Next Generation Higher National Unit Specification

DevOps Principles and Practice (SCQF level 8)

Unit code:	J897 48
SCQF level:	8 (16 SCQF credit points)
Valid from:	session 2024 to 2025

Prototype unit specification for use in pilot delivery only (version 1.0) June 2024

This unit specification provides detailed information about the unit to ensure consistent and transparent assessment year on year. It is for lecturers and assessors and contains all the mandatory information you need to deliver and assess the unit.

This edition: June 2024 (version 1.0)

Unit purpose

This unit equips learners with fundamental DevOps knowledge and experience, helping them to understand how modern organisations reliably and efficiently deliver and maintain software and implement network infrastructure. DevOps also helps address critical challenges in domains such as cybersecurity and data science. The emphasis on collaboration, automation and reliability helps learners gain insights into efficient practices that go beyond traditional boundaries.

The unit introduces DevOps principles and practices that are applicable in software development and network infrastructure design and implementation. Learners gain exposure to the concepts and tools that enable DevOps workflows to foster cross-functional collaboration across development, quality assurance and operations teams. Through a combination of lectures and hands-on group projects, learners experience DevOps ways of working while leveraging key tools like Git, continuous integration, delivery and deployment (CI/CD) pipelines, infrastructure as code (IaC), and containerisation.

This is a specialist unit intended for learners with an interest in modern approaches to computing in areas such as software development and network infrastructure. Entry is at your centre’s discretion. We recommend that learners have a good understanding of computing concepts, as well as basic experience in writing, testing and deploying computer code, before starting the unit. Basic experience of working with computer networks and cloud computing technologies is also useful.

The unit covers concepts and tools that lay a foundation for learners to continue developing their interpersonal skills and their understanding of modern information technology (IT) operations culture. Learners may progress to other units in software development and network infrastructure at SCQF level 9.

Unit outcomes

Learners who complete this unit can:

1 explain the core concepts relating to DevOps
2 demonstrate version control and repository management
3 describe tools for automating build and test processes
4 describe the processes for deployment, operations, and monitoring
5 create a CI/CD pipeline
6 implement the principles of infrastructure as code (IaC)
7 deploy a containerised application (app)

Evidence requirements

Learners must provide knowledge and product evidence to demonstrate their knowledge and skills across all outcomes.

Knowledge evidence

The knowledge evidence relates to outcomes 1, 3 and 4. Learners must provide evidence for all knowledge and skills statements in these outcomes. The amount of evidence may be the minimum required to infer competence.

The knowledge evidence can be written (for example, a logbook) or oral (audio recordings) or a combination of these. Evidence can be captured, stored and presented in a range of media (including audio and video) and formats.

Product evidence

The product evidence relates to outcomes 2, 5, 6 and 7. It demonstrates that learners have the competence to apply a range of DevOps techniques and tools to implement a full or partial DevOps process, such as:

- screenshots that show they can:
  - commit code changes to a repository
  - create branches
  - merge branches
- a link to a public or private repository with relevant commits
- screenshots that show they can:
  - set up a CI/CD tool
  - configure automated builds and tests
  - deploy stages
  - run pipelines
- logs showing successful builds and deployments

- code snippets relating to:
  - infrastructure code or scripts
  - provisioning resources (such as virtual machines, databases, networks or cloud services) using IaC
  - managing infrastructure changes through code
- documentation explaining the IaC approach
- a Dockerfile for an application
- a Docker image
- yet another markup language (YAML) files defining Kubernetes deployments or similar
- screenshots of running containers

Knowledge and skills

Knowledge	**Skills**
Learners should understand: ♦ software life cycle development ♦ DevOps as the union of people, processes and products to enable continuous delivery of value to end users ♦ general approaches to systems development before DevOps ♦ tensions between development and operations teams (change versus stability) ♦ the limitations of traditional models such as Waterfall ♦ the significance of DevOps in the context of quality assurance, transforming silos, adding value, and continuous development and improvement ♦ the purpose, general functionality and features of Git version control ♦ comparative functionality for different repository systems ♦ the development process and key tooling ♦ the iterative nature of modern development planning ♦ the role of tools such as Azure DevOps and GitHub in the development phase ♦ boards, backlogs, sprints and dependencies ♦ the tools for code creation and management (for example, Maven and Gradle) ♦ iterative testing techniques and tools for test automation (for example, Selenium) ♦ the importance of checking dependencies for security	Learners can: ♦ describe the phases of a typical DevOps project (plan, code, build, test, release deploy, operate, monitor) ♦ describe the basic principles of Agile, as set out in the Agile manifesto, including Scrum and Kanban ♦ create and configure a local or remote Git repository ♦ demonstrate Git actions including init, add, commit, status, config, checkout and remove ♦ demonstrate remote actions with Git, such as clone, push and pull ♦ perform administrative tasks in Git, such as clean, garbage collection and file system check ♦ demonstrate branches and merging with Git ♦ describe tools for build and test automation ♦ describe the processes for release, deployment, operations, and monitoring ♦ access and configure CI/CD tools (for example, GitHub and Azure pipelines) ♦ create a simple Azure DevOps pipeline with sample code from a repository ♦ build a CI/CD environment with appropriate jobs and stages ♦ perform a pipeline run and review the run details ♦ configure an IaC tool (for example Terraform or OpenTofu) ♦ create a project using a suitable IaC tool ♦ manage and provision infrastructure with IaC ♦ automate a process with IaC ♦ perform an IaC monitoring review

Knowledge	**Skills**
Learners should understand: ♦ release pipelines and product verification ♦ activities that take place when preparing a product for the production environment ♦ techniques for testing the product in the production environment ♦ monitoring and telemetry, including product usage, performance indicators and user feedback ♦ the importance of identifying and documenting issues for planning the next iteration ♦ processes and tools to build CI/CD pipelines ♦ review and management of pipeline details ♦ tool configuration (IaC) ♦ managing and provisioning infrastructure ♦ process automation ♦ the monitoring review process ♦ how to differentiate between containerisation versus virtual machines ♦ the advantages of the containerised approach, including security, continuous deployment and portability	Learners can: ♦ perform basic configuration of a containerisation platform (for example, Docker or Kubernetes) ♦ configure and build a simple containerised application using a containerisation platform ♦ share a containerised application using a repository environment (for example Docker Hub or GitHub)

Meta-skills

You must give learners opportunities to develop their meta-skills throughout this unit. We've suggested how to incorporate the most relevant ones into the unit content, but you may find other opportunities.

Self-management

This includes focusing, integrity, adapting and initiative. The most relevant are:

- focusing:
  - prioritising tasks
  - managing deadlines
  - allocating resources
- adapting:
  - learning and adopting new tools
  - responding to shifting priorities
  - accommodating evolving project requirements
- initiative:
  - motivating and influencing others
  - setting a positive example
  - driving initiatives forward

Social intelligence

This includes communicating, feeling, collaborating and leading. The most relevant are:

- communicating:
  - conveying technical information clearly to both technical and non-technical stakeholders
  - discussing issues and development ideas
- feeling:
  - creating a positive environment by understanding the perspectives and feelings of others, particularly in relation to the experiences and concerns of colleagues
- collaborating:
  - discussing different aspects of the project and actively listening to the opinions of others
  - collaborating with team members and working effectively with people from different backgrounds, respecting their knowledge and achieving shared goals
- leading:
  - motivating and influencing others
  - setting a positive example
  - driving initiatives forward

Innovation

This includes curiosity, creativity, sense-making and critical thinking. The most relevant are:

- sense-making:
  - problem solving, such as researching and learning the appropriate tooling for a particular stage of the DevOps process
- critical thinking:
  - objectively assessing situations, evaluating information and making informed decisions

Learning for Sustainability

Throughout this unit, you should encourage learners to develop their skills, knowledge and understanding of sustainability.

This includes:

- a general understanding of social, economic and environmental sustainability
- a general understanding of the United Nations Sustainable Development Goals (SDGs)
- a deeper understanding of subject-specific sustainability
- the confidence to apply the skills, knowledge, understanding and values they develop in the next stage of their life

DevOps enables efficient and effective development of software or building network infrastructure, with positive contributions to one or more of the UN SDGs.

Delivery of unit

This unit is an optional unit in Higher National Diploma (HND) Software Development (SQCF level 8) and HND Network Infrastructure and Cloud Computing (SCQF level 8).

The notional time for delivery and assessment is 80 hours. The amount of time you allocate to each outcome is at your centre's discretion. We suggest the following distribution of time, including assessment:

Outcome 1 — Explain the core concepts relating to DevOps
(10 hours)

Outcome 2 — Demonstrate version control and repository management
(10 hours)

Outcome 3 — Describe tools for automating build and test processes
(10 hours)

Outcome 4 — Describe the processes for deployment, operations, and monitoring
(10 hours)

Outcome 5 — Create a CI/CD pipeline
(15 hours)

Outcome 6 — Implement the principles of infrastructure as code (IaC)
(10 hours)

Outcome 7 — Deploy a containerised application (app)
(15 hours)

Additional guidance

The guidance in this section is not mandatory.

Content and context for this unit

Explain the core concepts relating to DevOps (outcome 1)

This outcome provides learners with a foundational understanding of DevOps. It focuses on:

- the key concepts, philosophy and history of DevOps
- the stages of a typical DevOps project
- constituent development methodologies, such as Agile and Kanban

You could tackle common misconceptions around DevOps at this stage. You could explain that DevOps is not an actual product, but a flexible set of tools and methodologies that development teams can select and apply as needed for continuous product delivery and improvement. You could set out a working definition, such as that of Microsoft DevOps Program Manager Donovan Brown: 'DevOps is the union of people, process, and products to enable continuous delivery of value to our end users.'

Demonstrate version control and repository management (outcome 2)

The ability to manage files and repositories using version control is a core aspect of DevOps. In outcome 2, learners understand how to manage source code and other important files using a version control system. Popular 'repo' systems include open-source options such as Git and GitHub. Other options include GitLab and Gitea.

Describe tools for automating build and test processes (outcome 3)

Outcome 3 is a knowledge-based outcome where learners understand key development processes and tooling, and where these fit in the development phase of a DevOps project. At the time of writing, contemporary tools for managing the development phase include Azure DevOps and GitHub Actions, and tools for code creation and test automation such as Maven, Gradle and Selenium. This is not a prescriptive or exhaustive list, and new tools and features are likely to develop over time. Although practical application of these tools is not essential, learners should be able to describe their general purpose and understand where and how they are likely to be implemented in a typical DevOps project.

Describe the processes for deployment, operations, and monitoring (outcome 4)

Outcome 4 is a knowledge-based outcome that focuses on the operational side of DevOps, including processes and tools for preparing a product for release and deployment, and monitoring and reporting mechanisms for continuous product improvement.

Create a CI/CD pipeline (outcome 5)

Outcome 5 focuses on CI/CD. It builds on the DevOps principles of continuous and cyclical planning, development, collaboration and delivery. It provides a framework on which incremental changes — particularly to code files — can be made frequently and reliably in the wider context of a DevOps project. You should highlight the benefits of CI/CD pipelines, particularly that they can promote improved efficiency, leading to faster product development and lower costs.

Implement the principles of infrastructure as code (IaC) (outcome 6)

Outcome 6 is a practical outcome in which learners demonstrate the principles of IaC.

Deploy a containerised application (app) (outcome 7)

Outcome 7 introduces learners to containerisation. They should be able to define containerisation and differentiate between containerisation and virtual machines. They should also understand the advantages of a containerised approach over virtualisation, including enhanced security, continuous deployment and portability.

Resources

You should consider open-source software alternatives including, but not limited to:

- GitLab (GitHub alternative)
- Gitea (Git repository)
- Podman (container tooling)
- WeKan (Kanban alternative)
- OpenTofu (TerraForm alternative)

Approaches to delivery

Explain the core concepts relating to DevOps (outcome 1)

You could provide learners with a historical overview of general systems development before DevOps. You could highlight traditional tensions and conflicting goals between development and operations teams — developers tend to like creating new things, whereas those in operations veer towards maintaining continuity and stability.

You could also discuss the limitations of more rigid approaches to systems development, such as the Waterfall methodology. Early approaches to systems development tended to have long project delivery timelines with limited customer involvement. This sometimes led to issues such as misinterpreted requirements, scope creep and increased costs.

Learners benefit from a foundational understanding of the stages of a DevOps project, and you could highlight these stages. For a release, these typically include:

- development stages:
  - planning
  - coding
  - building
  - testing

- operational stages:
  - deployment
  - ongoing operational monitoring.

Demonstrate version control and repository management (outcome 2)

You should explain the reasons for version control, particularly in relation to how repositories enable collaboration and consistency for teams who work on multiple versions of the same file over time. You should highlight capabilities that allow teams to centrally store and track files with metadata such as labels, version numbers, comments and notes relating to historical changes.

This outcome is practical in nature. Learners should be able to:

- set up and configure a local or remote repository
- demonstrate actions appropriate to the chosen platform (for example, add, commit, checkout, push and pull)
- demonstrate branching and merging
- perform administrative and clean-up actions such as file system checks and garbage collection

Depending on the context of delivery, you could supply learners with sample files to manage. If they are working as a collaborative team, they could produce their own source files to manage. An innovative approach could include configuring a version control system to work alongside a code editor such as Visual Studio Code. Depending on the context of delivery, the type of source files used may vary, for example, students studying networking may choose to work with PowerShell scripts, or other source files relevant to them.

In addition to using the general functionality and features of a version control system, learners should be able to identify basic differences between popular version control systems. They should know the difference between Git repositories which can be installed and run locally, and cloud-based Git services.

Describe tools for automating build and test processes (outcome 3)

You should introduce learners to the iterative nature of DevOps planning, processes and terminology, particularly those that fit with the Agile methodology. This should include boards, backlogs, sprints and dependencies. At the time of writing, security of dependencies is a relevant topic, and learners benefit from understanding the importance of maintaining them.

Describe the processes for deployment, operations, and monitoring (outcome 4)

You should teach learners the purpose and significance of a product release pipeline and where it features in the release and deployment stage. Learners should be able to describe general activities that take place when preparing a product for integration into the production environment, including product verification and testing.

You should teach learners the processes that take place during the latter stage of the operating phase, particularly when a product has been deployed and released. These may include different mechanisms for monitoring how users are interacting with the product, and other feedback, telemetry and reporting mechanisms. You should emphasise the value of gathering this information, particularly in terms of its significance for continuous product development and improvement.

Learners should be able to identify and describe contemporary tooling for the operational phases of DevOps. At the time of writing, tools associated with pipeline and release management include pipeline features in Azure DevOps and GitHub Actions. Although practical application is not essential, learners should be able to identify contemporary tools and describe how they support the operational phase.

Create a CI/CD pipeline (outcome 5)

This outcome is practical in nature. Learners should be able to construct a simple example CI/CD pipeline using an appropriate development tool. The tooling used is at your centre's discretion, but it should allow learners to:

- create a simple workflow or pipeline with sample code from a repository (supplied to learners or created themselves)
- define an environment with relevant jobs and stages
- review and manage the pipeline details
- perform and review a pipeline run

At the time of writing, tools that support development of CI/CD pipelines include GitHub Actions, Azure Pipelines and GitLab.

Implement the principles of infrastructure as code (IaC) (outcome 6)

When demonstrating IaC principles, you could provide learners with sample code, or they can use code that they have developed themselves (for example, as part of a group project). The choice of IaC tool is at your centre's discretion. At the time of writing, contemporary tooling includes tools such as Terraform and open-source alternatives such as OpenTofu. The nature of the infrastructure could vary depending on the context for delivery, for example, networking students could define and build a simple cloud infrastructure with a virtual private cloud (VPC) and subnets using code.

Learners should demonstrate that they can:

- create a project using a suitable IaC tool
- manage and provision code
- automate a sample process
- perform suitable monitoring and review processes

Deploy a containerised application (app) (outcome 7)

Outcome 7 is a practical outcome, and to get started learners should grasp the basics of configuring a containerisation platform. At the time of writing, popular platforms include Docker and Kubernetes. In addition, centres are free to explore open-source alternatives such as Podman.

When familiar with a suitable environment, learners should create a simple containerised application by configuring and building it using their chosen platform. They should then share the application with others using a repository environment such as Docker Hub or GitHub.

Approaches to assessment

Learners can produce knowledge evidence over an extended period in lightly-controlled conditions. The knowledge evidence may be written (for example, a logbook), oral (audio recordings) or a combination of these. Learners can capture, store and present evidence in a range of media (including audio and video) and formats. All actions taken in a repository are captured and logged automatically, so the link to the repository will suffice as evidence.

Alternatively, you can produce evidence using testing, for example, with a closed-book test consisting of multiple-choice questions. In this case, learners must produce evidence under controlled conditions in terms of location, timing and access to reference materials. The sampling frame must cover outcomes 1, 3 and 4, and cover a major part of the knowledge, although it doesn’t have to include all knowledge statements.

Generating product evidence

When generating the product evidence, you can take a stand-alone or integrated approach, at your centre’s discretion.

Stand-alone approach

Learners can work independently or in a group to conduct a series of pre-defined practical tasks that form a small DevOps project. You should design the tasks to generate evidence of the required skills and incorporate appropriate processes and tooling (for example Azure DevOps, GitHub, Terraform and Docker). Learners can capture evidence in a variety of ways, such as annotated screenshots and videos, and present it in the form of a written logbook or presentation.

Integrated approach

Learners can generate product evidence through a collaborative project. This can integrate the skills requirements of the practical outcomes (2, 5, 6 and 7) with other project work from learners’ programme of study. This could be achieved through combined assessment with other units from the HND framework. When designing an integrated assessment, we recommend that mapping is done to cross-reference and check that it covers all the knowledge and skills of the constituent units.

Equality and inclusion

This unit is designed to be as fair and as accessible as possible with no unnecessary barriers to learning or assessment.

You must consider the needs of individual learners when planning learning experiences, selecting assessment methods or considering alternative evidence.

Guidance on assessment arrangements for disabled learners and those with additional support needs is available on the assessment arrangements web page.

Information for learners

DevOps Principles and Practice (SCQF level 8)

This information explains:

- what the unit is about
- what you should know or be able to do before you start
- what you need to do during the unit
- opportunities for further learning and employment

Unit information

This unit equips you with fundamental DevOps knowledge and experience, to help you understand how modern information technology (IT) organisations deliver software and implement network infrastructure rapidly, reliably and collaboratively.

You are introduced to DevOps principles and practices that are applicable in software development and network infrastructure design and implementation. You learn the concepts and tools that enable DevOps workflows to foster cross-functional collaboration across development, quality assurance and operations teams. Through a combination of conceptual overviews and hands-on group projects, you experience DevOps ways of working while using key tools like Git, continuous integration, delivery and deployment (CI/CD) pipelines, Infrastructure as Code (IaC), and containerisation.

Before starting the unit, you should have a good understanding of computing concepts and basic experience in writing, testing and deploying computer code. Basic experience of working with computer networks and cloud computing technologies is also useful.

You are assessed through knowledge evidence and product evidence. You demonstrate knowledge evidence in a manner determined by your centre, using approaches such as a portfolio that you assemble or an exam. Product evidence includes elements such as screenshots of your implementation of DevOp methods and tools, a link to the repository log, code snippets, and documentation.

You gain a sound foundational knowledge of DevOps concepts and methods, and experience using DevOps tools to build products that support collaboration and process automation. You can build on this foundation to further develop your DevOps skills as you advance in your studies in computing and in your career. You may progress to other units in software development and network infrastructure at SCQF level 9.

Meta-skills

Throughout this unit, you develop meta-skills for the computing sector.

Meta-skills are transferable behaviours and abilities that help you adapt and succeed in life, study and work. There are three categories of meta-skills: self-management, social intelligence and innovation.

Self-management

This meta-skill includes:

- ♦ focusing:
  - — prioritising tasks
  - — managing deadlines
  - — allocating resources
- ♦ adapting:
  - — learning and adopting new tools
  - — responding to shifting priorities
  - — accommodating evolving project requirements
- ♦ initiative:
  - — motivating and influencing others
  - — setting a positive example
  - — driving initiatives forward

Social intelligence

This meta-skill includes:

- ♦ communicating:
  - — conveying technical information clearly to both technical and non-technical stakeholders
  - — discussing issues and development ideas
- ♦ feeling:
  - — creating a positive environment by understanding the perspectives and feelings of others, particularly in relation to the experiences and concerns of colleagues
- ♦ collaborating:
  - — discussing different aspects of the project and actively listening to the opinions of others
  - — collaborating with team members and working effectively with people from different backgrounds, respecting their knowledge and achieving shared goals
- ♦ leading:
  - — motivating and influencing others
  - — setting a positive example
  - — driving initiatives forward

Innovation

This meta-skill includes:

- ♦ sense-making:
  - — problem solving, such as researching and learning the appropriate tooling for a particular stage of the DevOps process

- critical thinking:
  - objectively assessing situations, evaluating information and making informed decisions

Learning for Sustainability

Throughout this unit, you develop skills, knowledge and understanding of sustainability.

You learn about social, economic and environmental sustainability principles and how they relate to the computing sector. You also develop an understanding of the United Nations Sustainable Development Goals.

DevOps enables efficient and effective development of software or building network infrastructure, with positive contributions to one or more of the UN SDGs.

Administrative information

Published: June 2024 (version 1.0)

Superclass: CB

History of changes

Version	Description of change	Date

Please check SQA’s website to ensure you are using the most up-to-date version of this document.