LAPT

LONDON ACADEMY OF PROFESSIONAL TRAINING

CERTIFICATE

Certificate in Web Development

C

LAPT — London Academy of Professional Training

Address 85 Great Portland Street, W1W 7LT, London, United Kingdom

Email info@lapt.org

Telephone +44 7513 283044

Web lapt.org

Reference ITC-C03 • IT Centres

UKPRN 10067351 • Registered in England and Wales, company 4984798

4	77	500	400	60%	Lifetime
SUBJECTS	CHAPTERS	CLASSES	TOTAL MARKS	PASS MARK	VALIDITY

AT A GLANCE

REFERENCE	**ITC-C03**
AWARD	**Certificate**
ASSESSMENT	**400 marks — 240 to pass (60%)**
PREREQUISITES	**None**
VALIDITY	**Lifetime**
AWARDING BODY	**LAPT — London Academy of Professional Training**

CURRICULUM

1

Programming Fundamentals and Logic

31 chapters · 140 classes **100 marks**

- **1 Chapter 1 — Fundamentals of Propositional Logic** — 4 classes
  - 1.1 Understanding Basic Propositions and Their Components
  - 1.2 Exploring Logical Connectives: AND, OR, NOT
  - 1.3 Constructing Truth Tables and Evaluating Statements
  - 1.4 Applying Propositional Logic in Real-World Scenarios
- **2 Chapter 2 — Techniques for Logical Deduction** — 4 classes
  - 2.1 Understanding Logical Statements
  - 2.2 Identifying Premises and Conclusions
  - 2.3 Applying Deductive Reasoning
  - 2.4 Solving Problems with Logical Deduction
- **3 Chapter 3 — Introduction to Predicate Logic and Applications** — 4 classes
  - 3.1 Understanding the Basics of Predicate Logic
  - 3.2 Exploring Quantifiers and Their Roles
  - 3.3 Translating English Sentences into Predicate Logic
  - 3.4 Applying Predicate Logic to Real-World Scenarios
- **1 Fundamentals of Basic Algorithms** — 4 classes
  - 1.1 Understanding Basic Algorithm Concepts
  - 1.2 Exploring Simple Sequence Structures
  - 1.3 Implementing Conditional Logic in Algorithms
  - 1.4 Building Iterations for Problem Solving
- **2 Algorithm Design and Efficiency** — 4 classes
  - 2.1 Understanding Basic Algorithm Structures
  - 2.2 Analyzing Algorithm Efficiency: Time and Space Complexities
  - 2.3 Designing Simple Algorithms: Step-by-Step Approach
  - 2.4 Improving Algorithm Performance through Optimization
- **3 Implementing and Adapting Basic Algorithms** — 4 classes
  - 3.1 Understanding Algorithm Structures
  - 3.2 Implementing Simple Algorithms
  - 3.3 Adapting Algorithms for Different Scenarios
  - 3.4 Testing and Optimizing Algorithm Efficiency
- **1 Chapter 1 — Understanding Basic Data Structures** — 4 classes
  - 1.1 Exploring Fundamental Data Structures
  - 1.2 Comparing Arrays and Linked Lists
  - 1.3 Implementing Stacks and Queues
  - 1.4 Analyzing Performance in Data Structure Operations
- **2 Chapter 2 — Complex Data Structures and Their Uses** — 4 classes
  - 2.1 Understanding the Characteristics of Complex Data Structures
  - 2.2 Exploring Applications of Trees and Graphs
  - 2.3 Implementing Hash Tables in Real-World Scenarios
  - 2.4 Analyzing Performance and Efficiency of Complex Data Structures
- **3 Chapter 3 — Data Structures in Software Development** — 4 classes
  - 3.1 Understand the Role of Data Structures in Software Development
  - 3.2 Explore Common Data Structures: Arrays, Lists, and Maps
  - 3.3 Implement Data Structures in Programming: Hands-On Exercises
  - 3.4 Analyze the Efficiency of Data Structures in Real-World Applications
- **1 Understanding Fundamental Logical Concepts** — 4 classes
  - 1.1 Grasping Basic Logical Operators
  - 1.2 Applying Truth Tables in Problem Solving
  - 1.3 Analyzing Logical Statements and Arguments
  - 1.4 Implementing Logic in Computational Thinking
- **2 Building Structured Problem-Solving Steps** — 4 classes
  - 2.1 Understanding the Problem: Defining Objectives and Constraints
  - 2.2 Breaking Down the Problem: Creating Sub-Tasks
  - 2.3 Planning Solutions: Developing Logical Sequences
  - 2.4 Evaluating Outcomes: Testing and Refining Solutions
- **3 Applying Logic to Real-World Computational Problems** — 4 classes
  - 3.1 Understanding Logic in Everyday Contexts
  - 3.2 Analyzing Computational Problems with Logical Frameworks
  - 3.3 Creating Logical Models for Problem Solving
  - 3.4 Applying Logical Models to Real-World Scenarios
- **1 Understanding Computational System Structures and**
- **2 Data Types, Variables, and Operators** — 5 classes
  - 2.1 Understanding Basic Data Types
  - 2.2 Exploring Variables and Their Uses
  - 2.3 Mastering Arithmetic and Logical Operators
  - 2.4 Applying String and Boolean Data Types
  - 2.5 Solving Problems with Variables and Operators
- **3 Control Structures: Sequence, Selection, and Iteration** — 5 classes
  - 3.1 Understanding Sequence in Programming
  - 3.2 Implementing Conditional Selection
  - 3.3 Utilizing Iteration Constructs
  - 3.4 Combining Sequence, Selection, and Iteration
  - 3.5 Applying Control Structures in Real-world Problems
- **4 Basic Input/Output and Error Handling** — 5 classes
  - 4.1 Understanding Standard Input and Output
  - 4.2 Implementing Input Functions in Programming
  - 4.3 Exploring Output Techniques and Methods
  - 4.4 Identifying and Managing Common I/O Errors
  - 4.5 Applying Error Handling Strategies in Code
- **1 Understanding Basic Algorithmic Concepts** — 5 classes
  - 1.1 Exploring What Algorithms Are
  - 1.2 Understanding Algorithm Components: Inputs and Outputs
  - 1.3 Identifying and Describing Basic Control Structures
  - 1.4 Applying Pseudocode to Design Simple Algorithms
  - 1.5 Analyzing Algorithm Efficiency and Effectiveness
- **2 Core Logic Structures and Constructs** — 5 classes
  - 2.1 Understanding Basic Logical Constructs
  - 2.2 Implementing Conditional Statements
  - 2.3 Exploring Loop Structures
  - 2.4 Applying Logical Operators in Algorithms
  - 2.5 Constructing and Analyzing Control Flow
- **3 Introduction to Algorithmic Efficiency** — 5 classes
  - 3.1 Understanding Algorithmic Efficiency
  - 3.2 Exploring Time Complexity Concepts
  - 3.3 Analyzing Space Complexity in Algorithms
  - 3.4 Comparing Algorithms: Best, Average, and Worst Case Scenarios
  - 3.5 Applying Efficiency Principles to Optimize Code
- **4 Designing and Analyzing Data Structures** — 5 classes
  - 4.1 Understanding Data Structures: Core Concepts and Types
  - 4.2 Comparing Data Structures: Strengths and Weaknesses
  - 4.3 Creating Data Structures: Basic Design Techniques
  - 4.4 Analyzing Data Structures: Big O Notation and Complexity
  - 4.5 Implementing Solutions: Choosing the Right Data Structure
- **1 Understanding Basic Syntax Elements** — 5 classes
  - 1.1 Identifying Common Syntax Elements
  - 1.2 Exploring Variable Declarations
  - 1.3 Understanding Control Structures
  - 1.4 Utilizing Functions and Procedures
  - 1.5 Applying Syntax Rules in Practice
- **2 Control Flow Constructs** — 5 classes
  - 2.1 Understanding Conditional Statements
  - 2.2 Exploring Loop Structures
  - 2.3 Implementing Branching with Conditionals
  - 2.4 Comparing Iterative Constructs Across Languages
  - 2.5 Applying Control Flow to Solve Problems
- **3 Functions and Modular Syntax** — 5 classes
  - 3.1 Understanding Functions: Concepts and Terminology
  - 3.2 Exploring Function Syntax: Parameters and Return Values
  - 3.3 Implementing Modular Code: Breaking Down Problems
  - 3.4 Applying Functions: Real-World Examples and Uses
  - 3.5 Reviewing Function Syntax: Best Practices and Common Pitfalls
- **4 Error Handling and Exceptions** — 5 classes
  - 4.1 Understanding Error Types and Exceptions
  - 4.2 Identifying and Interpreting Error Messages
  - 4.3 Implementing Basic Try-Catch Blocks

2 **Web Development: HTML, CSS and JavaScript** 14 chapters · 120 classes **100 marks**

- **1 Mastering Modern JavaScript Syntax and ES6+ Features** — 6 classes
  - 1.1 Exploring ES6+ Syntax and Features
  - 1.2 Understanding Let, Const, and Block Scoping
  - 1.3 Harnessing Arrow Functions and Implicit Returns
  - 1.4 Mastering Template Literals and String Interpolation
  - 1.5 Utilizing Destructuring for Arrays and Objects
  - 1.6 Implementing Async/Await for Asynchronous Programming
- **2 Exploring the Core Concepts of React.js** — 6 classes
  - 2.1 Understanding JSX: Bridging HTML and JavaScript
  - 2.2 Building Components: The Backbone of React
  - 2.3 Managing State: Enhancing Interactivity
  - 2.4 Handling Events: Capturing User Actions
  - 2.5 Implementing Props: Facilitating Component Communication
  - 2.6 Utilizing React Hooks: Functional Programming in React
- **3 Building Interactive User Interfaces with Vue.js** — 6 classes
  - 3.1 Understanding Vue.js Basics and Its Core Concepts
  - 3.2 Setting Up the Development Environment for Vue.js Projects
  - 3.3 Creating Reactive Data Bindings and Vue Instance
  - 3.4 Designing Dynamic Components and Vue Templates
  - 3.5 Implementing Vue Directives for Enhanced Interactivity
  - 3.6 Integrating Vue.js with External APIs for Dynamic Data
- **4 Harnessing the Power of Angular for Complex Applications** — 6 classes
  - 4.1 Exploring Angular Architecture and Core Concepts
  - 4.2 Building Dynamic Interfaces with Angular Components
  - 4.3 Managing Data with Angular Services and Dependency Injection
  - 4.4 Navigating Complex Applications Using Angular Routing
  - 4.5 Optimizing State Management with Angular Forms and Reactive Programming
  - 4.6 Integrating Third-Party Libraries and APIs in Angular
- **5 Integrating RESTful APIs with Advanced Frameworks** — 6 classes
  - 5.1 Understanding RESTful API Concepts
  - 5.2 Setting Up API Requests in Advanced Frameworks
  - 5.3 Implementing Authentication for Secure API Access
  - 5.4 Parsing and Managing API Responses
  - 5.5 Error Handling in API Integration
  - 5.6 Optimizing Performance in Framework-API Communication
- **6 Web Basics and HTML Document Structure** — 10 classes
  - 6.1 How the Web Works
  - 6.2 Browsers, Servers and Requests
  - 6.3 Setting Up a Code Editor and Workspace
  - 6.4 Anatomy of an HTML Document
  - 6.5 Headings and Paragraphs
  - 6.6 Lists and Nested Elements
  - 6.7 Comments, Entities and Clean Formatting
  - 6.8 Validating HTML Markup
  - 6.9 Organising Project Files and Folders
  - 6.10 Building a First Web Page
- **7 Text, Links, Images and Semantic HTML** — 10 classes
  - 7.1 Links and Navigation Paths
  - 7.2 Working with Images and Alt Text
  - 7.3 Audio and Video Embedding
  - 7.4 Tables for Tabular Data
  - 7.5 Semantic Page Sections
  - 7.6 Header, Footer and Main Content Areas
  - 7.7 Article and Aside Elements
  - 7.8 Meta Tags and Page Titles
  - 7.9 Favicons and Page Icons
  - 7.10 Building a Multi-Page Website
- **8 HTML Forms and Input Validation** — 10 classes
  - 8.1 Form Structure and Actions
  - 8.2 Text Inputs and Labels
  - 8.3 Choices with Radio Buttons and Checkboxes
  - 8.4 Drop-Down Lists and Text Areas
  - 8.5 Built-In Browser Validation
  - 8.6 Required Fields and Patterns
  - 8.7 Date, Number and File Inputs
  - 8.8 Form Layout and Usability
  - 8.9 Client-Side Validation with Scripts
  - 8.10 Building a Contact Form
- **9 CSS Fundamentals and Styling** — 10 classes
  - 9.1 Linking Style Sheets to Pages
  - 9.2 Selectors and Specificity
  - 9.3 Colours and Backgrounds
  - 9.4 Fonts and Text Styling
  - 9.5 Box Model and Spacing
  - 9.6 Borders, Shadows and Rounded Corners
  - 9.7 Styling Links, Lists and Tables
  - 9.8 Pseudo-Classes and States
  - 9.9 CSS Variables for Reusable Themes
  - 9.10 Styling a Complete Page
- **10 CSS Layout with Flexbox and Grid** — 10 classes
  - 10.1 Display and Positioning Basics
  - 10.2 Flexbox Containers and Items
  - 10.3 Aligning and Distributing Content
  - 10.4 Wrapping and Ordering Flex Items
  - 10.5 Grid Rows and Columns
  - 10.6 Grid Areas and Gaps
  - 10.7 Building a Navigation Bar
  - 10.8 Building a Card Layout
  - 10.9 Sticky Headers and Footers
  - 10.10 Building a Landing Page Layout
- **11 Responsive and Mobile First Design** — 10 classes
  - 11.1 Why Responsive Design Matters
  - 11.2 Viewport Settings
  - 11.3 Fluid Widths and Relative Units
  - 11.4 Media Queries and Breakpoints
  - 11.5 Responsive Images
  - 11.6 Mobile First Strategy
  - 11.7 Responsive Navigation Menus
  - 11.8 Testing on Different Devices
  - 11.9 Touch Friendly Controls
  - 11.10 Making a Site Fully Responsive
- **12 JavaScript DOM and Events** — 10 classes
  - 12.1 What the DOM Is
  - 12.2 Selecting Page Elements
  - 12.3 Changing Text and Styles
  - 12.4 Creating and Removing Elements
  - 12.5 Click and Input Events
  - 12.6 Form Handling with Scripts
  - 12.7 Toggling Menus and Tabs
  - 12.8 Storing Data in the Browser
  - 12.9 Debugging with Browser Tools
  - 12.10 Building an Interactive To-Do List
- **13 Web Accessibility and Search Visibility** — 10 classes
  - 13.1 Why Accessibility Matters
  - 13.2 Contrast and Readable Text
  - 13.3 Keyboard Navigation
  - 13.4 Screen Reader Friendly Markup
  - 13.5 Accessible Forms and Error Messages
  - 13.6 Roles and Labels for Components
  - 13.7 Search Engine Basics
  - 13.8 Titles, Descriptions and Headings for Search
  - 13.9 Site Maps and Crawling Rules
  - 13.10 Running an Accessibility Check
- **14 Front-End Testing, Performance and Deployment** — 10 classes

3 **Back-End Development: PHP, MySQL and Node.js** 14 chapters · 120 classes **100 marks**

- **1 Node.js Basics and Setup** — 6 classes
  - 1.1 Understanding Node.js: An Introduction to Its Architecture and Features
  - 1.2 Installing Node.js: Setting Up Your Development Environment
  - 1.3 Exploring Node.js Modules: Built-in and Third-Party Options
  - 1.4 Running Your First Node.js Script: Basic Commands and Execution
  - 1.5 Managing Node.js Packages: Using npm for Dependencies
  - 1.6 Utilizing Node.js REPL: Interactive Coding and Debugging
- **2 Core Modules and File System Operations** — 6 classes
  - 2.1 Understanding Node.js Core Modules
  - 2.2 Exploring the File System Module
  - 2.3 Reading Files with Node.js
  - 2.4 Writing and Updating Files in Node.js
  - 2.5 Managing Directories and Paths
  - 2.6 Error Handling in File System Operations
- **3 Building HTTP Servers and RESTful APIs** — 6 classes
  - 3.1 Understanding HTTP: The Basics of Web Communication
  - 3.2 Setting Up a Node.js Server: First Steps
  - 3.3 Handling HTTP Requests and Responses in Node.js
  - 3.4 Building RESTful APIs: Designing Resource Endpoints
  - 3.5 Implementing CRUD Operations in RESTful Services
  - 3.6 Securing and Testing Node.js APIs for Robustness
- **4 Database Integration with Node.js** — 6 classes
  - 4.1 Understanding Databases in Node.js Context
  - 4.2 Setting Up a Local Database Environment
  - 4.3 Connecting Node.js to Your Database
  - 4.4 Performing CRUD Operations in Node.js
  - 4.5 Implementing Data Models and Schemas
  - 4.6 Handling Database Errors and Optimizing Queries
- **5 Security Practices and Deployment Strategies** — 6 classes
  - 5.1 Understanding Security Fundamentals in Node.js
  - 5.2 Implementing Authentication and Authorization
  - 5.3 Securing Data with Encryption Techniques
  - 5.4 Managing Server and Database Security
  - 5.5 Optimizing Node.js Applications for Deployment
  - 5.6 Deploying and Monitoring Node.js Applications
- **6 Server-Side Basics and PHP Syntax** — 10 classes
  - 6.1 Client Server Model Revisited
  - 6.2 Setting Up a Local Server Stack
  - 6.3 Writing the First PHP Script
  - 6.4 Variables and Data Types in PHP
  - 6.5 Operators and Expressions
  - 6.6 Echoing Output and Embedding in HTML
  - 6.7 Constants and Type Conversion
  - 6.8 Comments and Coding Standards
  - 6.9 Reading Error Messages
  - 6.10 Mini Exercise Building a Dynamic Page
- **7 PHP Control Flow, Functions and Arrays** — 10 classes
  - 7.1 Conditional Statements
  - 7.2 Switch and Match Logic
  - 7.3 Loops for Repeating Tasks
  - 7.4 Indexed Arrays
  - 7.5 Associative Arrays
  - 7.6 Array Functions for Sorting and Searching
  - 7.7 Writing User Defined Functions
  - 7.8 Function Arguments and Return Values
  - 7.9 Including and Reusing Files
  - 7.10 Building a Simple Price Calculator
- **8 Handling Forms, Sessions and Cookies in PHP** — 10 classes
  - 8.1 Receiving Form Data
  - 8.2 Get and Post Methods
  - 8.3 Sanitising and Validating Input
  - 8.4 Showing Error Messages to Users
  - 8.5 Cookies for Remembering Visitors
  - 8.6 Starting and Using Sessions
  - 8.7 Managing a Shopping Cart Session
  - 8.8 Redirects and Page Flow
  - 8.9 Preventing Duplicate Submissions
  - 8.10 Building a Registration Form Handler
- **9 Relational Database Design for Web Applications** — 10 classes
  - 9.1 Why Applications Need Databases
  - 9.2 Tables, Rows and Columns
  - 9.3 Choosing Data Types
  - 9.4 Primary and Foreign Keys
  - 9.5 One to Many Relationships
  - 9.6 Many to Many Relationships
  - 9.7 Normalising a Design
  - 9.8 Drawing an Entity Relationship Diagram
  - 9.9 Designing a Student Records Database
  - 9.10 Reviewing and Refining a Schema
- **10 SQL Queries with MySQL** — 10 classes
  - 10.1 Installing and Using a Database Server
  - 10.2 Creating Databases and Tables
  - 10.3 Inserting Records
  - 10.4 Selecting and Filtering Records
  - 10.5 Sorting and Limiting Results
  - 10.6 Updating and Deleting Records
  - 10.7 Joining Related Tables
  - 10.8 Aggregate Functions and Grouping
  - 10.9 Indexes for Faster Queries
  - 10.10 Exporting and Restoring Database Backups
- **11 Connecting PHP to a Database Securely** — 10 classes
  - 11.1 Database Connection Settings
  - 11.2 Running Queries from PHP
  - 11.3 Fetching Results into Pages
  - 11.4 Prepared Statements
  - 11.5 Preventing SQL Injection
  - 11.6 Insert, Update and Delete Pages
  - 11.7 Pagination of Large Result Sets
  - 11.8 Search Features with Safe Queries
  - 11.9 Handling Database Errors Gracefully
  - 11.10 Building a Complete Records Manager
- **12 User Authentication and Authorisation** — 10 classes
  - 12.1 Registration and Login Flow
  - 12.2 Storing Passwords with Hashing
  - 12.3 Verifying Login Credentials
  - 12.4 Session Based Login State
  - 12.5 Logout and Session Expiry
  - 12.6 User Roles and Permissions
  - 12.7 Protecting Pages from Unauthorised Access
  - 12.8 Password Reset Process
  - 12.9 Preventing Cross-Site Scripting
  - 12.10 Preventing Cross-Site Request Forgery
- **13 File Handling, Uploads and Email Sending** — 10 classes
  - 13.1 Reading and Writing Text Files
  - 13.2 Working with Directories
  - 13.3 Uploading Files Safely
  - 13.4 Checking File Types and Sizes
  - 13.5 Storing and Serving Images
  - 13.6 Generating CSV Exports
  - 13.7 Generating Simple PDF Reports
  - 13.8 Sending Email from Scripts
  - 13.9 Email Templates and Attachments
  - 13.10 Building a Document Upload Module
- **14 Hosting and Maintaining a PHP Application** — 10 classes

4 **APIs, DevOps and Continuous Integration** 18 chapters · 120 classes **100 marks**

- **1 Understanding APIs: Types and Uses in Modern Software** — 6 classes
  - 1.1 Exploring API Fundamentals and Principles
  - 1.2 Identifying Different Types of APIs
  - 1.3 Examining RESTful API Architecture
  - 1.4 Understanding GraphQL and Its Applications
  - 1.5 Analyzing Real-world API Use Cases
  - 1.6 Evaluating API Integration Methods and Strategies
- **2 Building RESTful APIs: Best Practices and Protocols** — 6 classes
  - 2.1 Understanding RESTful API Architecture
  - 2.2 Identifying and Defining API Resources
  - 2.3 Designing Consistent API Endpoints
  - 2.4 Implementing HTTP Methods Effectively
  - 2.5 Securing RESTful APIs with Best Practices
  - 2.6 Testing and Debugging RESTful APIs
- **3 Advanced API Authentication and Security** — 6 classes
  - 3.1 Understanding Advanced Authentication Protocols
  - 3.2 Implementing OAuth2 for Enhanced Security
  - 3.3 Securing APIs with JWT and Token Verification
  - 3.4 Applying Role-Based Access Control (RBAC) in APIs
  - 3.5 Enforcing Rate Limiting and Quotas for API Protection
  - 3.6 Monitoring and Auditing API Security Postures
- **4 Integrating APIs: Connecting Services and Data** — 6 classes
  - 4.1 Exploring API Fundamentals and Concepts
  - 4.2 Analyzing RESTful API Architecture
  - 4.3 Configuring API Endpoints for Seamless Integration
  - 4.4 Authenticating and Securing API Interactions
  - 4.5 Implementing API Calls in Web Applications
  - 4.6 Troubleshooting and Optimizing API Performance
- **5 Testing and Documenting APIs: Ensuring Reliability** — 6 classes
  - 5.1 Understanding the Importance of API Testing
  - 5.2 Exploring Different Types of API Testing
  - 5.3 Implementing Automated API Testing
  - 5.4 Designing Effective API Test Cases
  - 5.5 Creating Comprehensive API Documentation
  - 5.6 Utilizing Tools for API Documentation and Testing
- **1 Understanding DevOps and Its Role in Modern Software Development** — 6 classes
  - 1.1 Exploring the Fundamentals of DevOps
  - 1.2 Identifying Key DevOps Practices and Tools
  - 1.3 Analyzing the Benefits of Continuous Integration
  - 1.4 Implementing DevOps Culture in Software Development
  - 1.5 Integrating Automated Testing in CI/CD Pipelines
  - 1.6 Evaluating DevOps Success through Metrics and Feedback
- **2 Continuous Integration: Tools and Best Practices** — 6 classes
  - 2.1 Understanding Continuous Integration in DevOps
  - 2.2 Exploring Popular CI Tools: Jenkins, Travis CI, CircleCI
  - 2.3 Setting Up Your First CI Pipeline
  - 2.4 Integrating Automated Testing in CI
  - 2.5 Analyzing Best Practices for CI Success
  - 2.6 Troubleshooting Common CI Issues
- **3 Automating Deployment and Infrastructure with Continuous Delivery** — 6 classes
  - 3.1 Understanding Continuous Delivery Concepts and Benefits
  - 3.2 Exploring Tools for Continuous Deployment
  - 3.3 Setting Up Automated Deployment Pipelines
  - 3.4 Implementing Infrastructure as Code with Terraform
  - 3.5 Integrating CI/CD with Containerization Technologies
  - 3.6 Monitoring and Optimizing Automated Deployments
- **4 Monitoring, Logging, and Feedback Loops in DevOps** — 6 classes
  - 4.1 Understanding the Importance of Monitoring in DevOps
  - 4.2 Exploring Key Metrics and Tools for Effective Logging
  - 4.3 Implementing Monitoring Solutions to Enhance System Performance
  - 4.4 Developing Feedback Loops to Improve Development Processes
  - 4.5 Integrating Logging and Monitoring in CI/CD Pipelines
- **5 Security and Compliance in DevOps Practices** — 6 classes
  - 5.1 Understanding Security Principles in DevOps
  - 5.2 Identifying Compliance Requirements in DevOps
  - 5.3 Implementing Security Best Practices in CI/CD
  - 5.4 Integrating Automated Security Testing into Pipelines
  - 5.5 Managing Access and Permissions within DevOps Teams
  - 5.6 Monitoring Compliance through DevOps Tools
- **1 Understanding DevOps Principles and Culture** — 6 classes
  - 1.1 Exploring the Core Principles of DevOps
  - 1.2 Identifying Key Components of a DevOps Culture
  - 1.3 Understanding Continuous Integration and Continuous Delivery
  - 1.4 Examining the Role of Automation in DevOps
  - 1.5 Fostering Collaboration between Development and Operations
  - 1.6 Aligning DevOps Practices with Business Goals
- **2 Continuous Integration and Deployment Strategies** — 6 classes
  - 2.1 Understanding Continuous Integration
  - 2.2 Exploring Continuous Deployment Pipelines
  - 2.3 Implementing Automated Testing in CI/CD
  - 2.4 Integrating CI/CD with Version Control Systems
  - 2.5 Monitoring and Feedback in Deployment
  - 2.6 Optimizing CI/CD for DevOps Efficiency
- **3 Infrastructure as Code and Configuration Management** — 6 classes
  - 3.1 Understanding Infrastructure as Code Principles
  - 3.2 Exploring Configuration Management Tools
  - 3.3 Setting Up Version Control for Infrastructure
  - 3.4 Automating Infrastructure Deployment Pipelines
  - 3.5 Implementing Configuration Management Strategies
  - 3.6 Evaluating Best Practices for IaC and Configuration Management
- **4 Monitoring, Logging, and Observability in DevOps** — 6 classes
  - 4.1 Understanding Monitoring Fundamentals in DevOps
  - 4.2 Exploring Logging Techniques and Tools
  - 4.3 Implementing Observability Concepts
  - 4.4 Integrating Monitoring and Logging
  - 4.5 Utilizing Observability for Proactive Operations
  - 4.6 Analyzing Data for Continuous Improvement in DevOps
- **5 DevOps Security Practices and Compliance** — 6 classes
  - 5.1 Understanding the Importance of DevOps Security
  - 5.2 Identifying Common Security Vulnerabilities in DevOps
  - 5.3 Implementing Secure Coding Practices in DevOps
  - 5.4 Integrating Automated Security Testing in CI/CD Pipelines
  - 5.5 Navigating Compliance Requirements and Best Practices
  - 5.6 Developing a Culture of Security within DevOps Teams
- **16 Version Control and Team Collaboration with Git** — 10 classes
  - 16.1 Why Version Control Matters
  - 16.2 Installing and Configuring Git
  - 16.3 Creating a Repository and Commits
  - 16.4 Reading History and Differences
  - 16.5 Branching for Features
  - 16.6 Merging and Resolving Conflicts
  - 16.7 Remote Repositories and Pushing
  - 16.8 Pull Requests and Code Review
  - 16.9 Tags, Releases and Ignored Files
  - 16.10 Team Branching Workflows
- **17 Containers and Application Packaging** — 10 classes
  - 17.1 Why Containers Are Used
  - 17.2 Images and Containers Explained
  - 17.3 Running Your First Container
  - 17.4 Writing a Container Build File
  - 17.5 Managing Container Storage
  - 17.6 Container Networking Basics
  - 17.7 Running Multiple Containers Together
  - 17.8 Storing and Sharing Images in a Registry
  - 17.9 Container Security Basics
  - 17.10 Packaging a Sample Web Application
- **18 Release Management, Rollbacks and Post Release Review** — 10

LAPT — 85 Great Portland Street, W1W 7LT, London, United Kingdom
info@lapt.org • +44 7513 283044 • lapt.org

UKPRN 10067351 • Company 4984798
Curriculum current at the date of issue.