PROJECT NOTES (JS-HTML-CSS)

BY : ABDELILLAH NAMIL

DD101

2021/2022

First i creat html :

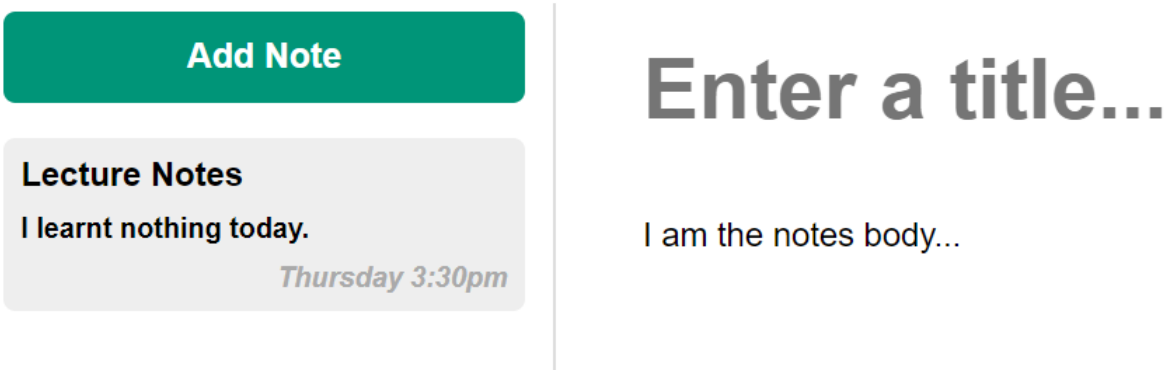
In html i creat div and i give them class-name .*I*nside the div i creat buttom i give them class name like divs with type of this buttom .

And in the last of this part html i creat text area we can write anything .

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible"
content="IE=edge">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Notes App</title>
    <link rel="stylesheet" href="css/notep.css">
</head>
<body>
    <div class="notes" id="app">
        <div class="notes__sidebar">
            <button class="notes__add"
type="button">Add Note</button>
            <div class="notes__list">
                <div class="notes__list-item
notes__list-item--selected">
                    <div class="notes__small-
title">Lecture Notes</div>
                    <div class="notes__small-body">I
learnt nothing today.</div>
```

```
                    <div class="notes__small-
updated">Thursday 3:30pm</div>
                </div>
            </div>
        </div>
        <div class="notes__preview">
            <input class="notes__title" type="text"
placeholder="Enter a title...">
            <textarea class="notes__body">I am the
notes body...</textarea>
        </div>
    </div>
    <script src="js/main.js" type="module"></script>
</body>
</html>
```

Seconde i creat css for designe web site (html) :




```
body {
    height: 100%;
    margin: 0;
}

.notes {
    display: flex;
    height: 100%;
}
```

```css
.notes * {
    font-family: sans-serif;
}

.notes__sidebar {
    border-right: 2px solid #dddddd;
    flex-shrink: 0;
    overflow-y: auto;
    padding: 1em;
    width: 300px;
}

.notes__add {
    background: #009578;
    border: none;
    border-radius: 7px;
    color: #ffffff;
    cursor: pointer;
    font-size: 1.25em;
    font-weight: bold;
    margin-bottom: 1em;
    padding: 0.75em 0;
    width: 100%;
}

.notes__add:hover {
    background: #00af8c;
}

.notes__list-item {
    cursor: pointer;
}
```

```css
.notes__list-item--selected {
    background: #eeeeee;
    border-radius: 7px;
    font-weight: bold;
}

.notes__small-title,
.notes__small-updated {
    padding: 10px;
}

.notes__small-title {
    font-size: 1.2em;
}

.notes__small-body {
    padding: 0 10px;
}

.notes__small-updated {
    color: #aaaaaa;
    font-style: italic;
    text-align: right;
}

.notes__preview {
    display: flex;
    flex-direction: column;
    padding: 2em 3em;
    flex-grow: 1;
}

.notes__title,
.notes__body {
```

```
    border: none;
    outline: none;
    width: 100%;
}

.notes__title {
    font-size: 3em;
    font-weight: bold;
}

.notes__body {
    flex-grow: 1;
    font-size: 1.2em;
    line-height: 1.5;
    margin-top: 2em;
    resize: none;
}
```

And the last one i creat js folder :

In this one he is name NotesAPI.js you can do edditing or update and you can save notes in the local storage .

```
export default class NotesAPI {
    static getAllNotes() {
        const notes =
JSON.parse(localStorage.getItem("notesapp-notes") ||
"[]");

        return notes.sort((a, b) => {
            return new Date(a.updated) > new
Date(b.updated) ? -1 : 1;
        });
    }

    static saveNote(noteToSave) {
```

```
        const notes = NotesAPI.getAllNotes();
        const existing = notes.find(note => note.id ==
noteToSave.id);

        // Edit/Update
        if (existing) {
            existing.title = noteToSave.title;
            existing.body = noteToSave.body;
            existing.updated = new
Date().toISOString();
        } else {
            noteToSave.id = Math.floor(Math.random() *
1000000);
            noteToSave.updated = new
Date().toISOString();
            notes.push(noteToSave);
        }

        localStorage.setItem("notesapp-notes",
JSON.stringify(notes));
    }

    static deleteNote(id) {
        const notes = NotesAPI.getAllNotes();
        const newNotes = notes.filter(note => note.id
!= id);

        localStorage.setItem("notesapp-notes",
JSON.stringify(newNotes));
    }
}
```

PART JS :

In this part of js he is name NOTESView.js you will able to delet any note .

```
export default class NotesView {
    constructor(root, { onNoteSelect, onNoteAdd,
onNoteEdit, onNoteDelete } = {}) {
        this.root = root;
        this.onNoteSelect = onNoteSelect;
        this.onNoteAdd = onNoteAdd;
        this.onNoteEdit = onNoteEdit;
        this.onNoteDelete = onNoteDelete;
        this.root.innerHTML = `
            <div class="notes__sidebar">
                <button class="notes__add"
type="button">Add Note</button>
                <div class="notes__list"></div>
            </div>
            <div class="notes__preview">
                <input class="notes__title" type="text"
placeholder="New Note...">
                <textarea class="notes__body">Take
Note...</textarea>
            </div>
        `;

        const btnAddNote =
this.root.querySelector(".notes__add");
        const inpTitle =
this.root.querySelector(".notes__title");
        const inpBody =
this.root.querySelector(".notes__body");

        btnAddNote.addEventListener("click", () => {
            this.onNoteAdd();
```

```
        });

        [inpTitle, inpBody].forEach(inputField => {
            inputField.addEventListener("blur", () => {
                const updatedTitle =
inpTitle.value.trim();
                const updatedBody =
inpBody.value.trim();

                this.onNoteEdit(updatedTitle,
updatedBody);
            });
        });

        this.updateNotePreviewVisibility(false);
    }

    _createListItemHTML(id, title, body, updated) {
        const MAX_BODY_LENGTH = 60;

        return `
            <div class="notes__list-item" data-note-
id="${id}">
                <div class="notes__small-
title">${title}</div>
                <div class="notes__small-body">
                    ${body.substring(0,
MAX_BODY_LENGTH)}
                    ${body.length > MAX_BODY_LENGTH ?
"..." : ""}
                </div>
                <div class="notes__small-updated">
                    ${updated.toLocaleString(undefined,
{ dateStyle: "full", timeStyle: "short" })}
```

```
                </div>
            </div>
        `;
    }

    updateNoteList(notes) {
        const notesListContainer =
this.root.querySelector(".notes__list");

        // Empty list
        notesListContainer.innerHTML = "";

        for (const note of notes) {
            const html =
this._createListItemHTML(note.id, note.title,
note.body, new Date(note.updated));

            notesListContainer.insertAdjacentHTML("befo
reend", html);
        }

        // Add select/delete events for each list item
        notesListContainer.querySelectorAll(".notes__li
st-item").forEach(noteListItem => {
            noteListItem.addEventListener("click", ()
=> {
                this.onNoteSelect(noteListItem.dataset.
noteId);
            });

            noteListItem.addEventListener("dblclick",
() => {
                const doDelete = confirm("Are you sure
you want to delete this note?");
```

```
                if (doDelete) {
                    this.onNoteDelete(noteListItem.data
set.noteId);
                }
            });
        });
    }

    updateActiveNote(note) {
        this.root.querySelector(".notes__title").value
= note.title;
        this.root.querySelector(".notes__body").value =
note.body;

        this.root.querySelectorAll(".notes__list-
item").forEach(noteListItem => {
            noteListItem.classList.remove("notes__list-
item--selected");
        });

        this.root.querySelector(`.notes__list-
item[data-note-
id="${note.id}"]`).classList.add("notes__list-item--
selected");
    }

    updateNotePreviewVisibility(visible) {
        this.root.querySelector(".notes__preview").styl
e.visibility = visible ? "visible" : "hidden";
    }
}
```