Next Generation Higher National Unit Specification

Code Security (SCQF level 8)

Unit code: J7EB 48

SCQF level: 8 (16 SCQF credit points)

Valid from: session 2023–2024

Prototype unit specification for use in pilot delivery only (version 1.0) June 2023

This unit specification provides detailed information about the unit to ensure consistent and transparent assessment year on year.

This unit specification is for teachers and lecturers and contains all the mandatory information required to deliver and assess the unit.

This edition: June 2023 (version 1.0)

Unit purpose

This unit enables learners to develop knowledge of the theoretical concepts, underlying principles, scope, and role of systems analysis and design, focusing on security requirements use cases. Learners also develop problem solving, documentation and programming skills through planning and implementation of secure coding practices.

Learners will demonstrate their proficiency in these skills by implementing an appropriate threat model and documenting the process.

This is a specialist unit, suitable for learners with an interest in taking a more secure approach to programming in a software development environment. It is particularly suitable for learners with a vocational interest in programming and cyber security subjects, or who wish to progress to higher education and/or vendor qualifications. Before starting the unit, learners should have experience in computer programming at SCQF level 7 or above, and good written communication, critical thinking, and analytical skills. Previous experience of software development is desirable.

On completion of the unit, learners may progress to more specialised topics, such as software engineering, data engineering, data science, machine learning or secure programming-based subjects at SCQF level 8 or above. Learners may also progress to a computing related vendor qualification or certification.

Unit outcomes

Learners who complete this unit can:

1 create security-focused software design documentation from a given software scenario
2 create a threat modelling plan
3 code a software application using secure coding practices
4 implement a threat modelling plan
5 apply secure threat mitigation solutions to the codebase of the software application
6 perform security testing on a secure threat mitigation solution

Evidence requirements

Learners must provide product evidence. Knowledge is inferred from the product evidence.

Product evidence will be generated by learners in their work on a software development project, with a focus on security-based use cases. The project must involve a system based on a real-life problem, and the scenario must require a security-based design and secure coding. The centre will provide the project scenario. The product evidence must be the work of the individual learner.

Learners must provide the following evidence:

- user requirements with a focus on security use cases
- a threat model that includes the analysis of each of the security-focused use cases identified in the user requirements
- a software application that wholly depends on the security use cases
- identification of threat mitigation and application of threat mitigation to the codebase
- a test plan and results to show that the learner has effectively implemented threat mitigation

The tutor's role in the conduct of the software project is to:

- provide the software development brief for learners
- clarify assumptions and problem statements from the brief
- validate the tests provided by learners

Learners can produce evidence over an extended period in lightly-controlled conditions or generate it in conjunction with other software development units in a group award. Evidence produced in lightly-controlled conditions must be authenticated. The Guide to Assessment provides further advice on methods of authentication.

We recommend that the software brief is changed regularly. The standard of evidence should be consistent with the SCQF level of the unit.

Knowledge and skills

The following table shows the knowledge and skills covered by the unit outcomes:

Knowledge	Skills
Learners should understand: ♦ the software development lifecycle model ♦ software design documentation ♦ use case and descriptions ♦ threat modelling ♦ code minification and obfuscation ♦ access control ♦ version control ♦ the Open Web Application Security Project (OWASP) Foundation ♦ how to identify known vulnerabilities ♦ open-source components and libraries ♦ how to prevent vulnerabilities ♦ writing and compiling code ♦ testing strategies ♦ threat mitigation measures	Learners can: ♦ determine scope and depth of analysis by capturing security requirements ♦ model attack possibilities ♦ identify security threats ♦ perform code minification and obfuscation ♦ enforce encryption ♦ use version control ♦ prevent vulnerabilities ♦ perform code reviews ♦ perform auditing and logging ♦ identify and replace components with known vulnerabilities ♦ create use cases and use case descriptions with consideration to security ♦ apply code validation and exception handling ♦ use a range of testing strategies

Meta-skills

Throughout this unit, learners develop meta-skills to enhance their employability in the software development sector.

Self-management

This meta-skill includes:

- focusing: analysing a specification to filter and sort information on security requirements; understanding which information is non-essential to the problem at hand; applying solutions to a client's security requirements with varying degrees of complexity
- adapting: critically reflecting on new knowledge to attain a deeper understanding and embed and extend learning
- initiative: displaying independent thinking

Social intelligence

This meta-skill includes:

- communicating: receiving information through various means and interpreting it; sharing information about software solutions to a range of audiences, both technical and non-technical
- collaborating: listening and conveying information
- leading: being a change catalyst

Innovation

This meta-skill includes:

- curiosity: questioning constructively to identify requirements for a software application; information sourcing; problem recognition
- creativity: idea generation; recognising problem types and constructing solutions; problem solving
- sense-making: synthesis and analysis; seeing the bigger picture
- critical thinking: breaking down large complex problems to support daily task management and contribution to projects; logical and computational thinking

Delivery of unit

The time required varies depending on the previous experience of individual learners.
Based on 80 hours delivery and assessment time, we suggest the following distribution:

Outcome 1 — Create security-focused software design documentation from a given software scenario
(10 hours)

Outcome 2 — Create a threat modelling plan
(10 hours)

Outcome 3 — Code a software application using secure coding practices
(20 hours)

Outcome 4 — Implement a threat modelling plan
(10 hours)

Outcome 5 — Apply secure threat mitigation solutions to the codebase of the software application
(20 hours)

Outcome 6 — Perform security testing on a secure threat mitigation solution
(10 hours)

Learners require access to appropriate hardware and software throughout the unit.

Additional guidance

The guidance in this section is not mandatory.

Content and context for this unit

In the unit, you teach learners how to:

- build security into the software development lifecycle
- write code more securely in a version-controlled environment
- perform threat modelling
- identify vulnerabilities
- use threat prevention techniques

The unit provides learners with an opportunity to:

- design a software solution, focusing on the security requirements
- create a threat modelling plan to identify, communicate and understand threats
- apply threat mitigation techniques to the implementation
- perform testing

Learners understand the importance of creating more secure code and the benefit this can add to a software development project. The unit should provide a hands-on approach to implementing a threat modelling plan. It should also examine a range of testing strategies and highlight the importance of testing while developing and deploying software applications. Learners understand the value of planning, and the application of more secure coding techniques. They experience the challenges of working as a software developer.

The unit provides detailed underpinning knowledge that learners need to carry out practical activities when progressing to advanced computing subjects.

On delivering the unit, you should allow for a range of scenarios and practical activities whenever possible to allow learners to gain skills in developing a computing problem, selecting a programming language, and then implementing and testing code based on a threat modelling plan. Learners should have access to a range of programming languages and associated development environments.

Approaches to assessment

Learners can complete the unit stand-alone or as part of a group award. Where the unit is part of a group award, you can combine assessment with other units in the award.

The unit's general context is for learners to apply secure software development principles to security-focused use cases. Assessment is based on a core learning activity, which is a software development project broken into the stages identified in the evidence requirements.

We recommend that you assess the unit with a single project covering all six unit outcomes. Your choice of project brief must permit the full range of evidence to be generated. Of possible you should offer more than one brief to give learners a degree of choice.

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 should take into account the needs of individual learners when planning learning experiences, selecting assessment methods or considering alternative evidence.

Guidance on assessment arrangements for disabled learners and/or those with additional support needs is available on the assessment arrangements web page: www.sqa.org.uk/assessmentarrangements.

Information for learners

Code Security (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 provides you with the knowledge and skills that you need to design and develop software code that is secure and resilient to known threats. You learn how to:

- build security into the software development lifecycle
- write code more securely in a version-controlled environment
- perform threat modelling to identify code vulnerabilities
- apply threat prevention techniques

To benefit from the unit, you should have a knowledge of software development, and experience of developing and testing programme code at SCQF level 7 or above.

You learn the importance of creating secure code and its benefits for a software development project. You gain the practical skills you need to implement a threat modelling plan and apply testing strategies through a hands-on approach to learning. You experience the challenges of working as a software developer.

You are assessed on product evidence you produce in the context of a software development project, with a focus on security-based use cases. You will be supplied with a brief for this project, covering all the outcome of the unit.

Throughout the unit, you develop meta-skills covering self-management, social intelligence and innovation.

Successful completion of the unit enables you to progress to higher level units in software development and cyber security.

Administrative information

Published: June 2023 (version 1.0)

Superclass: CB

History of changes

Version	Description of change	Date

Note: please check SQA's website to ensure you are using the most up-to-date version of this document.