Smartphone security

*Please refer to the online lesson plan on the DT Hub to access all website links and additional resources.*

### Year levels: 5–6

This lesson provides an opportunity to investigate security measures, including those powered by artificial intelligence (AI), that are used to protect users from unauthorised (unapproved, unwanted) access to their digital devices.

Students can create computer programs to demonstrate a security measure such as using a personal identification number (PIN); for example, a four-digit code. Alternatively, students can incorporate a type of image recognition to mimic AI. This lesson was developed in collaboration with the Digital Technologies Institute.

Unauthorised use of our digital devices is an everyday threat and is a key aspect of cyber security. We need to ensure measures are in place to protect our personal information and privacy. There are a range of security measures used, depending on the device. On a smartphone, AI has opened the way to use biometrics (for example, thumbprint or face recognition) to protect the authorised user of a phone and deny access to unauthorised users.

### Preparation

Tools and resources required:

* video AI in everyday life
* Scratch 3.0 or a similar visual programming environment to create a digital solution
* Google Teachable Machine: Familiarise yourself with the [Teachable Machine](https://teachablemachine.withgoogle.com/)  application. View the supporting videos. Note: The teachable machine requires an internet connection, Google Chrome on Windows or Macintosh and a webcam. A version for iPad can be downloaded at TeachableMachine for mobile.

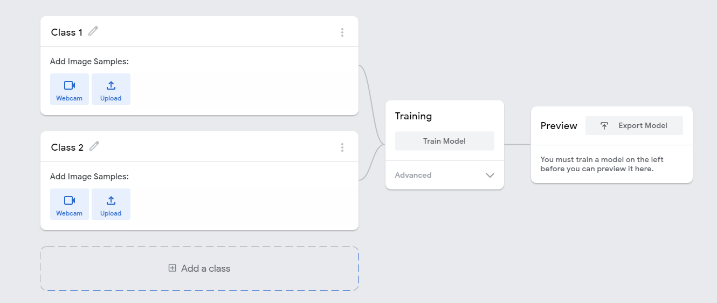


Image 1: Teachable Machine application screenshot

Image 1 shows the view of a project created in the Teachable Machine AI application. On the left, the classes are shown. Once the model is created and trained, a preview will be shown.

### Safety

Privacy and personal information

Discuss the potential misuse of personal images when uploading images of ourselves or friends on websites. Instruct students not to record images of themselves or others via the webcam or uploading images.

The conditions of use for Teachable Machine state that images are not stored on external servers if the teachable machine program is closed when completed and the project is not saved. If students close the tab, nothing is saved in their browser or on any servers. View the conditions when saving projects to the cloud on what data is stored and ‘who’ can view the model.

Risk assessment

Use this risk-assessment tool to assess risks and benefits before introducing any new online platforms or technologies.

# Learning hook

Use the video AI in everyday life to introduce the many ways AI is used by people every day. Ask students to predict and then look for ways in the video that AI is used to protect a smartphone user from unauthorised access.

Screen capture of video screen AI in everday life 


Discuss the many things that people use their smartphones for in daily life. Then ask: ‘How do users make sure their smartphone is secure?’ List security measures used to protect a user from someone else’s unwanted (unauthorised) use of their smartphone.

Examples include:

* personal identification number (PIN)
* thumbprint or fingerprint scan and recognition
* eye (iris) scanning and recognition.

Discuss which of these would use AI to perform the security measure. Explain that biometric security often uses image recognition to perform this function.

# Learning map and outcomes

In this lesson, students will:

* describe ways to protect use of digital devices using passphrases or biometric security
* create a program using a visual programming language that protects the user against unauthorised access
* describe the process used to train a simple AI using machine learning, and how it recognises images
* create, train and test their machine learning model using an online AI tool
* evaluate their model and describe how well it works.

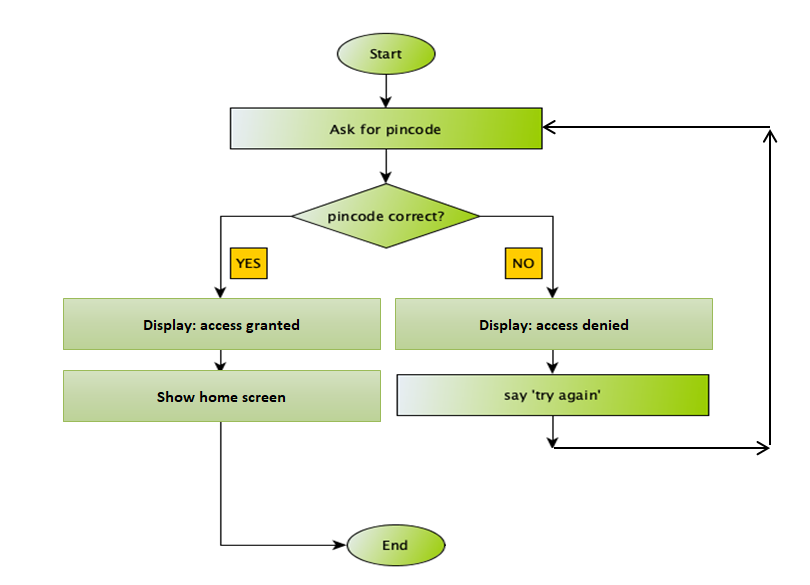
# Learning input

Ask students to list or draw the steps a user would follow to unlock their smartphone. (Bear in mind that many primary-age students may not have their own smartphone.) Consider a flowchart or similar as a suitable representation of the process. To encourage thinking, pose the following questions:

* How do users access their smartphone?
* What prompts might people see on-screen to unlock their device?
* How does the user gain authorised access?
* What happens when access is granted?
* What happens when access is denied?

Share students’ algorithms and model for the class how to represent the steps as a flowchart, discussing your thinking and inviting student input as you model.

An example flowchart for a pincode security measure might look something like this.



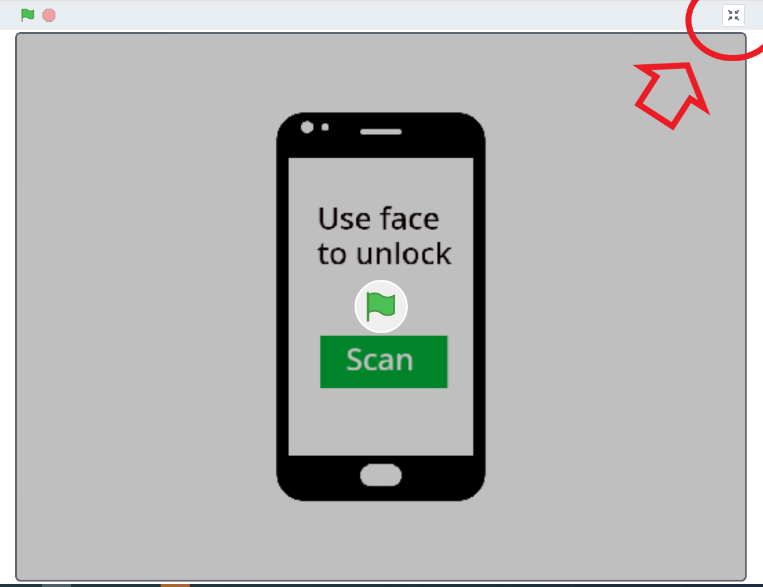
# Learning input and construction

Use the PRIMM approach to teaching programming. This is where students follow the process: **Predict** what the program does, **Run** the program to see if you were correct, **Investigate** the program blocks and how they work, **Modify** parts of the program, **Make** your own program.

**Differentiation**

Give three examples of providing security measures to grant access to a smartphone. The examples cater for a range of programming abilities.

* This basic program uses the logic of first asking for a PIN as an ‘answer’. An ‘ask’ block is selected for this purpose. It can be initiated when the green flag is selected. PIN for the sample program is 1234. Basic Phone lock/unlock on Scratch (mit.edu)
* This basic program starts with a home screen with animated home button. PIN for the sample program is 1234. Phone lock/unlock on Scratch (mit.edu)
* This program has more complex coding and includes randomisation. The Start screen asks the user to scan their facial image. It uses a sprite labelled ‘Scan’. The program randomly chooses 1 or 2 to display the image to be scanned. Basic Phone lock/unlock IMAGE on Scratch (mit.edu)



To view the project page select **Full screen control** in the top right corner. From the project page students can select 'See inside’ to tinker with the programming blocks.

Share students’ remixed versions and ask what blocks they used and what functionality they wanted to include and how successful they were in making it work as expected.

# Learning input and construction

As an alternative to programming or as an extra lesson, provide the opportunity for students to train an AI to recognise one avatar as the owner and to grant them access to the device, and any other avatar as unauthorised and classified as denied access.

Outline the steps to train the AI model using Google Teachable Machine following these steps:

* Select **Get Started**, then select **Image Project** as students are teaching their AI based on images.
* For each Class add a label. For example, ‘Access granted’ for the one avatar that unlocks access to the device. The other class can be ‘Access denied’. It is also useful to record a background for when no image is presented. These are the buckets into which students will be adding the training images.
* Give students access to images from This gallery of test faces. They can select from the two folders which avatars they will include in training their AI model.
* Select **Webcam** to record printed images of avatars held in front of the device’s camera. Alternatively, students can upload the image from the folder, which saves printing images. Record the images for each of their classes.
* When all images have been used for training, select **Train Model**.
* When training is complete select **Preview**. Students then test the model using the training images cut into individual images. If the model does not work as expected, students can retrain the model by adding their training images.
* **Export Model** enables students to have a copy of their model to refer to.

Provide students with access to Teachable Machine. They will need a device with a webcam. Remind students of safety and to avoid recording their image via the webcam. Working as a pair or small group is advisable to take turns and collaborate.

# Learning construction

Once the students have tested their AI, they swap with another group to test each other’s AI.

With a partner, swap images and test the AI model with images representing a range of users including the owner, who is the only one who is granted access. Observe the confidence bar. It tells us how well the AI believes it recognises the image. Is it correct? Is it certain? Are there cases when the AI is wrong? Students use a three-star or five-star rating and justify the rating they have given.

# Learning reflection

**Programming a smartphone security program**

* What changes were made? How successful were the changes? Did they work out as expected? Why or why not?

**Training an AI**

* Observe when the AI gets a result wrong. Discuss with the students what might be the underlying cause. Were the avatars too similar for the AI to recognise the owner from another avatar?
* Also discuss how we used the AI tool safely. What protocols did we set up to make sure an image of our face is not recorded by the webcam?

# Curriculum links

Digital Technologies (AC V9.0)

* Design algorithms involving multiple alternatives (branching) and iteration AC9TDI6P02
* Implement algorithms as visual programs involving control structures, variables and input AC9TDI6P05
* Access multiple personal accounts using unique passphrases and explain the risks of password re-use AC9TDI6P09

# Assessment

### Digital Technologies: Years 5–6

Achievement standard

By the end of Year 6 students design algorithms involving complex branching and iteration and implement them as visual programs including variables.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 1 (limited) | 2 (basic) | 3 (proficient) | 4 (advanced) |
| **Knowledge of visual programming blocks** | demonstrates a limited understanding of visual programming blocks | names some visual programming blocks or describes their purpose | describes the purpose of the visual programming blocks and gives relevant examples to support their understanding | explains how programming blocks are organised by type, their purpose, and gives relevant examples to support their understanding; their examples illustrate blocks to enable input, decisions, repeat steps and data stored as a variable |
| **Identifying control structures: investigating sample visual programs** | with guidance demonstrates a limited ability to describe or identify blocks in given examples and interpret the sequence of steps | identifies most block types in given examples; and interprets the sequence of steps and identifies user input, decisions (branching) or repeat steps (iteration) | consistently identifies block types correctly in various examples; and interprets the sequence of steps, predicts outcomes of blocks and identifies user input, decisions (branching) and repeat steps (iteration) | demonstrates a high level of accuracy in identifying block types of a range of complex visual programs; interprets the sequence of steps, predicts outcomes of blocks and identifies user input, branching and iteration; and explains ways a variable serves as a 'container' to hold and manipulate different data types within a program |
| **Producing and implementing visual programs** | with guidance creates a program in a sequence of steps to perform a basic task with limited opportunity for user input | creates a program with user input in a sequence of logical steps that includes a decision (branching) to provide the user with a choice | creates a program with user input in a sequence of logical steps that includes multiple decisions (complex branching) and iteration (repeat steps) | creates a program with multiple user input in a sequence of logical steps that includes multiple decisions (complex branching) and iteration (repeat steps); includes the use of variables to hold and manipulate different data types within a program |