

ACCA5033 - Assignment 2: Practical & Professional Development

2421062

Hasan Sivas

Introduction	2
1. Practical Development / Implementation	2
1.1 Initial Feedback	2
1.2 Implementation	4
2. Enterprise and Innovation	6
2.1 Potential Tech Start-up Plan	6
2.2 PEST Analysis	7
3. Project Feedback and Reflection	8
3.1. Practical Feedback	8
3.2 Project Progress Diary	10
3.3 Skills	11
3.4 Positive Social Impact	12
Conclusion	12
References	12
Appendix	13
URL:	13
Weekly Progress Sheets:	13
HTML + Javascript:	16
CSS:	33
Continued Professional Development:	40

Introduction

This report will detail and chronicle the development and implementation of the 'Draw Site' project. During the planning phases, Draw Site (Originally Drawsite) was pitched as a web application/game where the user would be prompted draw a random subject within a given amount of time, amongst other constraints such as a limited palette. Gamemodes such as Daily challenges, art training modes, and possibly a multiplayer mode were also considered as part of the design aspect. For the most part, these features were successfully implemented. In this report the programming and other aspects of development including feedback collection, practical progress, enterprise or innovation analysis, and reflection on skills developed necessary to implement the features, will be explained.

1. Practical Development / Implementation

1.1 Initial Feedback

Early on in development, when only a very basic prototype existed, it was decided that gaining feedback would be a suitable way to identify flaws with the project or seek out potential avenues to improve the design. The decided method of feedback capture was done using a form.

The feedback form consisted of 6 questions:

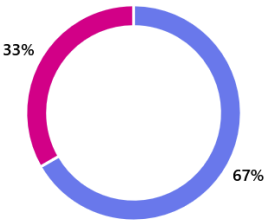
Q1. Please give the proposed project a rating based on how fun or useful it seems. (Rate /5)
Q2. What sort of person do you believe might enjoy using this application?
Q3. Does the proposed website layout seem suitable?(Yes/No/Unsure)
Q4. Are there any aspects of the application or its implementation that you believe may hinder its experience or usefulness
Q5. Do you have any suggestions for features that could improve the application

Q6. Would the inclusion of advertisements (e.g. banners at the side of the page) affect your ability to enjoy the application? (Yes/No/Unsure)

3. Does the proposed website layout seem suitable?

[More details](#)

- Yes 2
- No 1
- Unsure 0



4. Are there any aspects of the application or its implementation that you believe may hinder its experience or usefulness

2 Responses

ID ↑	Name	Responses
1	anonymous	Being particular, perhaps users on desktop could struggle to draw to their capabilities with a mouse, or even users on touch screen devices may struggle without a digital pen. Possibly something to think about
2	anonymous	The timer being too short/too long.

5. Do you have any suggestions for features that could improve the application

2 Responses

ID ↑	Name	Responses
1	anonymous	What I stated above ^
2	anonymous	Different difficulty levels.

6. Would the inclusion of advertisements (e.g. banners at the side of the page) affect your ability to enjoy the application?

2 Responses

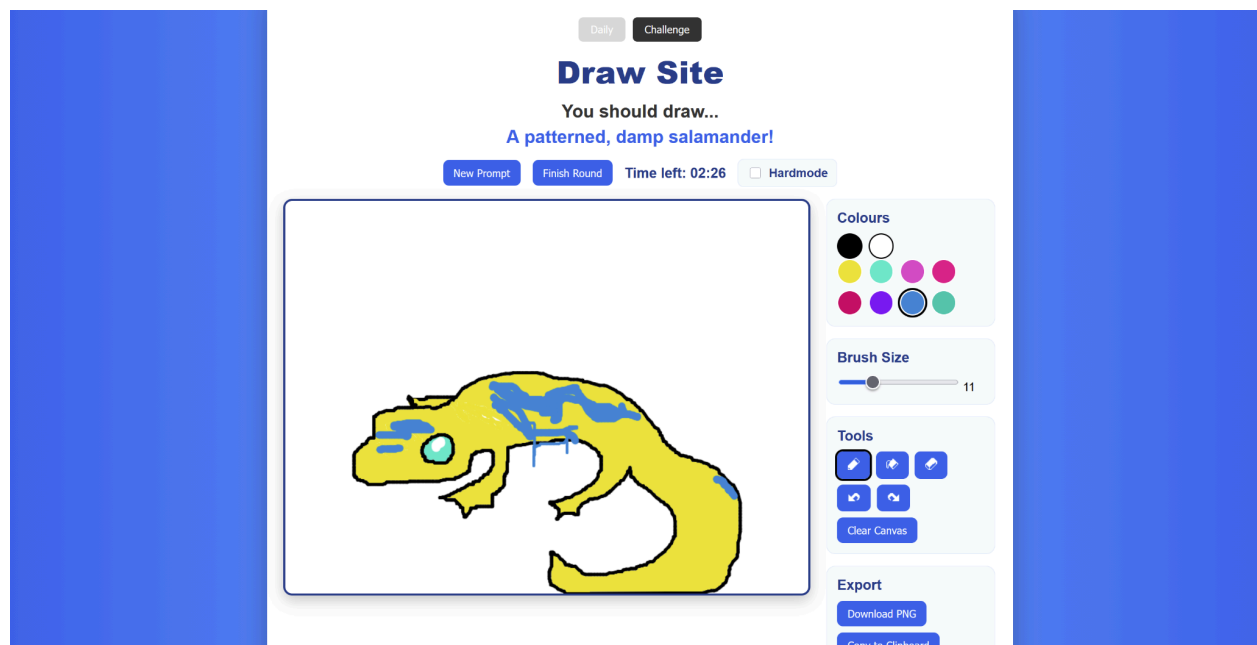
ID ↑	Name	Responses
1	anonymous	No
2	anonymous	Yes

Screenshots showing results for Questions 3,4,5 and 6

Although only a small number of people were given this initial survey, 3 people, it was able to produce some useful information. Questions 3, 4, 5 and 6 yielded the most useful information and would have a significant influence on the development of the project such as the suggestion for difficulty modes and the timer being too short possibly detracting from the experience.

In addition to the initial survey, feedback was obtained throughout the development process by showing the site to others in order to gain a better understanding of the suitability of features as they were added.

1.2 Implementation



Screenshot of [DrawSite's](#), final design.

For the practical implementation, Draw Site was developed using HTML5, JavaScript, CSS and supporting image/JSON assets.

The main interface was structured around a central drawing canvas, with a control panel for colours, brush size, tools and export options. This layout was chosen to keep the drawing area clear while still giving users quick access to the main functions needed during a timed drawing round and was left mostly unchanged from the original design based on the feedback that it seemed suitable (albeit, the final product did have many more buttons than initially planned).

Later additions included features such as undo and redo buttons which were deemed necessary for error prevention and to give users as much freedom as possible [5], tooltips were added for the same reason.

```
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115

    <button onclick="undoCanvas()">
      
    </button>

    <button onclick="redoCanvas()">
      
    </button>

    <button onclick="clearCanvas()">Clear Canvas</button>
  </div>
</div>

<div class="control-group">
  <h3>Export</h3>
  <div class="button-row">
    <button onclick="downloadImage()">Download PNG</button>
    <button onclick="copyToClipboard()">Copy to Clipboard</button>
  </div>
</div>

</div>

</section>

<!-- TIMER MODAL -->
<div id="timerModal">
  <div id="timerModalContent">
    <h2 id="timerModalTitle">Time's up!</h2>
    <button onclick="downloadImage()">Download</button>
    <button onclick="copyToClipboard()">Copy</button>
    <button onclick="handleModalNewPrompt()">New Prompt</button>
    <button onclick="closeModal()">Continue Drawing</button>
  </div>
</div>

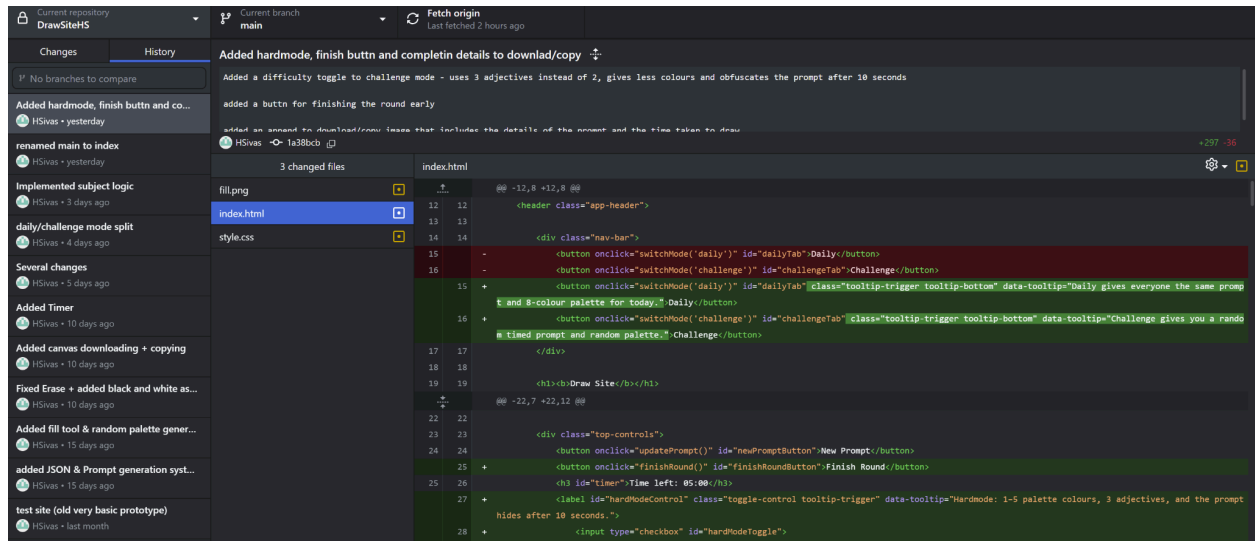
</main>

<script>
//to test
//python -m http.server
//http://localhost:8000
const canvas = document.getElementById("canvas");
const ctx = canvas.getContext("2d");

let timerInterval = null;
```

Notepad++ was used to write the HTML, JSONs and JavaScript thanks to its ease of use and

In particular, HTML5's canvas feature[1][2] was heavily utilised in order to make the whole image drawing system possible. The pencil tool draws continuous lines based on mouse movement, while the eraser uses the destination-out canvas operation to remove pixels. A fill tool was added using a flood-fill algorithm, which checks nearby pixels and replaces matching areas with the selected colour. This provided a minimal set of tools needed for users who wanted both freehand drawing and quick colouring options without overburdening the application.



Screenshot of GitHub Desktop

Throughout development, GitHub was used to manage DrawSite's code repository. The usage of version-control platforms are important in modern-day programming in order to ensure data is backed up and integrated seamlessly with the work done by others, they can also support CI/CD workflows, where code changes are automatically integrated, tested and deployed [9]. As at this stage the program was only being worked on by one person, frequent commits were not needed.

Testing was carried out throughout development, with each new feature tested alongside existing features that could be affected. This is known as incremental testing. By doing this, errors identified as they occurred rather than only at the end of the project, resulting in an overall better product [3].

For example, after the fill tool was added, it was quickly noticed that boundaries drawn using the default black colour were being ignored. This happened because the fill function was not clearly distinguishing between transparent canvas areas and fully opaque black pixels. I improved this by checking the alpha value of pixels and using colour-matching tolerance, which made the fill tool behave more reliably around drawn lines.

The conceptualised multiplayer was not implemented for this version of the project for several reasons: lack of an objective scoring system, the need for moderation and the lack of means to permanently host a backend server.

2. Enterprise and Innovation

2.1 Potential Tech Start-up Plan

Below is a potential plan for the sort of business that startup that might grow from Draw Site.

Startup name: Pen²/Pen²

Business type: Private Sector



Private Limited company (Ltd)

Business Strategy

Mission Statement: To produce software to grow and enhance human creativity.

Objectives:

- Make enough money to cover at a minimum the hosting of its websites
- Make enough to fund future, more ambitious projects
- Produce and host web-based applications
- Further develop desktop-based art software and potentially mobile ports

Pen²'s values:

- The needs and input of its users
- Creativity
- Consistency of services

Pen²'s business strategy would be to begin with a low-cost web-based product and gradually expand based on user feedback. Draw Site would act as the first product, offering users quick creative drawing challenges through a browser without requiring installation and building a small user base. This lean approach would reduce financial risk because the product can be developed and hosted with relatively low start-up costs.

2.2 PEST Analysis

Factor	Analysis
Political Factors	As a UK-based web application, Draw Site would need to consider digital regulation, data protection and online safety expectations. If the site is likely to be accessed by children, the ICO Children's Code is relevant because it applies to online services such as apps, games and websites that may process children's data. This means

	future features such as accounts, leaderboards or multiplayer would need privacy-by-design, limited data collection and clear user protections.
Economic Factors	Draw Site has no initial development costs because it is built using HTML, CSS, JavaScript and browser canvas technology, which allows it to run as a web application without specialist hardware. This would be suitable for a lean start-up model [6] where the only real cost would be hosting, domain registration, maintenance and future development.
Social Factors	Draw Site is aimed at users who want quick drawing challenges rather than complex professional art software, and follows the trend of short daily games like Wordle. Adding mobile support will be the most important way to reach a wider audience. Draw Site uses gamification through timed challenges, daily prompts and difficulty modes, which may improve user engagement[10] and make the product more addictive.
Technological Factors	The main technological advantage is that Draw Site works directly in the browser using the HTML5 canvas API, meaning users do not need to install software. Future development could include mobile support, cloud saving or multiplayer, although multiplayer would require backend hosting, moderation and stronger security controls. Web accessibility should also be considered, as WCAG 2.2[4] recommends making web content perceivable, operable, understandable and robust.

A PEST analysis table, showing the various factors that might affect Draw Site.

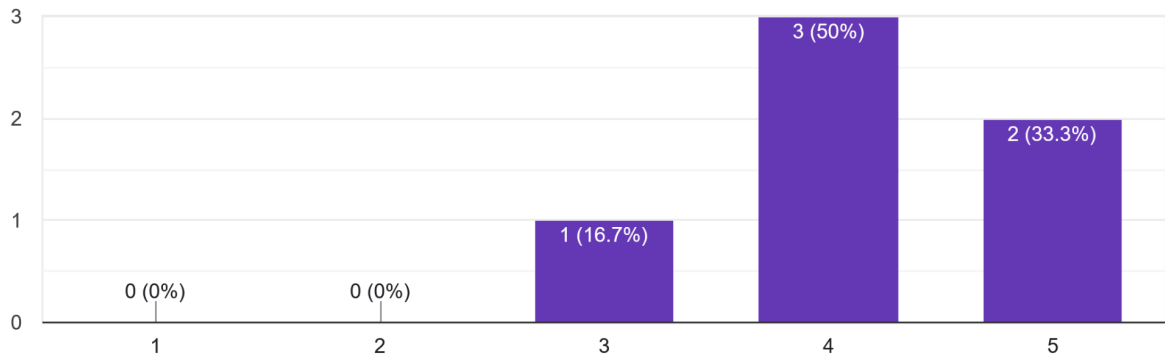
3. Project Feedback and Reflection

3.1. Practical Feedback

After the webpage was relatively complete, obtaining more feedback in order to refine it further was the natural next step. This time, it was conducted in person in order to receive more detailed feedback in addition to the survey. Participants were given an opportunity to try using the website and then asked to fill out a survey.

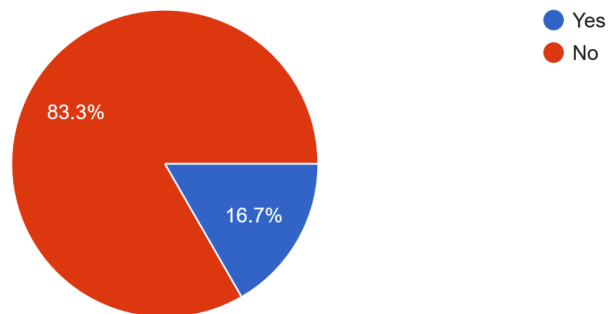
Out of 5, rate the Ease of Use of the website

6 responses



Did you encounter any obvious errors whilst using the application?

6 responses



Did you encounter anything that made the experience of using Drawsite worse?

3 responses

The random colors are sometimes too restrictive.

fill tool leaves small white gaps

random timer in challenges

Do you have any suggestions for features that can improve the website?

5 responses

drawing layers

Make sure the colors are more varied.

add sound effects

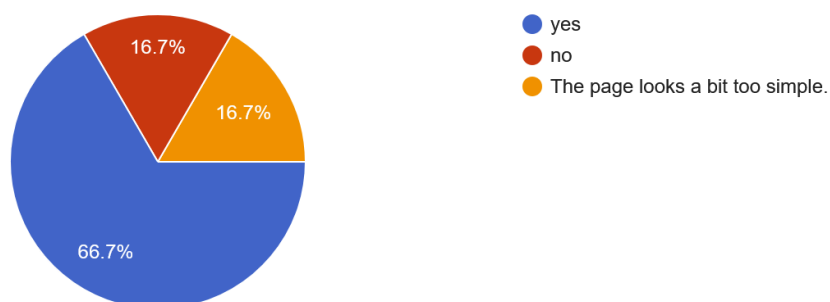
Let the colours vary in saturation too

add shortcuts and keybindings (ctrl + z to undo)

Is the appearance of the webpage pleasant to look at?

 [Copy chart](#)

6 responses



Charts and screenshots showing the results of the survey

The feedback has provided many potential avenues that could be used to improve the application and elucidated its flaws and strengths. Should Draw Site's development continue, these are things that need to be considered.

In particular, some users found the colours restrictive; the suggestion to add saturation levels would be a great way to improve the palette selection, allowing users to better tackle the prompts. Another suitable idea would be to implement drawing layers so that users have more leeway to colour.

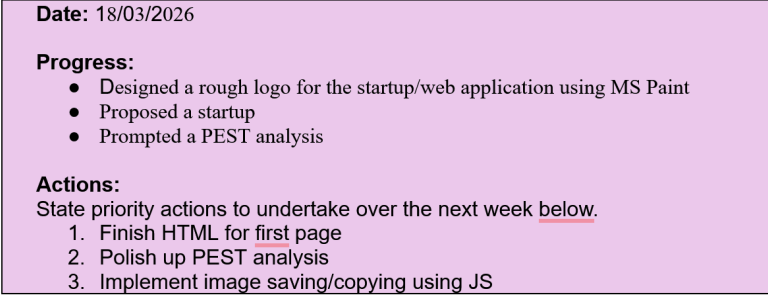
There were also many quality of life suggestions that could be implemented such as adding sound effects for when a round is over or keybinds to let users switch between tools more seamlessly.

3.2 Project Progress Diary

A weekly Project Progress Report Form was maintained to record completed work, remaining tasks and objectives for the following week [see Appendix]. This follows project management

guidance on scheduling work, monitoring progress, identifying risks and documenting plans [7]. Breaking the project into weekly tasks made development more manageable and supported staged planning of design, coding, testing and feedback.

Weekly Project progress form 2



Date: 18/03/2026

Progress:

- Designed a rough logo for the startup/web application using MS Paint
- Proposed a startup
- Prompted a PEST analysis

Actions:

State priority actions to undertake over the next week below.

1. Finish HTML for first page
2. Polish up PEST analysis
3. Implement image saving/copying using JS

Example image, entry 2 of the Weekly Project progress form

3.3 Skills

Throughout the planning, design, development and implementation of DrawSite, I have developed many new skills and improved old ones.

New Project Management Skills

During the initial conception of my project, I gained a better understanding of project designing and planning by outlining several potential avenues to pursue as projects, constructing SMART objectives based on my chosen project and making Gantt charts to assist me in maintaining a schedule during the actual development period.

I also learned how to refine a project through prototypes and user feedback. Early feedback helped me identify which features were most important to users, such as difficulty levels. This encouraged me to improve the project gradually rather than treating the first design as final.

New technical skills

A major technical skill I developed was using JavaScript to create interactive webpage features. Before this project, my understanding of JavaScript was limited, but I learned how to use it to control canvas drawing, timers, buttons, prompts, colour palettes and export functions.

I also improved my general web development skills by combining HTML, CSS, JavaScript and JSON into a functional browser-based application. This showed me how different front-end technologies work together to create a web application.

Additionally, I developed stronger bug testing and troubleshooting skills. Testing features as they were added helped me identify and fix problems early. Solving these issues improved my confidence in debugging and adapting code.

3.4 Positive Social Impact

There are a few ways in which Draw Site may have a Positive Social Impact (PSI). As a game, its main purpose would be to help people have fun and reduce stress. Draw Site also has a PSI by helping people practise their artistic skills by giving them short daily challenges which can improve people's happiness and mental health. H. Keyes et al. finds that engagement with creating arts and crafting predicted higher life satisfaction and happiness[8].

Conclusion

Overall, the Draw Site project was successful in producing a functional browser-based drawing application with random prompts, timed challenges, drawing tools and export options. Feedback collected thought the development helped shape improvements such as difficulty options, while practical implementation improved my skills in JavaScript, HTML, CSS, testing and project management. Although some features, such as multiplayer, were not implemented due to technical and moderation limitations, the final project demonstrates value through an accessible tool that encourages short, enjoyable drawing practice.

References

- [1] W3schools, "HTML5 Canvas," W3schools.com, 2019.
https://www.w3schools.com/html/html5_canvas.asp
- [2] Mozilla, "Canvas API," MDN Web Docs, Dec. 02, 2019.
https://developer.mozilla.org/en-US/docs/Web/API/Canvas_API
- [3] A. M. Kazerouni, C. A. Shaffer, S. H. Edwards, and F. Servant, "Assessing Incremental Testing Practices and Their Impact on Project Outcomes," Feb. 2019, doi:
<https://doi.org/10.1145/3287324.3287366>.
- [4] W3C, "Web content accessibility guidelines (WCAG) 2.2," www.w3.org, Dec. 12, 2024.
<https://www.w3.org/TR/WCAG22/>
- [5] J. Nielsen, "10 Heuristics for User Interface Design," Nielsen Norman Group, Apr. 24, 1994.
<https://www.nngroup.com/articles/ten-usability-heuristics/>
- [6] OpenAI's ChatGPT, private communication, April 17th 2025
- [7] Department for Business, Innovation and Skills, *Guidelines for Managing Projects: How to organise, plan and control projects*. London, UK: BIS, 2010. [Online]. Available: GOV.UK.
- [8] H. Keyes et al., "Creating arts and crafting positively predicts subjective wellbeing," *Frontiers in Public Health*, vol. 12, Aug. 2024, doi: <https://doi.org/10.3389/fpubh.2024.1417997>.
- [9] C. Cowell, N. Lotz, and C. Timberlake, *Automating DevOps with GitLab CI/CD Pipelines*. Packt Publishing Ltd, 2023.
- [10] P. Bitrián, I. Buil, and S. Catalán, "Enhancing user engagement: The role of gamification in mobile apps," *Journal of Business Research*, vol. 132, no. 1, pp. 170–185, Aug. 2021, doi: <https://doi.org/10.1016/j.jbusres.2021.04.028>.

Appendix

URL:

You can try the site here.

drawsitetest.neocities.org

Weekly Progress Sheets:

Name: Hasan Sivas 2421062

Programme: Changemakers

Project Aim: To research, design and develop a web-based art application that gives users the opportunity to compete with others and improve their skills.

Project Objectives:

1. Create the webpage with all of the necessary HTML, CSS and JavaScript.
2. Create a basic server for demonstration and making sure that multiplayer is implemented properly.
3. Thoroughly test and refine the application.

Weekly Project progress form 1

Date: 11/03/2026

Progress:

- Produced a plan for the website layout
- Produced a small test page with a useable canvas
- Designed a questionnaire and received 2 responses

Actions:

State priority actions to undertake over the next week below.

1. Finish HTML for first page
2. Consider how to integrate feedback
3. Implement image saving/copying using JS

Next review date: 18/03/2026.

Weekly Project progress form 2

Date: 18/03/2026

Progress:

- Designed a rough logo for the startup/web application using MS Paint
- Proposed a startup
- Prompted a PEST analysis

Actions:

State priority actions to undertake over the next week below.

1. Finish HTML for first page
2. Polish up PEST analysis
3. Implement image saving/copying using JS
4. Refine logo
5. Make a GitHub repo for the project

Next review date: 25/03/2026.

Weekly Project progress forms 3

Date: 25/03/2026

Progress:

- Created a GitHub repo for the project
- Researched and practised using WebSockets
- Made PEST analysis into table

Actions:

State priority actions to undertake over the next week below.

1. Implement image saving/copying using JS
2. Design & Implement prompting algorithm
3. Implement webpages

Next review date: 02/04/2026.

Weekly Project progress forms 4

Date: 14/04/2026

Progress:

- Implemented prompt generation
- Drew icon assets

Actions:

State priority actions to undertake over the next week below.

1. Add canvas tools
2. Work on image copy/downloading
3. Work on palette randomisation

Next review date: 22/04/2026.

Weekly Project progress forms 5

Date: 22/04/2026

Progress:

- Implemented canvas tools & tool switching
- Implemented palette randomisation
- Added more canvas tools

Actions:

State priority actions to undertake over the next week below.

4. Implement image saving/copying using JS
5. Make the webpage look nicer
6. Secondary mode

Next review date: 22/04/2026.

Weekly Project progress forms 6

Date: 29/04/2026

Progress:

- Added saving and copying
- Expanded tools
- Added the timer
- Made the CSS look nicer

Actions:

State priority actions to undertake over the next week below.

7. Daily/Challenge Mode split
8. hardmode

Next review date: 05/05/2026.

Date: 05/05/2026

Progress:

- Shifted from set colours to random colours
- Further improved UI
- Added Daily/ Challenge mode
- Added hardmode

HTML + Javascript:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Draw Site</title>
<link rel="stylesheet" href="style.css">
</head>

<body>

<main class="app">
  <header class="app-header">

    <div class="nav-bar">
      <button onclick="switchMode('daily')" id="dailyTab" class="tooltip-trigger
tooltip-bottom" data-tooltip="Daily gives everyone the same prompt and 8-colour palette for
today.">Daily</button>
      <button onclick="switchMode('challenge')" id="challengeTab" class="tooltip-trigger
tooltip-bottom" data-tooltip="Challenge gives you a random timed prompt and random
```

```

palette.">Challenge</button>
</div>

<h1><b>Draw Site</b></h1>
<h2>You should draw...</h2>
<h2 id="prompt">Loading prompt...</h2>

<div class="top-controls">
  <button onclick="updatePrompt()" id="newPromptButton">New Prompt</button>
  <button onclick="finishRound()" id="finishRoundButton">Finish Round</button>
  <h3 id="timer">Time left: 05:00</h3>
  <label id="hardModeControl" class="toggle-control tooltip-trigger"
data-tooltip="Hardmode: 1–5 palette colours, 3 adjectives, and the prompt hides after 10
seconds.">
    <input type="checkbox" id="hardModeToggle">
    Hardmode
  </label>
</div>

</header>

<!-- WORKSPACE: canvas + right panel -->
<section class="workspace">

  <div class="canvas-section">
    <canvas id="canvas" width="800" height="600"></canvas>
  </div>

  <div class="controls-panel">

    <div class="control-group">
      <h3>Colours</h3>
      <div id="fixedColours" class="button-row"></div>
      <div id="colourPalette" class="button-row"></div>
    </div>

    <div class="control-group">
      <h3>Brush Size</h3>
      <input type="range" id="brushSize" min="1" max="40" value="5">
      <span id="brushSizeLabel">5</span>
    </div>

    <div class="control-group">
      <h3>Tools</h3>
      <div class="button-row tools">
        <button onclick="setTool('pencil', this)">
          
        </button>

        <button onclick="setTool('fill', this)">

```

```

        
    </button>

    <button onclick="setTool('eraser', this)">
        
    </button>

    <button onclick="undoCanvas()">
        
    </button>

    <button onclick="redoCanvas()">
        
    </button>

    <button onclick="clearCanvas()">Clear Canvas</button>
</div>
</div>

<div class="control-group">
    <h3>Export</h3>
    <div class="button-row">
        <button onclick="downloadImage()">Download PNG</button>
        <button onclick="copyToClipboard()">Copy to Clipboard</button>
    </div>
</div>

</div>

</section>

<!-- TIMER MODAL -->
<div id="timerModal">
    <div id="timerModalContent">
        <h2 id="timerModalTitle">Time's up!</h2>
        <button onclick="downloadImage()">Download</button>
        <button onclick="copyToClipboard()">Copy</button>
        <button onclick="handleModalNewPrompt()">New Prompt</button>
        <button onclick="closeModal()">Continue Drawing</button>
    </div>
</div>

</main>

<script>
//to test
//python -m http.server
//http://localhost:8000
const canvas = document.getElementById("canvas");
const ctx = canvas.getContext("2d");

```

```

let timerInterval = null;
let promptHideTimeout = null;
let timeLeft = 300; // 5 minutes in seconds
let roundDuration = 300;
let isModalOpen = false;
let isHardMode = false;
let currentPromptText = "";

let currentColor = "black"; //always initializes as black
let currentTool = "pencil"; //always initializes as pencil
let drawing = false;

let undoStack = [];
let redoStack = [];
const maxUndoSteps = 20;

//brush size
const brushSize = document.getElementById("brushSize");
const brushSizeLabel = document.getElementById("brushSizeLabel");
const hardModeControl = document.getElementById("hardModeControl");
const hardModeToggle = document.getElementById("hardModeToggle");

brushSize.addEventListener("input", () => {
  brushSizeLabel.textContent = brushSize.value;
});

hardModeToggle.addEventListener("change", () => {
  isHardMode = hardModeToggle.checked;
  updateTheme();

  if (currentMode === "challenge") {
    updatePrompt();
  }
});

//daily-challenge mode switch
let currentMode = "daily";

function switchMode(mode) {
  currentMode = mode;
  stopPromptHideTimer();
  hardModeControl.style.display = mode === "challenge" ? "inline-flex" : "none";
  updateTheme();

  document.getElementById("dailyTab").classList.remove("active");
  document.getElementById("challengeTab").classList.remove("active");

  const newPromptButton = document.getElementById("newPromptButton");

```

```

    if (mode === "daily") {
        document.getElementById("dailyTab").classList.add("active");
        newPromptButton.style.display = "none";
        generateDailyPrompt();
    } else if (mode === "challenge") {
        document.getElementById("challengeTab").classList.add("active");
        newPromptButton.style.display = "inline-block";
        updatePrompt(); // normal random prompt
    }
}

function updateTheme() {
    document.body.classList.toggle("hardmode-theme", currentMode === "challenge" &&
isHardMode);
}

function handleModalNewPrompt() {
    closeModal();

    if (currentMode === "daily") {
        switchMode("challenge");
    } else {
        updatePrompt();
    }
}

//daily mode
function seededRandom(seed) {
    let x = Math.sin(seed) * 10000;
    return x - Math.floor(x);
}

function seededItem(arr, seed) {
    return arr[Math.floor(seededRandom(seed) * arr.length)];
}

// prompt helpers for structured JSON
function getSubjectTypes() {
    return ["creatures", "objects"];
}

function getAdjectiveCategories() {
    return ["creatures", "objects", "both"];
}

function getSubjectPool(subjectType) {
    return wordData.subjects[subjectType];
}

function getAdjectivePool(subjectType) {

```

```

const adjectives = wordData.descriptors.adjectives;

if (subjectType === "creatures") {
  return [...adjectives.creatures, ...adjectives.both];
}

if (subjectType === "objects") {
  return [...adjectives.objects, ...adjectives.both];
}

return adjectives.both;
}

function getRandomAdjective(subjectType) {
  const adjectives = wordData.descriptors.adjectives;

  // 30% chance to ignore whether it is a creature/object and pull from any adjective category
  if (Math.random() < 0.3) {
    const categories = getAdjectiveCategories();
    const randomCategory = randomItem(categories);
    return randomItem(adjectives[randomCategory]);
  }

  return randomItem(getAdjectivePool(subjectType));
}

function getSeededAdjective(subjectType, seed) {
  const adjectives = wordData.descriptors.adjectives;

  // 30% chance to ignore whether it is a creature/object and pull from any adjective category
  if (seededRandom(seed) < 0.3) {
    const categories = getAdjectiveCategories();
    const randomCategory = seededItem(categories, seed + 1);
    return seededItem(adjectives[randomCategory], seed + 2);
  }

  return seededItem(getAdjectivePool(subjectType), seed + 3);
}

function generateDailyPrompt() {
  if (!wordData) return;

  closeModal();
  stopPromptHideTimer();

  const today = new Date();
  const seed = today.getFullYear() * 1000 + today.getMonth() * 100 + today.getDate();

  const subjectTypes = getSubjectTypes();
  const subjectType = seededItem(subjectTypes, seed + 10);

```

```

const subject = seededItem(getSubjectPool(subjectType), seed + 20);

const adj1 = getSeededAdjective(subjectType, seed + 30);
const adj2 = getSeededAdjective(subjectType, seed + 40);

const prompt = `A ${adj1}, ${adj2} ${subject}!`;
showPrompt(prompt);

currentColor = "black";

// Daily mode always uses the same 8 colours for the calendar date
const dailyColours = getDailyColours(8, seed + 100);
renderColourPalette(dailyColours);

undoStack = [];
redoStack = [];
clearCanvas(false);

// Daily mode always uses the normal fixed 5 minute timer
startTimer(300);
}

//UNDO
function saveCanvasState() {
  undoStack.push(ctx.getImageData(0, 0, canvas.width, canvas.height));

  if (undoStack.length > maxUndoSteps) {
    undoStack.shift();
  }

  redoStack = []; //wipe redo instead of forking
}

function undoCanvas() {
  if (undoStack.length === 0) return;

  const currentState = ctx.getImageData(0, 0, canvas.width, canvas.height);
  redoStack.push(currentState); // save current for redo

  const previousState = undoStack.pop();
  ctx.putImageData(previousState, 0, 0);
  ctx.beginPath();
}

function redoCanvas() {
  if (redoStack.length === 0) return;

  const currentState = ctx.getImageData(0, 0, canvas.width, canvas.height);
  undoStack.push(currentState); // allow undo of redo

```

```

    const nextState = redoStack.pop();
    ctx.putImageData(nextState, 0, 0);
    ctx.beginPath();
}

//TIMER!
function startTimer(duration = 300) {
    clearInterval(timerInterval); // reset any existing timer
    roundDuration = duration;
    timeLeft = duration;
    updateTimerDisplay();

    timerInterval = setInterval(() => {
        timeLeft--;
        updateTimerDisplay();

        if (timeLeft <= 0) {
            clearInterval(timerInterval);
            onTimerEnd();
        }
    }, 1000);
}

function updateTimerDisplay() {
    timeLeft = Math.max(0, timeLeft);
    const minutes = Math.floor(timeLeft / 60);
    const seconds = timeLeft % 60;

    document.getElementById("timer").textContent =
        `Time left: ${String(minutes).padStart(2, "0")}:${String(seconds).padStart(2, "0")}`;
}

function onTimerEnd(title = "Time's up!") {
    isModalOpen = true;
    drawing = false;
    ctx.beginPath();
    document.getElementById("timerModalTitle").textContent = title;
    document.getElementById("timerModal").classList.add("show");
}

function finishRound() {
    clearInterval(timerInterval);
    stopPromptHideTimer();
    updateTimerDisplay();
    onTimerEnd("Round finished!");
}

function closeModal() {
    isModalOpen = false;
    document.getElementById("timerModal").classList.remove("show");
}

```

```

}

//TOOL SWITCHING
function setTool(tool, el) {
  currentTool = tool;

  document.querySelectorAll("button").forEach(b => {
    b.style.outline = "none";
  });

  el.style.outline = "3px solid black";//highlights currently selected tool (looks ugly though)
}

canvas.addEventListener("mousedown", (e) => {
  if (isModalOpen) return;

  if (currentTool === "pencil" || currentTool === "eraser") {
    saveCanvasState();
    drawing = true;
  } else if (currentTool === "fill") {
    saveCanvasState();
    const pos = getCanvasPos(e);
    floodFill(Math.floor(pos.x), Math.floor(pos.y));
  }
});

canvas.addEventListener("mouseup", () => {
  drawing = false;
  ctx.beginPath();
});

canvas.addEventListener("mousemove", draw);

//FILL TOOL
function floodFill(startX, startY) {
  const imageData = ctx.getImageData(0, 0, canvas.width, canvas.height);//gets full bounds
  of canvas
  const data = imageData.data;

  const targetColor = getPixel(data, startX, startY);
  const fillColor = cssColorToRgba(currentColor);

  // Use alpha too, so transparent black and solid black are NOT treated as the same
  if (exactColorsMatch(targetColor, fillColor)) return;

  const targetIsBlank = targetColor[3] < 20;
  const stack = [[startX, startY]];
  const visited = new Uint8Array(canvas.width * canvas.height);

```

```

while (stack.length) {
  const [x, y] = stack.pop();

  if (x < 0 || x >= canvas.width || y < 0 || y >= canvas.height) continue;

  const index = y * canvas.width + x;
  if (visited[index]) continue;
  visited[index] = 1;

  const current = getPixel(data, x, y);

  if (targetIsBlank) {
    // If filling empty canvas space, fill transparent / nearly transparent pixels.
    // This helps reduce stray pixels near anti-aliased lines.
    if (current[3] > 110) continue;
  } else {
    // If filling an already-coloured area, use normal tolerance matching.
    if (!colorsMatch(current, targetColor, 45)) continue;
  }

  setPixel(data, x, y, fillColor);

  stack.push([x + 1, y]);
  stack.push([x - 1, y]);
  stack.push([x, y + 1]);
  stack.push([x, y - 1]);
}

ctx.putImageData(imageData, 0, 0);

// small patch to cover tiny gaps at click point
ctx.globalCompositeOperation = "source-over";
ctx.fillStyle = currentColor;
ctx.beginPath();
ctx.arc(startX, startY, 2, 0, Math.PI * 2);
ctx.fill();
}

// fill helpers
function getPixel(data, x, y) {
  const i = (y * canvas.width + x) * 4;
  return [data[i], data[i+1], data[i+2], data[i+3]];
}

function setPixel(data, x, y, color) {
  const i = (y * canvas.width + x) * 4;
  data[i] = color[0];
  data[i+1] = color[1];
  data[i+2] = color[2];
  data[i+3] = color[3];
}

```

```

}

function exactColorsMatch(a, b) {
  return a[0] === b[0] &&
    a[1] === b[1] &&
    a[2] === b[2] &&
    a[3] === b[3];
}

function colorsMatch(a, b, tolerance = 30) {
  return Math.abs(a[0] - b[0]) <= tolerance &&
    Math.abs(a[1] - b[1]) <= tolerance &&
    Math.abs(a[2] - b[2]) <= tolerance &&
    Math.abs(a[3] - b[3]) <= tolerance;
}

// Convert CSS colour to RGBA
function cssColorToRgba(color) {
  const tempCanvas = document.createElement("canvas");
  const tempCtx = tempCanvas.getContext("2d");

  tempCtx.fillStyle = color;
  const computed = tempCtx.fillStyle;

  const hex = computed.replace("#", "");
  const bigint = parseInt(hex, 16);

  return [
    (bigint >> 16) & 255,
    (bigint >> 8) & 255,
    bigint & 255,
    255 // fully opaque otherwise ignores black pixels
  ];
}

//find position
function getCanvasPos(e) {
  const rect = canvas.getBoundingClientRect();

  return {
    x: (e.clientX - rect.left) * (canvas.width / rect.width),
    y: (e.clientY - rect.top) * (canvas.height / rect.height)
  };
}

//Draww
function draw(e) {
  if (!drawing || isModalOpen) return;

  const pos = getCanvasPos(e);

```

```

ctx.lineWidth = brushSize.value;
ctx.lineCap = "round";

if (currentTool === "eraser") {
  ctx.globalCompositeOperation = "destination-out";
} else {
  ctx.globalCompositeOperation = "source-over";
  ctx.strokeStyle = currentColor;
}

ctx.lineTo(pos.x, pos.y);
ctx.stroke();
ctx.beginPath();
ctx.moveTo(pos.x, pos.y);
}

let wordData = null;

// Load JSON
async function loadWords() {
  const res = await fetch('words.json?v=' + Date.now());
  wordData = await res.json();
}

// Random helper
function randomItem(arr) { //todo, use date and time derrived generation instead of completely
  random (random for challenge mode only)
  return arr[Math.floor(Math.random() * arr.length)];
}

// Random integer helper
function randomInt(min, max) {
  return Math.floor(Math.random() * (max - min + 1)) + min;
}

//always need black and white as basic colours
function renderFixedColours() {
  const fixedDiv = document.getElementById("fixedColours");
  fixedDiv.innerHTML = "";

  const fixed = ["black", "white"];

  fixed.forEach(colour => {
    const btn = document.createElement("button");

    btn.title = colour;
    btn.style.backgroundColor = colour;
    btn.style.width = "34px";
    btn.style.height = "34px";
  });
}

```

```

    btn.style.borderRadius = "50%";
    btn.style.border = "2px solid black";
    btn.style.cursor = "pointer";
    btn.style.padding = "0";

    btn.onclick = () => {
        currentColor = colour;

        document.querySelectorAll("#fixedColours button, #colourPalette button")
            .forEach(b => b.style.outline = "none");

        btn.style.outline = "3px solid black";
    };

    fixedDiv.appendChild(btn);
});
}

// Generate random colours
function getRandomColours(count) {
    const colours = [];

    for (let i = 0; i < count; i++) {
        const hue = Math.floor(Math.random() * 360);
        const saturation = Math.floor(60 + Math.random() * 35);
        const lightness = Math.floor(40 + Math.random() * 25);

        colours.push(`hsl(${hue}, ${saturation}%, ${lightness}%)`);
    }

    return colours;
}

// daily colours
function getDailyColours(count, seed) {
    const colours = [];

    for (let i = 0; i < count; i++) {
        const hue = Math.floor(seededRandom(seed + i * 10) * 360);
        const saturation = Math.floor(60 + seededRandom(seed + i * 20) * 35);
        const lightness = Math.floor(40 + seededRandom(seed + i * 30) * 25);

        colours.push(`hsl(${hue}, ${saturation}%, ${lightness}%)`);
    }

    return colours;
}

// Render palette buttons
function renderColourPalette(colours = null) {

```

```

const paletteDiv = document.getElementById("colourPalette");
paletteDiv.innerHTML = "";

const selectedColours = colours || getRandomColours(8);

selectedColours.forEach(colour => {
  const btn = document.createElement("button");

  btn.title = colour;
  btn.style.backgroundColor = colour;
  btn.style.width = "34px";
  btn.style.height = "34px";
  btn.style.borderRadius = "50%";
  btn.style.border = "2px solid white";
  btn.style.cursor = "pointer";
  btn.style.padding = "0";

  btn.onclick = () => {
    currentColor = colour;

    document.querySelectorAll("#fixedColours button, #colourPalette button")
      .forEach(b => b.style.outline = "none");

    btn.style.outline = "3px solid black";
  };

  paletteDiv.appendChild(btn);
});
}

//clear the canvas
function clearCanvas(shouldSave = true) {
  if (shouldSave) {
    saveCanvasState();
  }

  ctx.clearRect(0, 0, canvas.width, canvas.height);
  ctx.beginPath();
  undoStack = [];
  redoStack = [];
}

// Generate prompt
function generateArtPrompt(adjectiveCount = 2) {
  const subjectTypes = getSubjectTypes();
  const subjectType = randomItem(subjectTypes);
  const subject = randomItem(getSubjectPool(subjectType));

  const adjectives = [];
  for (let i = 0; i < adjectiveCount; i++) {

```

```

    adjectives.push(getRandomAdjective(subjectType));
  }

  return `A ${adjectives.join(", ")} ${subject}!`;
}

function stopPromptHideTimer() {
  clearTimeout(promptHideTimeout);
  promptHideTimeout = null;
}

function showPrompt(prompt) {
  stopPromptHideTimer();
  currentPromptText = prompt;
  document.getElementById("prompt").textContent = prompt;
}

function hidePromptAfterDelay() {
  stopPromptHideTimer();

  promptHideTimeout = setTimeout(() => {
    if (currentMode === "challenge" && isHardMode) {
      document.getElementById("prompt").textContent = "HIDDEN";
    }
  }, 10000);
}

//download image
function makeFilenameFromPrompt(prompt) {
  return prompt
    .toLowerCase()
    .replace(/[^a-z0-9\s]/g, "")
    .trim()
    .replace(/\s+/g, "_")
    .slice(0, 50);
}

function formatDuration(totalSeconds) {
  const safeSeconds = Math.max(0, totalSeconds);
  const minutes = Math.floor(safeSeconds / 60);
  const seconds = safeSeconds % 60;

  return `${String(minutes).padStart(2, "0")}:${String(seconds).padStart(2, "0")}`;
}

function getTimeTakenSeconds() {
  return Math.max(0, roundDuration - timeLeft);
}

function getRoundDetailsText() {

```

```

const prompt = currentPromptText || document.getElementById("prompt").textContent;

return [
  `Prompt: DRAW ${prompt.toUpperCase()}`,
  `Time Taken: ${formatDuration(getTimeTakenSeconds())}`
];
}

function makeCanvasWithRoundDetails() {
  const detailLines = getRoundDetailsText();

  const padding = 24;
  const lineHeight = 28;
  const detailsHeight = padding * 2 + 40 + detailLines.length * lineHeight;

  const exportCanvas = document.createElement("canvas");
  exportCanvas.width = canvas.width;
  exportCanvas.height = canvas.height + detailsHeight;

  const exportCtx = exportCanvas.getContext("2d");

  // White background for the whole exported image
  exportCtx.fillStyle = "white";
  exportCtx.fillRect(0, 0, exportCanvas.width, exportCanvas.height);

  // Draw the user's artwork
  exportCtx.drawImage(canvas, 0, 0);

  // Divider line
  exportCtx.strokeStyle = "#d1d5db";
  exportCtx.lineWidth = 2;
  exportCtx.beginPath();
  exportCtx.moveTo(0, canvas.height);
  exportCtx.lineTo(exportCanvas.width, canvas.height);
  exportCtx.stroke();

  // Draw Draw Site
  exportCtx.fillStyle = "#1e3a8a"; // dark blue
  exportCtx.font = "bold 28px Arial, Helvetica, sans-serif";
  exportCtx.textBaseline = "top";
  exportCtx.fillText("Draw Site", padding, canvas.height + padding);

  // Draw round details underneath
  exportCtx.fillStyle = "#111827";
  exportCtx.font = "20px Arial, Helvetica, sans-serif";

  detailLines.forEach((line, index) => {
    exportCtx.fillText(
      line,
      padding,

```

```

        canvas.height + padding + 40 + index * lineHeight
    );
});

return exportCanvas;
}

function canvasToBlob(sourceCanvas) {
    return new Promise((resolve, reject) => {
        sourceCanvas.toBlob((blob) => {
            if (blob) {
                resolve(blob);
            } else {
                reject(new Error("Could not create image blob."));
            }
        }, "image/png");
    });
}

function downloadImage() {
    const prompt = currentPromptText || document.getElementById("prompt").textContent;
    const filename = makeFilenameFromPrompt(prompt) || "drawing";

    const exportCanvas = makeCanvasWithRoundDetails();

    const link = document.createElement("a");
    link.download = filename + ".png";
    link.href = exportCanvas.toDataURL("image/png");
    link.click();
}

//copy to clipboard
async function copyToClipboard() {
    try {
        const exportCanvas = makeCanvasWithRoundDetails();
        const imageBlob = await canvasToBlob(exportCanvas);

        await navigator.clipboard.write([
            new ClipboardItem({
                "image/png": imageBlob
            })
        ]);

        alert("Copied image with round details to clipboard!");
    } catch (err) {
        console.error(err);
        alert("Copy failed (browser may not support copying images)");
    }
}

```

```

// Update UI
function updatePrompt() {
  if (!wordData) return;

  closeModal();
  stopPromptHideTimer();

  const prompt = generateArtPrompt(isHardMode ? 3 : 2);
  showPrompt(prompt);

  if (isHardMode) {
    hidePromptAfterDelay();
  }

  currentColor = "black";

  // Challenge mode gets a random palette size from 3 to 9 colours, or 1 to 5 in hardmode
  const challengePaletteSize = isHardMode ? randomInt(1, 5) : randomInt(3, 9);
  renderColourPalette(getRandomColours(challengePaletteSize));

  undoStack = [];
  redoStack = [];
  clearCanvas(false);

  // Challenge mode gets a random timer length from 2 to 5 minutes
  const challengeTimerLength = randomInt(120, 300);
  startTimer(challengeTimerLength);
}

// Init
async function init() {
  await loadWords();
  renderFixedColours();
  switchMode("daily");
}

init();

</script>

</body>
</html>

```

CSS:

```

root {

```

```
--theme-primary: #2563eb;
--theme-primary-dark: #1d4ed8;
--theme-primary-deep: #1e3a8a;
--theme-primary-light: #60a5fa;
--theme-panel-bg: #f8fafc;
--theme-panel-border: #dbeafe;
}

body.hardmode-theme {
  --theme-primary: #dc2626;
  --theme-primary-dark: #991b1b;
  --theme-primary-deep: #7f1d1d;
  --theme-primary-light: #f87171;
  --theme-panel-bg: #fef2f2;
  --theme-panel-border: #fecaca;
}

body {
  margin: 0;
  font-family: Arial, Helvetica, sans-serif;
  background: linear-gradient(90deg, var(--theme-primary) 0%, var(--theme-primary-light)
50%, var(--theme-primary) 100%);
  transition: background 0.25s ease;
  min-height: 100vh;

  display: flex;
  justify-content: center;
  align-items: flex-start;
}

.app {
  width: min(1000px, 95vw);
  background: white;
  min-height: 100vh;
  padding: 24px;
  box-sizing: border-box;
  box-shadow: 0 0 30px rgba(0,0,0,0.25);
}

.app-header {
  text-align: center;
}

h1 {
  margin: 0 0 10px;
  color: var(--theme-primary-deep);
  transition: color 0.25s ease;
  font-size: 42px;
}
```

```
h2 {
  margin: 6px 0;
  color: #333;
}

#prompt {
  color: var(--theme-primary);
  transition: color 0.25s ease;
}

.top-controls {
  display: flex;
  justify-content: center;
  align-items: center;
  gap: 16px;
  margin: 16px 0;
  flex-wrap: wrap;
}

.canvas-section {
  display: flex;
  justify-content: center;
  margin: 0;
}

canvas {
  background: white;
  border: 3px solid var(--theme-primary-deep);
  transition: border-color 0.25s ease;
  border-radius: 12px;
  box-shadow: 0px 6px 18px rgba(0,0,0,0.2);
  max-width: 100%;
}

.controls-panel {
  width: 260px;
  display: flex;
  flex-direction: column;
  gap: 16px;
}

.control-group {
  background: var(--theme-panel-bg);
  border: 1px solid var(--theme-panel-border);
  border-radius: 12px;
  padding: 14px;
  transition: background 0.25s ease, border-color 0.25s ease;
}

.control-group h3 {
```

```
margin: 0 0 10px;
color: var(--theme-primary-deep);
}

.button-row,
#fixedColours,
#colourPalette {
  display: flex;
  flex-wrap: wrap;
  gap: 8px;
  align-items: center;
}

button {
  border: none;
  border-radius: 8px;
  padding: 9px 14px;
  background: var(--theme-primary);
  transition: background 0.25s ease;
  color: white;
  cursor: pointer;
  font-size: 14px;
}

button:hover {
  background: var(--theme-primary-dark);
}

#timer {
  margin: 0;
  color: var(--theme-primary-deep);
  transition: color 0.25s ease;
}

.toggle-control {
  display: none;
  align-items: center;
  gap: 8px;
  border-radius: 8px;
  padding: 8px 12px;
  background: var(--theme-panel-bg);
  border: 1px solid var(--theme-panel-border);
  color: var(--theme-primary-deep);
  transition: background 0.25s ease, border-color 0.25s ease, color 0.25s ease;
  font-weight: bold;
  cursor: pointer;
}

.toggle-control input {
  cursor: pointer;
}
```

```
}

.tooltip-trigger {
  position: relative;
}

.tooltip-trigger::after {
  content: attr(data-tooltip);
  position: absolute;
  left: 50%;
  bottom: calc(100% + 10px);
  transform: translateX(-50%);
  width: max-content;
  max-width: 260px;
  padding: 8px 10px;
  border-radius: 8px;
  background: #111827;
  color: white;
  font-size: 13px;
  font-weight: normal;
  line-height: 1.3;
  text-align: center;
  opacity: 0;
  visibility: hidden;
  pointer-events: none;
  z-index: 20;
}

.tooltip-trigger::before {
  content: "";
  position: absolute;
  left: 50%;
  bottom: calc(100% + 4px);
  transform: translateX(-50%);
  border: 6px solid transparent;
  border-top-color: #111827;
  opacity: 0;
  visibility: hidden;
  pointer-events: none;
  z-index: 21;
}

.tooltip-trigger:hover::after,
.tooltip-trigger:focus-visible::after,
.tooltip-trigger:focus-within::after,
.tooltip-trigger:hover::before,
.tooltip-trigger:focus-visible::before,
.tooltip-trigger:focus-within::before {
  opacity: 1;
  visibility: visible;
}
```

```

}

.tooltip-bottom::after {
  top: calc(100% + 10px);
  bottom: auto;
}

.tooltip-bottom::before {
  top: calc(100% + 4px);
  bottom: auto;
  border-top-color: transparent;
  border-bottom-color: #111827;
}

/* simple modal styling */
#timerModal {
  position: fixed;
  top: 0;
  left: 0;
  width: 100%;
  height: 100%;
  background: rgba(0,0,0,0.6);
  display: flex;
  align-items: center;
  justify-content: center;

  opacity: 0;
  visibility: hidden;
  pointer-events: none;
  transition: opacity 0.3s ease, visibility 0.3s ease;
}

#timerModal.show {
  opacity: 1;
  visibility: visible;
  pointer-events: all;
}

#timerModalContent {
  background: white;
  padding: 24px;
  border-radius: 14px;
  text-align: center;
  transform: scale(0.95);
  transition: transform 0.3s ease;
}

#timerModal.show #timerModalContent {
  transform: scale(1);
}

```

```
.workspace {
  display: flex;
  gap: 24px;
  align-items: flex-start;
  justify-content: center;
}

@media (max-width: 1100px) {
  .workspace {
    flex-direction: column;
    align-items: center;
  }

  .controls-panel {
    width: 100%;
  }
}

.nav-bar {
  display: flex;
  gap: 10px;
  justify-content: center;
  margin-bottom: 10px;
}

.nav-bar button {
  padding: 8px 16px;
  border-radius: 6px;
  border: none;
  background: #ddd;
  cursor: pointer;
}

.nav-bar button.active {
  background: #333;
  color: white;
}
```

Continued Professional Development:

Cisco Networking Academy

INSTITUTE

JavaScript Essentials 2

Course Outline

Resources

Search course outline

Q

JSE2: Module 1: Classless Objects

30%

1.0. Welcome to JavaScript Essentials 2

6 / 6

1.1. Section 1 - What is an object?

3 / 3

1.2. Section 2 - Literals

2 / 2

1.3. Section 3 - Properties

6 / 6

1.3.1 Types

1.3.2 Nested properties

1.3.3 A function as a property type - a method

1.3.4 Adding a new property

1.3.5 Modifying a property

1.3.6 Deleting a property

1.4. Section 4 - Dot notation vs. bracket notation

0 / 2

1.5. Section 5 - Property existence test and property enumeration

0 / 4

1.6. Section 6 - References

0 / 4

1.7. Section 7 - Methods

0 / 4

1.3.3 A function as a property type - a method

The property of an object, as we have seen before, can also be a function.
A function that is the property of an object will be called a method.
Just as the properties of objects describe their characteristics, methods can be treated as their characteristic behavior, or ways of changing the state of an object.
As an example, let's consider an object describing a point on the surface. We'll only describe its two properties, which determine its position, that is, the position `x` and `y`. To this we'll add two methods, `moveVertically` and `moveHorizontally`.
Look at the code below:

```
1 let point = {  
2   x: 0,  
3   y: 0,  
4   moveHorizontally: function(distance) {  
5     this.x = this.x + distance;  
6   },  
7   moveVertically: function(distance) {  
8     this.y = this.y + distance;  
9   }  
}
```

Cisco Networking Academy

INSTITUTE

JavaScript Essentials 2

Course Outline

Resources

Search course outline

Q

2.2. Section 2 - Properties

3 / 3

2.3. Section 3 - Getters and setters

1 / 1

2.4. Section 4 - Inheritance

3 / 3

2.5. Section 5 - Static members

1 / 1

2.6. Section 6 - Classes vs. constructors

1 / 1

2.7. Module 2 Completion - Closing thoughts and hands-on exercises

6 / 6

2.8. Module 2 Completion - Module Test

JSE2: Module 3: Built-In Objects

JSE2: Module 4: Advanced Function Usage

Earn JSA Certification

End of Course Survey

2.8. Module 2 Completion - Module Test

Q1 Q2 Q3 Q4 Q5

Question 5

There is one line missing in the code below:

```
1 class A {  
2   // Insert line of code here.  
3 }
```

Select the correct missing line in order to declare a property named test and initialize it with the value 10:

☐ `test = 10;`

☐ `test: 10;`

5 of 19 Questions

Skip All

Skip Question

Submit

Cisco
Networking
Academy

INSTITUTE

JavaScript Essentials 2

Course Outline

Resources

Search course outline

3.2.29 Handling elements

3.2.30 Walking through the map elements

3.2.31 The for ... of loop

3.2.32 The spread operator in maps

3.2.33 Object

3.2.34 Creating an object

3.2.35 Handling elements

3.2.36 Walking through elements

3.2.37 The spread operator for objects

> 3.3. Section 3 - JSON, Math and RegExp 0 / 12

> 3.4. Section 4 - Extending built-in types 0 / 1

> 3.5. Module 3 Completion - Closing thoughts and hands-on exercises 0 / 6

> 3.6. Module 3 Completion - Module Test

JSE2: Module 4: Advanced Function Usage

Earn JSA Certification

3.2.31 The for ... of loop

3.2.31 The for ... of loop

For the purpose of going through the collections (i.e. arrays, sets, and maps) another simple mechanism was created in JavaScript – the `for ... of` loop.

It is universal for all collections and its use is so intuitive it does not require any explanation.

We analyze its functioning on the basis of the code presented here and compare the results for the array, then for set and map.

Look at the code below:

```
1 let petsArray = ["cat", "dog", "hamster"];
2 for( let pet of petsArray) {
3   console.log(pet); // -> cat -> dog -> hamster
4 }
5 let petsSet = new Set(["cat", "dog", "hamster"]);
6 for( let pet of petsSet) {
7   console.log(pet); // -> cat -> dog -> hamster
8 }
9 let petsMap = new Map([["cats", 1], ["dogs", 3], ["hamsters", 2]]);
10 for( let pet of petsMap) {
11   console.log(pet); // -> ["cats", 1] -> ["dogs", 3] -> ["hamsters", 2]
12   console.log(pet[0]); // -> cats -> dogs -> hamsters
13 }
```

Screenshots of the JavaScript Essentials 2 course on Cisco Networking Academy, a useful course when learning to build a web application