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

Software engineering projectContents

[About this resource 3](#_Toc165552999)

[Purpose of resource 3](#_Toc165553000)

[Target audience 3](#_Toc165553001)

[When and how to use 3](#_Toc165553002)

[Rationale 4](#_Toc165553003)

[Overview 5](#_Toc165553004)

[Outcomes 7](#_Toc165553005)

[Lesson sequence and details 9](#_Toc165553006)

[Weeks 1 and 2 9](#_Toc165553007)

[Weeks 3 to 5 16](#_Toc165553008)

[Weeks 6 to 9 22](#_Toc165553009)

[Week 10 26](#_Toc165553010)

[Overall program evaluation 30](#_Toc165553011)

[Capturing student voice when evaluating a program 30](#_Toc165553012)

[Additional information 32](#_Toc165553013)

[Further implementation support 32](#_Toc165553014)

[Assessment for learning 32](#_Toc165553015)

[Differentiation 33](#_Toc165553016)

[Support and alignment 34](#_Toc165553017)

[Evidence base 37](#_Toc165553018)

[References 39](#_Toc165553019)

# About this resource

## Purpose of resource

The resource is a sample program of learning for teaching the Software project 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 the Software project. 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 2022). 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 focus area of the Software engineering project. The lessons and sequences in this program of learning are designed to guide students through the development of their software engineering project. This project is built to a client’s specifications, using one of the software development approaches and implementation methods outlined in the syllabus.

The chosen project process is underpinned by the project documentation, which is based on Software Engineering teaching outcomes and content areas. Students are encouraged to find clients and/or generate their own scenario for which this software solution is to be built.

Identifying and defining: students develop a project proposal This includes reflection upon personal interest and their success with projects from Year 11 focus areas (Programming fundamentals, The object-oriented paradigm and Programming mechatronics). Students preview syllabus content for upcoming focus areas (Secure software architecture, Programming for the web and Software automation). They identify, define and analyse the requirements of a problem (or opportunity). Students explore tools used to develop ideas, generate solutions, and investigate types of software implementation methods. They contact and interview a client as well as analyse software engineering case studies, scenarios and existing solutions.

Research and planning: students research and use the Waterfall, Agile and WAgile software development approaches. They apply project management skills to plan and conduct the development and implementation of their software engineering solution. Students explore social and ethical issues associated with project work, including working individually, collaboratively and responding to stakeholders. They explore communication issues associated with project work and investigate how software engineering solutions are quality assured. Students demonstrate the use of modelling tools and explain the contribution of back-end engineering to the success and ease of software development.

Producing and implementing: students construct and implement a solution to a software problem or opportunity using appropriate development approach(es). They construct and document algorithms, demonstrate the use of programmed data backup, implement version control and explore strategies to respond to difficulties when developing their solution. Students propose an additional innovative solution using a prototype and user interface (UI) design.

Testing and evaluating: students apply methodologies to test and evaluate code. They use a language-dependent code optimisation technique, analyse and respond to feedback and evaluate the effectiveness of a software engineering solution.

The software engineering solution is evaluated and tested to ensure its features meet the success criteria outlined in the problem definition and the clients’ identified needs.

Students present their software engineering solution using presentation software to the class (and client) and submit the project documentation and code.

**Duration**: the content for this focus area should be delivered over 30 hours across 3 terms. It should be integrated with other focus areas to support students to apply deeper knowledge, understanding and skills. The Software engineering project should be introduced during the first term of the Year 12 course and run in parallel with theory from the focus areas in this course.

**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 software project **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 software engineering project development.**

## Weeks 1 and 2

Table 1 – identifying and defining 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-12-09**  **Identifying and defining**  Students:   * define and analyse problem requirements * assess the scheduling and financial feasibility * define boundaries * explore tools to develop ideas and generate solutions * investigate types of software implementation methods. | **Learning intentions**  **Understand the software engineering project purpose.**  Identify a client.  Develop a clear and concise project proposal.  Establish clear explanations for the use of selected project management tools.  **Success criteria**   * I can generate a detailed project proposal broken down into actual functional requirements, quality assurance, needs and boundaries. * I can justify the use of project management tools and ways of working.   **Teaching and learning activities**  Students are introduced to the software project assessment task to understand the outcomes and expectations.  As a class students read through the [course specifications and NESA teaching advice](https://curriculum.nsw.edu.au/learning-areas/tas/software-engineering-11-12-2022/teaching-and-learning) for the Year 12 focus areas: Programming for the web, Secure software architecture and Software automation.  Survey student interest.  Revise Year 11 focus areas and successful projects. Preview Year 12 focus areas.  Brainstorm real-world problems and clients.  Mind map above concepts to identify an appropriate software engineering project and the scale and scope of the project.  Students describe how the tools and processes will be used in their software engineering project.  As a class discuss the following questions:   * What are the key components of a software engineering project? * How can software engineering address real-world problems for clients? * What are your interests or concerns that could be addressed through this project? * What skills and understanding do you have? * What skills and understanding will you need?   Students decide on either a client or a self-generated problem to generate a software solution.  Teacher-led discussion on problem definition, project scope and requirements.  As a class discuss the ‘Identifying and defining’ stage of development.  Teacher-led discussion unpacking the following questions:   * What are the functional and performance requirements of your project? * (What must it do and how well should it do it?) * How will you determine the scale and scope of the project? * How will you measure the success of your project?   Teacher introduces the following types of software implementation methods:   * pilot implementation * phased implementation * parallel implementation * direct implementation   Students contact and interview a client.  Teacher provides software engineering case studies and scenarios.  Students research existing solutions that may be applicable to their project proposal. | Students:   * answer questions on the requirements of the software engineering project assessment task * reflect upon personal interest and their success with projects from Year 11 focus areas * preview syllabus content for upcoming focus areas * create a mind map to link concepts from personal interest, background Year 11 knowledge, upcoming Year 12 focus areas and potential clients.   Students:   * brainstorm ideas for inclusion in their software project * document via annotated screenshots their exploration of tools to develop ideas and generate solutions * document a clear and concise problem definition and needs analysis in their project documentation * document via annotated screenshots the tools and processes used to manage and document the development of a software engineering project.   Students start project documentation including a project proposal defining the problem, objectives, and expected outcomes.  Students contribute to class discussions and respond to revision quiz at the beginning of each lesson.  Students engage in class discussions and respond to questions.  Student documentation demonstrates existing solutions that may be applicable to their project. | Suggested adjusted activities. This section is also for use in school when making adjustments to support all students to achieve in their learning.  Students are given course specifications, glossaries, writing scaffolds, first-language dictionaries and exemplar answers or responses so that they know what and how to write for each section of the project documentation.  Motivate students by negotiating a task and real-world client that has personal interest, and demands knowledge and skills commensurate with students’ experience and capacity to develop |  |

## Weeks 3 to 5

Table 2 – researching and planning 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-12-05**  **SE-12-03**  **SE-12-04**  **Research and planning**  **Project management**  Students:   * apply project management to plan and conduct the development and implementation of a project and software engineering solution   Including   * using collaboration tools * explore the social and ethical aspects of software engineering projects * explore communication issues associated with project work.   **Quality assurance**  Students:   * ensure requirements are met using a continual checking process.   **Systems modelling**  Students:   * explain the contribution of back-end engineering to the success and ease of software development. | **Learning intentions**  Identify software development approaches and apply them appropriately to different scenarios.  Explore social and ethical issues associated with project work.  Apply knowledge of how-to quality assure a project.  Use modelling tools to assist a client understand the proposed solution.  Understand the contribution of back-end engineering to software engineering.  **Success criteria**   * I can choose an appropriate software development approach for a given scenario. * I can discuss social and ethical issues around my project. * I can establish a quality assurance criterion and use modelling tools to assist clients’ understanding. * I can explain back-end engineering.   **Teaching and learning activities**  **Students complete Jigsaw activity on software development approaches from the Teacher Support Resource (TSR).**  **Research, discuss and take notes on the following questions:**   * **What considerations do you need to give to privacy, security, accessibility, transparency, fairness, intellectual property (IP), collaboration and feedback when developing your software project?** * **How can a Gantt chart help in planning your project?** * **Can you identify key milestones and tasks for your project plan?** * **How could online collaboration tools enhance your project development?**   Explicit teaching will unpack the modelling tools in the NESA course specifications and how to create each one including:   * **data flow diagrams, data dictionaries, structure diagrams, class diagrams and decision trees. These aid in the planning process.**   Teacher-led discussion of open-ended questions that encourage deeper thinking and problem-solving, for example:  Computational thinking   * How do you think breaking down a problem into smaller, more manageable parts (decomposition) helps in the design and development of a software solution? * Can you describe a scenario where pattern recognition would be crucial in optimising a software solution's performance? * How might algorithms be utilised in managing data flow within a software solution? * In what ways do you think abstraction simplifies the complexity of a software solution for clients and developers? * How important is iteration in developing a solution? * How can I make this software secure? * How could machine learning be used to develop a solution? | Students:   * justify their selection of software development approaches for different scenarios * complete a description for each key issue of privacy, security, accessibility, transparency, fairness, intellectual property, collaboration and feedback in the TSR, and discuss its relevance for their project * explain quality criteria based upon identified need and functional requirements. These quality criteria should contain qualities, characteristics or components that need to be included or visible by the end of the current project * maintain a GANTT chart with milestones that indicate continual checking process * document via screenshots the use of collaboration tools * maintain a process diary, including ongoing evaluation for their software project * document a feasibility study for their software project.   Students:   * draw data flow diagrams for their software project * develop **data dictionaries** for their software project * develop a decision tree for their software project * engage in class discussions and answer questions posed on the researching and planning stage of project development.   Students complete all relevant activities within the TSR including:   * systems modelling * data flow diagrams * structure charts * class diagrams * storyboards * decision trees * algorithm design * back-end engineering. | Students are given NESA course specifications, writing scaffolds, glossaries, first-language dictionaries, and templates against which they can develop their software solutions analysis diagrams.  Provide visual and/or multimedia examples and check understanding of concepts.  Prompt student discussion with real-world scenarios and examples.  Include multiple opportunities to respond, for example:   * verbally * individually * partners turn and talk * nonverbal * gesture * response cards. |  |

## Weeks 6 to 9

Table 3 – producing and implementing lesson sequence and details

| **Outcomes and content** | **Teaching and learning activities** | **Evidence of learning** | **Differentiation and adjustments** | **Registration and evaluation notes** |
| --- | --- | --- | --- | --- |
| **Outcome**  **SE-12-02**  **SE-12-07**  **SE-12-08**  **Producing and implementing**  Students:   * design, construct and implement a solution to a software problem using appropriate development approach(es) * present a software engineering solution using presentation software * develop, construct and document algorithms * allocate resources to support the development of a software engineering solution * demonstrate the use of programmed data backup * implement version control when developing a software engineering solution * explore strategies to respond to difficulties when developing a software engineering solution * propose an additional innovative solution using a prototype and user interface (UI) design. | **Learning intentions**  Produce an individual software project.  Develop complete implementation plans to ensure the software project functions as intended.  **Success criteria**  I can:   * develop, construct and document algorithms * allocate resources to support the development of a software engineering solution * demonstrate the use of programmed data backup * implement version control when developing a software engineering solution * explore strategies to respond to difficulties when developing a software engineering solution * propose an additional innovative solution using a prototype and user interface (UI) design * produce a software project that addresses the problem definition and needs identified in the project documentation.   **Teaching and learning activities**  Students are tested regularly by the classroom teacher and/or the client on their software project functions and use.  Class discussion of the ‘Producing and implementing’ stage of development. Encourage deeper thinking and problem-solving by asking open-ended questions, including:   * Why is selecting the right development approach important for your project? * How will you justify your choice of tools and resources? * Are the clients' requirements being met? * How does your testing strategy ensure the software solution’s functionality and client/user experience? * What training will be necessary for users of your software solution? * How will you address potential compatibility issues? * What software implementation strategy will you use and why?   Teacher to lead class through real or simulated client meetings and typical items for discussion at these as the software project progresses. | Students:   * apply tools to meet the functional requirements of a software engineering project * convert algorithms into code in an approved programming language * identify and allocate resources to their project * discuss a backup plan and version control * use strategies to respond to issues and problems * create a storyboard of an innovative solution or prototype.   Students:   * use client interviews, to provide feedback on their software project * work iteratively to adjust their software project * generate a responsive, just-in-time feedback loop so that problems can be fixed quickly and iteratively.   Students:   * complete their software engineering project * document the production of their project including screenshots for their presentation * respond to real or simulated changes to project requirements or deliverables * develop their final presentation. | Students given PRIMM strategies during programming and coding activities:   * Predict * Run * Investigate * Modify * Make.   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. |  |

## 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-12-08**  **Testing and evaluating**  Students:   * apply methodologies to test and evaluate code to ensure its quality, reliability, and functionality. * highlight the methodologies to test and evaluate code thatare most relevant to your project * use a language-dependent code optimisation technique * analyse and respond to feedback * evaluate the effectiveness of a software engineering solution. | **Learning intentions**  Understand the relationship between problem definition and functional requirements and testing methods in the development of a software engineering project.  Communicate the effectiveness of the software project solution to a general audience.  **Success criteria**  I can:   * apply methodologies to test and evaluate code * use a language-dependent code optimisation technique * analyse and respond to feedback * evaluate the effectiveness of a software engineering solution * develop a report to synthesise feedback * develop a test plan * compare actual output with expected output * test paths and boundaries * generate a list of modifications to the current software solution based on test data * present my software project to the class simulating a client handover with a walkthrough of the project.   **Teaching and learning activities**  As a class discuss the ‘Testing and evaluating’ stage of development.  Encourage deeper thinking and problem-solving by asking open-ended questions, such as:   * How will you use test data to evaluate your software solution’s performance? * How has peer feedback contributed to your project refinement? * What modifications are/have been needed based on your testing outcomes? * How will you document and address the feedback received during testing?   Students complete TSR activities  Students deliver a 4-minute presentation which simulates the handover of their software solution to their client as part of their assessment task. | Students will be able to identify and explain:   * functional testing * acceptance testing * live data * simulated data * beta testing * volume testing.   Students participate in a jigsaw activity on common language-dependent code optimisation techniques.  Students:   * analyse their software solution against quality success criteria and write a report that synthesises feedback * define the criteria for evaluating the results of tests, such as pass/fail criteria, acceptance criteria, and performance, and provide a basis for determining whether the software meets its requirements * identify any discrepancies and determine the cause of any failures.   Students complete accompanying TSR activities.  Students participate in class discussion on the testing and evaluating stage of development.  Students complete a presentation in class replicating a client handover with a walkthrough of the project.  Students respond to a Q&A forum about their project including how they achieved it and the challenges they experienced. | 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. |  |

# 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, EAL/D students, students with a disability and/or additional needs and High Potential and gifted 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-11-01, SE-11-02, SE-11-03, SE-11-04, SE-11-05, SE-11-06, SE-11-07, SE-11-08, SE-11-09

**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) (2022) ‘[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.

NESA (NSW Education Standards Authority) (2022) ‘[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.

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