CP1295 —Advanced JavaScript
Mini-Project (20 %) "QuickNotes" – Browser-Based Sticky-Note Board

Individual Project – Due August 10, 2025 at 11:59 p.m. (submit on D2L)

Students must work **individually**; collaboration is limited to discussion.

1 Purpose

QuickNotes is a compact, single-page web app that allows users to create, move, edit, and delete virtual sticky notes directly in the browser. The workload is worth 20 % of the course grade while exercising every major topic in CP1295:

- DOM manipulation & events
- Custom objects & classes
- Browser storage & DevTools
- Asynchronous JavaScript (fetch, async/await)
- ES-module organisation
- JSON parsing & generation
- Core Git / GitHub practices

2 Learning Outcomes Demonstrated

1. Build and manipulate DOM elements with JavaScript events.
2. Encapsulate state in a custom class (Note).
3. Persist client-side data with localStorage and inspect it in DevTools.
4. Write and consume asynchronous code that fetches external data.
5. Structure a small codebase with ES modules (import/export).

6. Serialize and deserialize data to/from JSON.
7. Apply disciplined version control in a public GitHub repository.

3 Mandatory Features (MVP — already implemented)

#	Feature	Specification
1	Add note	Typing text (or double-clicking the board) creates a coloured note at the cursor.
2	Edit note	Click note inline textarea or modal; blur / Esc saves.
3	Drag & drop	Click-drag moves a note; position persists.
4	Delete note	The “ ” button removes it with a fade-out.
5	Persistence	Notes (text, colour, x, y) auto-save to localStorage; restored on refresh.
6	Random-quote enhancer	The “ ” icon retrieves a productivity quote from an open API (using async/await and error handling).
7	ES-module layout	Minimum files: main.js, ui.js, notes.js, storage.js.
8	JSON export	The “Export” button downloads my-notes.json, containing all notes.

3A Add-On Requirements (What needs to be done in this project by coding)

#	New Feature	What you must build
A	Images inside notes	Provide an “ ” icon (or drag-and-drop) so users can attach an image to a sticky note. • Store the image so it survives reloads (use a Data-URL in localStorage or a Blob in IndexedDB). • Scale it to fit neatly and keep the note’s text editable.
B	Timestamp	Automatically stamp each note with its creation date-and-time (e.g., “2025-07-09 14:37”) and display it. Persist the timestamp with the note.
C	Sort buttons	Add **Ascending** and **Descending** buttons that reorder all notes by timestamp and re-render the board.

These enhancements are **required**; omitting any will reduce Functionality marks. Demonstrate each one in the narrated demo video (5 min).

4 GitHub Workflow Requirement

- Create a public repo before writing any code with a clear README.
- Use at least one feature branch (e.g., feature/drag) and merge via Pull Requests.
- Record 8 meaningful commits (no single “final” dump).
- Comment on each PR and tag the final commit **v1.0.0** before submission.
- A weak commit history will cap your GitHub workflow credit.

5 D2L Submission Requirements

- Working GitHub link — tagged release URL (https://github.com/.../releases/tag/v1.0.0).
- Demo video **(at least five minutes)** — screen recording with narration showing:
  - User workflow (add, edit, drag, delete, export, quote fetch, image, timestamp, sorting)
  - Code structure (modules, class, storage).

No additional written report is required.

6 Marking Rubric (20 %)

Category	Criteria	Marks
Functionality	MVP + three enhancements operate as specified	10
Code quality	Clear naming, modular structure, comments	4
Demo video	5 min; audible narration; clear UI & code	6

7 Recommended Timeline

Week	Milestone
1	Repo + README; basic layout; add-note; start drag-and-drop
2	Complete drag-and-drop & delete; Note class; persistence
3	ES-modules refactor; quote fetch; error handling
4	Images, timestamp, sorting; JSON export; UI polish; record demo; tag release; submit

8 Technical Constraints & Advice

- Plain HTML, CSS, and JS only (no frameworks).

- Target evergreen browsers (Chrome, Edge, Firefox).
- Keep styles light; dark mode optional.
- Follow the Murach JavaScript style guide.
- Cite significant external code inspirations in the README.

9 Academic Integrity

Discuss ideas freely, but write your own code. Uncredited copying will trigger disciplinary procedures at the college level.

Questions? Post in the course forum or ask during the lab — and start your GitHub repo today!