# DIPLOMA IN SOFTWARE ENGINEERING

Institute of Developers Stack

**DETAILED COURSE OUTLINE**

**New Syllabus – 2025**

Developers Stack Master Program (DSMP)

**(INTAKE-6)**

**Hours: 1200** [120 UK], [48 ECTS], [80 SL], [26 U.S] **Medium: Sinhala**

**Institute of Developers Stack**
740 B Galle - Colombo Rd,
Panadura 12500, Sri Lanka
Tel: +94 714 911 257 | +94 701 469 965
Email: academic@developersstack.com | Web: www.developersstack.com



## Curriculum of the Diploma Program

| | |
|---|---|
| DSMP-M1501 - Module 1 | Software Development Foundation |
| DSMP-M1502 - Module 2 | Database Management System |
| DSMP-M1503 - Module 3 | Distributed application development |
| DSMP-M1504 - Module 4 | Enterprise application development & Testing |
| DSMP-M1505 - Module 5 | Mobile application development |
| DSMP-M1506 - Module 6 | Architectures & System Designing |
| DSMP-M1507 - Module 7 | System security & Performance Optimization |
| DSMP-M1508 - Module 8 | Final project Development |
| DSMP-M1509 - Module 9 | DevOps engineering, Deployment & Cloud computing |
| DSMP-M1510 - Module 10 | Interview preparation |

## 1. Introduction

This **Diploma in Software Engineering (DSMP)** is designed to equip students with the essential skills and knowledge required to thrive in the tech industry. The course offers an intensive, hands-on learning experience in the fields of software development, mobile programming, system architecture, and cloud technologies.

In addition to comprehensive training, the program offers an exclusive opportunity for the top 5 students who score the highest in assessments. These students will be connected with job opportunities at reputed local tech companies, providing a valuable chance to gain real-world industry experience and enhance their career prospects in the competitive tech market.

By the end of the program, students will have gained proficiency in various programming languages, frameworks, databases, and modern software development practices, preparing them to take on dynamic roles in the technology sector

## 2. Program Objectives

*What will students learn?*

Upon successful completion of the *Diploma in Software Engineering*, students will:

- Understand core software engineering principles, design patterns, and system architectures.
- Gain hands-on experience in developing **desktop**, **web**, **mobile**, and **enterprise** applications.
- Master key programming languages and tools including **Java, C#, JavaScript, Dart, Spring Boot, Node.js, .NET, Angular, React, Flutter,** and more.
- Learn modern software development practices such as **Agile**, **DevOps**, **CI/CD**, **Cloud Computing**, and **Containerization**.
- Understand and implement **secure coding practices**, system performance optimization, and industry-standard testing methodologies.
- Acquire practical experience through **real-world projects**, including full-stack systems and cloud-deployed solutions.
- Prepare for **AWS Certified Developer – Associate** and other recognized certifications.

*What skills will they develop?*

Students will develop both **technical and professional competencies**, including:

### Technical Skills:

- Full-stack development (frontend + backend)
- Mobile app development (cross-platform)
- RESTful API design & microservices architecture
- Database design, querying, and optimization (SQL + NoSQL)
- Secure application development and performance tuning
- Software testing and automation (unit, integration, and E2E)

- Cloud computing, CI/CD pipelines, containerization (Docker, Kubernetes)
- DevOps engineering & deployment strategies

### Professional Skills:

- Analytical thinking and problem-solving
- Project planning and management
- Business and system analysis
- Technical documentation and version control (GIT)
- Communication and teamwork in Agile environments
- Interview and career readiness

*What jobs or further studies will this course prepare them for?*

### Career Pathways:

- Software Developer / Engineer
- Full-Stack Developer
- Mobile App Developer
- Web Developer (Frontend/Backend)
- DevOps Engineer
- Database Administrator
- QA / Test Automation Engineer
- Cloud Application Engineer

## 3. Entry Requirements

To enroll in the *Diploma in Software Engineering* offered by **Institute of Developers Stack**, applicants must meet one of the following entry criteria:

### Minimum Educational Qualifications:

- Successfully completed **G.C.E. Advanced Level (A/L)** examination in any stream (Technology, Science, Commerce, or Arts).
- OR successfully completed **G.C.E. Ordinary Level (O/L)** with **a recognized foundation program** or equivalent IT-related qualification.

### Target Audience / Eligible Applicants:

This program is designed for:

- **School leavers** after A/Ls seeking a career in software engineering.
- **University undergraduates** pursuing degrees in IT, Computer Science, or Engineering.
- **Recent graduates** from non-IT backgrounds looking to enter the software industry.
- **Interns** and **trainees** in IT companies aiming to upskill for developer roles.
- **Associate Software Engineers (ASE)** looking for structured learning and certification.
- **Software Engineers (SE)** and **Senior Software Engineers (SSE)** who want to enhance their architecture, DevOps, or cloud computing skills.

### Additional Requirements:

- Basic computer literacy.
- A keen interest in software development and technology.
- Willingness to commit to **1,200 hours** of intensive practical and theoretical training.

## 4. Course Duration

### Total Course Hours:

- **1,200 hours** of guided learning and practical training.

### Program Duration:

- **12 months** (Full-Time)

The program includes lectures, practical sessions, project work, industry practices, and continuous assessments, designed to ensure a balance between theory and hands-on experience.

## 5. Delivery Mode

### Available Study Modes:

- **Full-Time** (Recommended)
- **Online** (Live Instructor-led)
- **Hybrid** (Combination of online & on-site sessions)

Students may choose the mode that best suits their personal, academic, or professional commitments. All delivery modes offer the same course content, assessments, and certification.

### Teaching & Learning Methods:

The program will be delivered using a combination of the following methods to ensure deep understanding and real-world application:

- 📖 **Lectures** – Theory sessions with concept explanations, demonstrations, and live coding.
- 💻 **Hands-on Labs** – Practical sessions in software development environments.
- 🛠️ **Mini Projects** – Modular projects to apply knowledge learned in each phase.
- 🎯 **Capstone Project** – Industry-grade final project integrating full-stack and cloud technologies.
- 🧠 **Workshops** – Special sessions on tools, frameworks, or current trends (DevOps, AI, Cloud).
- 📝 **Assessments** – Quizzes, assignments, code reviews, and presentations for each module.
- 👩‍🏫 **Mentoring** – One-on-one sessions and career guidance by industry professionals.
- 💼 **Internship Guidance** – (Optional) Support with industrial training or placement leads.

## 6. Course Content / Modules

The *Diploma in Software Engineering* consists of 10 comprehensive modules totaling **1,200 learning hours**. Each module is designed to build foundational and advanced skills in software development, system design, and industry deployment.

### Credit Breakdown for Each Module

| Module Name | Hours | SL Credits | ECTS Credits | U.S. Credits | UK Credits |
|---|---|---|---|---|---|
| Software Development Foundation | 120 | 8 | 4.8 | 2.6 | 12 |
| Database Management System | 120 | 8 | 4.8 | 2.6 | 12 |
| Distributed Application Development | 120 | 8 | 4.8 | 2.6 | 12 |
| Enterprise Application Development & Testing | 120 | 8 | 4.8 | 2.6 | 12 |
| Mobile Application Development | 120 | 8 | 4.8 | 2.6 | 12 |
| Architectures & System Designing | 120 | 8 | 4.8 | 2.6 | 12 |
| System Security & Performance Optimization | 120 | 8 | 4.8 | 2.6 | 12 |
| Final Project Development | 120 | 8 | 4.8 | 2.6 | 12 |
| DevOps Engineering, Deployment & Cloud Computing | 120 | 8 | 4.8 | 2.6 | 12 |
| Interview Preparation | 120 | 8 | 4.8 | 2.6 | 12 |
| **Total** | 1200 | 80 SLC | 48 ECTS | 26 U. S | 120 UK |

$$\text{UK Credits} = \frac{\text{Total Learning Hours}}{10}$$

$$\text{SL Credits} = \frac{\text{Total Learning Hours}}{15}$$

$$\text{ECTS Credits} = \frac{\text{Total Learning Hours}}{25}$$

$$\text{U.S. Credits} = \frac{\text{Total Learning Hours}}{45}$$

**Assessments Criteria**

| Module | Assignments | Exams | Projects | Viva Sessions | Interviews | Weight |
|---|---|---|---|---|---|---|
| ***Theory*** | | | | | | ***300H*** |
| ***Practical*** | | | | | | ***300H*** |
| Software Development Foundation | 5 | 1 | 1 | 1 | 0 | 18.63% |
| Database Management System | 3 | 1 | 1 | 1 | 0 | 13.56% |
| Distributed Application Development | 3 | 1 | 2 | 1 | 0 | 13.56% |
| Enterprise Application Development & Testing | 2 | 1 | 1 | 1 | 0 | 13.56% |
| Mobile Application Development | 2 | 1 | 1 | 1 | 0 | 12.33% |
| Architectures & System Designing | 0 | 0 | 1 | 1 | 0 | 8.63% |
| System Security & Performance Optimization | 0 | 0 | 1 | 1 | 0 | 8.63% |
| Final Project Development | 0 | 0 | 1 | 1 | 0 | 2.47% |
| DevOps Engineering, Deployment & Cloud Computing | 0 | 1 | 0 | 1 | 0 | 8.63% |
| Interview Preparation | 0 | 0 | 0 | 0 | 3 | - |
| Attendance | | | | | | 80% |

**DSMP-M1501 - Module 1**
(Software Development Foundation)

| Stage Index | Stage Content | Hours | SL Credits | ECTS credits | U.S credits | UK credits |
|---|---|---|---|---|---|---|
| 1 | IntelliJ Idea | 65 | 4.3 | 2.17 | 1.4 | 6.5 |
| | Java (V: 8,11,17,21) | | | | | |
| | JavaFX | | | | | |
| | Scene Builder | | | | | |
| 2 | UI/UX | 15 | 1 | 0.5 | 0.33 | 1.5 |
| | Software Engineering | | | | | |
| 3 | Business Analysis | 16 | 1.07 | 0.53 | 0.36 | 1.6 |
| | Project Management | | | | | |
| 4 | C# | 12 | 0.8 | 0.4 | 0.27 | 1.2 |
| 5 | GIT/GIT-Hub | 12 | 0.8 | 0.4 | 0.27 | 1.2 |
| | | **120** | **8** | **4.8** | **2.6** | **12** |

**Learning Outcomes for Module 1**

1. Demonstrate proficiency in using **IntelliJ IDEA** for Java development.
2. Develop Java applications using multiple versions of **Java** (8, 11, 17, 21).
3. Create interactive applications using **JavaFX** and implement UIs with **Scene Builder**.
4. Design and implement user-friendly **UI/UX** for software applications.
5. Apply **Software Engineering** principles to Java-based projects.
6. Conduct **Business Analysis** and define requirements for software projects.

7. Utilize **Project Management** tools to plan, track, and manage software development projects.
8. Write, test, and debug **C#** applications to implement programming concepts.
9. Use **Git** for version control and manage code repositories on **GitHub**.
10. Collaborate effectively on software projects using **GitHub** for code sharing and version management.

**DSMP-M1502 - Module 2**
(Database Management System)

| Stage Index | Stage Content | Hours | SL Credits | ECTS credits | U.S credits | UK credits |
|---|---|---|---|---|---|---|
| 1 | MySQL | 50 | 3.3 | 1.67 | 1.11 | 5 |
| | DBMS | | | | | |
| | PostgreSQL | | | | | |
| | SQL Server | | | | | |
| | Heidi SQL | | | | | |
| 2 | MongoDB | 20 | 1.3 | 0.67 | 0.44 | 2 |
| 3 | Design Patterns | 50 | 3.3 | 1.67 | 1.11 | 5 |
| | MVC, Layard Architecture Final Project | | | | | |
| | Jasper Report | | | | | |
| | | **120** | **8** | **4.8** | **2.6** | **12** |

## Learning Outcomes for Module 2

1. Demonstrate proficiency in using **MySQL** for database management and query optimization.
2. Understand the fundamentals and principles of **Database Management Systems (DBMS)**.
3. Design and manage databases using **PostgreSQL** and implement advanced features.
4. Utilize **SQL Server** for database management and query optimization.
5. Work effectively with **HeidiSQL** for database design and querying.
6. Understand and implement **MongoDB** for NoSQL database management.
7. Apply **Design Patterns** (MVC, Layered Architecture) in database-centric applications.
8. Develop and deploy a **final project** using **Database Management Systems**.
9. Generate reports using **Jasper Reports** for data presentation and visualization.

**DSMP-M1503 - Module 3**
(Distributed application development)

| Stage Index | Stage Content | Hours | SL Credits | ECTS credits | U.S credits | UK credits |
|---|---|---|---|---|---|---|
| 1 | HTML | 60 | 4 | 2.4 | 1.3 | 6 |
| | CSS | | | | | |
| | Final App (Responsive) | | | | | |
| | SASS | | | | | |
| | Bootstrap | | | | | |
| | Semantic UI | | | | | |
| | Tailwind CSS | | | | | |
| | Final App | | | | | |
| 2 | Java Script | 40 | 2.67 | 1.6 | 0.89 | 4 |
| | J-Query/J-Query UI | | | | | |
| | Three JS | | | | | |
| | Final App (POS) | | | | | |
| | Deployment | | | | | |
| 3 | SEO | 20 | 1.33 | 0.8 | 0.44 | 2 |

| | | | | | |
|---|---|---|---|---|---|
| | **120** | **8** | **4.8** | **2.6** | **12** |

## Learning Outcomes for Module 3

1. Demonstrate proficiency in **HTML** for building and structuring web pages.
2. Apply **CSS** to design and style responsive web pages.
3. Develop a **responsive web application** using modern web technologies.
4. Implement **SASS** for efficient and maintainable CSS styling.
5. Utilize **Bootstrap**, **Semantic UI**, and **Tailwind CSS** to create modern, responsive user interfaces.
6. Develop a fully functional **Final Application** using front-end frameworks.
7. Implement interactive features and client-side logic using **JavaScript**.
8. Use **JQuery/JQuery UI** to enhance user interface and interactivity.
9. Integrate **Three.js** for creating 3D graphics and animations on the web.
10. Develop a **Final Application (POS)** using JavaScript and relevant frameworks.
11. Deploy applications and manage their hosting effectively.
12. Apply **SEO** (Search Engine Optimization) techniques to enhance the visibility of web applications.

**DSMP-M1504 - Module 4**

(Enterprise application development & Testing)

| Stage Index | Stage Content | Hours | SL Credits | ECTS credits | U.S credits | UK credits |
|---|---|---|---|---|---|---|
| 1 | Maven | 5 | 1 | 0.6 | 0.33 | 1.5 |
| | JSP && EE (Servlet) | | | | | |
| | Final App | | | | | |
| 2 | Spring | 20 | 1.33 | 0.8 | 0.44 | 2 |
| | Spring Boot | | | | | |
| | Security | | | | | |
| | Final Project | | | | | |
| 3 | Node Js | 20 | 1.33 | 0.8 | 0.44 | 2 |
| | Express JS | | | | | |
| | Passport JS | | | | | |
| | Loopback | | | | | |
| | Prisma | | | | | |
| | Nest JS | | | | | |
| | Final Project | | | | | |
| 4 | .NET | 20 | 1.33 | 0.8 | 0.44 | 2 |
| | Razor Page | | | | | |
| | Final Project | | | | | |
| 5 | Type Script | 20 | 1.33 | 0.8 | 0.44 | 2 |
| | Angular | | | | | |
| | Prime Ng | | | | | |
| | NGRX | | | | | |
| | Firebase | | | | | |
| | Final Project | | | | | |
| 6 | React | 20 | 1.33 | 0.8 | 0.44 | 2 |
| | Material UI | | | | | |
| | Axios | | | | | |
| | Redux | | | | | |
| | Final App | | | | | |
| | Vue JS | 15 | 0.33 | 0.2 | 0.11 | 0.5 |
| | Vuetify & Final Project | | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | **120** | **8** | **4.8** | **2.6** | **12** |

## Learning Outcomes for Module 4

1. Utilize **Maven** for project management and automation in enterprise application development.
2. Develop web applications using **JSP** (Java Server Pages) and **EE (Servlets)**.
3. Create and deploy a **Final Application** using **JSP/Servlets**.
4. Implement and integrate **Spring Framework** for enterprise-level application development.
5. Use **Spring Boot** to develop microservices-based applications.
6. Apply **Security** techniques and best practices in enterprise application development (e.g., authentication and authorization).
7. Complete a **Final Project** demonstrating skills in **Spring** and **Spring Boot**.
8. Develop server-side applications using **Node.js** and integrate with **Express.js**.
9. Implement user authentication with **Passport.js** in Node.js applications.
10. Utilize **Loopback** and **Prisma** for backend development and database management.
11. Create applications using **NestJS**, a Node.js framework for building scalable and maintainable enterprise applications.
12. Complete a **Final Project** using **Node.js**, **Express**, **Loopback**, and **NestJS**.
13. Develop web applications with **.NET Framework** using **Razor Pages**.
14. Implement and deploy **Final Projects** using **.NET** technologies.
15. Develop modern web applications using **TypeScript** and **Angular**.
16. Utilize **PrimeNG**, **NGRX**, and **Firebase** to build rich, dynamic front-end applications.
17. Complete a **Final Project** incorporating **Angular**, **PrimeNG**, and **NGRX**.
18. Develop web applications using **React**, **Material UI**, **Axios**, and **Redux**.
19. Build responsive and high-performance web applications using **Vue.js** and **Vuetify**.
20. Complete a **Final Project** using **React**, **Redux**, and **Vue.js**.

**DSMP-M1505 - Module 5**
(Mobile application development)

| Stage Index | Stage Content | Hours | SL Credits | ECTS credits | U.S credits | UK credits |
|---|---|---|---|---|---|---|
| 1 | React Native | 60 | 4 | 2.4 | 1.3 | 6 |
| | Native Base | | | | | |
| | Final Project | | | | | |
| 2 | Dart | 60 | 4 | 2.4 | 1.3 | 6 |
| | Flutter | | | | | |
| | Flutter Gems | | | | | |
| | Final Project | | | | | |
| | | **120** | **8** | **4.8** | **2.6** | **12** |

## Learning Outcomes for Module 5

1. Develop mobile applications using **React Native** for cross-platform app development.
2. Utilize **Native Base** to build responsive and native-like mobile apps using React Native.
3. Create and deploy a **Final Project** demonstrating proficiency in **React Native** and **Native Base**.
4. Develop mobile applications using **Dart** programming language.
5. Utilize **Flutter** for building high-performance, cross-platform mobile applications.
6. Integrate **Flutter Gems** to extend the functionality of Flutter apps.
7. Complete a **Final Project** demonstrating the skills in **Dart**, **Flutter**, and integrating Flutter packages.

**DSMP-M1506 - Module 6**
(Architectures & System Designing)

| Stage Index | Stage Content | Hours | SL Credits | ECTS credits | U.S credits | UK credits |
|---|---|---|---|---|---|---|
| 1 | What is Software Architecture? | 40 | 2.67 | 1.6 | 0.89 | 4 |
| | Difference between Architecture & Design | | | | | |
| | Monolithic vs. Microservices Architecture | | | | | |
| | Software Quality Attributes (Scalability, Maintainability, Performance, Security, etc.) | | | | | |
| | Architectural Trade-offs & Decision Making | | | | | |
| | Design an E-commerce system with a monolithic architecture. | | | | | |
| | Redesign the same system using a microservices approach. | | | | | |
| 2 | Creational Patterns (Singleton, Factory, Builder) | 40 | 2.67 | 1.6 | 0.89 | 4 |
| | Structural Patterns (Adapter, Composite, Decorator) | | | | | |
| | Behavioral Patterns (Observer, Strategy, Command) | | | | | |
| | Event-Driven Architecture | | | | | |
| | Domain-Driven Design (DDD) | | | | | |
| | Implement a modular e-learning platform using design patterns. | | | | | |
| | Build a real-time chat system using event-driven architecture. | | | | | |
| 3 | High-Level & Low-Level Design (HLD & LLD) | 20 | 1.33 | 0.8 | 0.44 | 2 |
| | Load Balancing, Caching & CDNs | | | | | |
| | Database Sharding & Replication | | | | | |
| | Distributed Systems & CAP Theorem | | | | | |
| | Message Queues (RabbitMQ, Kafka) | | | | | |
| | Design a ride-sharing system (Uber-like) with scaling strategies. | | | | | |
| | Build a food delivery service handling high traffic. | | | | | |
| 4 | Netflix (Microservices, Chaos Engineering) | 20 | 1.33 | 0.8 | 0.44 | 2 |
| | Uber (Scalability & Real-time Architecture) | | | | | |
| | WhatsApp (Message Queue & Database Replication) | | | | | |
| | Amazon (Event-driven E-commerce) | | | | | |
| | Redesign an existing legacy application with a modern architecture. | | | | | |
| | | **120** | **8** | **4.8** | **2.6** | **12** |

## Learning Outcomes for Module 6

1. Understand the fundamental concepts of **Software Architecture** and distinguish it from **Software Design**.
2. Compare and contrast **Monolithic Architecture** with **Microservices Architecture**.
3. Identify and apply key **Software Quality Attributes** such as **Scalability**, **Maintainability**, **Performance**, and **Security** in system design.
4. Make informed **Architectural Trade-offs** and decisions in the context of system design.
5. Design an **E-commerce System** using **Monolithic Architecture** and redesign it using a **Microservices Approach**.

6. Apply **Creational Design Patterns** such as **Singleton**, **Factory**, and **Builder** to enhance system modularity.
7. Implement **Structural Design Patterns** like **Adapter**, **Composite**, and **Decorator** in software applications.
8. Utilize **Behavioral Design Patterns** including **Observer**, **Strategy**, and **Command** for effective communication and behavior in systems.
9. Apply **Event-Driven Architecture** for scalable and responsive system design.
10. Implement **Domain-Driven Design (DDD)** to design software systems that reflect real-world domains.
11. Build a **modular e-learning platform** using various **Design Patterns**.
12. Develop a **real-time chat system** using **Event-Driven Architecture**.
13. Apply **High-Level Design (HLD)** and **Low-Level Design (LLD)** principles to structure and detail system designs.
14. Implement **Load Balancing**, **Caching**, and **Content Delivery Networks (CDNs)** for high performance and scalability.
15. Understand and apply **Database Sharding** and **Replication** strategies to optimize database performance.
16. Design and implement **Distributed Systems** considering the **CAP Theorem**.
17. Utilize **Message Queues** such as **RabbitMQ** and **Kafka** to design asynchronous, reliable systems.
18. Design a **ride-sharing system (Uber-like)** incorporating **scaling strategies** for high traffic handling.
19. Build a **high-traffic food delivery service** system with an emphasis on performance and reliability.
20. Analyze and redesign an existing **legacy application** with a modern architecture approach, such as **Microservices** or **Event-Driven Architecture**.
21. Understand and apply concepts from **Netflix**'s **Microservices** and **Chaos Engineering**, **Uber**'s scalability strategies, **WhatsApp's** message queues, and **Amazon's event-driven e-commerce architecture**.

**DSMP-M1507 - Module 7**

(System security & Performance Optimization)

| Stage Index | Stage Content | Hours | SL Credits | ECTS credits | U.S credits | UK credits |
|---|---|---|---|---|---|---|
| 1 | CIA Triad (Confidentiality, Integrity, Availability) | 120 | 8 | 4.8 | 2.6 | 12 |
| | Defense in Depth | | | | | |
| | Zero Trust Security Model | | | | | |
| | Security vs. Performance Trade-offs | | | | | |
| | Common Security Threats & Attacks | | | | | |
| | OWASP Top 10 Vulnerabilities (SQL Injection, XSS, CSRF, etc.) | | | | | |
| | Man-in-the-Middle (MITM) Attacks | | | | | |
| | Denial-of-Service (DoS & DDoS) Attacks | | | | | |
| | Ransomware & Malware | | | | | |
| | Implement role-based access control (RBAC) in a web application | | | | | |
| | | **120** | **8** | **4.8** | **2.6** | **12** |

## Learning Outcomes for Module 7

1. Understand and apply the principles of the **CIA Triad** (Confidentiality, Integrity, Availability) in securing systems.
2. Implement a **Defense in Depth** strategy to secure systems through layered security controls.
3. Understand and apply the **Zero Trust Security Model** for improved security posture.
4. Evaluate and balance **Security vs. Performance** trade-offs in system design and optimization.
5. Identify and mitigate common **Security Threats** and **Attacks**, including SQL Injection, Cross-Site Scripting (XSS), Cross-Site Request Forgery (CSRF), and others.
6. Apply countermeasures to **Man-in-the-Middle (MITM)** attacks and prevent unauthorized interception of communications.
7. Prevent and defend against **Denial-of-Service (DoS)** and **Distributed Denial-of-Service (DDoS)** attacks.

8. Implement strategies to defend against **Ransomware** and **Malware** attacks and minimize their impact.
9. Implement **Role-Based Access Control (RBAC)** in a web application to restrict access based on user roles and permissions.
10. Recognize the top **OWASP Vulnerabilities** and learn how to defend against them, including **SQL Injection**, **XSS**, **CSRF**, etc.
11. Optimize system performance while ensuring security compliance and reducing risks from vulnerabilities.

**DSMP-M1508 - Module 8**
(Final project Development)

| Stage Index | Stage Content | Hours | SL Credits | ECTS credits | U.S credits | UK credits |
|---|---|---|---|---|---|---|
| 1 | Multi-vendor e-commerce system with user authentication, order management, payment gateway, and inventory control. | 120 | 8 | 4.8 | 2.6 | 12 |
| | Security-first & performance-optimized system. | | | | | |
| | Frontend: Angular/React (Optimized UI & lazy loading) | | | | | |
| | Backend: Node.js (NestJS) / Spring Boot | | | | | |
| | Database: PostgreSQL / MongoDB (Sharding & Indexing) | | | | | |
| | Authentication: OAuth2 + JWT + MFA | | | | | |
| | Caching & Optimization: Redis + Cloudflare CDN | | | | | |
| | Message Queue: Kafka / RabbitMQ | | | | | |
| | Cloud Deployment: AWS/GCP/Azure | | | | | |
| | CI/CD: Jenkins/GitHub Actions | | | | | |
| | Microservices-based backend architecture | | | | | |
| | Event-driven design (Using Kafka or RabbitMQ) | | | | | |
| | Secure API Gateway for routing & rate limiting | | | | | |
| | Global CDN & Load Balancer for fast content delivery | | | | | |
| | | **120** | **8** | **4.8** | **2.6** | **12** |

## Learning Outcomes for Module 8: Final Project Development

1. **Design and Implement a Multi-Vendor E-Commerce System**

- Develop a secure and high-performance multi-vendor e-commerce platform that includes user authentication, order management, payment gateway integration, and inventory control.
- Ensure the system is security-first and optimized for performance with proper error handling, data validation, and scalability considerations.

2. **Frontend Development Using Angular/React**

- Build the frontend of the system using **Angular** or **React**, applying advanced techniques like **UI optimization** and **lazy loading** to enhance user experience and system performance.
- Implement a responsive design suitable for both desktop and mobile devices.

3. **Backend Development with Node.js or Spring Boot**

    - Develop the backend services using **Node.js (NestJS)** or **Spring Boot**, implementing the necessary business logic and APIs for handling system operations such as user registration, order processing, and payment transactions.
    - Implement **microservices-based backend architecture** for better scalability and maintainability.

4. **Database Design and Optimization**

    - Use **PostgreSQL** or **MongoDB** for efficient database management, applying techniques like **sharding** and **indexing** to optimize performance for large-scale operations and fast data retrieval.

5. **User Authentication and Authorization**

    - Integrate **OAuth2**, **JWT**, and **Multi-Factor Authentication (MFA)** to ensure secure user authentication and authorization across the platform, providing protection against unauthorized access.

6. **Caching and Content Delivery Optimization**

    - Use **Redis** for caching to minimize database load and improve response time.
    - Leverage **Cloudflare CDN** for fast content delivery globally, ensuring a seamless user experience.

7. **Message Queuing with Kafka/RabbitMQ**

    - Implement **Kafka** or **RabbitMQ** for handling asynchronous events, such as order notifications and inventory updates, improving system performance and scalability.

8. **Cloud Deployment**

    - Deploy the system to popular cloud platforms like **AWS**, **Google Cloud Platform (GCP)**, or **Azure**, ensuring high availability, fault tolerance, and the ability to scale the system as needed.
    - Use cloud services for database, compute, and storage management, integrating them with the application for seamless performance.

9. **Continuous Integration/Continuous Deployment (CI/CD)**

    - Set up and automate **CI/CD pipelines** using tools such as **Jenkins** or **GitHub Actions**, enabling rapid deployment, version control, and automated testing.

10. **Event-Driven Architecture**

    - Implement **event-driven design** using **Kafka** or **RabbitMQ** to enable decoupled and scalable communication between microservices within the system.

11. **API Gateway and Rate Limiting**

    - Build a **secure API Gateway** for routing requests and ensuring rate limiting, preventing abuse and ensuring the system handles traffic efficiently.

12. **Global Content Delivery and Load Balancing**

- Set up **Global CDN** and **load balancers** to ensure content is delivered quickly to users worldwide, improving overall system performance and availability.

**DSMP-M1509 - Module 9**

(DevOps engineering, Deployment & Cloud computing)

| Stage Index | Stage Content | Hours | SL Credits | ECTS credits | U.S credits | UK credits |
|---|---|---|---|---|---|---|
| 1 | **AWS Cloud Computing & AWS Certified Developer – Associate Training** | 60 | 4 | 2.4 | 1.3 | 6 |
| | ***AWS Fundamentals & Core Services*** | | | | | |
| | Compute: EC2, Lambda, Elastic Beanstalk | | | | | |
| | Storage: S3, EFS, Glacier | | | | | |
| | Database: RDS, DynamoDB, Redshift | | | | | |
| | Networking: VPC, Route 53, CloudFront, ELB | | | | | |
| | IAM & Security: IAM Roles, KMS, Secrets Manager, WAF | | | | | |
| | Monitoring & Logging: CloudWatch, CloudTrail, AWS Config | | | | | |
| | Deployment & CI/CD: CodePipeline, CodeDeploy, CodeBuild | | | | | |
| | Serverless: API Gateway, Lambda, Step Functions | | | | | |
| 2 | ***AWS Exam Topics & Preparation*** | 20 | 1.33 | 0.8 | 0.44 | 2 |
| | IAM & Security Best Practices | | | | | |
| | EC2 & Load Balancing (Auto Scaling, ALB, NLB) | | | | | |
| | Databases & Data Warehousing (DynamoDB, RDS, Redshift) | | | | | |
| | Event-Driven & Serverless Architecture (SNS, SQS, Lambda) | | | | | |
| | Monitoring & Optimization (CloudWatch, CloudTrail, Cost Optimization) | | | | | |
| | Deploying Applications (Elastic Beanstalk, ECS, EKS, CodePipeline, CloudFormation) | | | | | |
| | Networking (VPC, Route 53, NAT, VPN) | | | | | |
| 3 | **DevOps Engineering – Tools, Techniques & Deployment** | 20 | 1.33 | 0.8 | 0.44 | 2 |
| | ***Version Control & Collaboration*** | | | | | |
| | Git (GitHub, GitLab, Bitbucket) | | | | | |
| | GitFlow, Branching Strategies | | | | | |
| | ***CI/CD Pipelines*** | | | | | |
| | Jenkins (Pipeline Automation) | | | | | |
| | GitHub Actions | | | | | |
| | GitLab CI/CD | | | | | |
| | CircleCI, Travis CI | | | | | |
| | ArgoCD (Kubernetes CI/CD) | | | | | |
| | ***Configuration Management & Automation*** | | | | | |
| | Ansible (Infrastructure as Code - IaC) | | | | | |
| | Terraform (Cloud Resource Management) | | | | | |
| | Puppet & Chef (Configuration Management) | | | | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | ***Containers & Orchestration*** | | | | | |
| | Docker (Containerization) | | | | | |
| | Kubernetes (K8s) (Container Orchestration) | | | | | |
| | Helm (K8s Package Management) | | | | | |
| | OpenShift, Rancher (Enterprise Kubernetes) | | | | | |
| | ***Infrastructure Monitoring & Logging*** | | | | | |
| | Prometheus & Grafana (Real-time Monitoring) | | | | | |
| | ELK Stack (Elasticsearch, Logstash, Kibana) | | | | | |
| | New Relic, Datadog (Application Performance Monitoring) | | | | | |
| | Cloud Logging (AWS CloudWatch, GCP Stackdriver, Azure Monitor) | | | | | |
| | ***Security & Compliance*** | | | | | |
| | HashiCorp Vault (Secrets Management) | | | | | |
| | AWS Security Hub, IAM, GuardDuty | | | | | |
| | OWASP ZAP (Vulnerability Scanning) | | | | | |
| | SonarQube (Code Quality & Security Analysis) | | | | | |
| 4 | **Deployment & Cloud Infrastructure Management** | 20 | 1.33 | 0.8 | 0.44 | 2 |
| | ***AWS Deployment & Scaling*** | | | | | |
| | Elastic Beanstalk (Automated Deployment) | | | | | |
| | ECS/EKS (Containerized Deployment) | | | | | |
| | EC2 & Auto Scaling (Instance-based Deployment) | | | | | |
| | AWS Lambda & API Gateway (Serverless Deployment) | | | | | |
| | CloudFormation & Terraform (Infrastructure as Code) | | | | | |
| | AWS CodePipeline + CodeDeploy (CI/CD Pipeline Deployment) | | | | | |
| | ***VPS Deployment*** | | | | | |
| | Linux-based VPS (Ubuntu, CentOS, Debian) Setup | | | | | |
| | Nginx & Apache Web Server Configuration | | | | | |
| | SSL Certificates & Security Hardening | | | | | |
| | Database Hosting (MySQL, PostgreSQL, MongoDB on VPS) | | | | | |
| | Backup & Disaster Recovery Strategies | | | | | |
| | ***Mobile App Deployment*** | | | | | |
| | Google Play Console (Android APK & AAB Deployment) | | | | | |
| | Apple App Store (iOS Deployment via Xcode & TestFlight) | | | | | |
| | Firebase App Distribution (Beta Testing & CI/CD) | | | | | |
| | Push Notification Setup (FCM, APNs) | | | | | |
| | ***Domain & Hosting Setup*** | | | | | |
| | GoDaddy, Namecheap, AWS Route 53 (Domain Management) | | | | | |
| | Cloudflare (CDN & DNS Security) | | | | | |
| | SSL/TLS Certificate Installation | | | | | |
| | Static Site Hosting (S3, Netlify, Vercel, GitHub Pages) | | | | | |
| | | **120** | **8** | **4.8** | **2.6** | **12** |

## Learning Outcomes for Module 9

1. **AWS Cloud Computing and Core Services**:

   - Understand and implement core AWS services such as **EC2**, **Lambda**, **Elastic Beanstalk**, **S3**, **EFS**, **Glacier**, **RDS**, **DynamoDB**, **Redshift**, and **VPC** for compute, storage, and networking purposes.
   - Develop and manage **serverless architectures** using **API Gateway**, **Lambda**, and **Step Functions**.
   - Learn about **IAM roles** and **security management** using AWS services like **KMS**, **Secrets Manager**, and **WAF** to ensure secure access to resources.
   - Utilize **AWS monitoring tools** like **CloudWatch**, **CloudTrail**, and **AWS Config** to track system performance and security events.

2. **AWS Certification Preparation**:

   - Prepare for the **AWS Certified Developer – Associate** exam by covering advanced topics like **IAM**, **EC2**, **Load Balancing**, **Event-Driven & Serverless Architecture**, and **Database Management**.
   - Gain knowledge in **cost optimization**, **networking**, and **deploying applications** using tools such as **Elastic Beanstalk**, **ECS**, **EKS**, **CodePipeline**, and **CloudFormation**.

3. **DevOps Engineering – Tools and Techniques**:

   - Understand and apply **Version Control** with **Git** (GitHub, GitLab, Bitbucket), **GitFlow**, and branching strategies for efficient collaboration.
   - Set up and manage **CI/CD Pipelines** using tools like **Jenkins**, **GitHub Actions**, **GitLab CI/CD**, **CircleCI**, **Travis CI**, and **ArgoCD**.
   - Automate infrastructure and configuration management using **Ansible**, **Terraform**, **Puppet**, and **Chef** to enable **Infrastructure as Code** (IaC).
   - Gain hands-on experience in **containerization** with **Docker** and **Kubernetes (K8s)** for orchestration and **Helm** for package management.
   - Utilize **Prometheus** and **Grafana** for **real-time monitoring**, and manage logs with **ELK Stack** (Elasticsearch, Logstash, Kibana) and **Datadog** for **Application Performance Monitoring**.
   - Implement **security and compliance** practices using tools like **HashiCorp Vault**, **AWS Security Hub**, **GuardDuty**, **OWASP ZAP**, and **SonarQube** for security scanning and vulnerability management.

4. **Deployment and Cloud Infrastructure Management**:

   - Implement **AWS deployment and scaling** solutions using **Elastic Beanstalk**, **ECS/EKS**, **EC2**, and **AWS Lambda** for serverless applications.
   - Utilize **CloudFormation** and **Terraform** for **Infrastructure as Code** and automate the deployment of cloud resources.
   - Set up and deploy applications using **VPS** on **Linux-based systems** (Ubuntu, CentOS, Debian), configure **Nginx** and **Apache web servers**, and manage **SSL certificates** for secure communication.
   - Implement **backup** and **disaster recovery strategies** to ensure high availability and data safety.

5. **Mobile App Deployment**:

- Deploy mobile applications on the **Google Play Console** for **Android** and **Apple App Store** for **iOS** using **Xcode** and **TestFlight** for beta testing.
- Set up **Firebase App Distribution** for continuous integration and continuous deployment (CI/CD) in mobile app development.
- Implement **push notifications** using **FCM (Firebase Cloud Messaging)** and **APNs (Apple Push Notification Service)**.

6. **Domain and Hosting Setup**:

- Manage **domains** with services like **GoDaddy**, **Namecheap**, and **AWS Route 53**, and configure **DNS** settings using **Cloudflare** for enhanced security and performance.
- Set up **SSL/TLS certificates** and ensure secure communication across the platform.
- Host **static sites** using platforms like **AWS S3**, **Netlify**, **Vercel**, and **GitHub Pages** for quick deployment.

**DSMP-M1510 - Module 10**
(Interview preparation)

| Stage Index | Stage Content | Hours | SL Credits | ECTS credits | U.S credits | UK credits |
|---|---|---|---|---|---|---|
| 1 | ***Resume & Portfolio Preparation*** | 20 | 1.33 | 0.8 | 0.44 | 2 |
| | Optimized Resume for ATS (Applicant Tracking Systems) | | | | | |
| | Building a Strong GitHub Profile & Contributions | | | | | |
| | Portfolio Website with Real-World Projects | | | | | |
| | LinkedIn Optimization & Professional Networking | | | | | |
| 2 | ***Common Interview Rounds & How to Prepare*** | 40 | 2.67 | 1.6 | 0.89 | 4 |
| | *Round 1: Screening & HR Round (Soft Skills)* | | | | | |
| | Self-introduction & Career Goals | | | | | |
| | Behavioral Questions (STAR Method) | | | | | |
| | Expected Salary & Company Fit | | | | | |
| | *Round 2: Technical Assessment (Hands-on & Theory)* | | | | | |
| | Coding Challenges (LeetCode, HackerRank, CodeSignal) | | | | | |
| | Data Structures & Algorithms (DSA) | | | | | |
| | System Design Scenarios (Scalable & Secure Architecture) | | | | | |
| | AWS & DevOps Case Studies (Infrastructure Setup, Security, CI/CD, Deployment) | | | | | |
| | *Round 3: Advanced DevOps & Cloud Questions* | | | | | |
| | AWS Best Practices (Security, IAM, Networking, Performance Optimization) | | | | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | Infrastructure as Code (Terraform, CloudFormation) | | | | | |
| | Kubernetes & Docker Orchestration | | | | | |
| | CI/CD & Automation (Jenkins, GitHub Actions, AWS CodePipeline) | | | | | |
| | Monitoring & Logging (ELK, Prometheus, AWS CloudWatch) | | | | | |
| | *Round 4: Problem-Solving & System Design* | | | | | |
| | Designing a Scalable E-Commerce Platform | | | | | |
| | Optimizing a High-Traffic Microservices System | | | | | |
| | Handling Server Downtime & Disaster Recovery | | | | | |
| | *Round 5: Offer Negotiation & Career Growth* | | | | | |
| | Salary & Benefits Negotiation | | | | | |
| | Work-Life Balance & Remote Work Options | | | | | |
| | Long-Term Career Roadmap in DevOps & Cloud Engineering | | | | | |
| 3 | **Mock Interviews & Real-World Case Studies** | 40 | 2.67 | 1.6 | 0.89 | 4 |
| | Live Mock Interviews (Technical & Behavioral) | | | | | |
| | Whiteboard System Design & Real-World Case Studies | | | | | |
| | Reviewing Past Interview Questions from FAANG, Startups & Enterprises | | | | | |
| | Industry-Specific Interview Questions (FinTech, E-Commerce, SaaS, AI, etc.) | | | | | |
| 4 | Final Deliverables | 20 | 1.33 | 0.8 | 0.44 | 2 |
| | Updated Resume & Portfolio | | | | | |
| | GitHub Profile with Real-World DevOps & AWS Projects | | | | | |
| | Interview Strategy & Mock Interview Experience | | | | | |
| | Personalized Career Roadmap & Growth Plan | | | | | |
| | | **120** | **8** | **4.8** | **2.6** | **12** |

## Learning Outcomes for Module 10

1. **Resume & Portfolio Preparation**

- Prepare an **optimized resume** for **ATS (Applicant Tracking Systems)** ensuring your qualifications are easily discoverable by recruiters.
- Build a **strong GitHub profile** with active contributions that demonstrate your coding and development expertise.
- Develop a **portfolio website** that highlights **real-world projects** to showcase your technical skills and practical experience.
- Optimize your **LinkedIn profile** for **professional networking** and career growth, ensuring it reflects your technical abilities and career aspirations.

2. **Common Interview Rounds & How to Prepare**

- Learn how to approach **Screening & HR rounds**, focusing on **soft skills**, self-introduction, career goals, and company fit.
- Master the **STAR method** for answering **behavioral questions**, improving your communication and self-presentation during interviews.
- Prepare for **Technical Assessment rounds** with focus areas like **coding challenges**, **data structures and algorithms (DSA)**, and **system design scenarios** (scalable and secure architecture).
- Gain familiarity with **AWS & DevOps case studies** related to **infrastructure setup**, **security**, **CI/CD**, and **deployment** in technical interviews.
- Master **advanced DevOps & cloud questions**, focusing on **AWS best practices**, **Kubernetes & Docker orchestration**, **CI/CD & automation**, and **monitoring & logging**.
- Learn how to approach **problem-solving and system design questions**, such as designing scalable platforms, optimizing microservices systems, and handling server downtime and disaster recovery.
- Understand how to negotiate offers and plan for **career growth**, including **salary and benefits negotiations**, work-life balance, and long-term career roadmap in **DevOps & Cloud Engineering**.

3. **Mock Interviews & Real-World Case Studies**

- Participate in **live mock interviews**, covering both **technical** and **behavioral** aspects of the interview process.
- Practice **whiteboard system design** exercises and real-world case studies, simulating the conditions of actual interviews.
- Review past **interview questions** from top tech companies (FAANG) and enterprises, as well as industry-specific questions (e.g., **FinTech**, **E-Commerce**, **SaaS**, **AI**).
- Gain exposure to **industry-specific interview questions**, tailoring your interview preparation to various sectors.

4. **Final Deliverables**

- Submit an **updated resume and portfolio**, with a **GitHub profile** showcasing **real-world DevOps & AWS projects**.
- Develop an **interview strategy** and gain experience from mock interviews to refine your interview performance.
- Create a **personalized career roadmap** and **growth plan**, outlining steps for long-term success in the **DevOps** and **cloud engineering** fields.

## Course Content

### Soft Skills & Employability

1. **Communication & Teamwork**

- Develop effective **communication skills** for both technical and non-technical audiences.
- Practice **team collaboration**, including how to contribute to discussions, give and receive constructive feedback, and work towards shared goals.
- Improve **cross-functional communication**, ensuring clear and concise delivery of technical information to diverse teams (developers, product managers, designers, etc.).

- Cultivate **active listening** skills to improve understanding and reduce miscommunication during group discussions.

2. **Technical Writing**

- Learn how to write **clear and concise documentation** for software, including code comments, user guides, API documentation, and system manuals.
- Gain experience in **writing technical blog posts**, **tutorials**, and **case studies** that explain complex technical topics in an easy-to-understand way.
- Develop skills to **document systems**, processes, and codebases in a structured format that is accessible for other developers, testers, and end users.

3. **Interview Skills & Resume Building**

- Prepare for **technical interviews** by mastering how to present yourself, explain your thought process, and solve problems on the spot.
- Learn the art of crafting a **professional resume** that highlights your technical skills, experience, and achievements in a format that stands out to hiring managers.
- Practice for **behavioral interviews** using techniques like the **STAR method** (Situation, Task, Action, Result) to answer questions about past experiences and personal strengths.
- Build a **strong LinkedIn profile** and portfolio, showcasing your key projects, contributions, and achievements, ensuring it's optimized for recruiters.

4. **Professional Ethics & Workplace Etiquette**:

- Understand **professional ethics** in the workplace, including confidentiality, intellectual property rights, and the ethical use of technology.
- Develop a strong understanding of **workplace etiquette**, including time management, punctuality, respect for colleagues, and maintaining a professional demeanor in various scenarios.
- Learn how to navigate **conflict resolution**, manage disagreements constructively, and foster a collaborative work environment.
- Cultivate the ability to maintain a **healthy work-life balance** and manage stress in a fast-paced and demanding work environment.

## Assessment Criteria

1. **Written Exams (30%)**:

- This is a standard way to assess theoretical understanding and knowledge retention.
- Typically covers topics like algorithms, system design, security, architecture principles, etc.
- Can be divided into **multiple-choice questions** (MCQs), **short-answer questions**, and **problem-solving questions**.

2. **Practical Assessments (30%)**:

- This focuses on testing real-world skills, where students apply knowledge to solve practical problems.
- Could include coding challenges, debugging tasks, or small software projects.
- **Hands-on tasks** such as configuring cloud environments, deploying systems, or setting up DevOps pipelines.

3. **Projects / Assignments (20%)**:

- These help assess the students' ability to execute projects and assignments based on real-world industry problems.
- Projects could include **system design**, **software development**, or **cloud architecture** assignments.
- Assignments may be individual or group-based, simulating collaborative work in real-life settings.

4. **Final Viva or Presentation (20%)**:

- The **viva** or **presentation** is crucial for assessing the ability to communicate and explain projects clearly.
- Students would defend their project decisions, design choices, and demonstrate their understanding of the system and technologies used.
- It is also a good way to test problem-solving and critical thinking abilities.

## Trainer Requirements

To ensure the quality of instruction and provide students with the best learning experience, it's important to define clear and relevant qualifications and experience for instructors. Below is a suggested list of **qualifications and experience** that would be appropriate for trainers of each module in this course

### *General Requirements for All Trainers*

1. **Minimum Qualifications:**

- Bachelor's Degree (BSc) in Information Technology, Computer Science, Software Engineering, or a related field.
- Master's Degree (MSc) in relevant fields (e.g., Software Engineering, Computer Science) is highly preferred.

2. **Professional Certifications (Desirable):**

- **AWS Certified Solutions Architect, AWS Certified Developer, or other cloud-related certifications** for instructors in cloud computing and DevOps.
- **Certified Kubernetes Administrator (CKA)** or **Certified Kubernetes Application Developer (CKAD)** for trainers in containerization and orchestration.
- **Certified Java Developer** for instructors teaching Java programming.
- **Scrum Master Certification** (CSM) for instructors covering Agile/Scrum practices and project management.
- **Other certifications in relevant tools and technologies** like **Git, Docker, Spring Framework, React Native, Angular, and others**.

3. **Industry Experience:**

- **At least 3 to 6 years of hands-on experience** in software development, cloud computing, DevOps, or system architecture, depending on the specific module.
- Practical knowledge of working with modern technologies like cloud platforms (AWS, GCP, Azure), DevOps tools, CI/CD pipelines, containerization (Docker, Kubernetes), and various programming languages (Java, JavaScript, Python, C#).
- **Proven track record of implementing real-world systems and projects**, as this ensures the instructor can relate the course content to actual industry scenarios.

4.**Teaching Experience:**

- Experience in **teaching or training** in higher education, technical institutes, or industry-specific training programs.
- Ability to **adapt teaching methods** to different learning styles (e.g., lectures, practical sessions, project-based learning).
- Familiarity with **learning management systems (LMS)** and **online teaching platforms** (especially for hybrid or online courses).

5.**Soft Skills:**

- Strong **communication skills**, with the ability to explain complex concepts clearly.
- **Teamwork** skills for collaborating with other instructors, students, and departments.
- Ability to **mentor and guide students**, providing career insights and professional growth advice.
- **Critical thinking** and **problem-solving skills**, enabling instructors to help students tackle real-world challenges.

## Conclusion

This Diploma in Software Development and Mobile Programming (DSMP) provides a well-rounded curriculum, blending theoretical knowledge with hands-on practical skills. By the end of the program, students will be fully equipped to tackle the challenges in software development, mobile programming, system architecture, and cloud-based technologies. Through a combination of lectures, projects, and industry-relevant assessments, students will build a solid foundation for a successful career in the fast-evolving tech industry.

The course's focus on modern technologies, best practices, and real-world application development will help students stand out to employers and advance in their careers. Whether you are looking to kick-start your career or enhance your skills, this program offers comprehensive training and certifications that meet international standards.

This course is designed to provide an in-depth learning experience with a focus on practical skills and industry-relevant knowledge. The detailed curriculum ensures that each learner gains the technical expertise and real-world problem-solving capabilities required to excel in full-stack development.

**Note: Please note that specific details and internal resources related to the program are confidential and should not be shared publicly. *We are committed to delivering a premium educational experience while maintaining the integrity and privacy of the materials provided.***