Software Engineering Stage 6 (Year 12) – sample program of learning

Secure software architectureContents

[About this resource 3](#_Toc183426400)

[Purpose of resource 3](#_Toc183426401)

[Target audience 3](#_Toc183426402)

[When and how to use 3](#_Toc183426403)

[Rationale 4](#_Toc183426404)

[Overview 5](#_Toc183426405)

[Outcomes 7](#_Toc183426406)

[Lesson sequence and details 9](#_Toc183426407)

[Weeks 1 and 2 9](#_Toc183426408)

[Weeks 3–4 14](#_Toc183426409)

[Weeks 5–6 18](#_Toc183426410)

[Weeks 7–9 24](#_Toc183426411)

[Week 10 30](#_Toc183426412)

[Overall program evaluation 33](#_Toc183426413)

[Capturing student voice when evaluating a program 33](#_Toc183426414)

[Additional information 35](#_Toc183426415)

[Further implementation support 35](#_Toc183426416)

[Assessment for learning 35](#_Toc183426417)

[Differentiation 36](#_Toc183426418)

[Support and alignment 38](#_Toc183426419)

[Evidence base 40](#_Toc183426420)

[References 42](#_Toc183426421)

# About this resource

## Purpose of resource

The resource is a sample program of learning for teaching Secure software architecture in Year 12 during the Software Engineering 11–12 course.

## Target audience

This resource can be used by teachers to support effective syllabus implementation of Software Engineering 11–12.

## When and how to use

This resource is designed for implementing over 10 weeks or a term of learning on Secure software architecture. The resource can be adapted and contextualised to the school setting. Adjustments can be made to the program of learning to suit students in the teaching and learning cycle.

# Rationale

The NSW Department of Education publishes a range of curriculum support materials, including samples of lesson sequences, scope and sequences, assessment tasks, examinations, student and teacher resource booklets, and curriculum planning and curriculum evaluation templates. The samples are not exhaustive and do not represent the only way to complete or engage in each of these processes. Curriculum design and implementation is a dynamic and contextually specific process. While the mandatory components of syllabus implementation must be met by all schools, it is important that the approach taken by teachers is reflective of their needs and faculty or school processes.

NESA defines [programming](https://educationstandards.nsw.edu.au/wps/portal/nesa/k-10/understanding-the-curriculum/programming) as the process of ‘selecting and sequencing learning experiences which enable students to engage with syllabus outcomes and develop subject specific skills and knowledge’ (NESA 2022a). A program is developed collaboratively within a faculty. It differs from a unit in important ways, as outlined by NESA on their [Advice on units](https://educationstandards.nsw.edu.au/wps/portal/nesa/k-10/understanding-the-curriculum/programming/advice-on-units) page. A unit is a contextually specific plan for the intended teaching and learning for a particular class for a particular period. The organisation of the content in a unit is flexible and it may vary according to the school, the teacher, the class and the learning space. They should be working documents that reflect the thoughtful planning and reflection that takes place during the teaching and learning cycle. There are mandatory components of programming and unit development, and this template provides one option for the delivery of these requirements. The NESA and department guidelines that have influenced this template are elaborated upon at the end of the document.

This resource has been developed to assist teachers in NSW Department of Education schools to create learning that is contextualised to their classroom. It can be used as a basis for the teacher’s own program, assessment, or scope and sequence, or be used as an example of how the new curriculum could be implemented. The resource has suggested timeframes that may need to be adjusted by the teacher to meet the needs of their students.

# Overview

**Description**: this program of learning addresses the Secure software architecture focus area. The lessons and sequences in this program of learning are designed to guide students through the development of secure software and a security report. Students are provided a scenario, client and code. They play the role of a security specialist to identify and correct the vulnerabilities in the code and produce a security report. This project is built to a client’s specifications, using one of the software development approaches and implementation methods outlined in the syllabus.

Teachers are provided options to differentiate the project depending upon their confidence and the experience of their students. At a minimum, students complete the security report template for the scenario. Students attempt coding exercises to make the unsecure progressive web app (PWA) secure.

Teachers planning to integrate this focus area with the Programming for the web and/or the Software Engineering project may offer the students the opportunity to develop a secure app from scratch using Python and the Flask framework. ‘How to’ guides are available for each of these differentiated tasks in the Appendix of the teacher support resource (TSR). Assessment tasks that accompany these projects, including: Steps to success, Marking guidelines and Student facing rubrics, can be found in the **Secure Software** folder under the **files** tab of the [Software Engineering channel](https://schoolsnsw.sharepoint.com/:f:/r/sites/TASNSWStatewideStaffroom/Shared%20Documents/13.%20%F0%9F%A7%91%E2%80%8D%F0%9F%92%BB%20Software%20Engineering%2011-12/Secure%20Software%20Architecture?csf=1&web=1&e=6nkPE3) via the TAS State wide staffroom.

Designing software: students apply the fundamental software development steps to develop secure code. They describe the benefits of developing secure software to the client (The Unsecure PWA Company). Students provide expert advice to the ‘The Unsecure PWA Company’ on the privacy and security of their PWA. They list all the security issues found and explain how bad they could be if someone takes advantage of them. Students write a requirements definition for the client. They identify the user specifications for the solution.

Developing secure code: students check the PWA for security issues using special tools and tests to find any problems. They investigate how the PWA is built to find any security weaknesses in the code and how data is handled. Students find any problems that cannot be fixed by changing the code, like how users behave, or how data is managed. They make changes to the PWA's code and settings to fix the security issues found, like checking user inputs, encrypting data, and controlling who can access what. Students use appropriate HTML/CSS/JS/SQL/JSON/Python code and web content changes to provide a close-to-industry standard solution that fully or near fully mitigates security and privacy vulnerabilities. They apply strategies to manage the security of programming code. Students test and evaluate the security and resilience of the software.

**Impact of safe and secure software development**: students present their solutions and security report to the client (role-played by teachers and classmates). They explain the benefits to the client of the implementation of safe and secure development practices. Students explain the social, ethical and legal issues as well as ramifications that affect people and enterprises resulting from the development and implementation of safe and secure software.

**Duration**: the content for this focus area (code samples and security report) can be delivered over 30 hours across one term. Alternatively, it could be integrated with the Programming for the web focus area (making the PWA secure and completing the security report) across 60 hours. A further option is to integrate this focus area throughout the Programming for the web and Software Engineering project (creating a secure PWA from scratch and completing the report) delivered over 90 hours. Each of these models are available in the Appendix of the accompanying teacher support resource.

**Explicit teaching**: suggested learning intentions and success criteria are available for some lessons provided. Learning intentions and success criteria are most effective when they are contextualised to meet the needs of students in the class. The examples provided in this document are generalised to demonstrate how learning intentions and success criteria could be created.

# Outcomes

A student:

* justifies methods used to plan, develop and engineer software solutions **SE-12-01**
* applies structural elements to develop programming code **SE-12-02**
* analyses how current hardware, software and emerging technologies influence the development of software engineering solutions **SE-12-03**
* evaluates practices to safely and securely collect, use and store data **SE-12-04**
* explains the social, ethical and legal implications of software engineering on the individual, society and the environment **SE-12-05**
* justifies the selection and use of tools and resources to design, develop, manage and evaluate software **SE-12-06**
* designs, develops and implements safe and secure programming solutions **SE-12-07**
* tests and evaluates language structures to refine code **SE-12-08**
* applies methods to manage and document the development of a Secure software architecture **SE-12-09**

[Software Engineering 11–12 Syllabus](https://curriculum.nsw.edu.au/learning-areas/tas/software-engineering-11-12-2022/overview) © NSW Education Standards Authority (NESA) for and on behalf of the Crown in right of the State of New South Wales, 2022.

**Prior to planning for teaching and learning, please consider the following**:

**Engagement**

* How will I provide authentic, relevant learning opportunities for students to personally connect with lesson content?
* How will I support every student to grow in independence, confidence and self-regulation?
* How will I facilitate every student to have high expectations for themselves?
* How will I identify and provide the support each student needs to sustain their learning efforts?

**Representation**

* What are some different ways I can present content to enable every student to access and understand it?
* How will I identify and address language and/or cultural considerations that may limit access to content for students?
* How will I make lesson content and learning materials more accessible?
* How will I plan learning experiences that are relevant and challenging for the full range of students in the classroom?

**Expression**

* How will I provide multiple ways for students to respond and express what they know?
* What tools and resources can students use to demonstrate their understanding?
* How will I know every student has understood the concepts and language presented in each lesson?
* How will I monitor if every student has achieved the learning outcomes and learning growth?

# Lesson sequence and details

**Teachers provide specific, actionable feedback throughout the learning process, not just at the end of a project. This could involve real-time feedback during practical tasks, or reflective discussions post-completion of stages in the Secure software architecture development.**

## Weeks 1 and 2

Table 1 – designing software lesson sequence and details

| **Outcomes and content** | **Teaching and learning activities** | **Evidence of learning** | **Differentiation and adjustments** | **Registration and evaluation notes** |
| --- | --- | --- | --- | --- |
| **Outcome**  **SE-12-01**  **SE-12-04**  **SE-12-06**  **Designing software**  Students:   * describe the benefits of developing secure software * interpret and apply fundamental software development steps to develop secure code * describe how the capabilities and experience of end users influence the secure design features of software. | Teacher determines whether to run this unit as a ‘standalone’ or integrate into other focus areas.  Students:   * complete a pre-test on fundamental concepts within Secure software architecture. Teachers may allocate groups preading the expertise based on the results of this pre-test. * develop an overview of the latest trends impacting the internet and cybersecurity.   Teacher introduces and explains the task:  Your client, ‘The Unsecure PWA Company’, has engaged you as a software engineering security specialist to provide expert advice on the security and privacy of their application. This progressive web app is currently in the testing and debugging phase of the software development lifecycle and can be accessed at [Unsecure PWA](https://github.com/TempeHS/The_Unsecure_PWA).  For the security report PWA project, students familiarise themselves with the existing [Unsecure PWA](https://github.com/TempeHS/The_Unsecure_PWA)  **Learning intentions**  **By the end of the pre-test, review and corrections, students will be able to:**   * define secure software architecture and list benefits of developing secure software * explain why data protection in secure software development is important and acknowledge how software architecture can help minimise cyber attacks and vulnerabilities * identify the fundamental software development steps and requirements definition in software development.   **By the end of their research activities and slide deck production, students will be able to recognise key software architecture concepts,** including: data protection, cyber attacks, static application security testing (SAST), dynamic application security testing (DAST), vulnerability assessment, penetration testing, application programming interface (API) security, cross-site scripting (XSS), cross-site request forgery (CSRF).  **Teachers introduce students to secure software architecture questions within the** [Familiarisation exam](https://fam.hsconline.nesa.nsw.edu.au/)  **Success criteria**  **I can:**   * **explain the importance of secure software architecture** * **recognise some of the vulnerabilities of insecure code.**   **Teaching and learning activities**  **Teachers distribute the Security report template (Appendix 1 of the TSR) and display** code and readme files from the [Unsecure PWA](https://github.com/TempeHS/The_Unsecure_PWA).  Students work through Activities 1–13 of the TSR. | Students:   * share a common technical language in discussing their understanding of secure software architecture * correctly match glossary keywords with definitions * teams create a single slide summary of each of the latest trends impacting the internet and cybersecurity including a definition, characteristics and an example. * contribute to class discussions and respond to the revision quiz at the beginning of each lesson and complete Activities 1–13 of the teacher support resource (TSR).   Students:   * begin work on the security report template – the documentation component of their project for this focus area * complete activities within the teacher support resource (TSR) on minimising cyber attacks and vulnerabilities * describe to their client, ‘The Unsecure PWA Company’ the benefits of developing secure software * use the terminology and language specific to this focus area including: defining, characterising and providing an example of key concepts. These are referred to as students complete their coding activities around each concept. * Complete Activities 1–13 of the TSR and updated security report. | Students are provided course specifications, glossaries, security report writing scaffold, first-language dictionaries and exemplar answers or responses so that they know what and how to write for each section of the project documentation.  Student teams match glossary terms with definitions. These teams could be led by students identified in pre-test results as having background knowledge.  Teacher introduces the students to [Lessons (hacksplaining.com)](https://www.hacksplaining.com/lessons) and Imperva Learning Center <https://www.imperva.com/learn/>. Teacher allocates extension activities based on the pre-test results.  Students complete the [How cyber secure are you? Quiz](https://www.cyber.gov.au/learn-basics#quizcontainer) and share findings with classmates.  Extension activities include students reporting on[Practical ways to protect yourself online](https://www.cyber.gov.au/learn-basics#:~:text=Practical%20ways%20to%20protect%20yourself%20online) and taking control of your cyber security to reduce the impact of an attack.  Teachers employ the ‘5 whys’ methodology to have students think deeply around the concepts and activities as well as produce more thorough answers in responding to examination questions. |  |

## Weeks 3–4

Table 2 – developing secure code lesson sequence and details

| **Outcomes and content** | **Teaching and learning activities** | **Evidence of learning** | **Differentiation and adjustments** | **Registration and evaluation notes** |
| --- | --- | --- | --- | --- |
| **Developing secure code**  **Outcome**  **SE-12-07**  **SE-12-08**  Students:   * explore fundamental software design security concepts when developing programming code * apply security features incorporated into software including data protection, security, privacy and regulatory compliance * use and explain the contribution of cryptography and sandboxing to the ‘security by design’ approach in the development of software solutions. | Students:   * describe the benefits of developing secure software to protect data and minimise cyber attacks and vulnerabilities. They study and respond to lessons from [Software Security Case Studies](https://www.blueoptima.com/4-lessons-from-software-security-case-studies/). Students establish and maintain a news board, journal or folder of hyperlinks to news stories from cybersecurity incidents to refer to in class as case studies. * complete the matching exercises from the TSR on requirements definition, determining specifications, design and development phases and apply these to ‘The Unsecure PWA Company’ scenario and their security report. * For their project, students interpret and apply fundamental software development steps to develop secure code. They research and implement two-factor authentication (2FA).   Students:   * complete activities on integration, testing and maintenance in the TSR and apply these to the development of their [Unsecure PWA](https://github.com/TempeHS/The_Unsecure_PWA) project * describe how the capabilities and experience of end users influence the secure design features of software. They apply this to the development of their secure [Unsecure PWA](https://github.com/TempeHS/The_Unsecure_PWA) projects * complete the Activities 14–25 in the TSR.   **Learning intentions**  **By the end of these activities students will be able to** describe the benefits of developing secure software to protect data and minimise cyber attacks and vulnerabilities. They will explain the use and value of 2FA and write a requirements definition for a client.  **Success criteria**  I can:   * **explain** the benefits of developing secure software to protect data and minimise cyber attacks and vulnerabilities including the use and value of 2FA to a client * write a requirements definition in the security report for a client. | Students:   * complete the jigsaw activity in the TSR and are quizzed upon each of these processes at the commencement of each lesson * make positive contributions to the jigsaw process.   Students:   * correctly complete the Activities 14–25 in the TSR * complete the requirements definition in the security report for the client. | Extension material includes students visiting [ZAP to perform security testing](https://www.zaproxy.org/getting-started/) and watching [OWASP Zap Tutorial](https://youtu.be/_VpFaqF0EcI?si=k28lIIX7EeH999ez).  Teachers monitor students’ progress through these online resources. Students provide explanation to the class of their findings. |  |

## Weeks 5–6

Table 3 – developing secure code lesson sequence and details

| **Outcomes and content** | **Teaching and learning activities** | **Evidence of learning** | **Differentiation and adjustments** | **Registration and evaluation notes** |
| --- | --- | --- | --- | --- |
| **Outcome**  **SE-12-06**  **SE-12-07**  **SE-12-08**  **Developing secure code**  Students:   * use and explain the ‘privacy by design’ approach in the development of software solutions * test and evaluate the security and resilience of software by determining vulnerabilities, hardening systems, handling breaches, maintaining business continuity and conducting disaster recovery * apply and evaluate strategies used by software developers to manage the security of programming code * design, develop and implement code using defensive data input handling practices, including input validation, sanitisation and error handling * design, develop and implement a safe application programming interface (API) to minimise software vulnerabilities. | Students:   * explore fundamental software design security concepts when developing programming code including confidentiality, integrity, availability, authentication, authorisation, accountability by ‘The Unsecure PWA Company’ scenario * research and define the security features incorporated into their software project including data protection, security, privacy, and regulatory compliance * use and explain the contribution of cryptography and sandboxing to the ‘security by design’ approach in the development of software solutions * visit the Australian Signals Directorate website and research ‘What is security by design?’ * visit the Office of the Australian Information Commissioner and the Information and Privacy Commission websites to summarise a definition of ‘privacy by design’ including key concepts of a proactive not reactive approach, embedding privacy into design, and respect for user privacy * design, develop and perform a role-play based on a security breach case study and simulate approaches to business continuity including the incident response, and disaster recovery * test and evaluate the security and resilience of software by determining vulnerabilities, hardening systems, handling breaches, maintaining business continuity and conducting disaster recovery * research case studies and use vulnerability scanning tools to identify security weaknesses in a sample application or system including their own secure PWA * visit the [Australian Signals Directorate](https://www.cyber.gov.au/resources-business-and-government/governance-and-user-education/secure-by-design) website to research [guidelines for systems hardening](https://www.cyber.gov.au/resources-business-and-government/maintaining-devices-and-systems/system-hardening-and-administration/system-hardening/iot-secure-design-guidance-manufacturers) and to identify best practices for securing software systems. * visit the Office of the Australian Information Commissioner and summarise a definition of [Privacy by design](https://www.oaic.gov.au/privacy/privacy-guidance-for-organisations-and-government-agencies/privacy-impact-assessments/privacy-by-design)   **Learning intentions**  **Students:**   * explain the ‘privacy by design’ approach in the development of software solutions * apply strategies used by software developers to manage the security of programming code * implement code using defensive data input handling practices, including input validation, sanitisation and error handling * implement a safe application programming interface (API) to minimise software vulnerabilities.   **Success criteria**  I can:   * explain the ‘privacy by design’ approach * manage the security of programming code * using defensive data input handling practices * implement a safe application programming interface (API)   **Teaching and learning activities**   * Follow instructions of Appendix 1 of the TSR * Complete Activities 26–47 of the TSR. | Students   * complete activities in the TSR and applied understanding in their presentation for ‘The Unsecure PWA Company’ (client) on the fundamental software design security concepts * audit application of security features incorporated into their software project * complete the activities related to cryptography and sandboxing within the TSR * complete TSR activities to define ‘security by design’ including providing an analogy and explaining why it is important * participation and engagement in a role-play based on a security breach case study * complete activities in the TSR * update security report documentation on the use of vulnerability scanning tools * create a [Login page using Python, Flask and sqlite3 DB](https://gist.github.com/jironghuang/24e0577e58844882604c0013407bf606). | Students engage in paired programming to peer-mentor and discuss problems and solutions.  Students are given glossaries, first-language dictionaries and access to previous worksheets which upskilled them in software engineering applications.  Provide visual and/or multimedia examples and check understanding of concepts.  Prompt student discussion with metaphors and analogies.  Include unplugged activity and use of semantic waves to introduce and reinforce concepts.  Extension activities including students learning more about threats via [Threats | Cyber.gov.au](https://www.cyber.gov.au/threats). |  |

## Weeks 7–9

Table 4 – developing secure code lesson sequence and details

| **Outcomes and content** | **Teaching and learning activities** | **Evidence of learning** | **Differentiation and adjustments** | **Registration and evaluation notes** |
| --- | --- | --- | --- | --- |
| **Outcome**  **SE-12-06**  **SE-12-07**  **SE-12-08**  **Developing secure code**  Students:   * design, develop and implement code considering efficient execution for the user * design, develop and implement secure code to minimise vulnerabilities in user action controls * design, develop and implement secure code to protect user file and hardware vulnerabilities from file attacks and side channel attacks. | Students:   * apply strategies used by software developers to manage the security of programming code including code review, static application security testing (SAST), dynamic application security testing (DAST), vulnerability assessment, penetration testing their [Unsecure PWA](https://github.com/TempeHS/The_Unsecure_PWA) project * implement code for defensive data input handling practices, including input validation, sanitisation, and error handling in their [Unsecure PWA](https://github.com/TempeHS/The_Unsecure_PWA) project * implement a safe application programming interface (API) to minimise software vulnerabilities * implement code that considers efficient execution for the user including memory management, session management, exception management * implement secure code that minimises vulnerabilities in user action controls including broken authentication and session management, cross-site scripting (XSS) and cross-site request forgery (CSRF), invalid forwarding and redirecting, and race conditions. They apply these to their [Unsecure PWA](https://github.com/TempeHS/The_Unsecure_PWA) project * implement secure code to protect user file and hardware vulnerabilities from file attacks and side channel attacks. They apply and describe the benefits of collaboration to develop safe and secure software * complete Activities 47–55 in the TSR and update their security report with finding from the activities.   **Learning intentions**  **Students:**   * manage the security of programming code * implement code for defensive data input handling practices * implement a safe application programming interface (API) * implement secure code that * considers efficient execution for the user and minimises vulnerabilities in user action controls * protects user file and hardware vulnerabilities from file attacks and side channel attacks.   **Success criteria**  I can recognise and use:   * static application security testing (SAST), dynamic application security testing (DAST), vulnerability assessment, penetration testing * input validation, sanitisation, and error handling * memory management, session management, exception management.   I can recognise and mitigate against:   * broken authentication and session management, cross-site scripting (XSS) and cross-site request forgery (CSRF), invalid forwarding and redirecting, race conditions * file attacks and side channel attacks.   **Teaching and learning activities**   * **Follow instructions of Appendix 1 of the TSR.** * Attempt Activities 47–55 of the TSR. | Students:   * complete Activities 47–55 of the TSR * update their security report with findings from TSR and coding activities. | Students are given templates to aid completion of the project documentation.  Writing scaffolds can also be used to write the scripted part of their presentation to their clients/peers.  Provide visual and/or multimedia examples and check understanding of concepts.  Students discuss real-world scenarios and examples. |  |

## Week 10

Table 4 – testing and evaluating lesson sequence and details

| **Outcomes and content** | **Teaching and learning activities** | **Evidence of learning** | **Differentiation and adjustments** | **Registration and evaluation notes** |
| --- | --- | --- | --- | --- |
| **Outcome**  **SE-12-04**  **SE-12-05**  **SE-12-06**  **Impact of safe and secure software development**  Students:   * apply and describe the benefits of collaboration to develop safe and secure software * investigate and explain the benefits to an enterprise of the implementation of safe and secure development practices * evaluate the social, ethical and legal issues and ramifications that affect people and enterprises resulting from the development and implementation of safe and secure software. | Students:   * investigate and explain the benefits to an enterprise of the implementation of safe and secure development practices including improved products or services, influence on future software development, improved work practices, productivity and business interactivity * evaluate the social, ethical, and legal issues and ramifications that affect people and enterprises resulting from the development and implementation of safe and secure software, including employment, data security, privacy, copyright, intellectual property and digital disruption * complete the Activities 56–57 in the TSR * present their [Unsecure PWA](https://github.com/TempeHS/The_Unsecure_PWA) solution using presentation software to the class (playing the role of ‘The Unsecure PWA Company’ (client) * submit their project documentation and code.   **Learning intentions**  **Students:**   * **confidently deliver an explanation of their PWA** * **demonstrate expertise in responding to Q&A from peer assessment of their PWA.**   **Success criteria**  I can:   * deliver my PWA project to peers, teachers and/or clients.   **Teaching and learning activities**  **Teachers model a professional presentation and the assist class in the development of relevant questions for the Q&A section of the PWA presentation.** | Students:   * present their more secure progressive web app solution using presentation software to the class (client) * submit their project security report * submit all assessable items for this task including all components * provide informed responses to Q&A session of presentation. | Students may negotiate with teachers a presentation method (for example, video) that still demonstrates a thorough understanding of their solution and ensures academic integrity including adhering to [All My Own Work.](https://www.nsw.gov.au/education-and-training/nesa/hsc/all-my-own-work) |  |

# Overall program evaluation

Collating ongoing evaluations and reflecting on the strengths and areas for development within the program creates opportunities to enhance student outcomes. The following prompts can be used to support your evaluation of the program:

* Did the program assist all students to improve in their learning?
* How could the sequencing of the program be improved?
* What did the student evaluations of the program indicate? How can these be actioned to improve the program?
* The strategies and resources that were most effective for student learning were …
* Teaching strategies and resources that would benefit from review and refinement are …

## Capturing student voice when evaluating a program

Student voice is useful in the evaluation process for programs. The statements below could be useful as a starting point when asking students to provide feedback on their learning experiences. These statements are derived from some of the themes from [What works best: 2020 update](https://education.nsw.gov.au/about-us/education-data-and-research/cese/publications/research-reports/what-works-best-2020-update) (CESE 2020a) and could be useful in teacher reflection on how these themes could be incorporated into a teaching program. The statements could also prompt student reflection on their metacognitive processes while learning.

**Please rate how much you agree with these statements:**

* My teacher had confidence that I could achieve and improve in my learning. (CESE 2020a Chapter 1: High expectations)
* I had a clear idea of what I was learning and why. (CESE 2020a Chapter 2: Explicit teaching)
* I used the feedback provided to improve my performance. (CESE 2020a Chapter 3: Effective feedback)
* I understood the feedback on the assessment task. (CESE 2020a Chapter 3: Effective feedback)
* I was able to predict the marks I achieved in the assessment tasks. (CESE 2020a Chapter 5: Assessment)
* The activities in the unit prepared me for the assessment task. (CESE 2020a Chapter 5: Assessment)
* I found the activities in the lessons interesting to me. (CESE 2020a Chapter 7: Wellbeing)
* I made valuable contributions to the class during this unit. (CESE 2020a Chapter 7: Wellbeing)
* I ask questions in class when I don’t understand yet. (CESE 2020a Chapter 7: Wellbeing)

**Optional open-ended prompts:**

* The lessons and/or activities that I most enjoyed were when we … because …
* When the learning was difficult, the strategy I used was …
* If I was giving advice to a student who was starting this unit, I would tell them to …
* If I was giving advice to a teacher who was teaching this unit, I would tell them to …

# Additional information

For additional support or advice, contact the TAS curriculum team by emailing [TAS@det.nsw.edu.au](mailto:TAS@det.nsw.edu.au).

## Further implementation support

Curriculum design and implementation is a dynamic and contextually specific process. The department is committed to supporting teachers to meet the needs of all students. The advice below on assessment and planning for the needs of every student may be useful when considering the material presented in this sample program of learning.

## Assessment for learning

Possible formative assessment strategies that could be included:

* Learning intentions and success criteria assist educators to articulate the purpose of a learning task to make judgements about the quality of student learning. These help students focus on the task or activity taking place and what they are learning and provide a framework for reflection and feedback. [Online tools](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Card/622) can assist implementation of this formative assessment strategy.
* Eliciting evidence strategies allow teachers to determine the next steps in learning and assist teachers in evaluating the impact of teaching and learning activities. Strategies that may be added to a learning sequence to elicit evidence include all student response systems, [exit tickets](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Card/543), mini whiteboards (actual or [digital](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Card/575)), [hinge questions](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Card/560), [Kahoot](https://app.education.nsw.gov.au/digital-learning-selector/LearningTool/Card/621), [Socrative](https://app.education.nsw.gov.au/digital-learning-selector/LearningTool/Card/587), or quick quizzes to ensure that individual student progress can be monitored and the lesson sequence adjusted based on formative data collected.
* Feedback is designed to close the gap between current and desired performance by informing teacher and student behaviour (AITSL 2017). AITSL provides a [factsheet to support evidence-based feedback](https://www.aitsl.edu.au/teach/improve-practice/feedback#:~:text=FEEDBACK-,Factsheet,-A%20quick%20guide).
* [Peer feedback](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Card/549) is a structured process where students evaluate the work of their peers by providing valuable feedback in relation to learning intentions and success criteria. It can be supported by [online tools](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Browser?cache_id=1d29b).
* Self-regulated learning opportunities assist students in taking ownership of their own learning. A variety of strategies can be employed and some examples include reflection tasks, [Think-Pair-Share](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Card/645), [KWLH charts](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Card/562), [learning portfolios](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Card/583) and [learning logs](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Card/583).

The primary role of assessment is to establish where individuals are in their learning so that teaching can be differentiated, and further learning progress can be monitored over time.

Feedback that focuses on improving tasks, processes and student self-regulation is the most effective. Students engaging with feedback can take many forms including formal, informal, formative, summative, interactive, demonstrable, visual, written, verbal and non-verbal (CESE 2020a).

## Differentiation

Differentiated learning can be enabled by differentiating the teaching approach to content, process, product and the learning environment. For more information on differentiation go to [Differentiating learning](https://education.nsw.gov.au/teaching-and-learning/professional-learning/teacher-quality-and-accreditation/strong-start-great-teachers/refining-practice/differentiating-learning) and [Differentiation](https://education.nsw.gov.au/campaigns/inclusive-practice-hub/primary-school/teaching-strategies/differentiation).

When using these resources in the classroom, it is important for teachers to consider the needs of all students in their class, including:

* **Aboriginal and Torres Strait Islander students**. Targeted [strategies](https://education.nsw.gov.au/teaching-and-learning/aec/aboriginal-education-in-nsw-public-schools) can be used to achieve outcomes for Aboriginal students in K–12 and increase knowledge and understanding of Aboriginal histories and cultures. Teachers should utilise students’ Personalised Learning Pathways to support individual student needs and goals.
* **EAL/D learners**. EAL/D learners will require explicit English language support and scaffolding, informed by the [EAL/D enhanced teaching and learning cycle](https://education.nsw.gov.au/teaching-and-learning/curriculum/literacy-and-numeracy/resources-for-schools/eald/enhanced-teaching-and-learning-cycle) and the student’s phase on the [EAL/D Learning Progression](https://education.nsw.gov.au/teaching-and-learning/curriculum/multicultural-education/english-as-an-additional-language-or-dialect/planning-eald-support/english-language-proficiency). In addition, teachers can access information about [supporting EAL/D learners](https://education.nsw.gov.au/teaching-and-learning/curriculum/multicultural-education/english-as-an-additional-language-or-dialect/planning-eald-support/english-language-proficiency) and [literacy and numeracy support specific to EAL/D learners](https://education.nsw.gov.au/teaching-and-learning/curriculum/literacy-and-numeracy/resources-for-schools/eald).
* **Students with additional learning needs**. Learning adjustments enable students with disability and additional learning and support needs to access syllabus outcomes and content on the same basis as their peers. Teachers can use a range of [adjustments](https://education.nsw.gov.au/teaching-and-learning/disability-learning-and-support/personalised-support-for-learning/adjustments-to-teaching-and-learning) to ensure a personalised approach to student learning. Subject specific curriculum considerations can be found on the [Inclusive Practice hub](https://education.nsw.gov.au/campaigns/inclusive-practice-hub).
* **High potential and gifted learners**. [Assessing and identifying high potential and gifted learners](https://education.nsw.gov.au/teaching-and-learning/high-potential-and-gifted-education/supporting-educators/assess-and-identify#Assessment1) will help teachers decide which students may benefit from extension and additional challenge. [Effective strategies and contributors to achievement](https://education.nsw.gov.au/teaching-and-learning/high-potential-and-gifted-education/supporting-educators/evaluate) for high potential and gifted learners help teachers to identify and target areas for growth and improvement. In addition, the [Differentiation Adjustment Tool](https://education.nsw.gov.au/teaching-and-learning/high-potential-and-gifted-education/supporting-educators/implement/differentiation-adjustment-strategies) can be used to support the specific learning needs of high potential and gifted students. The [High Potential and Gifted Education Professional Learning and Resource Hub](https://schoolsnsw.sharepoint.com/sites/HPGEHub/SitePages/Home.aspx) supports school leaders and teachers to effectively implement the High Potential and Gifted Education Policy in their unique contexts.

All students need to be challenged and engaged to develop their potential fully. A culture of high expectations needs to be supported by strategies that both challenge and support student learning needs, such as through appropriate curriculum differentiation (CESE 2020a:6).

## Support and alignment

**Resource evaluation and support**: all curriculum resources are prepared through a rigorous process. Resources are periodically reviewed as part of our ongoing evaluation plan to ensure currency, relevance and effectiveness. For additional support or advice contact the TAS curriculum team by emailing [TAS@det.nsw.edu.au](mailto:TAS@det.nsw.edu.au).

**Differentiation:** further advice to support Aboriginal and Torres Strait Islander students, English as an additional language or dialect (EAL/D) students, students with a disability and/or additional needs and High Potential and Gifted (HPG) students can be found on the [Planning, programming and assessing 7–12](https://education.nsw.gov.au/teaching-and-learning/curriculum/planning-programming-and-assessing-k-12/planning-programming-and-assessing-7-12) webpage. This includes the [Inclusion and differentiation 7–10 advice](https://education.nsw.gov.au/teaching-and-learning/curriculum/planning-programming-and-assessing-k-12/planning-programming-and-assessing-7-12/inclusion-and-differentiation-advice-7-10) webpage.

**Assessment**: further advice to support formative assessment is available on the [Planning, programming and assessing 7–12](https://education.nsw.gov.au/teaching-and-learning/curriculum/planning-programming-and-assessing-k-12/planning-programming-and-assessing-7-12) webpage. This includes the [Classroom assessment advice 7–10](https://education.nsw.gov.au/teaching-and-learning/curriculum/planning-programming-and-assessing-k-12/planning-programming-and-assessing-7-12/classroom-assessment-advice-7-10-). For summative assessment tasks, the [Assessment task advice 7–10](https://education.nsw.gov.au/teaching-and-learning/curriculum/planning-programming-and-assessing-k-12/planning-programming-and-assessing-7-12/assessment-task-advice-7-10) webpage is available.

**Consulted with**: Curriculum and Reform and subject matter experts

**Alignment to system priorities and/or needs**: [School Excellence Policy](https://education.nsw.gov.au/policy-library/policies/pd-2016-0468)

**Alignment to the School Excellence Framework**: this resource supports the [School Excellence Framework](https://education.nsw.gov.au/policy-library/policies/pd-2016-0468) elements of curriculum (curriculum provision) and effective classroom practice (lesson planning, explicit teaching).

**Alignment to Australian Professional Standards for Teachers**: this resource supports teachers to address [Australian Professional Standards for Teachers](https://educationstandards.nsw.edu.au/wps/portal/nesa/teacher-accreditation/meeting-requirements/the-standards/proficient-teacher) 1.1.2, 1.2.2, 1.3.2, 2.1.2, 2.2.2, 2.6.2, 3.2.2, 3.3.2, 3.4.2, 4.5.2, 6.2.2.

**NSW Syllabus**: Software Engineering 11–12

**Syllabus outcomes**: SE-12-01, SE-12-02, SE-12-03, SE-12-04, SE-12-05, SE-12-06, SE-12-07, SE-12-08, SE-12-09

**Author**: TAS, Curriculum Secondary Learners, Curriculum Reform

**Publisher**: State of NSW, Department of Education

**Resource**: program of learning

**Related resources**: further resources to support Software Engineering 11–12 can be found on the [TAS curriculum page](https://education.nsw.gov.au/teaching-and-learning/curriculum/tas).

**Professional learning**: relevant professional learning is available through [HSC Professional Learning](https://education.nsw.gov.au/teaching-and-learning/professional-learning/hsc-pl) or in the TAS statewide staffroom.

**Creation date:** 2024

**Rights**: © State of New South Wales, Department of Education

# Evidence base

[Software Engineering 11–12 Syllabus](https://curriculum.nsw.edu.au/learning-areas/tas/software-engineering-11-12-2022/overview) © NSW Education Standards Authority (NESA) for and on behalf of the Crown in right of the State of New South Wales, 2022.

[Software Engineering Course Specifications](https://curriculum.nsw.edu.au/learning-areas/tas/software-engineering-11-12-2022/overview#software-engineering-course-specifications-software_engineering_11_12_2022) © NSW Education Standards Authority (NESA) for and on behalf of the Crown in right of the State of New South Wales, 2022.

AITSL (Australian Institute for Teaching and School Leadership Limited) (n.d.) [*Learning intentions and success criteria* [PDF 251 KB]](https://www.aitsl.edu.au/docs/default-source/feedback/aitsl-learning-intentions-and-success-criteria-strategy.pdf?sfvrsn=382dec3c_2#:~:text=Learning%20Intentions%20are%20descriptions%20of,providing%20feedback%20and%20assessing%20achievement.), AITSL, accessed 3 April 2024.

AITSL (2017) [*Feedback Factsheet*](https://www.aitsl.edu.au/teach/improve-practice/feedback#:~:text=FEEDBACK-,Factsheet,-A%20quick%20guide), AITSL, accessed 3 April 2024.

Brookhart S (2011) How to Assess Higher-Order Thinking Skills in Your Classroom, Hawker Brownlow Education, Victoria.

CESE (Centre for Education Statistics and Evaluation) (2020a) [*What works best: 2020 update*](https://education.nsw.gov.au/about-us/education-data-and-research/cese/publications/research-reports/what-works-best-2020-update), NSW Department of Education, accessed 3 April 2024.

CESE (2020b) [*What works best in practice*](https://education.nsw.gov.au/about-us/education-data-and-research/cese/publications/practical-guides-for-educators-/what-works-best-in-practice), NSW Department of Education, accessed 3 April 2024.

NESA (NSW Education Standards Authority) (2022a) ‘[Programming](https://educationstandards.nsw.edu.au/wps/portal/nesa/k-10/understanding-the-curriculum/programming)’, *Understanding the curriculum*, NESA website, accessed 3 April 2024.

——(2022b) ‘[Advice on units](https://educationstandards.nsw.edu.au/wps/portal/nesa/k-10/understanding-the-curriculum/programming/advice-on-units)’, *Programming,* NESA website, accessed 3 April 2024.

——(2022c) ‘[Proficient Teacher Standard Descriptors](https://educationstandards.nsw.edu.au/wps/portal/nesa/teacher-accreditation/meeting-requirements/the-standards/proficient-teacher)’, The Standards, NESA website, accessed 3 April 2024.

Rosenshine B (2012) ‘[Principles of Instruction: Research-Based Strategies That All Teachers Should Know](https://eric.ed.gov/?id=EJ971753)’, American Educator, 36(1):12–19, ISSN-0148-432X, accessed 3 April 2024.

Wiliam D (2013) ‘[Assessment: The Bridge between Teaching and Learning](https://www.researchgate.net/publication/258423377_Assessment_The_bridge_between_teaching_and_learning)’, Voices from the Middle, 21(2):15–20, accessed 3 April 2024.

Wiliam D (2017) Embedded Formative Assessment, 2nd edn, Solution Tree Press, Bloomington, IN.

Wisniewski B, Zierer K and Hattie J (2020) ‘[The Power of Feedback Revisited: A Meta-Analysis of Educational Feedback Research](https://doi.org/10.3389/fpsyg.2019.03087)’, *Frontiers* *In Psychology*, 10:3087, doi:10.3389/fpsyg.2019.03087, accessed 3 April 2024.

# References

This resource contains NSW Curriculum and syllabus content. The NSW Curriculum is developed by the NSW Education Standards Authority. This content is prepared by NESA for and on behalf of the Crown in right of the State of New South Wales. The material is protected by Crown copyright.

Please refer to the NESA Copyright Disclaimer for more information <https://educationstandards.nsw.edu.au/wps/portal/nesa/mini-footer/copyright>.

NESA holds the only official and up-to-date versions of the NSW Curriculum and syllabus documents. Please visit the NSW Education Standards Authority (NESA) website <https://educationstandards.nsw.edu.au> and the NSW Curriculum website <https://curriculum.nsw.edu.au>.

[Software Engineering 11–12 Syllabus](https://curriculum.nsw.edu.au/learning-areas/tas/software-engineering-11-12-2022/overview) © NSW Education Standards Authority (NESA) for and on behalf of the Crown in right of the State of New South Wales, 2022.

[Software Engineering Course Specifications](https://curriculum.nsw.edu.au/learning-areas/tas/software-engineering-11-12-2022/overview#software-engineering-course-specifications-software_engineering_11_12_2022) © NSW Education Standards Authority (NESA) for and on behalf of the Crown in right of the State of New South Wales, 2022.

**© State of New South Wales (Department of Education), 2024**

The copyright material published in this resource is subject to the Copyright Act 1968 (Cth) and is owned by the NSW Department of Education or, where indicated, by a party other than the NSW Department of Education (third-party material).

Copyright material available in this resource and owned by the NSW Department of Education is licensed under a [Creative Commons Attribution 4.0 International (CC BY 4.0) license](https://creativecommons.org/licenses/by/4.0/).

[](https://creativecommons.org/licenses/by/4.0/)

This license allows you to share and adapt the material for any purpose, even commercially.

Attribution should be given to © State of New South Wales (Department of Education), 2024.

Material in this resource not available under a Creative Commons license:

* the NSW Department of Education logo, other logos and trademark-protected material
* material owned by a third party that has been reproduced with permission. You will need to obtain permission from the third party to reuse its material.

**Links to third-party material and websites**

Please note that the provided (reading/viewing material/list/links/texts) are a suggestion only and implies no endorsement, by the New South Wales Department of Education, of any author, publisher, or book title. School principals and teachers are best placed to assess the suitability of resources that would complement the curriculum and reflect the needs and interests of their students.

If you use the links provided in this document to access a third-party's website, you acknowledge that the terms of use, including licence terms set out on the third-party's website apply to the use which may be made of the materials on that third-party website or where permitted by the Copyright Act 1968 (Cth). The department accepts no responsibility for content on third-party websites.