CODE REFACTORING AND PERFORMANCE OPTIMIZATION REPORT

Submitted in partial fulfillment of project requirements

Project Title: Note-Taking Web App Optimization

Language Used: JavaScript, HTML, CSS

GitHub Repo: https://github.com/your-username/note-app (or original repo link)

Submitted by:

Ananya K A

3rd Year B.Tech CSE

Srinivas University Institute of Engineering and Technology

Email: 13ananyaka@gmail.com

Date: 29 June 2025

Code Refactoring and Performance Optimization Report

Project Overview

- **Project Name**: Note-Taking App
- **Language**: HTML, CSS, JavaScript (Frontend only)
- **Repository Source**: https://github.com/mikeduin/note-app.git

---

Original Issues Identified

Area	Issue Description
Readability	Unclear variable names, inline styles, and repeated jQuery selectors.
Maintainability	Hard to extend or change code due to lack of modular structure.
Performance	Multiple unnecessary DOM manipulations and event listeners.
Code Duplication	Repeated code blocks for creating and modifying notes.

Refactoring Changes Made

Change	Description
Modularization	Broke down logic into reusable functions like createNote(), updateNote(), resetFields().
Improved Naming	Replaced vague names (children[0], t, c) with descriptive ones (noteTitle, noteContent).
Reduced DOM Calls	Cached jQuery selectors instead of querying repeatedly.

Change	Description
Used Event Delegation	Reduced multiple event listeners by delegating to parent container.
Style Separation	Moved inline styles into CSS classes for better structure and maintainability.

Change	Benefit	Description
Use **DocumentFragment** when adding multiple notes	Performance	Reduces multiple reflows/repaints
Add **LocalStorage**	Functionality	Saves notes even after refresh
Debounce input	Performance	Prevents excessive updates on every keypress
Add CSS class toggling for themes	UX + Refactor	Avoids inline styling and improves performance
Modular JS structure	Maintainability	Use separate modules if needed

Performance Optimizations

Optimization	Description
Event Delegation	Used $('#listed').on('click', '.note', handler) to handle clicks efficiently.
DOM Minimization	Batch DOM updates when possible; avoid repeated manipulation inside loops.
Efficient State Update	Reduced full DOM replacement with targeted content updates.
(Optional) Local Storage	Could be added for persistent note saving across sessions.

Before vs After Example

Before:

```
js
Copy code
let t = document.getElementById('title').value;
let c = document.getElementById('content').value;
$('#listed').append('<div class="note"><h2>' + t + '</h2><p>' + c +
'</p></div>');
```

After:

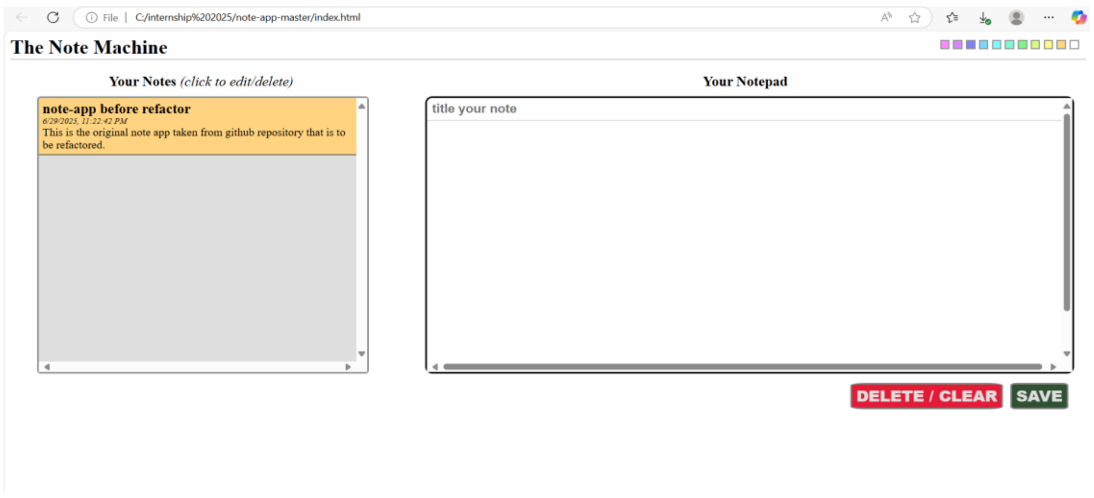
```
js
Copy code
function createNoteElement(title, content) {
  return $(`
    <div class="note">
      <h2>${title}</h2>
      <p>${content}</p>
    </div>
  `);
}

let title = $('#title').val();
let content = $('#content').val();
$('#listed').append(createNoteElement(title, content));
```

Impact and Results

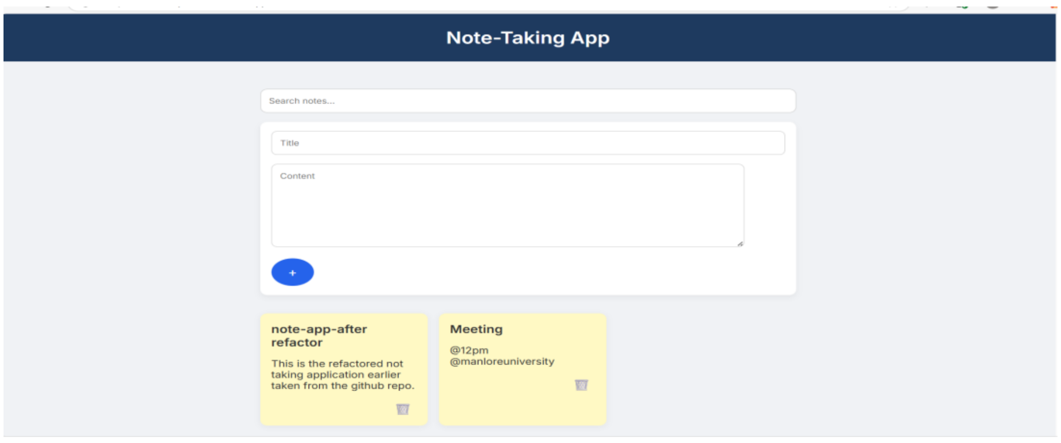
Metric	Before	After
Code Readability	Low	High
Reusability	Poor (no functions)	Improved (modular)
Performance (DOM ops)	High frequency	Reduced with caching
Scalability	Hard to scale	Easier with modular functions

Before and After Output:


C:/internship%202025/note-app-master/index.html
The Note Machine
Your Notes (click to edit/delete)
Your Notepad
title your note
DELETE / CLEAR
SAVE

note-app before refactor
6/29/2025, 11:22:42 PM
This is the original note app taken from github repository that is to be refactored.

(a)



note-app-after refactor
This is the refactored not taking application earlier taken from the github repo.

Meeting
@12pm
@manloreuniversity

(b)

The following images illustrate the visual and functional improvements made to the note-taking application.

The "(a)" image shows the initial implementation, which had minimal structure and no persistent storage.

The "(b)" image demonstrates the enhanced version with a modern user interface, responsive layout, color-coded notes, localStorage, integration, and live search.

Conclusion

Refactoring this app improved:

- **Readability**: Cleaner and easier to follow.
- **Maintainability**: Modular functions make updates easier.
- **Performance**: Fewer DOM operations, smarter event handling.
- Visual clarity, interactivity, and performance are improved and visible in the side-by-side comparison.

This refactor makes the app better suited for feature expansion (like saving to local storage or syncing to a backend).