School of Games and Creative Technology
BSc (Hons) Computer Science

University for the Creative Arts

FGCT6021 MOBILE APPLICATION DEVELOPMENT
LAB 4 – TOPIC 4 INTRODUCTION TO API

Submission

1. Submit your Learning Journal link (GitHub) and weekly journal entry on myUCA.
2. During the workshop, you will show your progress and share what you have learned.

Workshop Overview

You will extend your Learning Journal Progressive Web App (PWA) by applying APIs to make it more powerful and interactive. This week's focus is on storing data using a **Storage API**, enhancing your journal with a **Browser API**, and integrating an external service using a **Third-Party API**.

Your goal this week is to:

- Use a **Storage API** to save and retrieve data.
- Apply **at least one Browser API** of your choice to add a useful feature.
- Apply **at least one Third-Party API** to integrate external content or services.

This builds on your previous work with JavaScript and the DOM, now extending the PWA to use persistent storage, browser capabilities, and external platforms.

Goal

Add a **Storage API**, a **Browser API**, and a **Third-Party API** to your Learning Journal PWA.

All JavaScript files should be placed in the `/js` folder, linked correctly, and committed to your **GitHub** repository so progress is visible and version-controlled.

What to Do

The following tasks are suggestions. You may adapt them as appropriate to your own design.

Folder and File Structure

1. Continue using the folder structure from Week 3:
   - `/css` Stylesheets
   - `/js` JavaScript files
   - `/images` Media assets

2. Add new JavaScript files if needed for API-specific features, for example:
   - `/js/storage.js`
   - `/js/browser.js`
   - `/js/thirdparty.js`

Storage API

1. Save journal entries and display them after a page reload.
2. Store theme preference (light or dark mode) so they remain consistent across visits.
3. Use **at least one Storage API** to persist data in your PWA, such as:
   - **Local Storage:** Keep entries or theme settings even after closing the browser.
   - **Session Storage:** Store temporary data that resets when the browser closes.
   - **IndexedDB API:** Store larger or more complex data structures, such as multiple journal entries and images.

Browser API

1. Use **at least one Browser API** to add interactive functionality, such as:
   - **Validation API:** Add rules for journal form inputs.
   - **Clipboard API:** Provide a button to copy entries to the clipboard.
   - **Notifications API:** Alert the user when a new entry is saved.
   - **Geolocation API:** Show the user's location on a page.
2. For more available browser APIs, explore MDN Web APIs list.

Third-Party API

1. Integrate an external service into your journal.
2. Use **at least one Third-Party API**, such as:
   - **YouTube API:** Embed and control a video.
   - **Facebook API:** Enable sharing an entry to Facebook.
   - **Google Maps API:** Display a location on a map.
   - **Twitter API:** Show tweets based on a keyword.
   - **Spotify API:** Display or play music tracks or playlists.
3. For more options, explore Public APIs – A Collection of Free APIs.

DOM Manipulation Practice

1. Retrieve and display saved data from your chosen Storage API.
2. Use your Browser API to interact with page content dynamically.
3. Insert or control external content from your chosen Third-Party API.
4. Handle at least one relevant user event (click, submit, play/pause, or copy).

Journal Questions

For your weekly entry, **submit it on myUCA** and **post it on the Learning Journal Web**:

1. Which Storage, Browser, and Third-Party APIs did you choose, and why?
2. How did you integrate each API with DOM manipulation?
3. What challenges did you encounter, and how did you solve them?
4. In what ways do these APIs improve your Learning Journal PWA?