Shri Vile Parle Kelavani Mandal's

Dwarkadas J. Sanghvi College of Engineering

(Autonomous College Affiliated to the University of Mumbai)

Scheme and Detailed Syllabus (DJS23)

of

Honours Degree Program

in

DevOps (Development and Operations)

Revision: 2

With effect from the Academic Year: 2024-2025

Sr.	Course Code	Course	Teaching Scheme (hrs.)				Continuous Assessment (A) (marks)			Semester End Assessment (B) (marks)					(A+B)	Total Credits
			Th	P	T	Credits	Th	T/W	Total CA (A)	Th	O	P	O & P	Total SEA (B)		
Sem III																
1	DJS23IH1201	Development Frameworks	4	--	--	4	40	--	40	60	--	--	--	60	100	4
Sem IV																
2	DJS23IH1251L	Advanced Java Laboratory	--	4	--	2	--	25	25	--	--	--	25	--	50	2
Sem V																
3	DJS23IH1301	DevOps	3	--	--	3	40	--	40	60	--	--	--	60	100	3
4	DJS23IH1301L	DevOps Laboratory	--	2	--	1	--	25	25	--	--	--	--	--	25	1
Sem VI																
5	DJS23IH1351	MLOps	3	--	--	3	40	--	40	60	--	--	--	60	100	3
6	DJS23IH1351L	MLOps Laboratory	--	2	--	1	--	25	25	--	--	--	--	--	25	1
Sem VII																
7	DJS23IH1401	DevSecOps	4	--	--	4	40	--	40	60	--	--	--	60	100	4
		Total	14	8	0	18	160	75	235	240	0	0	25	240	500	18

Program: Information Technology **S.Y B. Tech Semester: III**

Course: Development Frameworks (DJS23IH1201)

Pre-requisite: Knowledge of any programming language and Database Management System

Course Objectives: The objective of this course is to familiarize learners with different development frameworks. The course also introduces students to the principles and process of software engineering.

Course Outcomes: On successful completion of this course, student should be able to:

1. Select appropriate frameworks for application development.
2. Apply software engineering principles for application development.

Detailed Syllabus: (Unit wise)		
Unit	**Description**	**Duration**
1	Introduction to Software Engineering and Process Model: Introduction to Software Engineering, Process framework, Software Development Life Cycle (SDLC), Process Models: Sequential, Incremental and Evolutionary models, Software Requirements - Functional and Non-Functional requirements, Software Requirements Specification (SRS) Introduction to Frameworks: Definition and characteristics of frameworks, Historical background and evolution of frameworks, Types of frameworks (e.g., web frameworks, application frameworks, testing frameworks).	**08**
2	Fundamentals of Agile Process: Concept of agility, Need of Agile software development, Agile Manifesto and Principles, Stakeholders and Challenges, Overview of Agile Development Models: Scrum, Extreme Programming, Feature Driven Development, Crystal, Kanban, and Lean Software Development, Methods, Values, Roles, Artifacts, Stakeholders, and challenges. Business benefits of software agility, ASD, DSD. Introduction to Scrum: Agile Scrum Framework, Scrum Artifacts, Meetings, Activities and Roles, Scrum Team Simulation, Scrum Planning Principles, Product and Release Planning, sprinting: Planning, Execution, Review and Retrospective; User story definition and Characteristics, Acceptance tests and Verifying stories, Burn down chart, Daily scrum, Scrum Case Study.	**12**
3	Introduction to Architectures: Introduction to Model View Controller (MVC) Framework: History of MVC, Features of MVC, MVC Architecture, MVC Examples, Popular MVC Frameworks, Advantages and Drawbacks of MVC, 3-Tier Architecture Vs MVC Architecture. The Reactive Manifesto: Introduction, Reactive Principles, Reactive Systems vs Reactive Programming Clean architecture: Introduction, The Dependency Rule, A Typical Scenario.	**10**
4	SOLID Design principles: Introduction, The Single Responsibility Principle, The Open-Closed Principle, The Liskov Substitution Principle, The Interface Segregation Principle, The Dependency Inversion Principle. Reactive architecture: Introduction, Design Principles of Reactive Systems, commands and Events, Commands, Events, Messages, Commands Versus Events: An Example Destinations and Space Decoupling, Time Decoupling, The Role of Nonblocking Input/Output, Blocking Network I/O, Threads, and Concurrency, How Does Nonblocking I/O Work? Reactor Pattern and Event Loop, Anatomy of Reactive Applications.	**10**
5	Core Technologies of Spring Framework: Introduction to Object oriented programming concept, Spring–Environment Setup, Spring beans and its scopes, Spring bean lifecycle, how to create a bean using Factory Bean? How to create a bean using static Factory	**06**

	Bean? Best Practices of spring Framework, Spring Dependency Injection and Inversion of Controls, Spring Java Configuration vs XML configuration.	
6	Spring Event Handling and Aspect Oriented Programming (AOP): Event Handling in Spring, Custom Events in Spring, AOP Concepts, Types of AOP, AOP in Spring, AOP Spring Architecture, Framework Services for AOP, Using @AspectJ-Style Annotations, AspectJ Integration, Spring - Transaction Management, Spring Web MVC Framework, Spring - Logging with Log4J. Spring Boot: Introduction to spring boot, spring boot Build systems, spring boot Code structure, Springs and dependency injection, spring boot Runners, Spring Boot – Application Properties	**06**

Books Recommended:

Textbooks:

1. Iuliana Cosmina Rob Harrop Chris Schaefer Clarence Ho," An In-Depth Guide to the Spring Framework and Its Tools", 5th Edition, Apress, 2017.
2. Roger S Pressman, "Software Engineering: A Practitioner's Approach", 8th Edition, McGraw-Hill, 2015.
3. Ian Sommerville, "Software Engineering", 9th Edition, Pearson Education, 2011.
4. Clement Escoffier , Ken Finnigan, "Reactive Systems in Java: Resilient, Event-Driven Architecture with Quarkus, 1st Edition, O'Reilly Media, 2021
5. Craig Walls. "Spring Boot in Action" 6th Edition, Manning, 2016.

Reference Books:

1. Ashish Sarin J Sharma, "Getting Started with Spring Framework", 2nd Edition, CreateSpace, 2012
2. Rod Johnson et al," Professional Java Development with the Spring Framework", John Wiley & Sons 2005.

Prepared by **Checked by** **Head of the Department** **Principal**

Program: Information Technology **S.Y B. Tech Semester: IV**

Course: Advanced Java Laboratory (DJS23IH1251L)

Pre-requisite: Structured programming using C, Object Oriented Programming using Java.

Course Objectives: The objective of the course is to introduce and familiarize students with advanced concepts that go beyond Core Java – most importantly the APIs defined in Java 8. Through this course, students will delve into Java Collections, exploring the intricacies of managing and manipulating data structures efficiently. Understand how to apply design patterns to solve common software design problems and improve code quality, reusability, and scalability.

Course Outcomes: On successful completion of this course, student should be able to:

1. Develop reusable codes using generics.
2. Use various APIs in Java for efficient application development.
3. Use appropriate design patterns.

Detailed Syllabus: (unit wise)		
Unit	**Description**	**Duration**
1	Java Collections: Collections in Java, basic data structures, arrays and lists, stacks, and queues, sets and maps. Generics: Basic generics, bounded type parameters, type inference, wildcards, type erasure	08
2	Java Reflection API: Modifiers and Security, Accessing Fields, Accessing Methods, Accessing Constructors, What About Arrays? Accessing Generic Type Information, Accessing Annotation Data, Dynamic Interface Adapters	08
3	Lambda Expression: Lambda expression fundamentals, Functional Interfaces, examples on Lambda Expressions, Block Lambda Expressions, Generic Functional Interfaces, Passing Lambda Expression as Arguments, Lambda Expression and Exceptions, Lambda Expression and variable Capture, Method References to static methods, Method references to Instance methods, Method references with Generics, Constructor References, Predefined Functional Interfaces, Comparing method references with lambda expressions.	12
4	The Stream API: Stream Basics, Stream Interfaces, How to Obtain a Stream, A Simple Stream Example, Reduction Operations, Using Parallel Streams, Mapping, Collecting, Iterators and Streams, Use an Iterator with a Stream, Use Spliterator	06
5.	Annotations: Annotations Basics, specifying a retention policy, Obtaining Annotations at run time by use of Reflections, The AnnotatedElement Interface, using default values, Marker Annotations, Single member Annotations, The Built in Annotations, Type Annotations, Repeating Annotations, some Restrictions. Comparable and Comparator, Optional Class: Date/Time API: Date, Calendar, GregorianCalender, TimeZone, SimpleTimeZone, Locale	12
6	Introduction to Design Patterns Creational: Singleton Pattern, Structural: Adapter Pattern, Behavioural: Observer Pattern	06

List of Practical's:

1. Creating JDBC application
2. Implementation of different collection types (stacks, queues, vectors etc)
3. Creation of generic classes, methods
4. Use reflection API to examine or modify the behaviour of methods, classes, and interfaces at runtime.
5. Using streams API to implement program logic by composing functions and executing them in a data flow.
6. Demonstration of lambda expressions.
7. Implementation of Functional Interfaces, Comparable and Comparator.
8. Implementation of Optional Class, Date/Time API.
9. Implementation of Annotations.
10. Implementation of Singleton Design Patterns.
11. Implementation of Structural Design Patterns.
12. Implementation of Behavioral Design Patterns.

Books Recommended:

Textbooks:

1. Anita Seth, B.L. Juneja, "JAVA: ONE STEP AHEAD", 1st Edition, Oxford University Press; (20 May 2017)
2. Patrick Niemeyer, Daniel Leuck, "Learning Java", 4th Edition, O'Reilly Media, Inc, June 2013

Reference Books:

1. Herbert Schildt, "Java: The Complete Reference", 9th edition, McGraw Hill.
2. Uttam K. Roy, "Advanced Java Programming, Oxford University Press, 2015.
3. D.T. Editorial Services, "Java 8 Programming Black Book", Dream Tech Press, 2015.
4. Cay S. Horstmann, Gary Cornell, "Core Java™ 2: Volume II–Advanced Features" 9th Edition, Prentice Hall PTR.

Prepared by **Checked by** **Head of the Department** **Principal**

Program: Information Technology **T.Y B. Tech Semester: V**

Course: DevOps (DJS23IH1301)

DevOps Laboratory (DJS23IH1301L)

Pre-requisite:

1. Knowledge of Linux Operating system, installation and configuration of services and command line basics.
2. Basics of Computer Networks and Software
3. Software Development Life cycle.

Course Objectives: The objective of this course is to understand the fundamentals of DevOps engineering and be fully proficient with DevOps terminologies, concepts, benefits, and deployment options to meet real world software development requirements.

Course Outcomes: On completion of the course, learner will be able to:

1. Apply DevOps principles to meet software development requirements.

Detailed Syllabus: (Unit wise)		
Unit	**Description**	**Duration**
1	Introduction to DevOps: Introduction, History of DevOps, DevOps Stakeholders, Goals, Important terminologies, DevOps Vs Agile, DevOps Tools, DevOps Lifecycle, Challenges with DevOps Implementation, Minimum Viable Product (MVP) & Cross-functional Teams. Case studies: Traditional software development approaches and DevOps culture.	04
2	Version Control: Introduction, Overview of Version Control Systems, Role of Version Control System, Types of Control Systems and their Supporting Tools Database Version Control: Introduction, Need of database version control, Methods of database version control, Challenges and benefits of Database version control, Database version control tools Security and Access Control in VCS: Role-based access. Audit trails and commit signing	08
3	Continuous Integration: Introduction, Definition and Purpose of CI/CD, Key Benefits and Challenges of CI, Development without CI vs. Development with CI, Tools for CI process , Introduction to Jenkins, Key features of Jenkins, Architecture of Jenkins, Jenkins in Agile and DevOps environments, Continuous Integration with Jenkins, Use Cases for Distributed Builds in Jenkins, Coding Standards and Best Practices in CI: Importance of coding standards, Code quality tools, Coding Standards and Best Practices in CI	08
4	Continuous Deployment: Introduction, Need of CD, Features and Benefits of continuous deployment, , Process of Continuous Deployment, Deployment Patterns, Continuous deployment tools , Continuous deployment vs. continuous delivery, Overview of Docker, Docker vs Virtual Machines, Architecture, Docker Containers, Docker Workflow, Anatomy of Docker Image, Role of Docker in the CI/CD Pipeline, Advantages of Docker, CD in monoliths vs. microservices, CD in legacy applications. Case studies: Netflix, Amazon, Google	08
5	Continuous Testing: Introduction, Continuous Testing vs. Traditional Testing, the role of continuous testing in DevOps, Types of Continuous Testing, Best Practices and Challenges in Continuous Testing, Steps to implement a continuous testing strategy into DevOps pipeline, Tools for Continuous Testing Introduction to Selenium, Selenium Tool Suite, Selenium IDE, Selenium RC, Selenium WebDriver, Architecture of Selenium Webdriver, Differences between Selenium 2 and Selenium 3, Page Object Model (POM) & Page Factory in Selenium	06
6	Continuous Management: Introduction, Key Aspects of Continuous Management, Benefits of Continuous Management in DevOps, Overview of Infrastructure as a code, Benefits of Infrastructure as Code, The Four Key Metrics, Three Core Practices for Infrastructure as Code, The Parts of an Infrastructure System, Infrastructure Platforms, Infrastructure Resources,	05

	Compute Resources, Storage Resources, Network Resources Puppet Architecture, The Puppet Server, Performance Optimizations, Ansible: Ansible Architecture, Ansible and Infrastructure Management, Local Infrastructure Development: Ansible and Vagrant.	

Suggested list of Laboratory Experiments (Tools):

1. To understand Version Control System / Source Code Management, install git and create a GitHub account.
2. To Perform various GIT operations on local and Remote repositories using GIT Cheat-Sheet.
3. To set up a CI pipeline using GitHub.
4. To integrate Git version control with project management tool (e.g. JIRA) to manage software development tasks.
5. To study deployment patterns on cloud, on premise and DevOps.
6. To implement team workflow using Git, GitHub, Jira where each team member has a specific role.
7. To understand Continuous Integration, install and configure Jenkins to setup a build Job.
8. To Setup and Run Selenium Tests in Jenkins.
9. To understand Docker Architecture and Container Life Cycle, install Docker and execute docker commands to manage images and interact with containers.
10. To learn Dockerfile instructions, build an image for a sample web application using Dockerfile.

Books Recommended:

Textbooks:

1. Iuliana Cosmina Rob Harrop Chris Schaefer Clarence Ho," An In-Depth Guide to the Spring Framework and Its Tools", 5th Edition, Apress, 2017.
2. Roger S Pressman, "Software Engineering: A Practitioner's Approach", 8th Edition, McGraw-Hill, 2015.
3. Ian Sommerville, "Software Engineering", 9th Edition, Pearson Education, 2011.
4. Clement Escoffier , Ken Finnigan, "Reactive Systems in Java: Resilient, Event-Driven Architecture with Quarkus, 1st Edition, O'Reilly Media, 2021
5. Craig Walls. "Spring Boot in Action" 6th Edition, Manning, 2016.
6. Clean Code: A Handbook of Agile Software Craftsmanship" by Robert C. Martin, 1st Edition, Pearson Education, 2008

Reference Books:

1. Ashish Sarin J Sharma, "Getting Started with Spring Framework", 2nd Edition, CreateSpace, 2012
2. Rod Johnson et al," Professional Java Development with the Spring Framework", John Wiley & Sons 2000.

Prepared by **Checked by** **Head of the Department** **Principal**