# GUJARAT TECHNOLOGICAL UNIVERSITY

**Program Name:** Engineering **Level:** Diploma **Branch:** Information & Communication Technology **Course / Subject Code:** [To be assigned] **Course / Subject Name:** Advanced Java Programming

## w. e. f. Academic Year: 2025-26

**Semester:** 4th **Category of the Course:** PCC (Program Core Course)

## Prerequisite:

Successful completion of **Semester 3 – Java Programming (DI03032021)** with foundational understanding of:

* Object-oriented programming concepts including classes, objects, inheritance, polymorphism, and interfaces
* Exception handling mechanisms and basic multithreading
* Collections framework (ArrayList, LinkedList, HashSet, HashMap)
* File input/output operations and basic serialization

## Rationale:

This course advances students from core Java programming to enterprise web application development. Building upon the object-oriented foundation established in Semester 3, students learn to create production-ready, database-driven web applications using industry-standard technologies.

The course adopts a **project-driven pedagogical approach** where students incrementally build a complete web application throughout the semester. This methodology ensures comprehensive understanding of how JDBC, Servlets, JSP, and REST APIs integrate to form modern web solutions. Students develop proficiency in database connectivity, server-side programming, dynamic web page generation, and RESTful API development.

The emphasis is on practical application development that prepares diploma graduates for immediate employment in Java web development roles. By course completion, students will have developed a complete, deployable web application that serves as a portfolio demonstration for career opportunities. This course provides the necessary foundation for advanced frameworks and prepares students for professional Java development positions.

## Course Outcome:

After Completion of the Course, Student will be able to:

| No | Course Outcomes | RBT Level |
| --- | --- | --- |
| 01 | Apply modern Java features (generics, lambdas, streams) to develop efficient and maintainable applications. | R,U,A |
| 02 | Develop secure database-driven applications using JDBC with proper transaction management and resource handling. | R,U,A |
| 03 | Create server-side web applications using Java Servlets with session management and security implementation. | R,U,A |
| 04 | Build dynamic web interfaces using JSP with Expression Language and JSTL following MVC architecture principles. | R,U,A |
| 05 | Design and implement RESTful web services and integrate development tools for complete application deployment. | R,U,A |

*Revised Bloom’s Taxonomy (RBT)*

## Teaching and Examination Scheme:

| Teaching Scheme (in Hours) |  |  | Total Credits L+T+(PR/2) | Assessment Pattern and Marks |  |  |  | Total Marks |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **L** | **T** | **PR** | **C** | **Theory** |  | **Tutorial / Practical** |  |  |
|  |  |  |  | ESE (E) | PA(M) | PA(I) | ESE (V) |  |
| 3 | 0 | 2 | 4 | 70 | 30 | 20 | 30 | 150 |

**Assessment Components:**

**Theory (100 marks):**

* ESE (E): 70 marks - End Semester Examination
* PA (M): 30 marks - Periodic Assessment (Mid-term examination)

**Tutorial/Practical (50 marks):**

* PA (I): 20 marks - Continuous lab assessment (attendance, participation, regular coding assignments)
* ESE (V): 30 marks - Practical examination (15 marks) + Mini-project demonstration and viva (15 marks)

## Course Content:

| Unit No. | Content | No. of Hours | % of Weightage |
| --- | --- | --- | --- |
| **1.** | **Modern Java Programming Features** | **06** | **13** |
|  | 1.1 Generics Implementation |  |  |
|  | 1.1.1 Type safety requirements in collections |  |  |
|  | 1.1.2 Generic classes and methods implementation |  |  |
|  | 1.1.3 Bounded type parameters with extends keyword |  |  |
|  | 1.2 Lambda Expressions and Functional Programming |  |  |
|  | 1.2.1 Lambda syntax and functional interfaces |  |  |
|  | 1.2.2 Predicate, Consumer, Function, and Supplier interfaces |  |  |
|  | 1.2.3 Method references (static, instance, constructor) |  |  |
|  | 1.3 Streams API for Data Processing |  |  |
|  | 1.3.1 Stream creation from collections |  |  |
|  | 1.3.2 Intermediate operations (filter, map, sorted, distinct) |  |  |
|  | 1.3.3 Terminal operations (collect, forEach, reduce, count) |  |  |
|  | 1.3.4 Collectors utility class implementation |  |  |
|  | 1.4 Annotations Framework |  |  |
|  | 1.4.1 Annotation purposes and applications |  |  |
|  | 1.4.2 Standard annotations (@Override, @Deprecated, @WebServlet) |  |  |
| **2.** | **Database Programming with JDBC** | **10** | **22** |
|  | 2.1 JDBC Architecture and Configuration |  |  |
|  | 2.1.1 JDBC components and driver implementation |  |  |
|  | 2.1.2 Connection establishment and management |  |  |
|  | 2.1.3 Database connectivity with MySQL |  |  |
|  | 2.1.4 Maven project configuration and dependency management |  |  |
|  | 2.2 Database Operations and Data Management |  |  |
|  | 2.2.1 Statement execution methodologies |  |  |
|  | 2.2.2 PreparedStatement for secure query execution |  |  |
|  | 2.2.3 ResultSet processing and data retrieval |  |  |
|  | 2.2.4 CRUD operations implementation |  |  |
|  | 2.3 Transaction Management |  |  |
|  | 2.3.1 Transaction processing concepts |  |  |
|  | 2.3.2 Commit and rollback operations |  |  |
|  | 2.3.3 Transaction consistency and integrity |  |  |
|  | 2.4 Security and Best Practices |  |  |
|  | 2.4.1 SQL injection prevention techniques |  |  |
|  | 2.4.2 Input validation and data sanitization |  |  |
|  | 2.4.3 Resource management and exception handling |  |  |
| **3.** | **Web Application Development with Servlets** | **13** | **29** |
|  | 3.1 Web Application Architecture |  |  |
|  | 3.1.1 Client-server communication protocols |  |  |
|  | 3.1.2 HTTP request and response mechanisms |  |  |
|  | 3.1.3 Web application project structure |  |  |
|  | 3.1.4 Apache Tomcat server configuration |  |  |
|  | 3.2 Servlet Programming |  |  |
|  | 3.2.1 Servlet API and HttpServlet implementation |  |  |
|  | 3.2.2 Servlet lifecycle management |  |  |
|  | 3.2.3 HTTP method handling (GET, POST) |  |  |
|  | 3.2.4 Request parameter processing |  |  |
|  | 3.2.5 Response generation and content delivery |  |  |
|  | 3.3 Servlet Configuration and Deployment |  |  |
|  | 3.3.1 Annotation-based servlet configuration |  |  |
|  | 3.3.2 Application deployment strategies |  |  |
|  | 3.4 Session Management |  |  |
|  | 3.4.1 HTTP stateless protocol challenges |  |  |
|  | 3.4.2 HttpSession interface implementation |  |  |
|  | 3.4.3 User authentication and authorization |  |  |
|  | 3.4.4 Cookie implementation for state management |  |  |
|  | 3.5 Security Implementation and Filters |  |  |
|  | 3.5.1 Input validation and security measures |  |  |
|  | 3.5.2 Password encryption and secure authentication |  |  |
|  | 3.5.3 Servlet filters for security enforcement |  |  |
|  | 3.5.4 Application logging and monitoring |  |  |
| **4.** | **Dynamic Web Development with JSP and MVC** | **09** | **20** |
|  | 4.1 JavaServer Pages Fundamentals |  |  |
|  | 4.1.1 JSP architecture and servlet comparison |  |  |
|  | 4.1.2 JSP lifecycle and processing |  |  |
|  | 4.1.3 JSP directives and implicit objects |  |  |
|  | 4.2 Expression Language Implementation |  |  |
|  | 4.2.1 EL syntax and expression evaluation |  |  |
|  | 4.2.2 Scope-based data access |  |  |
|  | 4.2.3 JavaBean property access |  |  |
|  | 4.2.4 EL operators and built-in functions |  |  |
|  | 4.3 JSTL (JavaServer Pages Standard Tag Library) |  |  |
|  | 4.3.1 JSTL library configuration and setup |  |  |
|  | 4.3.2 Conditional logic implementation (c:if, c:choose) |  |  |
|  | 4.3.3 Iteration control (c:forEach) |  |  |
|  | 4.3.4 Variable manipulation and data handling |  |  |
|  | 4.4 Model-View-Controller Architecture |  |  |
|  | 4.4.1 MVC design pattern principles |  |  |
|  | 4.4.2 Controller implementation with servlets |  |  |
|  | 4.4.3 View layer development with JSP |  |  |
|  | 4.4.4 Model layer design with JavaBeans |  |  |
|  | 4.4.5 Component integration and request dispatching |  |  |
| **5.** | **RESTful Web Services and Development Tools** | **07** | **15** |
|  | 5.1 REST Architecture Principles |  |  |
|  | 5.1.1 RESTful web service design principles |  |  |
|  | 5.1.2 HTTP methods for REST implementation |  |  |
|  | 5.1.3 JSON data format and structure |  |  |
|  | 5.1.4 HTTP status codes and response handling |  |  |
|  | 5.2 REST API Development |  |  |
|  | 5.2.1 RESTful endpoint creation with servlets |  |  |
|  | 5.2.2 JSON serialization and deserialization with Gson |  |  |
|  | 5.2.3 Request and response data handling |  |  |
|  | 5.2.4 Error handling and status code implementation |  |  |
|  | 5.3 API Testing and Consumption |  |  |
|  | 5.3.1 API testing methodologies with Postman |  |  |
|  | 5.3.2 HTTP client implementation for API consumption |  |  |
|  | 5.3.3 External API integration techniques |  |  |
|  | 5.4 Development Tools and Version Control |  |  |
|  | 5.4.1 Git version control system implementation |  |  |
|  | 5.4.2 Maven build automation and dependency management |  |  |
|  | 5.4.3 Unit testing with JUnit framework |  |  |
|  | **Total** | **45** | **100** |

## Suggested Specification Table with Marks (Theory):

| Distribution of Theory Marks (in %) |  |  |
| --- | --- | --- |
| **R Level** | **U Level** | **A Level** |
| 20 | 30 | 50 |

*Where R: Remember; U: Understanding; A: Application (as per Revised Bloom’s Taxonomy for Diploma Level)*

**Note:** Higher weightage on Application level (50%) reflects the practical implementation focus of this course where students develop complete web applications.

## References/Suggested Learning Resources:

**(a) Books:**

1. Head First Servlets and JSP by Bryan Basham, Kathy Sierra, Bert Bates, O’Reilly Media, 2nd Edition, 2008, ISBN: 9780596516680
2. Java: The Complete Reference by Herbert Schildt, McGraw Hill Education, 12th Edition, 2021, ISBN: 9781260440232
3. Core Servlets and JavaServer Pages by Marty Hall, Prentice Hall, 2nd Edition, 2008, ISBN: 9780131482609
4. Programming with Java by E Balagurusamy, McGraw Hill Education, 6th Edition, 2019, ISBN: 9789353161491
5. Effective Java by Joshua Bloch, Addison-Wesley, 3rd Edition, 2017, ISBN: 9780134685991

**(b) Open source software and website:**

**Software**

* [Java Development Kit – JDK 17 or above](https://www.oracle.com/java/technologies/downloads/)
* [Apache Tomcat 9.x or 10.x](https://tomcat.apache.org/)
* [MySQL 8.x](https://www.mysql.com/downloads/)
* [PostgreSQL (alternative)](https://www.postgresql.org/download/)
* [IntelliJ IDEA Community Edition](https://www.jetbrains.com/idea/download/)
* [Eclipse IDE for Enterprise Java and Web Developers](https://www.eclipse.org/downloads/)
* [Visual Studio Code with Java extensions](https://code.visualstudio.com/)
* [Maven 3.x](https://maven.apache.org/download.cgi)
* [Postman (API testing tool)](https://www.postman.com/downloads/)
* [Git (version control system)](https://git-scm.com/downloads)

**Official Documentation**

* [Official Java EE Documentation](https://docs.oracle.com/javaee/)
* [Apache Tomcat Documentation](https://tomcat.apache.org/tomcat-9.0-doc/)
* [MySQL Documentation](https://dev.mysql.com/doc/)
* [Maven Documentation](https://maven.apache.org/guides/)
* [Gson User Guide](https://github.com/google/gson/blob/master/UserGuide.md)

**Online Tutorials**

* [Baeldung - Java and Spring Tutorials](https://www.baeldung.com/)
* [JavaTpoint - Servlet and JSP Tutorials](https://www.javatpoint.com/servlet-tutorial)
* [GeeksforGeeks - Java Web Development](https://www.geeksforgeeks.org/java-web-development/)
* [TutorialsPoint - Advanced Java Tutorials](https://www.tutorialspoint.com/java/index.htm)
* [Oracle Java Tutorials - JDBC and Web Services](https://docs.oracle.com/javase/tutorial/)

**Video Courses**

* [Java Brains - Servlets and JSP Tutorial (YouTube Playlist)](https://www.youtube.com/playlist?list=PLqq-6Pq4lTTbx8p2oCgcAQGQyqN8XeA1x)
* [Telusko - Advanced Java Full Course (YouTube Playlist)](https://www.youtube.com/playlist?list=PLsyeobzWxl7pUPF2xjjJiG4BKC9x_GJ46)
* [Programming with Mosh - Java Web Development Tutorial](https://www.youtube.com/watch?v=eIrMbAQSU34)
* [Amigoscode - Full Stack Development with Java and React (YouTube)](https://www.youtube.com/playlist?list=PLwvrYc43l1MzeA2bBYQhCWr2gvWLs9A7S)

**Comprehensive Courses**

* [Java Web Development with Servlets and JSP by LearnQuest on Coursera](https://www.coursera.org/learn/java-servlet-pages)
* [Full Stack Java Developer Nanodegree by Udacity](https://www.udacity.com/course/java-developer-nanodegree--nd035)
* [Java Web Services and RESTful API with Spring Boot by Udemy](https://www.udemy.com/course/spring-boot-microservices-and-spring-cloud/)
* [Enterprise Java Development Specialization by University of California San Diego on edX](https://www.edx.org/professional-certificate/uc-san-diegox-java-programming)

**Professional Development Platforms**

* [GitHub (for project hosting and portfolio development)](https://github.com/)
* [Stack Overflow (community support and problem resolution)](https://stackoverflow.com/)
* [Maven Central Repository](https://search.maven.org/)
* [OWASP Web Security Testing Guide](https://owasp.org/www-project-web-security-testing-guide/)

## Suggested Course Practical List:

**Teaching Approach:** Progressive development of integrated “Student Management System” throughout all practical sessions

| Sr. No. | Practical Outcomes (PrOs) | Unit No. | Approx. Hrs. required |
| --- | --- | --- | --- |
| 1 | Implement generic classes and lambda expressions with student data collections. | I | 1 |
| 2 | Develop data processing applications using Streams API operations. | I | 2 |
| 3 | Configure Maven project structure and database dependencies. | II | 1 |
| 4 | Design database schema and establish JDBC connectivity. | II | 1 |
| 5 | Implement Data Access Object (DAO) with CRUD operations using PreparedStatement. | II | 2 |
| 6 | Develop transaction management for multi-table operations. | II | 1 |
| 7 | Configure Apache Tomcat and develop foundational servlet applications. | III | 1 |
| 8 | Implement secure user authentication with input validation and password encryption. | III | 2 |
| 9 | Develop session-based authentication and authorization system. | III | 2 |
| 10 | Implement servlet filters and listeners for application security and monitoring. | III | 1 |
| 11 | Integrate servlet applications with database operations. | III | 1 |
| 12 | Develop JSP pages using Expression Language for dynamic content presentation. | IV | 1 |
| 13 | Implement data input forms using JSTL with comprehensive validation. | IV | 2 |
| 14 | Refactor application architecture using MVC design pattern. | IV | 2 |
| 15 | Develop RESTful API endpoints for application data services. | V | 2 |
| 16 | Implement comprehensive API testing using Postman framework. | V | 1 |
| 17 | Establish Git version control workflow for project management. | V | 1 |
| 18 | Integrate external API services into web application. | V | 1 |
| 19 | Develop unit testing framework for application components. | V | 1 |
| 20 | Complete application integration, testing, and deployment. | All | 2 |
|  | **Total** |  | **26** |

## Suggested Project List:

**Mandatory Mini-Project (6-8 weeks):**

Students must develop **ONE comprehensive web application** demonstrating integration of all course technologies. Project development commences from Week 3 with structured weekly progress evaluations.

**Core Technical Requirements (Mandatory):**

* Normalized database design with minimum three related tables
* Complete CRUD operations accessible through web interface
* Implementation of MVC architecture (Servlets, JSP, JavaBeans)
* User authentication and session management system
* Implementation of at least one servlet filter and one listener
* Maven-based project structure with dependency management
* Comprehensive input validation and security measures
* Git version control with structured commit history
* Professional user interface design and user experience

**Advanced Technical Features (Optional - for distinction):**

* RESTful API endpoint implementation
* Unit testing framework integration
* External API service integration
* Responsive web design implementation
* Advanced features (search functionality, pagination, reporting)

**Recommended Project Domains:**

1. **Library Management System**
   * Entities: Books, Members, Authors, Transactions
   * Features: Catalog management, member registration, circulation tracking
2. **Student Information Management Portal**
   * Entities: Students, Courses, Enrollments, Academic Records
   * Features: Student profiles, course management, academic tracking
3. **Task and Project Management System**
   * Entities: Users, Projects, Tasks, Assignments
   * Features: Task allocation, progress tracking, project collaboration
4. **E-commerce Product Management System**
   * Entities: Products, Categories, Customers, Orders
   * Features: Product catalog, inventory management, order processing
5. **Content Management System**
   * Entities: Users, Posts, Comments, Categories
   * Features: Content creation, user management, publication workflow
6. **Employee Management System**
   * Entities: Employees, Departments, Attendance, Payroll
   * Features: Employee records, attendance tracking, administrative management

**Project Assessment Criteria (15 marks out of ESE V - 30 marks):**

* Functional implementation and completeness (40%)
* Code organization, structure, and documentation (25%)
* User interface design and user experience (20%)
* Security implementation and best practices (15%)

## List of Laboratory/Learning Resources Required:

| Sr. No. | Equipment Name with Broad Specifications | PrO. No. |
| --- | --- | --- |
| 1 | Computer with latest configuration (minimum 8GB RAM, i5 processor) with Windows/Linux/Mac Operating System | All |
| 2 | Java Development Kit (JDK) Version 17 or above | All |
| 3 | Apache Tomcat 9.x or 10.x (Servlet Container) | 7-20, Project |
| 4 | MySQL 8.x (Relational Database Management System) | 4-20, Project |
| 5 | Integrated Development Environment (IntelliJ IDEA Community Edition recommended) | All |
| 6 | Maven 3.x (Build automation and dependency management tool) | 3-20, Project |
| 7 | Postman (API development and testing platform) | 15-18, Project |
| 8 | Git (Distributed version control system) | 17, Project |
| 9 | Web browser (Chrome/Firefox/Edge) for application testing | 7-20, Project |

## Suggested Activities for Students:

1. **Technical Presentation Sessions:** Weekly structured presentations where students demonstrate implemented features and explain technical decisions to develop communication and documentation skills.
2. **Collaborative Programming Exercises:** Pair programming sessions for complex problem-solving exercises, fostering teamwork and knowledge sharing among students.
3. **Code Review and Quality Assurance:** Systematic peer code review sessions focusing on best practices, security considerations, and performance optimization.
4. **Industry Professional Guest Lectures:** Sessions with experienced Java developers sharing real-world development practices, career guidance, and industry trends.
5. **Progressive Project Development Reviews:** Structured weekly project evaluation sessions with faculty feedback and peer collaboration opportunities.
6. **Technical Debugging Workshops:** Hands-on sessions focusing on systematic debugging methodologies and troubleshooting techniques for web applications.
7. **API Development and Testing Laboratories:** Collaborative sessions for REST API development and comprehensive testing using professional tools and methodologies.
8. **Version Control and Portfolio Development:** Training sessions on Git workflow implementation and professional portfolio creation for career development.
9. **Security Implementation Workshops:** Focused sessions on web application security best practices, vulnerability assessment, and prevention techniques.
10. **Final Project Demonstration and Evaluation:** Comprehensive project showcase event with faculty evaluation, peer assessment, and potential industry professional participation.