Think, Compute, Grow!

A strategy for teaching computing and ICT through sustainable agriculture

PART 1- INTRODUCTION AND OVERVIEW

Licence and permissions:

You are free to:

Created by Owen Dobbing, 2026

CONTENTS

Introduction 4
Three stages 5
Outcome-driven approach 6
UN Sustainable Development Goals 7
Keeping children safe 8
Teaching effective ICT lessons 9
Computational thinking and metacognition 14
Progression 16
Guide to planning lessons 17
Example yearly-planning 19
Teacher training 21
Next steps and checklist 22
About the author 23
References and Acknowledgements 24

INTRODUCTION

Aims

This strategy is written specially to **help children in education in rural areas of the world.** It aims to provide ways that children can use laptops and the internet to develop the skills that can help them live and thrive in their communities, while also preparing them for an increasingly digital, worldwide society.

The lessons and planning provided by this strategy are built **on current educational thinking and practice**. Teachers using these lessons can be sure that they are using methodology and resources backed by the most effective research in teaching children to use technology.

The lessons in this strategy can be used to teach children how technology can be used in the context of **sustainability in agriculture**.

Schools in both agricultural and non-agricultural regions of the world can use the lessons in this strategy to teach children how technology can help make food production **greener and more effective**, supporting the aims laid out by the United Nations in the **Sustainable Development Goals.**

This is becoming more important, even in **urban settings**. New urban developments frequently include aspects of nature, both to benefit the environment and the health of people living there.

Teaching children to use technology in the context of nature and plants provides valuable opportunities for children to learn about how plants grow. It gives them opportunities to nurture and interact with plants, providing an engaging and wholesome context for their learning.

Many children find immense joy in growing their own plans and even selling the produce that they make!

The **resources** in this strategy have been selected with the aim of making computing skills accessible to all children. They are some of the best value and effective resources currently available in computing education.

THREE STAGES

Teaching is split into three stages of development These are:

Stage 1: Foundation stage (ages 7 – 9)

Description: Students become familiar with operating laptops. They enjoy being creative, solving problems and discovering familiar resources for research and learning.

Focus: Core computer problem solving, English language discovery, basic nature observation. Growing understanding of digital literacy.

Tools: Typing.com, E-Paath (grades 1 – 3), Scratch desktop (animation), Google Earth.

Stage 2: Intermediate stage (ages 10 – 12)

Description: Students grow in confidence at solving problems using technology. They make connections to their local environment and the technology they use. Children use digital tools to support their knowledge across subjects.

Focus: Structured document and presentation creation. Independent programming and problem solving using logical structures. Developing digital literacy.

Tools: Typing.com (focus on speed), E-Paath (grades 4 – 6), Scratch, Makecode, (virtual blocks), LibreOffice Writer.

Stage 3: Advanced stage (ages 13 – 16)

Description: Students apply their knowledge of hardware, software and online services to solve increasingly complex problems related to their environment.

Focus: Advanced programming logic (programming with Python), building automation hardware (Microbit), organising finances using spreadsheets, word-processing and presentational skills. Digital and AI literacy.

Tools: LibreOffice Calc (spreadsheets), Python programming (text-based programming), Microbits (programming using Microbit Python).

Development goals

Effective lessons contribute to a strategy of children achieving the most essential educational goals.

This strategy has identified **nine development goals**, split into **three strands**.

The **first strand** teaches children how they can use technology to develop the **essential academic skills** at will enable them to succeed in all aspects of their lives.

The **second strand** teaches children about **how technology works**, making them independent, creative, effective and critical users of technology.

The **third strand** focuses on **applying technology** in the context of business and agriculture.

Development goals:

Foundational academic skills	**Goal 1**	**Goal 2**	**Goal 3**
	Foundational academic excellence	**Functional and professional English**	**Geographic information and mapping**
Understanding and using technology	**Goal 4**	**Goal 5**	**Goal 6**
	Introduction to software logic	**Understanding computer systems and networks**	**Digital literacy, E-safety and AI**
Applying technology to agriculture	**Goal 7**	**Goal 8**	**Goal 9**
	Agricultural financial literacy	**Agricultural science**	**Modern marketing and business**

United Nations Sustainable Development Goals

The lessons in this strategy promote the outcomes of the United Nations (UN) Sustainable development Goals.

The 17 Sustainable Development Goals were created with a view to "ending all forms of poverty, fighting inequalities, tackling climate change and ensuring that no one is left behind". (UN Resolution A/RES/70.1, 25th September 2015).

Every lesson in this strategy contributes to one of the 17 goals. This teaches children that through the work they are doing and the knowledge they are learning, they can contribute to a fairer, more sustainable world.

Reflecting educational priorities

Education systems around the world are increasingly influenced by the aims present in the UN Sustainable Development Goals.

As an example, **Nepal's National Curriculum Framework (NCF 2076)** draws inspiration from the Sustainable Development Goals. The result of this is that children are increasingly engaged in practical problem solving, working in the context of pressing and real -world issues.

E-safety

Above all, when using technology, children should be safe. The most crucial aspect of this is supervision. Children should be supervised when using the internet at school and at home. Supervision can be in person and can involve using digital tools to monitor children's online activities. Children can also be kept safe online through effective teaching and discussion of E-safety, helping them to make good decisions when using technology and the internet.

Filtering and monitoring

Keeping children safe online requires effective monitoring. This should include there always being an adult present when children are using the internet and could include monitoring software.

Filtering means preventing harmful content from reaching children in the first place. Using software such as Cloudflare Families can block malware and adult content from reaching children's computers.

Artificial intelligence

Artificial intelligence is an emerging technology that is reshaping the world. It is a powerful technology and needs careful consideration for how it should be introduced to children in school. Some countries have decided to restrict children's access to AI over concerns around children forming relationships with AI and use of AI leading to decline of cognitive skills.

This strategy does not currently include provision for children using AI tools. Children will, however, develop their AI literacy skills through discussion of how AI works and examples of AI in the world around them. It should be noted that many children will have opportunities to use AI even without permission or instruction. For example, when performing an internet search, the top results often offer dialogue with AI chatbots. For this reason, teachers should set rules about how they would like children to use, or not use AI, supervise children and discuss AI tools with children as part of digital literacy lessons.

When teaching ICT or computing lessons, there are general principles of effective teaching to follow, as well as principles and techniques that are specific to teaching ICT and computing.

General principles of effective teaching

- **Preparation is key!** – Being prepared for the lesson is one of the most important parts of teaching good lessons. Here are ways that teachers of ICT can be prepared well for lessons:
    - Teachers should try computing projects themselves, prior to teaching them. (Video tutorials here!)
    - Teachers should have examples ready to show the children, to help them visualise success. (Scratch examples here!)
    - Ensure that laptops are ready and charged and that shortcuts are on desktops. Decide on a seating plan, make sure resources are organised, labelled and ready in the classroom.
    - Ensure that logins are set up for children, and that folders are ready for them to save work.
    - Make sure that you are clear on the time limits for the lesson, what should be achieved by which point. Plan for more than one lesson if needed.

- **Inclusion** – All children, no matter their abilities, special needs or backgrounds, should be included in the lesson and have the same chance to succeed. In practice, this means:
    - Considering barriers that children might face and making provision for them. This could be by seating children closer to the front, considering groupings, providing support in lessons (including video materials), assisting with organisation, regular check-ins and questioning.
    - Providing opportunities for gifted children to exceed the learning outcomes. This could be through access to extension work or extended learning outcomes, leading to

further challenge. Gifted children can also take on the role of 'experts' in the class and assist the teacher with teaching and managing the class, but this should not be to the detriment of opportunities to extend their own learning.

- **Growth mindset** – Where possible, discussion should take place about success being the result of hard work, repetition and commitment. This helps to dispel children's illusion that success is due of inherent ability, and as a result, lessens their tendency to give up!

- **Metacognitive strategies** – Metacognition means giving thought to the thinking processes we use. Children can be taught to solve problems by using effective thinking steps. These include:

  - Evaluating the problem, looking at requirements, materials available and constraints
  - Breaking the problem down
  - Looking for patterns, or thinking about where they have solved a similar problem before
  - Making a plan
  - Trying out a solution
  - Evaluating effectiveness

  To help children with this process children can use the 'S.M.A.R.T.S.' approach, which aims to provide a common metacognitive process to be used across subjects. Posters explaining this follow this section.

- **A culture of learning** – Learning should be celebrated in the class, whether this is success in a project or piece of work, a thoughtful answer to a question, effective planning or a valuable error or mistake.

- **Minimising cognitive load –** Cognitive lead theory recognises that children have a finite amount of processing power and attention

before they become tired and need to rest. Activities in lessons contribute to adding to cognitive load and depleting children's ability to produce work and solve problems. Teachers can minimise children's cognitive load by:

- Keeping teacher talk to a minimum. Short, concise inputs from teachers allow children to start their creative work quickly. Teachers can gather the class again for other instruction if needed.
- Breaking tasks down. Give children smaller, achievable tasks to perform instead of a larger project.
- Provide support materials. Like adults, children forget instructions. Consider printing support materials or giving children access to video tutorials such as these.
- In some cases, consider doing shared work as a class. The teacher guides the class through an activity one step at a time.
- Providing scaffolds for work. The teacher can provide the children with partially completed work, or with elements of the task that they will need. In computing, this can be writing an algorithm on the board or showing the children the coding blocks they will need.
-

- **Knowledge/skills v application in project work -** Just as in sport, it is important that children have the opportunity to develop and practise skills, as well as the chance to apply this to more open projects.

- **Active recall** – Challenge children frequently to recall their knowledge. This helps children to embed their knowledge into their long-term memory. This can take place by simply questioning children in class (try to question all children about the lesson), by proving short quizzes, by using flashcards or by using technology that tests children's knowledge. Children should get used to low-stakes testing and see this as an opportunity to 'see what they know'.

- **Agency** – Children should be active participants in their learning. They should take responsibility for their learning and success, rather than see learning and lesson as things that happen to them. Agency can be achieved by:

    - Providing children with opportunities to self-evaluate their work and set targets for next time (see the assessment section)
    - Giving children opportunities to verbalise their thoughts about lessons and their learning, and to say things that would help them
    - Showing children where they are on their learning journey, by sharing a long-term plan with them.

Principles specific to effective ICT/computing teaching

- **Preparation is key! -** (this has already been stated but is so important in computing/ICT Lessons!). Ensure Wi-Fi is working, laptops are charged, teacher materials are pre-loaded, digital materials are shared, usernames are ready and resources are labelled and organised. These things can make the difference between successful and non-successful ICT lessons.

- **Establish a lesson routine** – This should make time for children logging into devices, performing some independent work while they wait for others to be ready (such as touch typing, reflection on feedback or problem solving) and time to log off and put away equipment at the end of the lesson.

- **Establish and practise procedure for naming, saving and handing in work** – Children should know naming conventions for saving work and expectations about which folder they should save their work in. This will save teacher and student time in future lessons.

- **Focus on computational thinking and problem solving** – Children should be taught that an important aspect of computing is learning a process to solve problems. To do this, children can use an approach such as **computational thinking**, which is specific to computing. This includes the steps:

    - **Decomposing** the problem
    - **Looking for patterns**
    - **Abstraction** (focusing on the important part of the problem)
    - **Algorithm design**
    - **Debugging**

    Alternatively, children could use a common problem, solving approach, such as the **S.M.A.R.T.S. approach** (see below). Children should be taught that when they are stuck, there is always a process they should go through before asking the teacher. This can be called the **'4 B's'** approach:

    - Brain
    - Board
    - Buddy
    - Boss

    This approach encourages children to see the value of thinking, develops agency and independence and minimises the time spend by the teacher solving minor problems in the lesson.

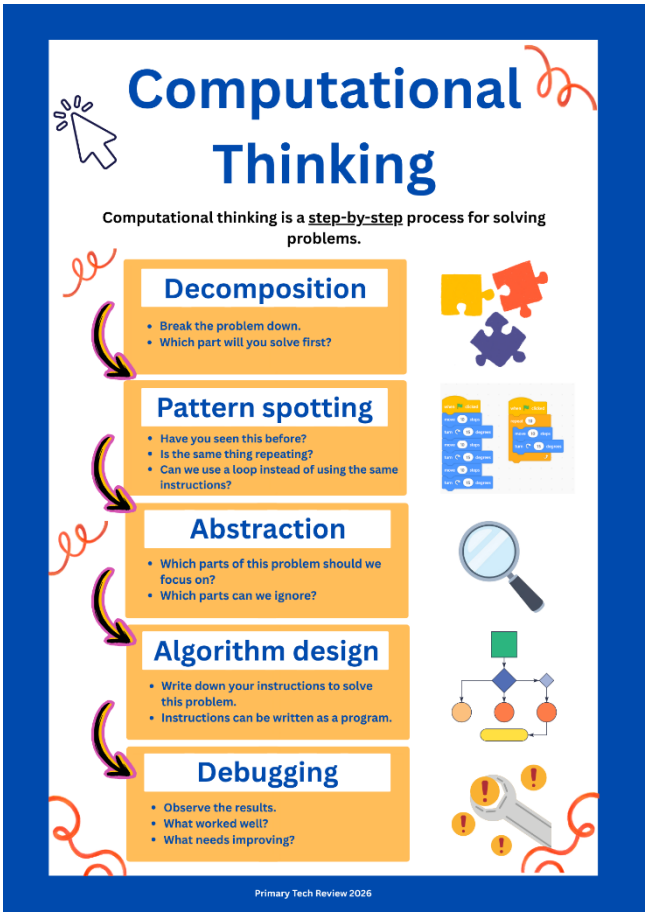
Computational thinking

Computational thinking is a valuable part of computing or ICT teaching. It is a strategy that teacher children a logical way to approach problems. This is particularly relevant to coding problems, but also relevant to areas including graphics design, 3D design and publishing. Computational thinking includes the following steps:

- **Decomposing** the problem
- **Looking for patterns**
- **Abstraction** (focusing on the important part of the problem)
- **Algorithm design**
- **Debugging**

This poster provides a child friendly version of an approach to computational thinking. This poster is available to download from

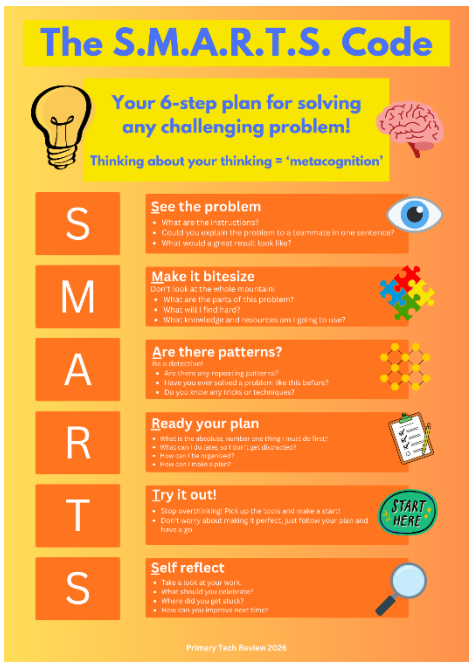
https://www.primarytechreview.co.uk/resources

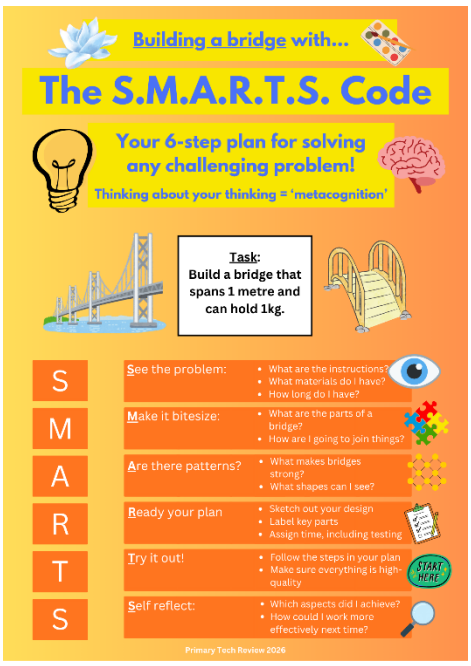


The S.M.A.R.T.S. code

The ‘S.M.A.R.T.S. code’ is a metacognitive approach designed to help children think about their thinking across the curriculum. The full set of A3 posters is available to download for free from

https://www.primarytechreview.co.uk/resources





PROGRESSION

The three stages of this strategy, **Foundation**, **Intermediate** and **Advanced** organise lessons according to children's skill and knowledge development.

The stages are linked to ages, since it is judged that children of those ages will be ready for the content of that stage. However, teachers should view the stage-structure as flexible and be willing to teach children according to their needs, rather than solely focusing on their age.

For example, a 13-year-old who has done little or no programming would benefit from studying at least some of the lessons in the foundation stage. These children might more at a faster pace than younger, beginner children. Older children might also learn differently and more independently and could be give more independent access to the video resources linked to this strategy.

In the next pages, milestones are provided to help teachers further understand the **Foundation**, **Intermediate** and **Advanced stages** and the skills they describe.

These milestones give teachers a broad overview of what children are aiming for in each stage of the nine development goals.

Part 4 of this strategy contains milestones and assessment materials. Teachers can use these assessment goals to inform their lesson planning and evaluate children's work, both within lessons (formative) and at the end of piece of work (summative).

Sharing milestones and assessment with children is an important part of making them active participants in the learning process and increasing their **agency**.

Children should know in each lesson what goal they are working towards, should see the success criteria for achieving this and should be able to make their own judgement about what they have achieved.

In **sections 4 - 6**, lesson plans are provided for the three stages of this strategy, **Foundation stage (age 7 – 9)**, **Intermediate stage (age 10 – 12)** and **Advanced stage (age 13 – 16).**

For each stage, lessons are provided, covering easch of the **nine development goals**:

Foundational academic skills	**Goal 1**	**Goal 2**	**Goal 3**
	Foundational academic excellence (via **E-Paath**)	**Functional and professional English**	**Geographic information and mapping**
Understanding and using technology	**Goal 4**	**Goal 5**	**Goal 6**
	Introduction to software logic	**Understanding computer systems and networks**	**Digital literacy, E-safety and AI**
Applying technology to agriculture	**Goal 7**	**Goal 8**	**Goal 9**
	Agricultural financial literacy	**Agricultural science** (via **E-Pustakalaya**)	**Modern marketing and business**

Some topics feature one or two lessons per goal, others feature more lessons, allowing teachers to explore topics in more depth.

The lessons **do not have to be taught in chronological order.** Teachers can select which lessons are most applicable to their students and add these to a yearly plan.

Teachers might decide to teach some of the lessons from a stage to one year group and save some for another year group.

Teachers might also decide that one of the given lessons merits spending several weeks on it. This might be particularly true for some of the programming projects, or electronics projects, or lessons where children make (and give) presentations.

As stated earlier, teachers should also feel free to teach lessons from one stage to children of a different age. Children just staring out programming for example, would not benefit from going straight to the advanced stage, even if they are 13 years old.

EXAMPLE YEARLY-PLANNING

In this **example yearly plan**, we can see how the teacher has split lessons from this strategy over 22 ICT lessons in a school year.

The teacher has decided to teach lesson **1-9 (skipping lesson 5)** to give an introduction to computing. The teacher has then taught **three coding projects**, teaching these coding lessons over two weeks each to give children time to develop and explore the projects fully.

Week	Lesson	Lesson focus
1	**N/A**	Introduction to the year. Agreeing ICT acceptable use. Lesson routines. Touch typing practice.
2	**Lesson 2**	Typing skills.
3	**Lesson 3**	Developing English vocabulary of everyday words
4	**Lesson 4**	Searching digital libraries
5	**Lesson 1**	Counting and grouping objects (academic skills)
6	**Lesson 10**	Programming a sprite to move left and right
7	**Lesson 10**	Programming a sprite to move left and right (continued)
8	**Lesson 10**	Programming a sprite to move left and right (continued)
9	**Lesson 11**	Programming the appearance of a sprite to change
10	**Lesson 11**	Programming the appearance of a sprite to change (continued)
11	**Lesson 12**	Programming the appearance of a sprite to change (continued)
12	**Lesson 15**	Programming a Microbit step counter (continued)
13	**Lesson 15**	Programming a Microbit step counter (continued)
14	**Lesson 15**	Programming a Microbit step counter (continued)
15	**Lesson 19**	E-safety
16	**Lesson 20**	E-safety
17	**Lesson 30**	English skills
18	**Lesson 31**	English skills
19	**Lesson 25**	Creating presentations
20	**Lesson 26**	Creating presentations
21	**N/A**	Preparation of a presentation about computing work this year
22	**N/A**	Delivering a presentation about computing work this year.

The teacher has then taught **E-safety** again for two weeks, to ensure that discussion of these topics remains fresh and that children have frequent opportunities to share their ideas and experience.

The teacher has then taught several lessons with a focus on **English** and **typing skills,** before giving the children a chance to develop their **presentation skills.**

The year ends with the children **creating and delivering a presentation** with screenshots of their computing work for the class.

Not all of the foundational lesson are taught in this year; the teacher has selected the lessons that the children need and arranged them into a logical sequence.

Touch typing will also be a part of most lessons, with the children doing typing at the starts of lessons for 10 – 15 minutes.

E-safety discussion should also regularly take place at the starts of lessons. An effective way to do this is with a 10-minute discussion of an E-safety scenario.

The teacher will deliver other foundation lessons in other academic years. The teacher may even decide to repeat or adapt some lessons if needed.

TEACHER TRAINING

When providing teachers with equipment and resources, it is important that teachers feel confident at using these resources. It is important that teachers have the chance to try and explore the resources they will use with children. An initial four-day program should allow teachers to become familiar with resources and discuss planning and routines.

Day	Training goals	Suggested activities
1 - Technology familiarisation	Introduce goals of the strategy Build basic teacher confidence with laptops, trackpads and LibreOffice	Turning on laptops, logging in. Navigating Windows and the internet. Saving bookmarks. Connecting to Wi-Fi. Creating a document and spreadsheet in Libre Office.
2 – Confidence with hardware and software	Navigate E-Paath, E-Pustakalaya and configuring network and devices	Time spent exploring the online resources. Teachers sign up to Canva, code.org, and Tinkercad. Create student accounts.
3 - Blended lesson design	Planning a week of classes combining digital tools with textbooks	Teachers examine lesson objectives and decide where digital tools are useful. Create lesson plans using Libre Office.
4 - Care and troubleshooting	Power management, device care, classroom routines and general troubleshooting	Teachers establish routines for device care. Connect and disconnect to networks. Set up folders on network and devices.

NEXT STEPS AND CHECKLIST

Activity	Check
Read an annotate this strategy critically, suggesting changes to resources, objectives and content.	☐
If the resource list is approved, install software on laptops. Add internet shortcuts and bookmarks where needed.	☐
Set up logins for computers, either as individual student logins, or generic user logins (i.e. user1). Make a record of the logins that students use.	☐
Create a student **ICT acceptable use agreement**, detailing rules that the students must follow. This could be in the format of 'I will' and 'I will never'. Provide a document for all students to sign to agree to follow these rules, once they have been discussed and agreed.	☐
Ensure that all laptops are connected to the Wi-Fi network and that filtering software is running (i.e. Cloudflare Families – seek advice with this).	☐
Create logins for children to use on the computers. Ensure that it is clear where and how they will save their work.	
Provide time for teachers to receive training on this strategy, using the laptops and the online services and apps. Give teachers time to experiment with the software and ask them to select and teach a lesson to each other. Ensure time for feedback and modification.	☐
Encourage teachers to draft their own lessons, based on the objectives in this strategy, or following on from the example lessons.	☐
In lessons, consider pairing up foundation students with advanced students for the first 15 minute of lessons. This helps foundation students set up and develops sense of community.	☐
Contact the author with any requests for remote training, guidance, or requests for follow-up materials.	☐

ABOUT THE AUTHOR

Owen Dobbing is a primary teacher from the UK. He works at a school in London where he teaches children and trains staff on using technology effectively.

Owen has also taught in South Korea and Thailand. He is passionate about helping children fulfil their potential through offering an effective and inspiring education.

Owen's philosophy of teaching is based around giving children opportunities to acquire knowledge and skills and giving them opportunities to practise these through project work. Children need to be taught the value of hard work, repetition and effective study, gradually taking increased agency for their learning and applying this to solve problems.

Owen advocates that technology presents incredible opportunities and challenges for children. Children can now access a vast range of resources and information immediately and at their own pace. This includes video resources, digital tools for learning and creative tools.

At the same time. children need guidance to make sure that they use technology safely and in ways that support their cognitive development.

Owen is delighted to be able to support children and teachers around the world by sharing expertise and facilitating development of additional learning materials.

REFERENCES AND ACKNOWLEDGEMENTS

This scheme of work references third-party interactive educational software, applications, platforms and media resources:

Productivity, Graphics design and desktop software

- **LibreOffice** (The Document Foundation) – Free and open-source office suite used under the Mozilla Public Licence v2.0 (MPLv2).
- **Microsoft Word, PowerPoint, Excel** (Microsoft Corporation) – Referenced for word processing, presentation creation and spreadsheet creation instruction

Coding, AI and Physical Computing

- **Code.org (AI for Oceans)** – Used under Code.org's free educational terms
- **Microsoft MakeCode** (Microsoft Corporation) – Used under Microsoft's free educational terms
  Microsoft makecode screenshots used with permission from Microsoft Corporation. Interface images © Microsoft. Microsoft and makecode are trademarks of the Microsoft group of companies.
- **Micro:bit Python Editor** (Microbit Educational Foundation) – Used for educational coding under the Micro:bit Educational Foundation terms
- BBC Microbit Python Editor screenshots used courtesy if the Micro:bit Educarional Foundation (microbit.org). Micropython is licences under the MIT licence. 'Python' is a registered trademark of the Python Software foundation.
- **Mu** (Code with Mu) – Free, open-source Python IDE used under the GNU General Public Licence v3 (GPLv3).
- **Scratch** (MIT Media lab) – Used under MIT's free educational terms and Creative Commons Attribution-ShareAlike 2.0 licence

Scratch is a project of the Scratch Foundation. It is available for free at http://scratch.mit.edu. Interface screenshots used under Creative Commons Atribution-ShareAlike terms.

Interactive Geography, Learning platforms, Research and Practise tools

- **E-Paath and E-Pustakalaya** (OLE Nepal) – Licenced under CC BY-NC-SA 3.0
- **Google Earth** (Google LLC) – mapping and spatial exploration tool used under Google Earth's standard permissions for educational reference
- **LearnEnglish Kids** (British Council) – Used for educational classroom reference
- **Typing.com** (Teaching.com) – used for educational touch-typing instruction and practice

Information and Media Resources

- **Wikipedia** (Wikimedia Foundation) – Content referenced under the Creative Commons Attribution ShareAlike 4.0 (CC BY-SA 4.0) Licence
- **Pixabay** – Stock images and visual assets used under the Pixabay Content Licence (free for educational and commercial use)
- **United Nations Sustainable Development Goals** – (United Nations) - https://www.un.org/sustainabledevelopment - Used according to the guidelines on the United Nations website, for non-commercial, educational purposes. The content of this publication has not been approved by the United Nations and does not reflect the views of the United Nations or its officials or Member States.
- **YouTube** (Google LLC) – Educational videos embedded or linked under YouTube's standard terms of service