OJT Project Design Template

Student Name(s): Anant Vardhan Pandey and Aditya Kuchiker
Roll No(s):
Year & Section: 2025 and 1A
Project Title (as assigned): Kanban-style sticky notes(drag and drop)
Project Type: (Product Developer / Application Developer / Open Source Developer)
Stack / Framework: HTML, CSS ,JS

---

1. Problem Understanding

1.1 What is the problem statement in your own words?
(Explain what you are trying to solve, not what you're trying to build.)

Teams and individuals often struggle to keep their tasks visually organized in a simple, intuitive way. A Kanban-style sticky notes board brings clarity to workflows by allowing tasks to move naturally across stages like 'To Do', 'In Progress', and 'Done'. This solves the need for a quick, flexible, visual task-tracking tool.

1.2 Why does this problem exist or matter?
(Who benefits from the solution — user, developer, community, etc.?)

Students, developers, and professionals all benefit from a clean system where tasks can be shifted with ease. A digital board ensures accessibility, reduces clutter, and mirrors the traditional physical sticky-note system that people trust and understand.

1.3 Key inputs and expected outputs:

Inputs	Process	Expected Outputs
User-created task cards, category/column selection	Create, drag, drop, edit, and delete task cards	Updated Kanban board reflecting current task stage
User interactions (click, drag, type)	DOM updates, event-based re-rendering	Visual task movement across columns

---

2. Functional Scope

2.1 What are the core features you plan to build (must-haves)?
(List 3–5 features you will definitely implement.)

- Create new sticky note tasks

- Drag & drop tasks across columns

- Edit and delete tasks

- Persistent board using localStorage

2.2 What stretch goals could you attempt if time permits?
(Features that would make it production-ready or App Store–ready.)

- User authentication

- Cloud-syncing tasks

- Color-coded priority tags

- Dark mode & theme personalization

2.3 Which libraries or tools will you use?
(Be specific — e.g., scikit-learn, React, Django ORM, Tailwind, etc.)

- JavaScript Drag & Drop API

- HTML/CSS for UI layout

- Optional: React.js, TailwindCSS

---

3. System & Design Thinking

3.1 Sketch or describe your app flow / pipeline:
(Draw boxes or bullet steps — e.g., Input → Processing → Output.)

User Input → Create Task Card → Store in Browser → Display on Board → Drag & Drop Interaction → Update Task State → Re-render Board

3.2 What data structures or algorithms are central to this project?
(e.g., arrays, trees, hash maps, nearest-neighbor search, regression tree.)

- Arrays to store lists of tasks

- Objects to represent each task

- DOM event listeners for drag & drop flow

- LocalStorage for persistence

3.3 How will you test correctness or performance?
(e.g., unit tests, metrics like accuracy, latency, etc.)

- Manual UI testing for drag-drop flow

- Checking DOM updates after card movement

- Ensuring persistence across refresh

---

4. Timeline & Milestones (4 Weeks)

Week	Planned Deliverables	Mentor Checkpoint
W1	Understand data / requirements / setup	☐
W2	Core logic / CRUD / model training	☐
W3	Evaluation / bug fixing / UI polish	☐
W4	Documentation / final demo / submission	☐

---

5. Risks & Dependencies

5.1 What's the hardest part technically for you right now?
(Be honest — data cleaning, unfamiliar library, deployment, etc.)

Mastering smooth drag-and-drop handling and ensuring tasks remain persistent across sessions is slightly challenging. UI edge-cases during dragging may also need careful debugging.

5.2 What dependencies or help do you need from mentors?
(E.g., feedback on model metrics, setup issues, data clarification.)

- Guidance on best drag-drop UX

- Suggestions for improving UI/UX polish

- Review of localStorage handling

---

6. Evaluation Readiness

6.1 How will you prove that your project "works"?
(Screenshots, metrics table, demo video, test cases, Deployment links and Git Links, etc.)

Screenshots of working board, video demo showing drag-drop, GitHub repo link, deployment link, and exportable test cases.

6.2 What success metric or goal will you aim for?
(E.g., accuracy > 90%, PWA Lighthouse > 85, 100% CRUD functionality.)

- 100% functional drag-drop flow

- Smooth card creation/deletion/editing

- No task loss after refresh

---

7. Responsibilities

7.1 Responsibilities

Task	Student 1	Student 2	Mentor Notes
Task 1	☐	☐	
Task 2	☐	☐	
Task 3	☐	☐	
Task 4	☐	☐	

Task 5	☐	☐	

Signatures (Students):
Mentor Approval:
Date: