

LAPT

LONDON ACADEMY OF PROFESSIONAL TRAINING

CERTIFICATE

ISO 12207SFT — Software Lifecycle Processes

ISO Certification Programme



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 IIT-SFT-12207SFT • ISO IT & Related Technologies

UKPRN 10067351 • Registered in England and Wales, company 4984798

AT A GLANCE

REFERENCE	IIT-SFT-12207SFT
AWARD	Certificate
ASSESSMENT	430 marks — 258 to pass (60%)
PREREQUISITES	None
VALIDITY	Lifetime
AWARDING BODY	LAPT — London Academy of Professional Training

CURRICULUM

1

Advanced Project Management Techniques

5 chapters · 30 classes 65 marks

• 1 Fundamentals of Project Management in Software Development

— 6 classes

› 1.1 Define Key Concepts in Software Project Management

› 1.2 Identify Phases of the Software Development Lifecycle

› 1.3 Analyze Stakeholder Roles and Responsibilities

› 1.4 Employ Risk Management Strategies in Software Projects

› 1.5 Utilize Agile Methodologies for Effective Project Execution

› 1.6 Develop a Project Plan with Measurable Outcomes

• 2 Integrating ISO 12207 into Project Management Practices

— 6 classes

› 2.1 Understand ISO 12207 Principles for Project Management

› 2.2 Map Software Lifecycle Processes to Project Phases

› 2.3 Identify Key Stakeholders and Their Roles in ISO 12207

› 2.4 Develop an Integrated Project Management Plan Using ISO 12207

› 2.5 Assess Project Risks through ISO 12207 Framework

› 2.6 Implement Continuous Improvement Practices in Project Management

• 3 Risk Management Techniques for Software Projects

— 6 classes

› 3.1 Identify Key Risk Factors in Software Projects

› 3.2 Assess the Impact of Risks on Project Objectives

› 3.3 Develop a Risk Mitigation Strategy for Software Delivery

› 3.4 Implement Risk Monitoring Techniques Throughout the Lifecycle

› 3.5 Evaluate the Effectiveness of Risk Management Approaches

› 3.6 Create a Comprehensive Risk Management Plan for a Case Study

• 4 Leadership and Team Dynamics in Software Projects

— 6 classes

› 4.1 Analyze Leadership Styles in Software Teams

› 4.2 Explore Team Dynamics and Communication Strategies

› 4.3 Assess Conflict Resolution Techniques in Project Management

› 4.4 Implement Decision-Making Frameworks for Teams

› 4.5 Evaluate the Impact of Team Culture on Project Success

› 4.6 Design a Leadership Development Plan for Software Projects

• 5 Evaluating Project Success and Continuous Improvement

— 6 classes

› 5.1 Define Key Metrics for Project Success

› 5.2 Analyze Stakeholder Feedback for Improvement Insights

› 5.3 Implement Best Practices in Project Evaluation

› 5.4 Develop a Continuous Improvement Plan

› 5.5 Conduct a Root Cause Analysis of Project Failures

› 5.6 Create a Framework for Ongoing Project Reviews

• 1 Understanding Software Lifecycle Models and ISO 12207 — 6 classes

- › 1.1 Define Key Concepts in Software Lifecycle Models
- › 1.2 Explore the Importance of ISO 12207 in Software Development
- › 1.3 Identify Different Software Lifecycle Models and Their Characteristics
- › 1.4 Analyze the Relationship Between ISO 12207 and Project Management
- › 1.5 Compare and Contrast Traditional vs. Agile Software Lifecycle Models
- › 1.6 Develop a Practical Implementation Plan Using ISO 12207 Guidelines

• 2 Identifying Stakeholders and Requirements in Process Design — 6 classes

- › 2.1 Define Key Stakeholders in Software Process Design
- › 2.2 Analyze Stakeholder Needs and Expectations
- › 2.3 Facilitate Stakeholder Collaboration and Communication
- › 2.4 Document Requirements Gathering Techniques
- › 2.5 Evaluate and Prioritize Stakeholder Requirements
- › 2.6 Develop a Comprehensive Stakeholder Requirements Matrix

• 3 Process Mapping and Documentation Techniques — 6 classes

- › 3.1 Identify Key Components of Process Mapping
- › 3.2 Analyze Different Documentation Techniques
- › 3.3 Create a Basic Process Map Using Flowcharts
- › 3.4 Employ SIPOC Diagrams for High-Level Process Overview
- › 3.5 Develop Detailed Process Documentation Standards
- › 3.6 Implement Process Mapping in Real-World Scenarios

• 4 Implementing Process Design for Continuous Improvement — 6 classes

- › 4.1 Analyze Current Processes for Improvement Opportunities
- › 4.2 Define Key Performance Indicators for Success
- › 4.3 Design a Framework for Continuous Improvement Initiatives
- › 4.4 Implement Process Changes Using Agile Methodologies
- › 4.5 Measure Outcomes and Gather Feedback for Refinement
- › 4.6 Foster a Culture of Continuous Improvement Across Teams

• 5 Measuring Process Performance and Governance in Software Development — 6 classes

- › 5.1 Define Key Performance Indicators for Software Processes
- › 5.2 Analyze Measurement Techniques for Process Performance
- › 5.3 Implement Governance Frameworks in Software Development
- › 5.4 Evaluate Data Collection Methods for Performance Metrics
- › 5.5 Design a Continuous Improvement Plan for Software Processes
- › 5.6 Present Findings and Recommendations for Process Enhancements

• 1 Understanding Risk in Software Development — 6 classes

- › 1.1 Define Risk in Software Development
- › 1.2 Identify Common Risks in Software Projects
- › 1.3 Analyze the Impact of Risks on Software Outcomes
- › 1.4 Evaluate Risk Assessment Techniques
- › 1.5 Develop a Risk Mitigation Plan
- › 1.6 Apply Risk Management Strategies in Real Case Scenarios

• 2 Identifying and Assessing Risks — 6 classes

- › 2.1 Define Risk in Software Development Context
- › 2.2 Identify Common Risks Associated with Software Projects
- › 2.3 Explore Tools and Techniques for Risk Identification
- › 2.4 Assess Risk Impact and Probability in Software Projects
- › 2.5 Prioritize Identified Risks for Effective Management
- › 2.6 Develop a Risk Assessment Report for Stakeholders

• 3 Risk Mitigation Strategies — 6 classes

- › 3.1 Identify Common Risks in Software Development
- › 3.2 Evaluate the Impact of Risks on Project Goals
- › 3.3 Develop Effective Mitigation Strategies
- › 3.4 Assess the Feasibility of Risk Mitigation Options
- › 3.5 Implement Risk Mitigation Plans in Agile Sprints
- › 3.6 Review and Adjust Mitigation Strategies Post-Implementation

• 4 Integrating Risk Management into Software Processes — 6 classes

- › 4.1 Identify and Define Key Risks in Software Development
- › 4.2 Analyze Risk Impact and Likelihood in Software Projects
- › 4.3 Develop Risk Mitigation Strategies for Software Processes
- › 4.4 Integrate Risk Management into Software Development Life Cycle
- › 4.5 Monitor and Review Risks Throughout Software Development
- › 4.6 Communicate and Document Risks for Stakeholder Engagement

• 5 Monitoring and Reviewing Risks — 6 classes

- › 5.1 Identify Key Risk Indicators in Software Development
- › 5.2 Develop a Risk Monitoring Plan for Software Projects
- › 5.3 Analyze Risk Data: Techniques for Effective Review
- › 5.4 Implement Tools for Continuous Risk Monitoring
- › 5.5 Evaluate the Effectiveness of Risk Management Strategies
- › 5.6 Communicate Risk Findings to Stakeholders Effectively

• 1 Fundamentals of Software Lifecycle Models — 6 classes

- › 1.1 Define Key Concepts in Software Lifecycle Models
- › 1.2 Explore the Historical Development of Software Lifecycle Models
- › 1.3 Identify Different Types of Software Lifecycle Models
- › 1.4 Compare Agile and Waterfall Models in Software Development
- › 1.5 Analyze the Benefits and Limitations of Various Models
- › 1.6 Apply a Software Lifecycle Model to a Real-World Project Scenario

• 2 Overview of Common Software Lifecycle Models — 6 classes

- › 2.1 Define Key Software Lifecycle Models
- › 2.2 Compare Waterfall and Agile Methodologies
- › 2.3 Examine Spiral Model Characteristics
- › 2.4 Identify Benefits of Iterative Development
- › 2.5 Assess the Role of DevOps in Software Lifecycles
- › 2.6 Apply Lifecycle Models to Real-World Scenarios

• 3 Comparative Analysis of Lifecycle Models — 6 classes

- › 3.1 Identify Key Characteristics of Software Lifecycle Models
- › 3.2 Compare and Contrast Traditional Waterfall and Agile Models
- › 3.3 Analyze the Pros and Cons of Iterative Development Approaches
- › 3.4 Evaluate the Impact of DevOps on Software Lifecycle Management
- › 3.5 Apply a Case Study to Examine Model Suitability for Specific Projects
- › 3.6 Discuss Future Trends and Innovations in Software Lifecycle Models

• 4 Tailoring Software Lifecycle Models for Projects — 6 classes

- › 4.1 Define Key Software Lifecycle Models in Project Context
- › 4.2 Assess Project Requirements to Choose Appropriate Lifecycle Model
- › 4.3 Identify Benefits and Limitations of Different Lifecycle Models
- › 4.4 Tailor Lifecycle Models for Specific Project Needs
- › 4.5 Integrate Stakeholder Perspectives into Lifecycle Model Selection
- › 4.6 Evaluate the Effectiveness of Tailored Lifecycle Models in Practice

• 5 Future Trends in Software Lifecycle Management — 6 classes

- › 5.1 Analyze Emerging Software Lifecycle Models
- › 5.2 Evaluate the Impact of AI on Software Lifecycle Management
- › 5.3 Explore Agile Methodologies in Future Contexts
- › 5.4 Assess the Role of DevOps in Evolving Software Processes
- › 5.5 Implement Best Practices for Continuous Integration and Delivery
- › 5.6 Predict Future Skills Needed for Software Lifecycle Leadership

• 1 Fundamentals of Software Quality Assurance — 6 classes

- › 1.1 Define Software Quality Assurance Principles
- › 1.2 Identify Key Software Quality Assurance Processes
- › 1.3 Examine Roles and Responsibilities in SQA
- › 1.4 Analyze Common Software Quality Metrics
- › 1.5 Evaluate Techniques for Quality Assurance in Software Development
- › 1.6 Implement a Basic Software Quality Assurance Plan

• 2 ISO 12207SFT Standards and Compliance — 6 classes

- › 2.1 Understand ISO 12207SFT: Overview of Software Lifecycle Processes
- › 2.2 Explore Key Principles of Software Quality Assurance
- › 2.3 Identify ISO 12207SFT Compliance Requirements for Projects
- › 2.4 Analyze the Role of Documentation in Quality Assurance
- › 2.5 Implement Best Practices for Ensuring Software Quality
- › 2.6 Evaluate Real-World Case Studies of ISO 12207SFT Compliance

• 3 Quality Assurance Processes and Methodologies — 6 classes

- › 3.1 Understand Quality Assurance Fundamentals in Software Development
- › 3.2 Explore Key Quality Assurance Processes in ISO 12207
- › 3.3 Analyze Quality Assurance Methodologies and Best Practices
- › 3.4 Implement Effective Quality Assurance Techniques in Software Projects
- › 3.5 Evaluate Quality Assurance Metrics and Performance Indicators
- › 3.6 Develop a Quality Assurance Plan for Software Lifecycle Management

• 4 Testing Strategies for Quality Assurance — 6 classes

- › 4.1 Analyze Testing Goals for Quality Assurance
- › 4.2 Identify Types of Testing Strategies in Software Development
- › 4.3 Develop Test Plans Aligned with Quality Assurance Standards
- › 4.4 Implement Automated Testing Tools for Efficiency
- › 4.5 Evaluate Test Results to Ensure Quality Compliance
- › 4.6 Optimize Testing Processes for Continuous Improvement

• 5 Continuous Improvement and Quality Metrics — 6 classes

- › 5.1 Analyze Current Quality Metrics for Improvement Opportunities
- › 5.2 Implement Data Collection Techniques for Software Quality
- › 5.3 Evaluate the Effectiveness of Quality Assurance Practices
- › 5.4 Develop Action Plans Based on Quality Assessment Results
- › 5.5 Foster a Culture of Continuous Improvement in Teams
- › 5.6 Utilize Feedback Loops to Enhance Software Development Processes

• 1 Foundations of Team Leadership in Software Projects — 6 classes

- › 1.1 Identify Key Attributes of Effective Team Leaders
- › 1.2 Explore Team Dynamics in Software Development
- › 1.3 Develop Communication Strategies for Leadership
- › 1.4 Implement Mentoring Techniques within Teams
- › 1.5 Assess Conflict Resolution Strategies in Team Settings
- › 1.6 Evaluate Team Performance Metrics and Improvement Plans

• 2 Building Effective Communication Strategies — 6 classes

- › 2.1 Assess Current Communication Styles in Teams
- › 2.2 Develop Clear Communication Objectives
- › 2.3 Implement Active Listening Techniques
- › 2.4 Utilize Feedback Loops for Improvement
- › 2.5 Create Tailored Communication Plans
- › 2.6 Evaluate Communication Strategy Effectiveness

• 3 Fostering a Positive Team Culture and Dynamics — 6 classes

- › 3.1 Identify Core Values that Shape Team Culture
- › 3.2 Assess Current Team Dynamics and Communication Styles
- › 3.3 Develop Strategies for Encouraging Collaboration and Inclusivity
- › 3.4 Implement Conflict Resolution Techniques to Enhance Team Cohesion
- › 3.5 Establish Mentorship Programs to Support Team Member Growth
- › 3.6 Evaluate the Impact of a Positive Team Culture on Performance Metrics

• 4 Mentoring Techniques for Software Leaders — 6 classes

- › 4.1 Define Mentoring and Its Importance in Software Leadership
- › 4.2 Identify Key Traits of an Effective Mentor in Tech
- › 4.3 Explore Different Mentoring Styles and Their Applications
- › 4.4 Develop Active Listening Skills for Better Mentoring
- › 4.5 Implement Goal-Setting Techniques in Mentoring Relationships
- › 4.6 Evaluate Mentoring Outcomes to Enhance Team Performance

• 5 Leading Teams Through Change and Conflict Resolution — 6 classes

- › 5.1 Understand the Dynamics of Change in Teams
- › 5.2 Identify Common Sources of Conflict in Team Settings
- › 5.3 Develop Strategies for Effective Change Communication
- › 5.4 Employ Conflict Resolution Techniques to Foster Team Cohesion
- › 5.5 Facilitate Team Discussions to Address Resistance and Concerns
- › 5.6 Create an Action Plan for Leading Teams Through Change