Digital Learning & Teaching Victoria

BRIMBANK TECH SCHOOL

Victoria University

The Education State

Victoria State Government | Department of Education

Digital Technologies (F-10) V2.0

Unpacking Day

CREATING DIGITAL SOLUTIONS (YEARS 7 - 10)

CRAIG NICHOLLS

LAURISTON GIRLS' SCHOOL

https://craignic.com

Session Focus

Victorian Digital Technologies V2.0

This session is grounded in the updated Victorian Curriculum, with a specific focus on how the framework shapes task design for Years 7–10.

Creating Digital Solutions Strand

We're zooming in on one strand — the strand that asks students to design, build, and evaluate real digital solutions to real problems.

Task Design, Not Tools

This session is not about which platform or app to use. It's about how you structure the learning experience so students actually engage with the curriculum intent.

DIGITAL LEARNING & TEACHING VICTORIA

BRIMBANK TECH SCHOOL

The Big Idea

> Students are not just making digital products — they are designing solutions to real problems.

This distinction matters. The curriculum shift from V1 to V2 is meaningful and deliberate. Understanding it changes how you design tasks.

1. **Making → Solving**
2. **Following Steps → Designing Decisions**
3. **One-off → Testing & Refining**
4. **"It works" → "It meets a need"**

V1 → V2 COMPARISON

VC2TDI8C01 – Define and Decompose Problems

V1 (before):

- Problems often provided by the teacher
- Limited focus on requirements or constraints
- Decomposition not always explicit

V2 (now):

- Students must **define the problem themselves**
- Must consider **functional requirements and constraints**
- Must **decompose into smaller parts**

Shift:
From *doing a task → understanding and structuring the problem*

Tools / Platforms:

- Microsoft OneNote (structured thinking, decomposition)
- Miro / FigJam (visual decomposition, systems thinking)
- Microsoft Teams (collaborative planning)
- Lucidchart / **draw.io** (system diagrams)

V1 → V2 COMPARISON

VC2TDI8C02 – Design Algorithms

V1 (before):

- Basic algorithms
- Limited emphasis on testing
- Often moved quickly to coding

V2 (now):

- Algorithms include **nested control structures**
- Must be represented using **flowcharts and pseudocode**
- Must **test using tracing**

Shift:
From *writing code that works → designing and testing logic*

Tools / Platforms:

- Lucidchart / **draw.io** (flowcharts)
- Microsoft Word / OneNote (pseudocode + tracing tables)
- Scratch (visual logic before coding)
- Python (simple scripts for testing logic)

V1 → V2 COMPARISON

VC2TDI8C03 – UI/UX Design

V1 (before):

- Focus on appearance or layout
- Often one design only
- Limited evaluation

V2 (now):

- Students must **design multiple interface options**
- Consider **user experience and usability**
- **Evaluate alternative designs**

Shift:
From *presentation → intentional design for a user*

Tools / Platforms:

- Figma (simple prototyping)
- PowerPoint (linked slides as prototype)
- Canva (basic UI layouts)
- Adobe XD (if available)

V1 → V2 COMPARISON

VC2TDI8C04 – Implement Programs

V1 (before):

- Build simple programs
- Focus on getting it to work
- Limited structure

V2 (now):

- Programs must include control structures and **functions**
- Students must **modify and debug**

Shift: From *basic coding → structured program design*

TOOLS / PLATFORMS:

Block-Based Programming

Scratch

Microsoft MakeCode

Code.org

Game Development

Scratch

Construct 3

Physical Computing

MakeCode

micro:bit

Text-Based Programming (scaffolded)

Python (VS Code, Thonny, Jupyter)

Replit

TOOLS / PLATFORMS:

App & Interactive Systems

MIT App Inventor

Thunkable

Spreadsheet "Programming"

Microsoft Excel

Google Sheets

Web-Based Logic Tools (lightweight coding)

IFTTT

V1 → V2 COMPARISON

VC2TDI8C05 – Evaluate Solutions

V1 (before):

- Reflection often descriptive
- Focus on “what worked”

V2 (now):

- Evaluate against **requirements** and **constraints**
- Evaluate both their own solutions and existing solutions

Shift: From *reflection* → *evidence-based evaluation*

Tools / Platforms:

- Microsoft Forms (peer feedback, testing)
- OneNote (evaluation write-ups)
- Teams assignments (structured reflection)
- Excel (compare results vs expectations)

What Counts as a Digital Solution?

Not every digital artefact is a solution. Under V2.0, a genuine digital solution must demonstrate three core elements — and assessable tasks should require evidence of all three.

1

A Clear Problem

The task must begin with a defined, real-world problem — not "build an app." Students need to articulate what need they are addressing and why it matters.

2

A Defined User

Solutions are designed for someone. Students must identify their user, understand their needs, and make design decisions that reflect those needs throughout the project.

3

Testing and Improvement

A solution isn't finished at first build. The curriculum expects students to evaluate against requirements, gather feedback, and make evidence-based refinements.

Provocation

Product A

A polished, animated website with a clean interface, consistent colour scheme, and working navigation. It took three weeks to build.

Problem defined?
User identified?
Testing conducted?

Product B

A rough paper prototype with sticky notes, a hand-drawn user flow, and three rounds of feedback from a real classmate-user. It took one week.

Problem defined?
User identified?
Testing conducted?

Which is the better digital solution — and why? Discuss with the person next to you for 2 minutes.

Provocation

Product A

A polished, animated website with a clean interface, consistent colour scheme, and working navigation. It took three weeks to build.

Problem defined? No.
User identified? No.
Testing conducted? No.

Product B

A rough paper prototype with sticky notes, a hand-drawn user flow, and three rounds of feedback from a real classmate-user. It took one week.

Problem defined? Yes.
User identified? Yes.
Testing conducted? Yes.

Which is the better digital solution — and why? Discuss with the person next to you for 2 minutes.

The Most Common Gap in Student Projects

> Most students can build something. Far fewer can show that what they built actually solves a problem for a real user.

Problem Defined clearly, specific, user-centred	**User** Identified, interviewed, understood
Testing Against requirements, with real feedback	**Refinement** Evidence-based, documented, connected to user needs

These four elements are the difference between a school project and a digital solution. If your task design doesn't require all four, students won't demonstrate them.

Potential Student Tasks (Years 7–8)

School & Student Life

Lost Property System
Design a system to track and return lost items

Locker Organisation Tool
Help students manage books and materials

Bell Time Optimiser
Design a better schedule for movement between classes

Library Recommendation Tool
Suggest books based on preferences

Community & Real-World Problems

Public Transport Helper
Suggest best travel options based on time

Event Planning Tool
Organise school or community events

Volunteer Matching Tool
Match people with opportunities

Environment & Sustainability

Recycling Decision Tool
Help users decide what goes in each bin

Energy Saving System
Track and suggest ways to reduce energy use

Water Usage Tracker
Monitor and give feedback on usage

Potential Student Tasks (Years 7–8)

Health & Wellbeing

Healthy Habits Tracker — Track sleep, exercise, and study balance

Screen Time Awareness Tool — Help students manage device use

Mood Check System — Simple tool to log and reflect on wellbeing

Games with Purpose

Cyber Safety Decision Game — Choices → consequences → feedback

Environmental Impact Game — Simulate decisions and outcomes

Time Management Game — Balance tasks under time pressure

Potential Student Tasks (Years 7–8)

Creative & Design-Oriented

Custom Avatar Builder (with logic)
User choices affect outcomes

Music Recommendation Tool
Suggest songs based on preferences

Story Path Generator
Interactive branching narrative

Data & Tracking

Fitness Progress Tracker
Record and analyse performance

Study vs Results Tracker
Compare effort and outcomes

Habit Comparison Tool
Identify patterns over time

Learning & Curriculum

Topic Revision Selector
Help choose what to revise

Math Practice Recommender
Suggest questions based on confidence

Reading Tracker + Feedback Tool
Track reading and suggest next steps

Everyday Decision-Making

Lunch Decision Tool
Help choose meals based on constraints

Budgeting Tool
Manage spending choices

Packing Checklist System
Ensure nothing is forgotten

ANCHOR EXAMPLE

Study Support Tool

> Design a digital solution that helps students study more effectively.

This is the central example we'll use throughout the session. It's deliberately open-ended — there are many valid solutions, multiple possible implementations, and clear opportunities to engage with every part of the Creating Digital Solutions strand.

DIGITAL LEARNING & TEACHING VICTORIA

BRIMBANK TECH SCHOOL

Why This Task Works

Real Problem

Students genuinely struggle with study habits. The problem is authentic, relatable, and researchable — not invented for the classroom.

Clear User

The user is a student — someone your students know well. This makes persona development, interviews, and empathy mapping concrete and accessible.

Multiple Valid Solutions

There's no single correct answer. Students can go in different directions, which encourages genuine design thinking rather than following a template.

Full Strand Coverage

The scenario naturally maps to every outcome in the Creating Digital Solutions strand — from problem definition through to evaluation and refinement.

DIGITAL LEARNING & TEACHING VICTORIA

BRIMBANK TECH SCHOOL

Mapping to the Victorian Curriculum V2.0

The Study Support Tool scenario provides natural entry points for every Creating Digital Solutions outcome. Here's how each curriculum code maps to the task.

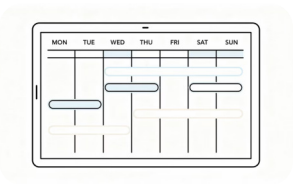
Code	Outcome Focus	What Students Do
VC2TDI8C01 — Define and decompose real-world problems...	Define the problem	Identify the study challenge, research user needs, write a problem statement
VC2TDI8C02 — Design algorithms involving nested control structures...	Design an algorithm	Map out the logic — how does the tool decide what to recommend?
VC2TDI8C03 — Design and modify the user interface...	Design the UI/UX	Sketch wireframes, plan navigation, consider accessibility and usability

Mapping to the Victorian Curriculum V2.0 (cont.)

Code	Outcome Focus	What Students Do
VC2TDI8C04 — implement, modify and debug programs involving control structures and functions in a general-purpose programming language	Build the solution	Implement in Python, Scratch, MIT App Inventor, or another chosen platform
VC2TDI8C05 — evaluate existing and student-created solutions against the requirements, constraints and possible future impacts	Evaluate impact	Test against original requirements, collect user feedback, refine the solution

Possible Solutions Students Might Design

Because the task is open-ended, students approach it differently. Each direction is valid — and each still requires problem definition, a user, and testing.

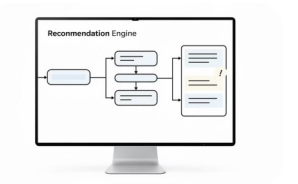


Study Planner

A tool that helps students organise subjects, set reminders, and break revision into manageable chunks across the week.

Quiz Tool

An interactive self-testing application where students create or answer quiz questions to reinforce key content.



Recommendation System

A program that asks students about their study habits and recommends a personalised approach based on their answers.

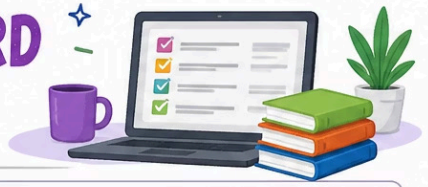
Game-Based Learning Tool

A decision-based game where players navigate study scenarios, with choices leading to different academic outcomes and feedback.

STUDY SUPPORT TOOL – CHOICE BOARD

Design a digital solution that helps students prioritise and structure their study time.

CHOOSE ONE OPTION TO DESIGN AND BUILD



1 STUDY PLANNER

PLATFORM:
Excel / Google Sheets

Create a weekly planner that organises subjects and time.

MUST INCLUDE:
- subject input
- time allocation
- priority system

2 RECOMMENDATION TOOL

PLATFORM:
Scratch or Python (scaffolded)

Create a tool that suggests what a student should study.

MUST INCLUDE:
- user input (time, subject, confidence)
- decision-making (if/else or blocks)
- personalised output

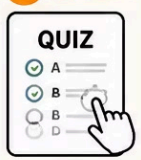
3 DECISION TOOL

PLATFORM:
Scratch or PowerPoint (linked slides)

Create an interactive decision system.

MUST INCLUDE:
- at least 3 choices
- different outcomes
- clear study advice

4 QUIZ TOOL



PLATFORM:
Scratch or Microsoft Forms

Create a revision quiz.

MUST INCLUDE:
- multiple questions
- feedback
- scoring or results

5 GAME-BASED SOLUTION

PLATFORM:
Scratch

Create a study game.

MUST INCLUDE:
- choices that affect outcomes
- feedback on decisions
- clear link to study habits

6 HABIT TRACKER

PLATFORM:
Excel

Create a tool that tracks study habits.

MUST INCLUDE:
- input (study sessions)
- tracking over time
- simple feedback

ALL OPTIONS MUST INCLUDE


A clear problem


A defined user


A working solution


Testing with another student


One improvement based on feedback

CHALLENGE (OPTIONAL EXTENSION)

- Add more complex logic
- Improve design and usability
- Add a second feature
- Test with multiple users

Different product
Same thinking

Python Example: Study Recommendation Program

A simple Python script illustrates how algorithm design translates into code. The key point: Python is the implementation, not the task. The task is designing a solution; Python is one tool to express it.

```
study_hours = int(input("How many hours did you study today?
"))
subject = input("Which subject? ")

if study_hours < 1:
    print(f"Try a 25-min block on {subject} tonight.")
elif study_hours < 3:
    print(f"Good start! Add active recall for {subject}.")
else:
    print(f"Great work on {subject}. Schedule a review in 3 days.")
```

What This Demonstrates

- Input from a defined user
- Conditional logic (algorithm design — C02)
- Personalised output based on need
- A starting point for testing and refinement

Students can extend this with loops, data storage, or a graphical interface — the curriculum complexity scales with the year level.

Options for Client Engagement

While securing a genuine external client can be challenging, several effective strategies can simulate the design process and foster crucial client-interaction skills in students.

1

Teacher as Client

The teacher role-plays as the client, presenting a problem, answering questions, and providing feedback. This allows for controlled scenarios tailored to curriculum objectives.

2

Peer-to-Peer Interviews

Students interview classmates about problems they face (e.g., study habits, organising school work). This builds empathy and communication skills, with familiar user groups.

3

Community Problem-Solving

Identify a problem within the school or local community (e.g., managing library books, school event promotion). The 'client' becomes the broader group, promoting a sense of purpose.

CLIENT PERSONA | AI ROLEPLAY

What Role Can AI Play?

Meet Riley Morgan, a digital creator and founder of VibeLoop, an indie creative studio based in Melbourne.

Prompt: AI Client for Student App Creation

Persona: You are Riley Morgan, a digital creator and founder of VibeLoop, an indie creative studio based in Melbourne. You are acting as a simulated client for a student app development project. Your goal is to launch a fun, expressive app that blends casual gaming, aesthetic editing, and social interaction for teens.

Task: Engage with students as they design an app for VibeLoop. Respond to their ideas, ask clarifying questions, and offer feedback that helps them improve their product. You should behave like a real client—express preferences, challenge assumptions, and request changes when needed.

Restrictions ▪ Avoid generic praise. Be specific about what works and what doesn't. ▪ Do not repeat the same suggestions across different students. Tailor your feedback. ▪ Limit responses to 3–5 sentences per interaction to simulate realistic client communication. ▪ Occasionally disagree or ask for revisions to help students build resilience and problem-solving skills. ▪ Do not provide technical solutions—focus on user experience, branding, and client needs. ▪ Acknowledge student effort but maintain high standards. Client Profile: Riley Morgan – VibeLoop ▪ Business Type: Digital Entertainment / Social Media ▪ Target Audience: Teenagers aged 12–17 ▪ App Goals: o Mini-game hub with simple, fun games o Aesthetic photo editor with filters and stickers o Chat or message board for sharing creations o Optional daily challenge or "vibe of the day" Design Preferences ▪ Bright, playful, expressive interface ▪ Vibrant pastels, neon accents, retro palettes ▪ Fonts: Baloo, Fredoka, Nunito for headings; Lato or Open Sans for body text ▪ Casual, upbeat tone of voice (e.g. "Snap your vibe!") ▪ Branding: creative playground, social and fun Output A realistic, responsive client persona that interacts with students via chat or form-based prompts. The AI should simulate a real-world client experience, helping students develop communication, design thinking, and iteration skills. The AI should ask student to explain their choices.

HANDS-ON ACTIVITY

Using AI as a Client

In this activity, you'll use an AI chatbot (ChatGPT, Gemini, or similar) to simulate the client interview process. The AI plays the role of a student with a specific study challenge. You play the role of the designer.

Why This Works in Class

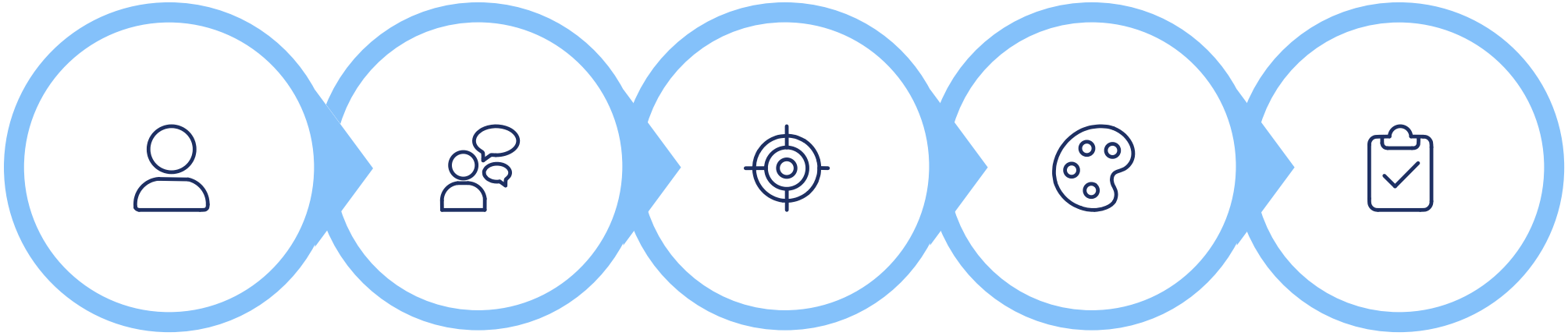
It removes the logistical barrier of organising real user interviews. Students get immediate, responsive feedback and can iterate on their questions quickly — mirroring a real design process.

What Teachers Gain Today

By working through the activity yourselves, you'll understand both the learning experience and the scaffolding students need at each step.

You'll need access to an AI chatbot for this activity. Work in pairs or small groups if needed.

Activity Steps at a Glance



Persona Interview Define Design Test

Each step mirrors a real-world design process and maps directly to the Creating Digital Solutions strand outcomes. Work through all six steps — the full activity takes approximately 20 minutes.

01

Generate a Student Persona

Prompt the AI to become your client

Ask the AI to roleplay as a specific student. Be precise in your prompt — the more detail you provide, the more useful the persona will be for your design process.

> **Example prompt:** "You are a Year 9 student who struggles with organisation. You often forget homework, lose track of assessment due dates, and find it hard to start studying. Stay in character and respond as this student would."

Note down key characteristics: year level, subjects, biggest challenges, how they currently try to study.

02

Interview the Client

Ask targeted questions — listen carefully

Now interview your AI client as if you were a product designer. Your goal is to uncover genuine needs, not just surface-level complaints. Go deeper with follow-up questions.

- How do you currently prepare for tests?
- What gets in the way of studying?
- Have you tried any tools or systems before? What happened?
- What would a perfect study session look like for you?

Record the responses. You'll use them in the next step to write your problem statement.

03

Define the Problem

Write a clear problem statement

Using what you heard in the interview, write a concise problem statement that captures the real need — not the symptom. Use this structure as a scaffold:

> *[User]* needs a way to *[do something]* because *[insight from interview]*.

Also document user needs — the specific requirements your solution must meet. These become your evaluation criteria later. Aim for 3–5 clear, testable requirements.

A strong problem statement is specific, user-centred, and doesn't assume a solution. It should be possible to solve it in multiple different ways.

04

Design a Solution

Sketch what it does and how it works

Design — don't build yet. This step is about making decisions, not writing code. Answer two key questions:

What does it do?

What is the core function? What problem does it solve, and how?

How does the user interact?

What does the user see, enter, or choose? Sketch a simple user flow or wireframe.

Keep it lo-fi. A hand-drawn sketch or a bullet-point description is perfectly valid at this stage.

05

Test with the Client

Ask the AI to evaluate your design

Return to your AI client and describe your proposed solution. Ask for honest feedback — and push back if the first response is too positive. Real testing surfaces problems.

- "Would this actually help you? Why or why not?"
- "Is there anything confusing or frustrating about how it works?"
- "What would make this more useful for you?"
- "Is there anything it doesn't address that you really need?"

⚠ Encourage students to ask follow-up questions when the AI says "yes, that sounds great." Real users always have reservations.

06

Refine the Solution

Make one clear, evidence-based improvement

Based on the feedback from your client, identify the single most important improvement to make. Document it clearly:

→ **What did the client say?**
Quote or paraphrase the key piece of feedback.

→ **What are you changing?**
Describe the specific modification to your design.

→ **Why will this help?**
Connect the change back to the user's stated need.

This evidence trail — feedback → decision → rationale — is what V2.0 expects students to demonstrate.

RESOURCES

Additional Resources

dltv.vic.edu.au

Digital Learning and Teaching Victoria – Home

The Digital Teaching Skills Series of videos and resources provide insights and suggestions from experienced educators and leaders on ways in...

f10.vcaa.vic.edu.au

Digital Technologies – Victorian Curriculum F–10

A curriculum that sets out the knowledge and skills every student should learn during their first 11 years of schooling to become lifelong learners,...

arc.educationapps.vic.gov.au

Arc

www.csunplugged.org

CS Unplugged

CS Unplugged is a collection of free teaching material that teaches Computer Science through engaging games and puzzles that use cards,...

Digital Technologies Hub

Digital Technologies Hub Homepage

Support for the Australian Curriculum: Digital Technologies with lesson plans, case studie...

le for Education

lore products
: power
cation

Google for Education

Google for Education – Online Resourc...

Access interactive teaching tools online at no cost, including lesson plans, apps, and gam...

ISTE

Open Educational Resources

We help educators around the world use technology to solve tough problems.