation and add collaborators Start buiding Project with proper coding guidelines
Introduction to Junit 5 Introduction to Mockito Discuss @MockBean,@Mock annotations and when() methods to test service layers Discuss @DataJpaTest,@AutoConfigureTestDatabase,@TestMethodOrder annotations @Rollback,@SpringBootTest annotations to test each layers Discuss assertj library and it's method assertThat()
Implements positive and negative test cases for every methods into controller,service and repository layers Project At the end of projects creation,start analyzing your projects with sonarqube tool

Project

SpringBoot based

Or
JPA-Hibernate based

Data Analytics (No + Low Code)(DANLC)

Index

Course Overview: Data Analytics (No + Low Code) (DANLC)

The Data Analytics (No + Low Code) course is designed to equip participants with the skills and knowledge necessary to conduct effective data analysis using tools that require minimal or no coding. In a rapidly evolving technological landscape, the course focuses on empowering individuals with the ability to leverage no-code and low-code platforms for robust data analytics.

Module	**Total Duration (Hours)**	**Self Learning (Hours)**	**ILT (Hours)**
No code data visualization	30	14	16
Python Module	120	20	100
SQL Module	24	4	20
Communicative Eng	60	20	40
Business Communication	62	32	30
Project with Mentoring by Industry Professionals	24	0	24
Total Hours	**320**	**90**	**230**

Curriculum

Excel Table of Contents

1. **VLOOKUP and HLOOKUP Mastery**
   - Overview of VLOOKUP and HLOOKUP
   - Nested VLOOKUP and HLOOKUP
   - Using INDEX and MATCH as alternatives
   - Advanced lookup scenarios
2. **IF Statements and Logical Functions**
   - Advanced IF statements
   - Nested IF functions
   - Logical functions (AND, OR, NOT)
3. **Array Formulas**
   - SUMPRODUCT and SUMIFS ,SUMIFS,COUNTIF,COUNTIFS
4. **Dynamic Named Ranges**
   - Creating dynamic named ranges
   - Using named ranges in formulas
   - Dynamic range in charts and data validation
5. **PivotTables and PivotCharts**
   - Introduction to PivotTables
   - Creating PivotTables and PivotCharts
   - Advanced PivotTable techniques
   - Slicers and Timelines
6. **Power Query Basics**
   - Introduction to Power Query
   - Importing and transforming data
   - Merging and appending queries
   - Basics of data modeling
7. **Data Validation and Validation Rules**
   - Setting up data validation
   - Custom validation rules
   - Using data validation for dynamic dropdowns

8. Data Consolidation

- Consolidating data from multiple sheets
- Consolidating data from different workbooks

9. Exporting and Importing Data

- Importing data from external sources (SQL, Web, etc.)

- Exporting data to different formats (CSV, PDF, etc.)
- Refreshing external data connections

Power BI Table of Contents

Python Table of Contents

- Definite Loops Using for
- Loop Patterns

7. **Working with Strings**
   - Defining Strings
   - Accessing a String
   - Special String Operators
   - Traversing a String
   - Built-in String Methods
   - String Comparison
   - Format Operator
8. **Working with Lists and Dictionaries**
   - Lists: Traversing, Operations, Slices, Methods, Functions
   - Dictionaries: Characteristics, Creating, Methods, The in Operator
   - List vs. Dictionary
9. **Understanding Tuples and Sets**
   - Tuples: Features, Creating, Operations, Functions
   - Sets: Features, Creating, Operations
10. **Data Persistence**
    - Opening Files
    - Reading Files
    - Searching Data in a File
    - Writing to a File
11. **Introduction to Object-Oriented Programming (OOP) in Python**
    - Inheritance
    - Handling Exceptions
12. **Introduction to NumPy**
    - NumPy Data Types
    - Creating Arrays
    - Array Indexing
    - Array Slicing
    - Array Copy vs. View
    - Array Shape and Reshaping
13. **NumPy Functions and Operations**
    - Important NumPy Functions
    - Searching, Splitting, Sorting, Joining, and Filtering
    - Random Functions, Where Function, Transpose, Mean
    - Statistical Functions and NumPy IO
14. **Mathematical Calculations and Logical Operations in NumPy**
    - Mathematical Calculations
    - Logical Operations
15. **Review and Interaction**

- Review of Topics Covered
- Q&A Discussion
- Multiple Choice Questions (MCQ)
- Interaction with Dean

16. **Data Visualization with Python**
    - Introduction to Data Visualization
    - Libraries for Data Visualization (Seaborn, Matplotlib)
    - Line Plot, Bar Plot, Histogram
    - Pie Plot, Scatter Plot, Subplot, Formatting Plots

17. **Working with Pandas**
    - Introduction to Pandas Library
    - Pandas vs. NumPy
    - Series in Pandas
    - DataFrames: Creating, Updating, Deleting, Inserting Values
18. **Reading and Cleaning Data with Pandas**
    - Reading CSV Files using Pandas
    - Saving Data to CSV Files
    - Cleaning Data
    - Handling Missing Values (dropna(), fillna())
    - Removing Duplicate Values
    - Objectives and Recap
19. **Pandas Data Analysis**
    - Exploratory Data Analysis with Pandas
    - Objectives and Recap
20. **Pandas Pivot Table**
    - Understanding Pivot Tables in Pandas
    - Objectives and Recap
21. **Review and Interaction**
    - Review of Topics Covered
    - Q&A Discussion
    - Multiple Choice Questions (MCQ)
    - Interaction with Dean
22. **Introduction to SciPy**
    - What is SciPy
    - SciPy Installation
    - Objectives and Recap
23. **SciPy Sub-Packages**
    - SciPy Cluster
    - SciPy Constant
    - Objectives and Recap

24. **Advanced SciPy Concepts**
    - Fast Fourier Transform
    - SciPy Interpolation
    - SciPy Input and Output
    - Objectives and Recap
25. **Linear Algebra and Ndimage in SciPy**
    - Linear Algebra in SciPy
    - Ndimage in SciPy
    - Objectives and Recap
26. **Sparse Matrix and Stats in SciPy**
    - Sparse Matrix in SciPy
    - Stats in SciPy
    - Objectives and Recap
27. **Review and Interaction**
    - Review of Topics Covered
    - Q&A Discussion
    - Multiple Choice Questions (MCQ)
28. **Project Work**
    - Project Kickoff and Planning
    - Setting Up Project Environment
    - Defining Project Structure
    - Data Exploration and Preparation
    - Implementation of Core Features
    - Integration and Testing
    - Finalizing Project and Documentation
29. **Final Project Review and Presentation**
    - Reviewing Project Code
    - Writing Comprehensive Documentation
    - Preparing a Project Presentation
30. **Conclusion and Sprint 1 Project**
    - Review of the Entire Course
    - Q&A Session
    - Interaction with Dean

SQL Table of Contents

1. **Introduction to RDBMS**
    - Overview of Relational Database Management System (RDBMS)
    - The Relational Model
    - Principles of Database Design

- Introduction to MySQL
- MySQL Data Types
- Creating Databases Using MySQL

2. **Basic MySQL Syntax**
   - Understanding Basic MySQL Syntax
   - Basic SQL Commands - SELECT
   - Basic SQL Commands - INSERT
   - Basic SQL Commands - UPDATE
   - Basic SQL Commands - DELETE
3. **Querying Data with SELECT Statement**
   - The SELECT List
   - SELECT List Wildcard (*)
   - The FROM Clause
   - How to Constrain the Result Set
   - DISTINCT and NOT DISTINCT
   - Filtering Results with the WHERE Clause
4. **Advanced Querying Techniques**
   - Other Boolean Operators (BETWEEN, LIKE, IN, IS, IS NOT)
   - Shaping Results with ORDER BY and GROUP BY
   - Set Functions
   - Boolean Operators (AND, OR)
   - Aggregate Functions
   - Set Function and Qualifiers
   - GROUP BY
   - Qualifiers
   - HAVING Clause
5. **Modifying Data in MySQL**
   - ALTER TABLE
   - MySQL Transactions
6. **Working with JOINs**
   - Matching Different Data Tables with JOINs
   - Table Aliases
   - INNER JOIN
   - OUTER JOINs
   - LEFT OUTER JOIN
   - RIGHT OUTER JOIN
   - FULL OUTER JOIN
   - SELF JOIN
   - Natural Join
   - CROSS JOIN

31. **Conclusion and Sprint 2 Project**

- Review of the Entire Schedule
- Q&A Session
- Interaction with Participants

Java Full Stack with Angular

Index

Java Full Stack with Angular program provides exposure to the entire spectrum of Java technologies starting from Core Java to Spring. It focuses on Web Application development using Angular and Spring Technology. The following table lists the course structure.

Sr. No.	Course	Duration (hrs)	Remarks
1	Core Java 8 + Database & MySQL with DevOps (Git, Sonarqube, Gradle, Jenkin) JPA with Hibernate 3.0	180	
2	Soft Skills	60	
1	Spring 5.0 , Spring Boot (Core + MVC + REST + Data JPA + Data REST)	40	
	Sprint 1 + JPA and Spring MCQ Test	30	Sprint 1 - Backend implementation using Spring REST and Spring Data JPA
2	HTML 5, CSS 3 with bootstrap 4, Javascript, Typescript Angular 15	40	
	Sprint 2 + HTML 5, CSS 3 with bootstrap 4, Javascript, Typescript & Angular 7 MCQ	30	Sprint 2 - Front end implementation using Angular
	Total Training Duration	**380**	4 hrs a day, 5 days a week Total: 19 weeks

Curriculum

Core Java 8

Contents:

- **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

- **Gradle Fundamentals**
  - Introduction
  - Folder Structure
  - Install and Setup Gradle on Windows
  - Dependencies in Build Scripts
  - Gradle Wrapper
  - Lifecycle Tasks: The Base Plug In
  - Using Project Info and the check command
  - Creating Variables and external properties
  - Creating a Build Scan
  - Dependencies

- **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

- **Lambda Expressions**
  - Introduction
  - Writing Lambda Expressions
  - Functional Interfaces
  - Types of Functional Interfaces
  - Method reference
- **Stream API**
  - Introduction
  - Stream API with Collections
  - Stream Operations

Introduction to Design Pattern

Self learning with online links and explanation by Trainer with Demos

- Creational Design Pattern
  - Factory Pattern
  - Singleton Pattern
  - Prototype Pattern
- Structural Design Pattern
  - Decorator Pattern
  - Facade Pattern
- Behavioral Design Pattern

        - Chain of Responsibility Pattern
        - Iterator Pattern
    - Presentation Layer Design Pattern
        - Intercepting Filter Pattern
        - Front Controller Pattern
    - Business Layer Design Pattern
        - Business Delegate Pattern
        - Transfer Object Pattern
    - Integration Layer Design Pattern
        - Data Access Object Pattern

• DevOps (Git, SonarQube, Maven, Jenkins)

• **Introduction to DevOps**

- Introduction of DevOps
- Dev And Ops
- Agile Vs DevOps
- Continuous Integration & Delivery pipeline
- Tools For DevOps
- Use-case walkthrough

• GIT Hub

- Working locally with GIT
- Working remotely with GIT
- Branching, merging & rebasing with GIT
- Use Case walkthrough

• Jenkins:

- Introduction to Jenkins
- Jenkins Objective
- Introduction to continuous integration deployment & Jenkins-ci

- Continuous Deployment & distribution builds with Jenkins • Sonar

- Introduction to Sonar
- Code quality Monitoring- Sonar
- Use Case walkthrough

Database Using MySQL

Contents:

- LEFT OUTER JOIN
- RIGHT OUTER JOIN
- FULL OUTER JOIN
- SELF JOIN
- Natural Join

- **Creating Database Tables**
  - CREATE DATABASE
  - CREATE TABLE
  - NULL Values
  - PRIMARY KEY
  - CONSTRAINT
  - ALTER TABLE
  - DROP TABLE
- **MySQL Transactions**
  - BEGIN, COMMIT, ROLLBACK
- **MySQL Constraints**
  - CHECK, UNIQUE, NOT NULL

- **Introduction to JDBC**
  - Connection, Statement, PreparedStatement, ResultSet

JPA with Hibernate 3.0

Contents:

- **Introduction**
  - Introduction & overview of data persistence
  - Overview of ORM tools
  - Understanding JPA
  - JPA Specifications
- **Entities**
  - Requirements for Entity Classes
  - Persistent Fields and Properties in Entity Classes
  - Persistent Fields
  - Persistent Properties
  - Using Collections in Entity Fields and Properties
  - Validating Persistent Fields and Properties
  - Primary Keys in Entities
- **Managing Entities**

- The EntityManager Interface
- Container-Managed Entity Managers
- Application-Managed Entity Managers
- Finding Entities Using the EntityManager
- Managing an Entity Instance's Lifecycle
- Persisting Entity Instances
- Removing Entity Instances
- Synchronizing Entity Data to the Database
- Persistence Units

▪ **Querying Entities**

- Java Persistence query language (JPQL)
- Criteria API

▪ **Entity Relationships**

- Direction in Entity Relationships
- Bidirectional Relationships
- Unidirectional Relationships
- Queries and Relationship Direction
- Cascade Operations and Relationships

Spring 5.0

Contents:

1. Spring Core

Spring Core Introduction / Overview

- Shortcomings of Java EE and the Need for Loose Coupling
- Managing Beans, The Spring Container, Inversion of Control
- The Factory Pattern
- Configuration Metadata - XML, @Component, Auto-Detecting Beans - Dependencies and Dependency Injection (DI) with the BeanFactory - Setter Injection

Spring Container

- The Spring Managed Bean Lifecycle
- Autowiring Dependencies

Dependency Injection

- Using the Application Context
- Constructor Injection
- Factory Methods
- Crucial Namespaces ‘p’ and ’c’
- Configuring Collections

Metadata / Configuration

- Annotation Configuration @Autowired, @Required, @Resource
- @Component, Component Scans. Component Filters
- Life Cycle Annotations
- Java Configuration, @Configuration, XML free configuration

- The Annotation Config Application Context

2. Spring Boot

SPRING BOOT Introduction

- Spring Boot starters, CLI, Gradle plugin
- Application class
- @SpringBootApplication
- Dependency injection, component scans, Configuration
- Externalize your configuration using application.properties
- Context Root and Management ports
- Logging

Using Spring Boot

- Build Systems, Structuring Your Code, Configuration, Spring Beans and Dependency Injection, and more.

Spring Boot Essentials

- Application Development, Configuration, Embedded Servers, Data Access, and many more
- Common application properties
- Auto-configuration classes
- Spring Boot Dependencies

3. Spring Data JPA

- Spring Data JPA Intro & Overview
- Core Concepts, @RepositoryRestResource
- Defining Query methods
- Query Creation
- Using JPA Named Queries
- Defining Repository Interfaces
- Creating Repository instances
- JPA Repositories
- Persisting Entities
- Transactions

4. Spring Data REST

- Introduction & Overview
- Adding Spring Data REST to a Spring Boot Project
- Configuring Spring Data REST
- Repository resources, Default Status Codes, Http methods
- Spring Data REST Associations
- Define Query methods

5. Introduction to Spring Security with Demo
6. Introduction to Spring Microservices with Demo

HTML 5, CSS 3 with Bootstrap, Javascript, TypeScript

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

- Implementing CSS3 in the "Real World" ○
  Modernizr

- HTML5 Shims
- SASS, and Other CSS Preprocessors
- CSS Grid Systems
- CSS Frameworks

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 Essentials**
- **ES6 & Typescript**
    - Var, Let and Const keyword
    - Arrow functions, default arguments
    - Template Strings, String methods
    - Object de-structuring
    - Spread and Rest operator
    - Typescript Fundamentals
    - Types & type assertions, Creating custom object types, function types - Typescript OOPS - Classes, Interfaces, Constructor, etc

Angular 15

Contents:

- **Introduction to Angular Framework**
    - Introduction to Angular Framework, History & Overview
    - Environment Setup, Angular CLI, Installing Angular CLI
    - NPM commands & package.json
    - Bootstrapping Angular App, Components, AppModule
    - Project Setup, Editor Environments
    - First Angular App & Directory Structure
    - Angular Fundamentals, Building Blocks
    - MetaData
- **Essentials of Angular**
    - Component Basics
    - Setting up the templates
    - Creating Components using CLI
    - Nesting Components
    - Data Binding - Property & Event Binding, String Interpolation, Style binding - Two-way data binding
    - Input Properties, Output Properties, Passing Event Data
- **Templates, Styles & Directives**
    - Template, Styles, View Encapsulation, adding bootstrap to angular app - Built-in Directives, Creating Attribute Directive
    - Using Renderer to build attribute directive
    - Host Listener to listen to Host Events

- Using Host Binding to bind to Host Properties

▪ **Pipes, Services & Dependency Injection**

- In-built Pipes, Creating a Custom Pipes
- Services & Dependency Injections
- Creating Data Service
- Understanding Hierarchical Injector

▪ **Template-Driven and Reactive Forms**

- Template-Driven vs Reactive Approach
- Understanding Form State
- Built-in Validators & Using HTML5 Validation
- Grouping Form Controls
- FormGroup, FormControl, FormBuilder
- Forms with Reactive Approach
- Predefined Validators & Custom Validators
- Showing validation errors

▪ **Components Deep Dive / Routing**

- Component Life Cycle Hooks
- Reusable components in angular using <ng-content>
- Navigating with Router links
- Understanding Navigation Paths
- Navigating Programmatically
- Passing Parameters to Routes
- Passing Query Parameters and Fragments
- Setting up Child (Nested) Routes
- Outsourcing Route Configuration (create custom module)

▪ **Http Requests / Observables**

- HTTP Requests
- Sending GET Requests
- Sending a PUT Request
- Using the Returned Data
- Catching Http Errors
- Basics of Observables & Promises

· **Authentication and Route Protection**

- How Authentication works in SPA
- JWT Module
- JSON Web Tokens
- Signup, Login and logout application
- Router Protection, Route Guards

- CanActivate interface
- Checking and using Authentication Status
- Router Protection and Redirection

Fundamentals of Java and Web Programming
(17 weeks)

Index

Java Fundamentals course provides exposure to Java and Web technologies. The following table lists the course structure.

Sr. No.	Course	Duration (hrs)	Remarks
1	Core Java 8 + Database & MySQL with DevOps (Git, Sonarqube, Gradle, Jenkin), JPA, Hibernate	250	
2	Sprint 1 project	30	Sprint 1 - Backend implementation using JPA
3	HTML, CSS, JavaScript	45	
4	Sprint 2 project	15	Sprint 1 - Frontend implementation using HTML, CSS, JS
5	Soft Skills	70	

Curriculum

Agile SCRUM

Execution:

- **Sprint 1 Implementation with code reviews**
  - o Implementing Core Java and Hibernate into the project
  - o Test case reviews
  - o Code reviews
  - o Performance monitoring during the sprint implementation and sharing the feedback o **Sprint – 1 Evaluation 30min/participant**

- **Sprint 2 Implementation with code reviews**
  - o Creating front end for the project using HTML, CSS, JavaScript
  - o Code reviews
  - o Performance monitoring during the sprint implementation and sharing the feedback
  - o **Sprint - 2 Evaluation 30min/participant**

Core Java 8

Contents:

- o Using a Variable or Array Element That Is Uninitialized and Unassigned o Local (Stack, Automatic) Primitives and Objects
- o Passing Variables into Methods
- o Passing Object Reference Variables
- o Does Java Use Pass-By-Value Semantics?
- o Passing Primitive Variables
- o Array Declaration, Construction, and Initialization
- o Declaring an Array
- o Constructing an Array
- o Initializing an Array
- o Initialization Blocks
- o Using Wrapper Classes and Boxing
- o An Overview of the Wrapper Classes
- o Creating Wrapper Objects
- o Using Wrapper Conversion Utilities
- o Autoboxing
- o Overloading
- o Garbage Collection
- o Overview of Memory Management and Garbage Collection
- o Overview of Java's Garbage Collector
- o Writing Code That Explicitly Makes Objects Eligible for Garbage Collection

• **Operators**

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

• **Flow Control, Exceptions**

- o if and switch Statements
- o if-else Branching
- o switch Statements
- o Loops and Iterators
- o Using while Loops
- o Using do Loops
- o Using for Loops
- o Using break and continue
- o Unlabeled Statements
- o Labeled Statements
- o Handling Exceptions
- o Catching an Exception Using try and catch
- o Using finally
- o 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

- **Gradle Fundamentals**
  - Introduction
  - Folder Structure
  - Install and Setup Gradle on Windows

  - Dependencies in Build Scripts
  - Gradle Wrapper
  - Lifecycle Tasks: The Base Plug In
  - Using Project Info and the check command
  - Creating Variables and external properties o Creating a Build Scan
  - Dependencies

- **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 o
Parallel Search on a Java 8 ConcurrentHashMap
o Parallel Map / Reduce on a Java 8 ConcurrentHashMap
o Parallel ForEach on a Java 8 ConcurrentHashMap
o Creating a Concurrent Set on a Java 8 ConcurrentHashMap
o Introducing Skip Lists to Implement ConcurrentMap
o Understanding How Linked Lists Can Be Improved by Skip Lists o
How to Make a Skip List Concurrent Without Synchronization

- **Lambda Expressions**
  - o Introduction
  - o Writing Lambda Expressions
  - o Functional Interfaces
  - o Types of Functional Interfaces
  - o Method reference
- **Stream API**
  - o Introduction
  - o Stream API with Collections
  - o Stream Operations

Introduction to Design Pattern

Self learning with online links and explanation by Trainer with Demos

- o Creational Design Pattern
  - ▪ Factory Pattern
  - ▪ Singleton Pattern
  - ▪ Prototype Pattern
- o Structural Design Pattern
  - ▪ Decorator Pattern
  - ▪ Facade Pattern
- o Behavioral Design Pattern
  - ▪ Chain of Responsibility Pattern
  - ▪ Iterator Pattern
- o Presentation Layer Design Pattern
  - ▪ Intercepting Filter Pattern
  - ▪ Front Controller Pattern
- o Business Layer Design Pattern
  - ▪ Business Delegate Pattern
  - ▪ Transfer Object Pattern
- o Integration Layer Design Pattern
  - ▪ Data Access Object Pattern

DevOps (Git, SonarQube, Maven, Jenkins)

- **Introduction to DevOps**
    - o Introduction of DevOps
    - o Dev And Ops
    - o Agile Vs DevOps
    - o Continuous Integration & Delivery pipeline
    - o Tools For DevOps
    - o Use-case walkthrough
- GIT Hub
    - o Working locally with GIT
    - o Working remotely with GIT
    - o Branching, merging & rebasing with GIT
    - o Use Case walkthrough
- Jenkins:
    - o Introduction to Jenkins
    - o Jenkins Objective
    - o Introduction to continuous integration deployment & Jenkins-ci o

Continuous Deployment & distribution builds with Jenkins • Sonar

- o Introduction to Sonar
- o Code quality Monitoring- Sonar
- o Use Case walkthrough

Database Using MySQL

Contents:

- **Introduction**
    - o The Relational Model
    - o What is MySQL?
    - o MySQL – Data Types
    - o Arrays Functions and Operators
- **Understanding Basic** MySQL **Syntax**
    - o The Relational Model
    - o Basic SQL Commands - SELECT
    - o Basic SQL Commands - INSERT
    - o Basic SQL Commands - UPDATE
    - o Basic SQL Commands – DELETE

- **Querying Data with the SELECT Statement**
    - o Wildcards (%, _)
    - o The SELECT List

- SELECT List Wildcard (*)
- The FROM Clause
- How to Constrain the Result Set
- DISTINCT and NOT DISTINCT

- **Filtering Results with the Where Clause**
  - WHERE Clause
  - Boolean Operators
  - The AND Keyword
  - The OR Keyword
  - Other Boolean Operators BETWEEN, LIKE, IN, IS, IS NOT

- **Shaping Results with ORDER BY and GROUP BY**

  ORDER BY
  - Set Functions
  - Set Function And Qualifiers
  - GROUP BY
  - HAVING clause

- **Matching Different Data Tables with JOINs**
  - Table Aliases
  - CROSS JOIN
  - INNER JOIN
  - OUTER JOINs
  - LEFT OUTER JOIN
  - RIGHT OUTER JOIN
  - FULL OUTER JOIN
  - SELF JOIN
  - Natural Join

- **Creating Database Tables**
  - CREATE DATABASE
  - CREATE TABLE
  - NULL Values
  - PRIMARY KEY
  - CONSTRAINT
  - ALTER TABLE
  - DROP TABLE
- **Transactions**
  - BEGIN, COMMIT, ROLLBACK

- **Constraints**
  - CHECK, UNIQUE, NOT NULL

▪ **Introduction to JDBC**

o **Connection, Statement, PreparedStatement, ResultSet**

JPA with Hibernate 3.0

Contents:

- **Introduction**
  - Introduction & overview of data persistence
  - Overview of ORM tools
  - Understanding JPA
  - JPA Specifications
- **Entities**
  - Requirements for Entity Classes
  - Persistent Fields and Properties in Entity Classes
  - Persistent Fields
  - Persistent Properties
  - Using Collections in Entity Fields and Properties
  - Validating Persistent Fields and Properties
  - Primary Keys in Entities
- **Managing Entities**
  - The EntityManager Interface
  - Container-Managed Entity Managers
  - Application-Managed Entity Managers
  - Finding Entities Using the EntityManager
  - Managing an Entity Instance's Lifecycle
  - Persisting Entity Instances
  - Removing Entity Instances
  - Synchronizing Entity Data to the Database
  - Persistence Units
- **Querying Entities**
  - Java Persistence query language (JPQL)
  - Criteria API
- **Entity Relationships**
  - Direction in Entity Relationships
  - Bidirectional Relationships
  - Unidirectional Relationships
  - Queries and Relationship Direction
  - Cascade Operations and Relationships

***** Sprint 1 Project using Core Java + JPA and Exception handling — 30 hrs**

HTML 5, CSS 3 with Bootstrap, Javascript, TypeScript

Contents:

HTML 5:

- HTML Basics
    - o Understand the structure of an HTML page.
    - o New Semantic Elements in HTML 5
    - o Learn to apply physical/logical character effects.
    - o Learn to manage document spacing.
- Tables
    - o Understand the structure of an HTML table.
    - o Learn to control table format like cell spanning, cell spacing, border
- List
    - o Numbered List
    - o Bulleted List
- Working with Links
    - o Understand the working of hyperlinks in web pages.
    - o Learn to create hyperlinks in web pages.
    - o Add hyperlinks to list items and table contents.
- Image Handling
    - o Understand the role of images in web pages
    - o Learn to add images to web pages
    - o Learn to use images as hyperlinks
- Frames
    - o Understand the need for frames in web pages.
    - o Learn to create and work with frames.
- HTML Forms for User Input
    - o Understand the role of forms in web pages
    - o Understand various HTML elements used in forms.
    - o Single line text field
    - o Text area
    - o Check box
    - o Radio buttons
    - o Password fields
    - o Pull-down menus
    - o File selector dialog box
- New Form Elements
    - o Understand the new HTML form elements such as date, number, range, email, search and datalist
    - o 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

- Implementing CSS3 in the "Real World"
  - Modernizr
  - HTML5 Shims
  - SASS, and Other CSS Preprocessors
  - CSS Grid Systems
  - CSS Frameworks

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 Essentials**
- **ES6 & Typescript**

- Var, Let and Const keyword
- Arrow functions, default arguments
- Template Strings, String methods
- Object de-structuring
- Spread and Rest operator
- Typescript Fundamentals
- Types & type assertions, Creating custom object types, function types - Typescript OOPS - Classes, Interfaces, Constructor, etc

***** Sprint 2 Project using HTML, CSS, JavaScript — 15 hrs**