IT Management & Audits

Practical Lab Manual

IT Audit Toolkit

Practical P11

Learning Domain

IT Audit & Internal Controls

Course Learning Outcomes

CLO11: Conduct information systems audits using structured methodologies

Unit

Unit VII: IT Audit & Compliance

Time Allocation: 3 hours

Learning Mode: Hands-on (70%) + Theory (30%)

Difficulty Level: Intermediate-Advanced

IT Audit Toolkit

Practical P11

Quick Reference

Practical Code	P11
Practical Name	IT Audit Toolkit
Slot	T/P-11
Duration	3 hours
CLO Mapping	CLO11
Unit	Unit VII: IT Audit & Compliance
Delivery Mode	Hands-on Lab
Target Audience	Intermediate-Advanced Level
India Integration	HIGH
Screenshot Count	5 Required

Prerequisites

- Basic IT governance concepts (P02 recommended)
- Understanding of audit principles and internal controls
- Familiarity with risk assessment concepts
- Python 3.8+ installed on your system
- Command-line interface (CLI) proficiency

Tools Required

Tool	Version	Free	Notes
Python	3.8+	✓	Required
pip	Latest	✓	Included with Python
Click	Latest	✓	CLI framework
Rich	Latest	✓	Terminal formatting
Jinja2	Latest	✓	HTML report templates
Web Browser	Latest	✓	For viewing HTML reports

Learning Objectives

- ✓ Understand the IT audit lifecycle: planning, fieldwork, reporting, and follow-up
- ✓ Set up and use a Python-based IT audit toolkit from a structured repository
- ✓ Create audit engagements and execute structured checklists across six control areas
- ✓ Document audit findings with severity ratings using the Condition-Criteria-Cause-Effect format
- ✓ Perform quantitative risk assessment using a 5x5 likelihood-impact matrix
- ✓ Generate professional HTML audit reports with executive summaries and risk heat maps
- ✓ Apply India-specific IS audit requirements (RBI, SEBI, Companies Act 2013)

What You Will Learn

1. Understand the four phases of an information systems audit
2. Clone and configure an IT audit toolkit with 6 structured checklists (67 controls)
3. Create audit engagements with defined scope, period, and objectives
4. Execute audit checklists and evaluate control compliance status
5. Document findings following the 5 Cs format with severity ratings
6. Calculate risk scores using likelihood-times-impact methodology
7. Generate comprehensive audit reports with remediation roadmaps
8. Relate audit practices to Indian regulatory requirements (RBI, SEBI, CAG)

Real-World Application

IT audits are mandatory for regulated industries in India. The RBI requires annual IS audits for all banks and NBFCs. SEBI mandates cybersecurity audits for stock brokers and market intermediaries. Under the Companies Act 2013, internal audits covering IT systems are required for listed companies. Organizations such as **HDFC Bank**, **Infosys**, and **TCS** undergo regular IT audits conducted by Deloitte, PwC, KPMG, and EY — following the structured methodologies you will practice in this lab.

Hands-On Procedure

Part A: Environment Setup

Step 1: Clone the IT Audit Toolkit Repository

Objective: Download and explore the IT audit toolkit project structure.

```
# Clone the repository
git clone <repository-url> it-audit-toolkit
cd it-audit-toolkit

# View the project structure
ls -R src/
# src/models.py            - Data models (audits, findings, risks)
# src/audit_engine.py      - Core audit execution engine
# src/risk_calculator.py   - Risk scoring and matrix calculations
# src/reporter.py          - Report generation (HTML, terminal)
# src/cli.py               - Click-based CLI interface
# src/checklists/          - 6 JSON audit checklists:
#   access_control.json        (15 controls)
#   change_management.json     (12 controls)
#   incident_response.json     (10 controls)
#   data_backup.json           (8 controls)
#   network_security.json      (12 controls)
#   compliance.json            (10 controls)

# View templates and sample data
ls templates/     # HTML report templates (Jinja2)
cat data/sample_audit.json | python -m json.tool | head -30
```

Clone and Explore the Repository

Expected Output

`src/` – 5 Python source files + `checklists/` directory with 6 JSON files
`templates/` – Jinja2 HTML report templates
`data/sample_audit.json` – Pre-configured sample audit engagement
Total: 67 controls across 6 audit areas

Step 2: Create Virtual Environment and Verify CLI

Objective: Set up the Python environment, install dependencies, and verify all five CLI commands.

```
# Create and activate virtual environment
python -m venv venv
source venv/bin/activate          # Linux/Mac
# .\venv\Scripts\Activate.ps1   # Windows PowerShell

# Install dependencies
```

```
pip install -r requirements.txt

# Verify the CLI tool -- should show 5 commands
python src/cli.py --help
# Commands: new-audit, run-checklist, add-finding,
#           calculate-risk, generate-report

# Verify individual commands
python src/cli.py new-audit --help
python src/cli.py run-checklist --help
python src/cli.py add-finding --help
python src/cli.py calculate-risk --help
python src/cli.py generate-report --help
```

Environment Setup and CLI Verification

Expected Output

```
Usage:  cli.py [OPTIONS] COMMAND [ARGS]...
IT Audit Toolkit - Structured IS Audit Management
Commands:
new-audit         Create a new audit engagement
run-checklist   Execute an audit checklist
add-finding     Document an audit finding
calculate-risk  Calculate risk scores for an audit
generate-report Generate audit report (HTML/terminal)
```

(1) Ensure virtual environment is activated (`(venv)` in prompt). (2) Try `python -m pip install -r requirements.txt`. (3) Check Python 3.8+: `python -version`. (4) On Windows, try `python3` instead.

Screenshot 1

What to paste: Terminal showing (a) repository structure with `ls -R src/` output listing all source files and 6 checklist JSONs, (b) CLI help output showing all 5 commands, and (c) a preview of `data/sample_audit.json`.

Paste your screenshot here

Part B: Audit Engagement Setup

Step 3: Create a New Audit Engagement

Objective: Initialize an IT audit engagement with defined scope, period, team, and objectives.

```
# Create a new audit engagement
python src/cli.py new-audit \
    --name "Digital Lending Platform Audit" \
    --scope "IT General Controls"

# The tool prompts for additional details:
# Audit Period: Q4 2025 (Oct-Dec 2025)
# Lead Auditor: [Your Name]
# Team Size: 3
# Objectives:
#   1. Evaluate access control effectiveness
#   2. Assess change management procedures
#   3. Review incident response capabilities
#   4. Verify data backup and recovery controls
#   5. Examine network security posture
#   6. Check regulatory compliance status

# View the generated audit configuration
cat data/sample_audit.json | python -m json.tool
```

Create New Audit Engagement

Expected Output

```
Audit ID: AUD-2025-001
Name:  Digital Lending Platform Audit
Scope:  IT General Controls   Status:  Planning
Checklists:  6 available   Findings:  0 (not yet started)
Configuration saved to:  data/sample_audit.json
```

In a real IT audit, the engagement letter defines scope, timeline, and responsibilities. Scope creep — expanding beyond agreed boundaries — is a common pitfall. Always document scope clearly at the start.

Step 4: Review Audit Checklists and Control Structure

Objective: Examine checklist structure to understand control definitions, test procedures, and expected evidence.

```
# View the access control checklist
cat src/checklists/access_control.json | python -m json.tool
# Each control has: id, category, description,
# test_procedure, expected_evidence, risk_weight,
```

```
# framework_mapping (COBIT, ISO 27001 references)

# Count controls in each checklist
for f in src/checklists/*.json; do
    count=$(python -c "import␣json;␣print(len(json.load(open('$f'))
        ['controls']))")
    echo "$(basename␣$f):␣$count␣controls"
done
# access_control.json:     15 controls
# change_management.json: 12 controls
# incident_response.json: 10 controls
# data_backup.json:         8 controls
# network_security.json:   12 controls
# compliance.json:         10 controls
# Total: 67 controls
```

Explore Audit Checklists

Expected Output

Sample control structure (AC-001):

```
ID: AC-001  Category:  Access Control  Risk Weight:  high
Description:  User access provisioning follows formal approval
Test Procedure:  Review access request forms, verify approvals
Expected Evidence:  Access request forms, approval emails, logs
Framework Mapping:  COBIT DSS05, ISO 27001 A.9
```

Each control maps to framework references (COBIT, ISO 27001, PCI-DSS). This traceability is essential — auditors must demonstrate testing covers required framework controls.

Part C: Audit Execution (Fieldwork)

Step 5: Execute Access Control Checklist

Objective: Run the access control checklist, evaluating each control as Compliant, Non-Compliant, Partially Compliant, or Not Applicable.

```
# Run the access control checklist
python src/cli.py run-checklist \
    --checklist access_control \
    --audit sample

# For each control the tool presents:
# - Control ID and description
# - Test procedure to follow
# - Expected evidence to collect
#
# You respond with:
#   Status: C (Compliant) / NC (Non-Compliant) /
#           PC (Partially Compliant) / NA (Not Applicable)
#   Evidence Reference: document/system reference
#   Notes: observation notes
#
# Example for AC-001:
#   Status: C
#   Evidence: ServiceNow SR-2024-1234, HR checklist v3.2
#   Notes: All sampled requests (20/20) had manager approval
```

Execute Access Control Checklist

Expected Output

Access Control Checklist – Execution Summary:

```
Compliant:  10 (66.7%)    Non-Compliant:  3 (20.0%)
Partially Compliant:  1 (6.7%)  Not Applicable:  1 (6.7%)

Non-Compliant Controls:
AC-003:  Privileged access review not performed quarterly
AC-007:  Password policy does not enforce complexity
AC-012:  Terminated accounts not disabled within 24h
```

When marking Non-Compliant, document specific evidence. Not “access controls are weak” but rather: “3 of 20 sampled terminated employees retained active accounts for 7+ days post-termination, verified against HR separation records.”

Screenshot 2

What to paste: Terminal showing access control checklist execution with interactive control evaluation prompts (2–3 controls visible) and the final summary with Compliant/Non-Compliant/Partially Compliant/Not Applicable counts.

Paste your screenshot here

Step 6: Execute Change Management and Incident Response Checklists

Objective: Run additional checklists to broaden audit coverage across multiple control domains.

```
# Run change management checklist (12 controls)
python src/cli.py run-checklist \
    --checklist change_management --audit sample
# Key controls: CM-001 (formal change request process),
# CM-004 (emergency change documentation within 48h),
# CM-008 (rollback procedure testing),
# CM-010 (CAB review for high-risk changes)

# Run incident response checklist (10 controls)
python src/cli.py run-checklist \
    --checklist incident_response --audit sample
# Key controls: IR-001 (IR plan current), IR-003 (severity
# classification), IR-006 (post-incident review within 5d),
# IR-009 (regulatory notification timelines)

# Run remaining checklists
python src/cli.py run-checklist --checklist data_backup --audit
   sample
python src/cli.py run-checklist --checklist network_security --
   audit sample
python src/cli.py run-checklist --checklist compliance --audit
   sample
```

Execute Additional Checklists

Expected Output

```
Change Management:  Compliant 8/12 (66.7%), Non-Compliant 3 (25.0%)
Incident Response:  Compliant 7/10 (70.0%), Non-Compliant 2 (20.0%)
Audit progress:  6/6 checklists completed, 67/67 controls assessed
```

Step 7: Document Detailed Audit Findings

Objective: Create structured findings for non-compliant controls using the Condition-Criteria-Cause-Effect-Recommendation format.

```
# Add a HIGH severity finding
python src/cli.py add-finding --audit sample \
    --control AC-003 --severity high \
    --description "Privileged␣access␣reviews␣not␣performed␣
       quarterly␣as␣required.␣Only␣1␣of␣4␣required␣reviews␣
       completed␣in␣past␣12␣months.␣15␣privileged␣accounts␣lack␣
       documented␣justification." \
    --recommendation "Implement␣automated␣quarterly␣privileged␣
       access␣certification.␣Remove␣or␣justify␣all␣15␣unreviewed␣
       accounts␣within␣30␣days."

# Add a CRITICAL severity finding
python src/cli.py add-finding --audit sample \
```

```
    --control AC-012 --severity critical \
    --description "Terminated accounts not disabled within 24-hour window. 8 of 30 sampled terminations (26.7%) had accounts active beyond 72 hours. Two accounts active for 30+ days." \
    --recommendation "Integrate HRMS with Active Directory for automated deprovisioning. Implement daily HR-AD reconciliation."

# Add a MEDIUM severity finding
python src/cli.py add-finding --audit sample \
    --control CM-008 --severity medium \
    --description "Change rollback procedures exist but untested in 12 months. Policy requires annual rollback testing for critical systems." \
    --recommendation "Schedule quarterly rollback drills for Tier-1 systems. Document results in change management KPI dashboard."

# Add a LOW severity finding
python src/cli.py add-finding --audit sample \
    --control IR-006 --severity low \
    --description "Post-incident review reports lack standardized format across teams, hindering trend analysis." \
    --recommendation "Develop standardized PIR template. Train incident commanders on consistent documentation."

# List all findings
python src/cli.py add-finding --audit sample --list
```

Document Audit Findings

Expected Output

```
All findings for audit AUD-2025-001:
FND-001  AC-003  HIGH     Privileged access review gaps
FND-002  AC-012  CRITICAL Terminated account deprovisioning
FND-003  CM-008  MEDIUM   Rollback procedures untested
FND-004  IR-006  LOW      Non-standardized PIR format
Total:  4 findings (1 Critical, 1 High, 1 Medium, 1 Low)
```

Screenshot 3

What to paste: Terminal showing `add-finding` command execution with severity and description visible, plus the `-list` output showing all findings with IDs, control references, and severity levels.

Paste your screenshot here

Part D: Risk Assessment & Reporting

Step 8: Calculate Risk Scores Using the 5x5 Matrix

Objective: Perform quantitative risk assessment using Likelihood x Impact to calculate overall risk, category-level risk, and compliance percentages.

```
# Calculate risk scores for the audit
python src/cli.py calculate-risk --audit sample

# Risk Matrix Scales:
# Likelihood: 1=Rare, 2=Unlikely, 3=Possible,
#             4=Likely, 5=Almost Certain
# Impact: 1=Negligible, 2=Minor, 3=Moderate,
#         4=Major, 5=Catastrophic
# Risk Score = Likelihood x Impact (1-25)
# Levels: 1-4=Low, 5-9=Medium, 10-15=High, 16-25=Critical
```

Calculate Risk Scores

Expected Output

```
================== RISK ASSESSMENT SUMMARY ==================
Overall Risk Score:  14.5 / 25.0 (HIGH)

Risk by Category:
Access Control:    16.0 (CRITICAL)  |  Compliance:  66.7%
Change Management: 12.0 (HIGH)      |  Compliance:  66.7%
Incident Response:  8.0 (MEDIUM)    |  Compliance:  70.0%
Data Backup:        6.0 (MEDIUM)    |  Compliance:  75.0%
Network Security:  10.0 (HIGH)      |  Compliance:  58.3%
Compliance:        15.0 (HIGH)      |  Compliance:  60.0%

Overall Compliance:  65.7% (44/67 controls compliant)

5x5 Risk Matrix (Likelihood x Impact):
5 |  5   10   15   20   25
4 |  4    8   12  [16]  20
3 |  3    6    9   12   15
2 |  2    4    6    8   10
1 |  1    2    3    4    5
+-1--2--3--4--5
```

Findings in the Critical zone (16–25) demand immediate executive attention. In RBI-regulated institutions, Critical findings must be reported to the Board Audit Committee and remediated before the next audit cycle.

Screenshot 4

What to paste: Terminal showing risk calculation output: overall risk score, risk by category, compliance percentages per audit area, and the 5x5 risk matrix visualization.

Paste your screenshot here

Step 9: Generate the Final Audit Report

Objective: Generate an HTML audit report with executive summary, detailed findings, risk heat map, and remediation roadmap.

```
# Generate HTML audit report
python src/cli.py generate-report \
    --audit sample --format html \
    --output audit_report.html

# Open in browser
start audit_report.html    # Windows
# open audit_report.html   # Mac/Linux

# Also generate terminal report for quick review
python src/cli.py generate-report \
    --audit sample --format terminal
```

Generate Audit Report

Expected Output

```
Generating audit report...
Executive Summary:  generated
Findings Detail (4 findings):  generated
Risk Heat Map:  generated
Compliance Scorecard:  generated
Remediation Roadmap:  generated
Report saved to:  audit_report.html (45 KB)
```

Report sections: (1) Executive Summary, (2) Scope & Methodology, (3) Findings Summary, (4) Detailed Findings (5 Cs format), (5) Risk Heat Map, (6) Compliance Scorecard, (7) Remediation Roadmap, (8) Appendices

The audit report is the primary deliverable of an IT audit engagement. In practice, this report is presented to the Board Audit Committee, shared with management for remediation planning, and may be submitted to regulators (RBI, SEBI) as evidence of compliance.

Screenshot 5

What to paste: HTML audit report opened in a web browser showing (a) the executive summary with overall risk score and key metrics, and (b) the risk heat map with findings plotted on the 5x5 matrix.

Paste your screenshot here

Conceptual Background

The IT Audit Lifecycle

An IT audit follows four structured phases:

Phase 1 – Planning: Define scope, assess inherent risks, allocate resources, develop the audit program (checklists and test procedures), and formalize the engagement letter with management.

Phase 2 – Fieldwork (Execution): Gather evidence through inquiry, observation, inspection, and re-performance. Execute checklists, test controls, document exceptions, and discuss interim findings with management.

Phase 3 – Reporting: Draft findings in the 5 Cs format, obtain management responses, assign risk ratings, and produce the final audit report with executive summary and remediation roadmap.

Phase 4 – Follow-Up: Track remediation progress, re-test fixed controls, escalate overdue findings to the Audit Committee, and incorporate lessons learned.

Audit Standards

ISACA ITAF (IT Audit Framework) defines three standards categories: General Standards (1000 series – independence, objectivity, competency), Performance Standards (1200 series – planning, evidence, reporting), and Reporting Standards (1400 series – content, format, distribution).

ISA 315/330: International Standards on Auditing. ISA 315 covers identifying and assessing risks of material misstatement through understanding internal controls. ISA 330 addresses the auditor's responses to assessed risks.

Control Framework Mapping for Audits

Audit Area	COBIT 2019	ISO 27001	PCI-DSS
Access Control	DSS05, APO13	A.9	Req. 7, 8
Change Mgmt	BAI06, BAI07	A.12.1.2	Req. 6.4
Incident Response	DSS02, DSS03	A.16	Req. 12.10
Data Backup	DSS04	A.12.3	Req. 9, 10
Network Security	DSS05	A.13	Req. 1, 2
Compliance	MEA03	A.18	Req. 12

Types of IT Audits

1. **IT General Controls (ITGC):** Foundational controls supporting all IT systems — access controls, change management, computer operations, program development. The most common type and the focus of this lab.
2. **Application Controls:** Controls within specific applications — input validation, processing accuracy, output completeness, authorization checks.
3. **Compliance Audit:** Adherence to specific regulations (RBI guidelines, PCI-DSS, DPDP Act). Scope is dictated by the regulation.
4. **Forensic IT Audit:** Conducted after suspected fraud or breach. Involves evidence preservation, chain of custody, and may support legal proceedings.

Risk Assessment Methodology

IT audit risk involves four components:

- **Inherent Risk:** Risk before controls — driven by environment complexity, data sensitivity, regulatory exposure, and incident history
- **Control Risk:** Risk that controls fail to prevent/detect issues — depends on design effectiveness and operating consistency
- **Detection Risk:** Risk that audit procedures miss material issues — driven by sample size, auditor competence, and procedure design
- **Residual Risk:** Risk remaining after controls: `Residual = Inherent - Control Effectiveness`

The 5x5 Risk Matrix

Level	Likelihood	Impact
5	Almost Certain (>75%)	Catastrophic (existential, >INR 100 Cr penalty)
4	Likely (50–75%)	Major (INR 10–100 Cr, regulatory action)
3	Possible (25–50%)	Moderate (INR 1–10 Cr, service disruption)
2	Unlikely (5–25%)	Minor (<INR 1 Cr, quickly recoverable)
1	Rare (<5%)	Negligible (minimal impact)

Risk Score = Likelihood x Impact. Levels: 1–4 Low (green), 5–9 Medium (yellow), 10–15 High (orange), 16–25 Critical (red).

Audit Evidence Types

1. **Inquiry:** Asking personnel questions. Weakest when alone; must be corroborated.

2. **Observation:** Watching processes in real time. Point-in-time evidence only.
3. **Inspection:** Examining documents, records, configurations, logs. Most common form.
4. **Re-performance:** Independently executing a control to verify results. Strongest form — confirms both design and operating effectiveness.

Audit Finding Structure (The 5 Cs)

Element	Definition	Example
Condition	What was found (current state)	8 of 30 terminated accounts active beyond 72 hours
Criteria	What should be (expected state)	Policy requires deactivation within 24 hours
Cause	Why the gap exists	No HRMS-Active Directory integration
Effect	Consequence (risk/impact)	Unauthorized access risk, potential data exfiltration
Recommendation	What to fix	Implement automated HRMS-AD deprovisioning

India-Specific IT Audit Context

RBI IS Audit Requirements for Banks

- **Annual IS Audit:** Mandatory for all scheduled commercial banks covering core banking, internet/mobile banking, payment systems (UPI, IMPS, NEFT), and ATM networks
- **Cyber Security Framework (2016):** Board-approved cybersecurity policy, C-SOC, regular vulnerability assessments and penetration testing
- **Coverage:** Operating systems, databases, applications, network devices, data centers, DR sites, third-party providers
- **Reporting:** IS audit findings reported to Board Audit Committee with management responses and remediation timelines
- **RBI Inspection:** RBI conducts its own IT examinations and may reference the bank's IS audit reports

SEBI Compliance Audits

- Stock exchanges and depositories: annual system audit by CERT-In empanelled auditors
- Stock brokers: annual cybersecurity audit covering trading systems and client data protection
- Cyber incidents must be reported to SEBI and CERT-In within 6 hours

- Audit trails must be maintained for a minimum of 5 years

Companies Act 2013 Internal Audit

Section 138 mandates internal audit for every listed company, unlisted public companies with paid-up capital of INR 50 Crore+, turnover of INR 200 Crore+, or outstanding loans/borrowings exceeding INR 100 Crore. The internal audit must cover IT systems processing financial transactions.

CAG IT Audits for Government

The Comptroller and Auditor General conducts IT audits of Aadhaar (UIDAI), GST Network, PFMS, and other national platforms. Focus areas include data integrity, system availability, access controls, value-for-money, and cybersecurity of critical infrastructure. CAG IT audit reports are presented to Parliament.

Real-World: Big 4 IT Audits for Indian Banks

Firms like Deloitte, PwC, KPMG, and EY follow a structured approach for Indian bank IT audits: (1) **Scoping** (2–3 weeks) — map the IT landscape including core banking (Finacle, Flexcube), payment gateways, and cloud services; (2) **Risk Assessment** (1–2 weeks) — prioritize based on RBI guidelines and past incidents; (3) **ITGC Testing** (4–6 weeks) — access controls, change management, computer operations, program development; (4) **Application Controls** (2–3 weeks) — input validation, processing, interface controls; (5) **Reporting** (2–3 weeks) — findings with RBI references, presentation to IT Risk Committee; (6) **Follow-Up** — quarterly tracking and re-testing.

Assessment & Deliverables

Assessment Questions

Q1. Describe the four phases of the IT audit lifecycle with key activities and deliverables for each phase.

Q2. Explain inherent risk, control risk, detection risk, and residual risk with examples from an online banking system.

Q3. What is the 5 Cs structure for audit findings? Write a complete sample finding for a password policy violation.

Q4. Compare IT General Controls (ITGC) audits and Application Controls audits — when would you perform each?

Q5. If a finding has Likelihood=4 and Impact=5, what is the risk score, level, and required action?

Q6. List at least four specific RBI IS audit requirements for Indian banks.

Q7. Rank the four audit evidence types from weakest to strongest and justify your ranking.

Q8. Draft a complete finding (all 5 Cs with severity and COBIT/ISO references) for a bank lacking Change Advisory Board review for high-risk changes.

Deliverables Checklist

Item	Description	Type	Status
Screenshot 1	Repo structure, CLI help, checklist sample	Paste	☐
Screenshot 2	Checklist execution (access control)	Paste	☐
Screenshot 3	Finding documentation with severity	Paste	☐
Screenshot 4	Risk matrix and compliance scores	Paste	☐
Screenshot 5	HTML report (exec summary + heat map)	Paste	☐
Findings	All documented findings with severity	Text	☐
Risk Scores	Category-level risk and compliance %	Text	☐
Q&A	Answers to 8 assessment questions	Text	☐

Verification Checklist

☐ Repository cloned and structure verified (6 checklists, 5 source files)

- ☐ Virtual environment created, dependencies installed (Click, Rich, Jinja2)
- ☐ CLI tool verified with all 5 commands via `-help`
- ☐ Audit engagement created with name, scope, and objectives
- ☐ Audit checklists reviewed — control structure understood
- ☐ Access control checklist executed (15 controls evaluated)
- ☐ Change management and incident response checklists executed
- ☐ At least 4 findings documented with severity and recommendations
- ☐ Risk scores calculated using the 5x5 matrix
- ☐ HTML audit report generated and reviewed in browser
- ☐ All 5 screenshots captured and all 8 questions answered

Grading Rubric

Criteria	Description	Points	Score
Setup	Repo cloned, environment configured, CLI verified	10	___/10
Audit Engagement	Engagement created with scope and objectives	10	___/10
Checklist Execution	Access control, change mgmt, incident response	20	___/20
Finding Documentation	Severity, evidence, recommendations (5 Cs)	15	___/15
Risk Assessment	5x5 matrix scores, compliance percentages	15	___/15
Report Generation	HTML report with exec summary and heat map	10	___/10
Assessment Questions	Answers demonstrate audit methodology understanding	15	___/15
Documentation	Screenshots complete, professional presentation	5	___/5
	TOTAL	**100**	___/100

Appendix A: Audit Checklist Quick Reference

Checklist	Controls	Key Areas Covered
Access Control	15	User provisioning/deprovisioning, privileged access, password policy, MFA, access reviews, segregation of duties, session management
Change Management	12	Change request process, CAB review, emergency changes, rollback testing, release management, environment separation, post-implementation review
Incident Response	10	IR plan, severity classification, escalation, communication, evidence preservation, post-incident review, regulatory notification, metrics
Data Backup	8	Backup schedule/retention, offsite storage, encryption, restoration testing, monitoring, DR integration
Network Security	12	Firewall management, segmentation, IDS/IPS, VPN, wireless security, vulnerability scanning, patch management, DDoS protection
Compliance	10	Regulatory mapping, policy framework, monitoring, audit trails, data privacy (DPDP), vendor compliance, reporting obligations

Appendix B: Risk Rating Methodology

Score	Level	Action Required	Remediation
1–4	Low	Monitor and accept	Within 180 days
5–9	Medium	Management attention	Within 90 days
10–15	High	Senior management action	Within 30 days
16–25	Critical	Immediate executive action	Within 7 days

L / I	1	2	3	4	5
5	5 (Med)	10 (High)	15 (High)	20 (Crit)	25 (Crit)
4	4 (Low)	8 (Med)	12 (High)	16 (Crit)	20 (Crit)
3	3 (Low)	6 (Med)	9 (Med)	12 (High)	15 (High)
2	2 (Low)	4 (Low)	6 (Med)	8 (Med)	10 (High)
1	1 (Low)	2 (Low)	3 (Low)	4 (Low)	5 (Med)

Appendix C: Finding Severity Definitions

Severity	Score	Definition	Example
Critical	16–25	Immediate risk of significant loss, breach, or regulatory penalty	Unpatched critical vuln on internet-facing system
High	10–15	Significant control weakness with material impact	Privileged access not reviewed quarterly
Medium	5–9	Moderate weakness requiring management attention	Backup restoration tests incomplete
Low	1–4	Minor observation or improvement opportunity	Inconsistent documentation format

Appendix D: Sample Finding Template

Audit Finding Template

Finding ID: FND-XXX **Control:** [ID] **Severity:** [Critical/High/Medium/Low]
Risk Score: Likelihood x Impact = Score

Condition: [What was found — specific numbers, dates, evidence]
Criteria: [Applicable policy, standard, or regulation]
Cause: [Root cause — process failure, resource gap, technology limitation]
Effect: [Potential consequence — financial loss, regulatory penalty, data exposure]
Recommendation: [Specific remediation with responsible party and timeline]
Management Response: [Agree/disagree, planned actions, target date]
Framework References: [COBIT, ISO 27001, PCI-DSS, RBI guideline]

Appendix E: Audit Report Structure

1. Cover Page (title, period, scope, classification)
2. Executive Summary (overall opinion, key risks, critical findings count)
3. Scope and Methodology (systems, standards, sampling, limitations)
4. Findings Summary Table (all findings by severity)
5. Detailed Findings (full 5 Cs with management responses)
6. Risk Heat Map (5x5 matrix with plotted findings)
7. Compliance Scorecard (per-area percentages with trends)
8. Remediation Roadmap (prioritized actions with owners and timelines)
9. Appendices (evidence inventory, methodology, glossary)

Appendix F: Troubleshooting Guide

(1) Ensure virtual environment is activated (`(venv)` in prompt). (2) Verify Click is installed: `pip list | grep -i click`. (3) Reinstall: `pip install -r requirements.txt`. (4) Check Python 3.8+: `python -version`. (5) Try `python3` instead.

(1) Run commands from the project root directory. (2) Verify files exist: `ls src/checklists/`. (3) Validate JSON: `python -m json.tool src/checklists/access_control.json`. (4) If missing, re-clone the repository.

(1) Verify Jinja2: `pip show jinja2`. (2) Check `templates/` directory exists with template files. (3) Open in Chrome or Firefox. (4) Try `-format terminal` first to verify data. (5) Check terminal for generation errors.

Appendix G: Resources

Standards and Guidelines

→ ISACA ITAF: `https://www.isaca.org/resources/it-audit`

→ ISACA COBIT 2019: `https://www.isaca.org/resources/cobit`

→ ISO/IEC 27001:2022: `https://www.iso.org/standard/27001`

→ PCI-DSS v4.0: `https://www.pcisecuritystandards.org`

→ RBI IT Framework: `https://www.rbi.org.in`

→ SEBI Cybersecurity Circular: `https://www.sebi.gov.in/legal/circulars`

→ Companies Act 2013: `https://www.mca.gov.in`

→ DPDP Act 2023: `https://www.meity.gov.in`

Tools Used in This Practical

Tool	Purpose	Cost
Python 3.8+	Programming language runtime	Free
Click	CLI interface framework	Free
Rich	Terminal formatting and tables	Free
Jinja2	HTML report template engine	Free
pip	Python package manager	Free
Git	Version control and repository cloning	Free

—END OF LAB MANUAL—

Document Version: 1.0

IT Management & Audits – Practical Lab Series