



Intent

At St Monica's Primary School, our intent is to equip every child with the knowledge, skills, and confidence to thrive in a digitally connected world. With a well thought out Computing Curriculum, we aim to foster computational thinking, digital literacy, and creativity from Early Years through Year 6. We want pupils to understand how digital systems work, how to use technology safely and responsibly, and how to become creators—not just consumers—of digital content.

We aim for pupils to:

- Develop logical thinking and problem-solving skills through programming and computer science.
- Use a range of technology confidently across different contexts and subjects.
- Understand and apply the fundamental principles of computing.
- Become responsible, competent, confident, and creative users of information and communication technology.
- Recognise the importance of e-safety and use technology respectfully and safely.

Implementation

Computing at St Monica's is taught through a well-sequenced and progressive curriculum developed by the Computing lead. This curriculum ensures full coverage of the National Curriculum for Computing, while allowing flexibility to integrate technology into other subjects.

Key aspects of implementation include:

- **Structured lessons** that follow a clear progression from EYFS to Year 6, building on prior knowledge and developing increasingly complex skills.
- Use of **practical, hands-on experiences** using devices such as iPads, Chromebooks, and desktop computers.
- Clear focus on three strands of the computing curriculum: **Computer Science, Information Technology, and Digital Literacy**.
- Regular opportunities for **coding** using age-appropriate tools (e.g., Bee-Bots, Scratch, and Python in upper KS2).
- **E-safety** is woven into lessons throughout the year, and whole-school events such as Safer Internet Day reinforce responsible use of technology.
- Cross-curricular use of computing supports learning in subjects such as maths, science, literacy, and art.

Impact

The impact of our computing curriculum is visible in the confidence and competence of our pupils when using technology. By the time children leave St Monica's, they will be able to:

- Apply their computational thinking and coding skills to solve real-world problems.
- Use a variety of software and devices to collect, present, and analyse information effectively.
- Make informed and safe choices when navigating digital environments.
- Demonstrate resilience, collaboration, and critical thinking when working with technology.

We assess computing both formatively (through observation, discussion, and outcomes) and summative (through end-of-unit tasks and digital portfolios). The St Monica's computing curriculum ensures that all learners, including SEND and EAL pupils, are supported and challenged appropriately.

Pupil voice, work samples, and teacher feedback show that computing is an enjoyable, relevant, and vital part of the St Monica's curriculum, preparing children to be active participants in a digital future.



Computing Expectations at St Monica's Primary School (EYFS–Year 6)

Based on the St Monica's Computing curriculum | Aligned with the National Curriculum

EYFS (Early Years Foundation Stage)

In the Early Years, computing is not taught as a discrete subject but through the strand of **Understanding the World** and **Expressive Arts and Design**. Children experience and explore technology in meaningful, hands-on ways.

Children are expected to:

- Recognise and name simple technology used at home and in school (e.g., tablets, interactive whiteboards, digital cameras).
- Begin to understand cause and effect using age-appropriate devices (e.g., pressing a button to produce sound or light).
- Use technology for a purpose (e.g., drawing pictures on a tablet, using a voice recorder).
- Begin to understand staying safe online through stories and discussions.

Links to EYFS Framework:

- Understanding the World: Recognise that a range of technology is used in places such as homes and schools.
- Expressive Arts: Use software to draw, create and share simple ideas.

Key Stage 1 (Years 1 & 2)

Children begin to develop foundational computing skills across the three strands: **Computer Science**, **Information Technology**, and **Digital Literacy**.

Children are expected to:

- Understand what algorithms are and how they can be implemented as programs (e.g., using Bee-Bots, Scratch Jr).
- Create and debug simple programs using logical reasoning to predict and correct errors.
- Use technology to create, organise, store, manipulate, and retrieve digital content (e.g., simple documents, drawings, photos).
- Use technology safely and respectfully, keeping personal information private.
- Know where to go for help if they feel unsafe online.

Links to the National Curriculum:

- Pupils should be taught to understand what algorithms are and how they are implemented.
- Use technology purposefully to create, organise, store, manipulate and retrieve content.
- Recognise common uses of IT beyond school (e.g., barcodes, tills, smart speakers).
- Use technology safely and respectfully.



Lower Key Stage 2 (Years 3 & 4)

Children deepen their understanding of programming, digital systems, and begin using software for more purposeful tasks.

Children are expected to:

- Design, write, and debug programs that accomplish specific goals (e.g., animations or games using Scratch).
- Use sequence, selection, and repetition in programs and work with variables.
- Understand how computer networks work, including the Internet and the World Wide Web.
- Use a variety of software (e.g., Microsoft Office, Google Workspace) to present data and information.
- Develop digital research skills and evaluate digital content.
- Recognise acceptable/unacceptable behaviour online and understand basic online safety rules.

Links to the National Curriculum:

- Use logical reasoning to explain how simple algorithms work.
- Understand computer networks, including the Internet.
- Select, use and combine a variety of software on a range of devices.
- Use technology safely and responsibly, recognising acceptable behaviour and knowing how to report concerns.

Upper Key Stage 2 (Years 5 & 6)

Pupils refine their programming and information-handling skills and become responsible, informed digital citizens.

Children are expected to:

- Use a range of programming languages (e.g., Scratch, Python basics) to design and debug complex programs.
- Understand how search engines work and how to use them effectively.
- Create, refine and present data and content using a wide range of multimedia tools.
- Understand the principles of data handling, spreadsheets, and databases.
- Demonstrate critical awareness of online safety, including the reliability of sources, digital footprint, and online privacy.
- Collaborate effectively on digital projects, demonstrating responsibility and creativity.

Links to the National Curriculum:

- Design, write, and debug programs controlling or simulating physical systems.
- Understand how the internet provides multiple services and how search engines work.
- Select, use, and combine software to accomplish given goals, including collecting and analysing data.
- Use technology safely, respectfully, and responsibly, including protecting online identity and reporting harmful content or contact.

Progression Overview

Strand	EYFS	Year 1	Year 2	Year 3	Year 4	Year 5	Year 6
Programming & Algorithms	Explore cause and effect (pressing	Give simple instructions to make	Create and debug simple programs	Use loops and events to make interactive programs. Begin to explain how	Use variables, repetition, and	Use more complex loops, variables, and	Design, write, and debug programs that accomplish

St-Monica's Roman Catholic Primary school
Computing



	buttons, making things move). Follow simple instructions.	something happen (e.g. Bee-Bot). Begin to understand that computers follow commands.	using step-by-step instructions. Sequence events to make simple animations (ScratchJr/ 2Code).	algorithms work. (Scratch / 2Code)	conditions. Design and test programs with more than one outcome.	nested conditions. Begin text-based coding (Python).	specific goals using both block and text-based coding (Scratch & Python). Use logical reasoning to detect and correct errors.
Data & Information	Collect and sort simple data (colours, shapes). Recognise symbols represent data.	Represent data in pictograms or block charts with support. Talk about what data shows.	Use simple databases and graphs to represent information. Sort and label digital data.	Collect, input, and present data in charts and tables. Use simple formulas (Excel/Sheets).	Analyse and present data using formulas and graphs. Compare data sets to identify patterns.	Use spreadsheets to analyse large data sets, apply filters, and use conditional formatting.	Collect, analyse, evaluate, and present data effectively for a specific purpose. Interpret complex data to make conclusions.
Multimedia / Digital Creativity	Use digital devices to take photos or draw pictures. Explore mark-making apps.	Create simple digital artwork and text (Paint, Jit). Add text to images.	Combine text and images to make posters or stories (2Publish, Paint 3D).	Create presentations and digital stories (PowerPoint, Google Slides). Add images, transitions, and sounds.	Combine media (text, sound, video) for specific purposes. Evaluate effectiveness.	Plan and produce multimedia projects such as videos or animations (Stop Motion Studio, Canva).	Design and present advanced multimedia projects integrating audio, video, and interactive elements. Evaluate and improve outcomes.
Digital Literacy & Online Safety	Recognise technology at home/school. Talk about using devices safely with an adult.	Understand that we should keep information private. Know what to do if something worries us online.	Understand personal information and why it must be kept private. Identify safe/unsafe online behaviour.	Recognise acceptable/unacceptable online behaviour. Evaluate the reliability of online information.	Explain how the internet works (basic networks). Know what to do about cyberbullying and unsafe content.	Understand digital footprints and how online actions can affect others. Create and follow e-safety rules.	Demonstrate responsible digital citizenship: evaluate online content critically, understand misinformation, and maintain positive digital relationships.
Computing Systems & Networks	Recognise technology used at home or school.	Identify common technology (computers, tablets, digital cameras).	Understand computers and devices need instructions. Recognise simple networks.	Identify the main parts of a computer. Understand the internet as a network of computers.	Understand how data is transmitted (network, router, server). Recognise importance of digital security.	Explain how networks and the internet work. Discuss reliability of digital communication.	Understand complex networks, cloud computing, and data transfer. Identify potential risks and safe practices.

Step-by-Step Structure of a Primary Computing Lesson

1. Starter / Review (5–10 mins)

Purpose: Activate prior knowledge and prepare pupils for new learning.



- Quick recap of previous learning (e.g., a short quiz, question discussion, or code snippet analysis).
- Use of mini whiteboards or visual aids to prompt recall.
- Introduce today's learning objective and success criteria (e.g., "Today we will create a program with a loop").

2. Teacher Input / Modelling (10–15 mins)

Purpose: Explicitly teach or demonstrate the new concept or skill.

- Model the skill or concept live using screen sharing or interactive boards (e.g., creating a Scratch program).
- Explain technical vocabulary clearly (e.g., "loop", "algorithm", "variable").
- Use questioning to check for understanding.
- Highlight key misconceptions or common mistakes.

Example:

In Year 4: "Watch how I use a loop to repeat this sound effect in Scratch."

3. Guided Practice (10–15 mins)

Purpose: Pupils try the task with teacher support.

- Pupils work in pairs or independently with the teacher circulating to support.
- Scaffold tasks or challenges (e.g., partially completed code for pupils to debug).
- Use of visual prompts, sentence stems, or success checklists.
- Support for SEND/EAL through visuals, paired work, or chunked instructions.

4. Independent Task (15–20 mins)

Purpose: Pupils apply their learning through a meaningful, hands-on task.

- Pupils create or build something based on the lesson objective (e.g., a program, digital poster, or animation).
- Tasks are open-ended or tiered by difficulty to challenge all learners.
- Pupils may work individually or collaboratively (e.g., pair programming or shared presentations).
- Teacher continues to support and assess through observation.

Example:

Year 5: Pupils design a quiz in Scratch using variables and user input.

5. Plenary / Reflection (5–10 mins)

Purpose: Consolidate learning, assess understanding, and celebrate success.

- Share examples of pupils' work on the screen.
- Discuss what went well and any difficulties encountered.
- Use questioning to assess understanding (e.g., "What is a bug and how do we fix it?").
- Pupils self-assess or peer-assess using success criteria or traffic lights.
- Reinforce online safety or digital citizenship if applicable.



6. Optional: Cross-Curricular or Extension Links

Purpose: Deepen learning through real-world or subject integration.

- Link computing concepts to maths (logic), science (systems), or literacy (digital storytelling).
- Provide optional challenges or enrichment for fast finishers (e.g., code a game level, explore a different platform).

An example of the progression of skills and Computing curriculum for Computing

St Monica's Computing Skills Progression Map

Strand	EYFS (Ages 3–5)	KS1 (Ages 5–7)	Lower KS2 (Years 3–4)	Upper KS2 (Years 5–6)
Computer Science	• Explore cause and effect using toys and digital devices (e.g., pressing buttons, simple coding toys). • Begin to understand that technology follows instructions. • Sequence actions in play (storytelling, routines).	• Understand what algorithms are and how they are implemented as programs. • Create and debug simple programs (e.g., Bee-Bots, ScratchJr). • Use logical reasoning to predict program behavior.	• Design, write, and debug programs using sequences, selection, and repetition. • Use variables and logical reasoning to explain and correct errors. • Understand that computers process data using precise instructions.	• Plan, test, and refine solutions in more advanced programming environments (Scratch, Python, micro:bit). • Apply decomposition and abstraction to structure larger programs. • Understand how networks and the internet enable communication and data exchange.
Information Technology	• Explore digital tools through play (drawing apps, digital cameras, voice recorders). • Use technology for a purpose (taking photos, playing music).	• Use technology to create, store, and retrieve digital content (text, images, sound). • Recognise common uses of IT beyond school.	• Combine applications to create purposeful content (text, images, audio, data). • Organise and manipulate information using spreadsheets or simple databases. • Understand how to manage files and use cloud storage.	• Select and evaluate digital tools for a purpose and audience. • Collect, analyse, and present data effectively. • Understand how digital systems are used in the real world and their impact on society.
Digital Literacy	• Explore technology safely with guidance. • Understand that devices need care and rules. • Learn about sharing and privacy through stories.	• Use technology safely and respectfully. • Keep personal information private. • Know who to tell if something feels wrong online.	• Understand responsible online behavior. • Recognise and manage online risks. • Question reliability and bias in online content.	• Demonstrate safe, responsible, and ethical use of technology. • Understand digital footprints, data privacy, and media influence. • Critically evaluate content, context, and credibility online.
Computational Thinking Skills (Cross-Strand)	• Explore patterns, sequences, and logical play (sorting, matching). • Follow and give simple instructions.	• Use step-by-step thinking to solve small problems. • Begin to break down tasks into smaller parts.	• Apply decomposition, pattern recognition, and logic to design solutions. • Test and debug systematically.	• Apply abstraction and evaluation to improve solutions. • Refine programs for efficiency and accuracy. • Collaborate effectively to plan and test digital projects.

Progression Summary (End of KS2)

By the end of **Key Stage 2**, pupils should be able to:

- **Design, write, and debug programs** that use sequence, selection, and repetition to achieve goals.



- **Use variables, inputs, and outputs** effectively, applying logical reasoning to detect and correct errors.
- **Understand how computer networks, including the internet, work**, and how data is transmitted securely.
- **Use a wide range of digital applications** to collect, analyse, and present information for a given audience.
- **Evaluate digital tools and outcomes**, choosing appropriate ones for specific purposes.
- **Demonstrate strong digital citizenship**, acting responsibly, respectfully, and safely online.
- **Think computationally** — breaking down complex problems, recognizing patterns, abstracting key ideas, and developing efficient solutions.
- **Collaborate effectively on digital projects**, showing creativity, critical thinking, and problem-solving skills that prepare them for Key Stage 3 computing.

Areas for development:

1. Increase Cross-Curricular Integration

Why: Computing is still often taught in isolation.

Improvement Action:

- Embed computing more consistently across subjects —e.g., data handling in science, story coding in English, presentations in history.
- Provide planning support for teachers to link computing objectives to real class topics.

2. Strengthen Assessment and Progress Tracking

Why: Computing progress is harder to quantify than in core subjects.

Improvement Action:

- Introduce a simple, consistent assessment framework or digital portfolio system to track individual skills.
- Use termly formative assessments or 'can-do' statements tied to the skills map.

3. Develop Staff Confidence and CPD

Why: Some staff may lack confidence, especially in teaching coding or managing unfamiliar software.

Improvement Action:

- Offer CPD focused on coding tools (e.g., Scratch, Python), using cloud platforms (Google Classroom), or differentiating computing for SEND.
- Provide model lessons or team teaching where confident staff can support others.

4. Enhance Support for SEND & EAL Learners

Why: Computing lessons may move quickly and rely on technical vocabulary.

Improvement Action:



- Provide more visual aids, pre-teaching of key vocabulary, and structured scaffolds (e.g., word banks, code cards).
- Adapt software tasks using step-by-step guides and screen recordings.

5. Improve Equipment Access and Maintenance

Why: Inconsistent access to hardware or outdated tech slows down learning.

Improvement Action:

- Audit devices and software across classes; ensure every class has regular, reliable access.
- Create a rolling program for updating devices and maintaining digital resources.

6. Expand Pupil Opportunities for Real-World Application

Why: Pupils need opportunities to see computing's real-life relevance.

Improvement Action:

- Introduce enrichment activities like coding clubs, robotics, digital media projects, or virtual school blogs.
- Link learning to careers and industries using real-world tech (e.g., graphic design, cybersecurity, animation).

7. Deepen Online Safety Education

Why: Online safety is essential but can feel superficial if taught only during Safer Internet Day.

Improvement Action:

- Integrate digital literacy and safety regularly through PSHE, assemblies, and all computing units.
- Create pupil-led safety campaigns or peer support roles (e.g., "Digital Leaders").

8. Celebrate Computing Achievements More Widely

Why: Computing work can feel hidden compared to literacy or art.

Improvement Action:

- Showcase pupil projects on school displays, newsletters, or class websites.
- Celebrate "Digital Star of the Week" or similar awards to boost engagement.

Successes in Autumn 2024-2025

At St Monica's, Computing has become a vibrant and valued part of our broad and balanced curriculum. Guided by the St-Monica's Computing curriculum (National curriculum for computing), we have made significant strides in ensuring our pupils are digitally literate, confident, and safe technology users.



1. Strong Curriculum Implementation

- We have fully embedded the St-Monica's Computing Curriculum across all year groups.
- Lessons are well-sequenced and progressively build on knowledge, from EYFS exploratory play to complex coding in Year 6.
- Teachers use consistent planning and vocabulary, ensuring clarity and coherence across the school.

2. High Pupil Engagement and Enjoyment

- Pupils consistently express enthusiasm for computing lessons, especially coding, animation, and creative digital tasks.
- Projects such as creating interactive stories, digital posters, and Scratch games foster creativity and problem-solving.
- Pupils are motivated and often extend their learning at home via platforms like Scratch or Google Workspace.

3. Development of Core Digital Skills

- Children from EYFS upwards are confident in using a range of devices including tablets, Chromebooks, and desktop computers.
- Pupils can confidently navigate age-appropriate apps and software to complete tasks across all three strands: Computer Science, Information Technology, and Digital Literacy.
- By Year 6, pupils can write structured code, debug errors, present data, and make informed digital decisions.

4. Online Safety Awareness

- Pupils demonstrate a strong understanding of e-safety principles, reinforced through PSHE links, assemblies, and Safer Internet Day.
- Children can articulate how to stay safe online, protect personal information, and respond appropriately to concerns.

5. Inclusive Computing Practice

- Our computing provision is accessible to all pupils, including SEND and EAL learners, with differentiated tasks and scaffolding built in.
- Staff use visual aids, paired programming, and step-by-step models to ensure all children can access core content and succeed.

6. Teacher Confidence and Consistency

- Staff are increasingly confident in delivering computing lessons, supported by trust-wide CPD and shared resources.
- Lessons show clear modelling, use of technical vocabulary, and growing confidence in teaching more advanced coding and digital concepts.

Support and planning of SEND during computing lessons



1. Differentiation and Scaffolding

- Break down tasks into small, manageable steps.
- Use visual instructions (e.g. screenshot-based guides or video tutorials).
- Provide simplified versions of tasks with clear success criteria.
- Offer templates or partially completed code to help pupils start.

2. Multi-Sensory & Visual Approaches

- Use block-based coding tools like **Scratch Jr** and **Scratch** to support sequencing with colour and shapes.
- Integrate visual learning tools (e.g., icons, diagrams, GIFs).
- Offer unplugged computing activities to reinforce concepts physically (e.g., programming a partner or drawing flowcharts).

3. Use of Assistive Technology

- Speech-to-text or text-to-speech tools for pupils with literacy needs.
- Magnifiers, alternative keyboards, or switch access for pupils with physical disabilities.
- Dictation tools or audio instructions for learners who struggle with reading.

4. Collaborative Learning

- Pair SEND pupils with supportive peers in mixed-ability pair programming or group tasks.
- Encourage peer modelling and shared control over devices.

5. Clear Instructions and Vocabulary

- Pre-teach computing vocabulary with visual support and simplified definitions.
- Revisit new terms regularly and link them to practical use (e.g. "algorithm = set of instructions").

6. Supportive Environment

- Reduce sensory distractions (e.g., headphones to minimise noise, screen filters).
- Allow flexible timing for tasks.
- Use calm, predictable routines and give early warnings before transitions.

Type of Need	Support Example
Cognitive/ Learning	Simplify tasks, provide visual checklists, use drag-and-drop interfaces.
Communication & Interaction	Use symbols, visuals, or short videos to explain concepts; model code verbally.
Social, Emotional & Mental Health	Provide choice and autonomy; praise effort and creativity; reduce performance pressure.
Sensory/Physical	Use accessible hardware (e.g. touchscreens, large keyboards); ensure physical space is navigable.



Who are the lowest 20%?

Pupils who:

- Consistently perform below age-related expectations.
- Struggle with reading, writing, or following instructions.
- Lack confidence using digital tools independently.
- May have additional barriers such as language needs or disrupted learning.

Key Strategies to Support the Lowest 20%

1. Pre-Teaching and Vocabulary Support

- Introduce key computing terms (e.g., "algorithm", "debug", "loop") *before* lessons using visuals and simplified language.
- Use **dual coding**: pair words with icons, animations, or real-life objects.

2. Chunked and Scaffolder Tasks

- Break tasks into small, manageable steps with clear, numbered instructions.
- Use checklists, templates, and worked examples.
- Provide partially completed code, drag-and-drop options, or scaffolder inputs.

3. Modelled and Repeated Instruction

- Explicitly model each new skill on-screen or physically (unplugged).
- Repeat instructions slowly and clearly, allowing time for processing.
- Offer video tutorials or visuals to reinforce steps pupils can revisit independently.

4. Increased Adult Support

- Target TA or teacher support towards the lowest 20% during independent tasks.
- Use guided groups for coding, typing or navigating software.
- Provide patient encouragement to build independence over time.

5. Use of High-Engagement Tools

- Use visual, block-based platforms (e.g., Scratch, Scratch Jr) to reduce cognitive load.
- Introduce unplugged activities or physical computing (e.g., Bee-Bots, micro: bits) to support hands-on learners.

6. Time and Repetition

- Allow more time for tasks, avoiding rushing.
- Repeat core computing concepts across multiple weeks to support retention.
- Use spaced retrieval (e.g., weekly mini-quizzes or "do now" retrieval questions).

7. Celebrate Small Wins

- Recognise and reward progress (not perfection): e.g., completing a step, solving a bug, showing resilience.
- Provide leadership roles where possible (e.g., tech helper, code checker) to build self-esteem.



Examples in Practice at St Monica's

- **Year 1** pupils struggling with sequencing use large floor arrows and real-world tasks before using Bee-Bots.
- **Year 3** low-attaining pupils create a Scratch animation using a visual checklist with pre-loaded sprites.
- **Year 5** pupils revisit loops and variables through guided code-along each week until confident.

Monitoring Progress

- Use simple assessment rubrics with visual cues (e.g., smiley faces, coloured dots).
- Keep a digital or paper portfolio of work samples showing progress over time.
- Liaise with the SENDCo, literacy/maths leads and class teachers to align interventions.

Whole-School Actions to Support Lowest 20%

- Identify pupils in the lowest 20% using data and teacher input each term.
- Ensure they receive additional support in computing lessons or targeted clubs.
- Provide CPD for staff on adapting computing for low attainers.

Professional development

CPD Overview (Annual Cycle)

Term	Focus Area	Target Audience	Delivery Format
Autumn 1	Curriculum Overview & Progression in Computing (What computing looks like from EYFS to Y6)	All teachers, TAs	Staff meeting / Inset day
Autumn 2	Effective Computing Lessons (Modelling, vocabulary, questioning, assessment)	KS1 & KS2 teachers	Team meetings / Peer observations
Spring 1	Introduction to Coding: Scratch & ScratchJr (Variables, loops, debugging, pair programming)	Years 2–6 teachers	Hands-on twilight / workshop
Spring 2	Inclusion in Computing (Adapting for SEND, EAL, and lowest 20%)	All staff	CPD session / TA training
Summer 1	Online Safety & Digital Literacy (Embedding safety into the curriculum)	All teachers, DSL	DSL-led CPD / PSHE links
Summer 2	Showcase & Review (Sharing pupil projects, reflecting on strengths/gaps)	All staff	Staff meeting / Learning walk debrief

Ongoing CPD Themes



1. Curriculum Confidence

- Deepen understanding of the three strands:
Computer Science
Information Technology
Digital Literacy
- Use St Monica's computing overview and progression map/National curriculum as a core reference.

2. Technical Skills Development

- Practice using core tools used in lessons:
 - Scratch / Scratch Jr
 - Bee-Bots / micro:bits
 - Google Docs/Slides
 - Typing and multimedia tools
- Offer optional drop-in sessions or recorded walkthroughs.

3. Inclusive Teaching in Computing

- Strategies for supporting:
 - SEND (scaffolds, visuals, assistive tech)
 - EAL (pre-teaching, vocabulary support)
 - Low confidence pupils (guided steps, repetition)

4. Assessment for Learning

- How to assess computing without levels:
 - Pupil self-assessment
 - Skills checklists
 - Digital portfolios (e.g., Google Classroom)
 - Peer code reviews

Measuring Impact

- Pupil voice: confidence, enjoyment, vocabulary use
- Lesson drop-ins: teaching quality, engagement, use of tech
- Staff surveys: confidence before/after CPD
- Digital work samples across year groups

Looking to the future of Computing at St Monica's

At St Monica's, we aim to prepare our pupils for a fast-changing digital world by equipping them with the computing skills, knowledge, and mind-set to become **confident, creative, and safe digital citizens**. As we look to the future, we are committed to deepening computing across the curriculum, enhancing staff expertise, and ensuring no child is left behind in digital learning.

Key Priorities for Future Development

1. Deepen Cross-Curricular Computing

- Embed computing meaningfully into subjects such as Science, Geography, and English (e.g., coding simulations in Science, digital storytelling in English).
- Use digital tools (e.g., spreadsheets, presentations, data loggers) to enhance learning outcomes across the curriculum.



2. Build a Culture of Digital Creativity

- Encourage project-based learning in upper KS2, allowing pupils to create games, animations, or websites linked to real-world topics.
- Introduce multimedia units involving video editing, podcasting, and digital art.

3. Strengthen Progression and Assessment

- Refine progression tracking using pupil portfolios or skills rubrics from EYFS to Year 6.
- Develop moderation opportunities across the Trust to align expectations and share good practice.

4. Promote Digital Leadership and Pupil Voice

- Establish a team of **Digital Leaders** from KS2 to support technology use, mentor younger pupils, and lead e-safety initiatives.
- Involve pupils in evaluating and improving computing provision through regular surveys and showcases.

5. Ensure Equity of Access for All Learners

- Further support the **lowest 20%, SEND, and EAL** pupils through tailored scaffolds and assistive tech.
- Build pupil confidence in typing, digital fluency, and navigation skills.

6. Enhance Staff CPD and Digital Confidence

- Develop internal digital champions to mentor staff in using coding tools, AI, and blended learning platforms.
- Introduce bite-size CPD options (e.g., "10-minute tech tips") for practical classroom ideas.

7. Leverage New Technologies

- Explore the use of AI tools in KS2 (e.g., machine learning basics, ethics in AI).
- Pilot physical computing (e.g., using micro:bits, Crumble or LEGO robotics kits).
- Consider introducing **cloud collaboration** (e.g., Google Workspace) more widely for independent and group work.

8. Enhance Home–School Digital Partnerships

- Strengthen parental engagement in digital safety through workshops, newsletters, and student-led demos.
- Promote safe home use of technology and bridge the digital divide through access to devices or online platforms.

Long-Term Aspirations (3–5 Years)

- Be recognised as a **centre of excellence** for primary computing within the Lux Mundi Academy Trust.
- Ensure 100% of pupils leave St Monica's digitally literate, confident coders, and responsible online citizens.
- Achieve national awards such as the **NAACE ICT Mark** or **Digital Schools Award**.