PROJECT ABSTRACT

Markdown Editor Desktop App with Live Preview

A distraction-free desktop Markdown editor with live preview, file sidebar, formatting toolbar, snippet library and one-click export to PDF, HTML and DOCX — built-to-order with source code, report, PPT and viva kit.

01. Title

Markdown Editor Desktop App with Live Preview

02. Introduction / Background

Engineering students write an enormous amount of Markdown-adjacent text — project reports, synopses, weekly logs, viva notes, README files — usually in a plain text editor with no idea how the rendered document will look, or in a heavyweight word processor that fights them on formatting. Markdown hits the sweet spot: plain-text source that is version-control friendly, with clean rendered output. This project builds a desktop Markdown editor around that workflow: a three-pane layout (file sidebar, source editor, live preview), a formatting toolbar that inserts markup without requiring memorized syntax, a snippet library for repeated structures like report templates and tables, and one-click export to PDF, standalone HTML and DOCX for university submission formats.

03. Problem Statement

Students writing project documentation toggle between raw Markdown in a text editor and a separate preview, or surrender to word processors whose formatting breaks across machines. None of the lightweight editors bundle the student-specific extras — report templates, export to DOCX for university formats, word counts for report requirements — in one offline desktop tool. This project fills that gap.

04. Objectives

- Build a Markdown source editor with syntax highlighting and line numbers.
- Render a live preview pane that updates as the user types (debounced).
- Provide a file sidebar for managing a folder of documents with quick switching.
- Implement a formatting toolbar (bold, italic, headings, lists, code, quote, table, image) that inserts markup at the cursor.
- Ship a snippet library with report, table and code-block templates.
- Implement export to PDF (print-ready), standalone HTML and DOCX.
- Show live word/character counts in the status bar.

05. Proposed Methodology

The editor is built on a standard Markdown parsing pipeline: source text → abstract syntax tree → rendered HTML for the preview pane. Typing is debounced (approximately 300 ms) before re-rendering so large documents stay responsive; only the preview DOM is patched, not rebuilt, on each keystroke. The toolbar operates on the editor’s selection model — wrapping the selection in markup or inserting block templates at the caret. Export reuses the same AST:

PDF via a print stylesheet in a headless renderer, HTML as a single file with embedded CSS, DOCX via document-object mapping of the AST nodes. Files are plain .md on disk, so the whole folder works with git.

06. System Architecture / Working

Three components: (1) Editor core — text buffer, syntax highlighting, selection-aware toolbar actions; (2) Render pipeline — Markdown parser to AST to live HTML preview with scroll-sync between panes; (3) Project layer — file sidebar, snippet library, export module and status bar. The working flow is: open a document folder → edit in the source pane → watch the live preview → insert snippets/templates as needed → export to PDF/HTML/DOCX for submission.

07. Hardware / Software Requirements

- Windows/macOS/Linux desktop or laptop — no special hardware
- Electron or Tauri (web-tech desktop shell), or Python with a Markdown library + Qt
- Markdown parser, HTML sanitizer for the preview pane
- Headless browser/PDF engine for PDF export; DOCX library for Word export
- Fully offline; no account or internet needed

08. Expected Outcomes

A working desktop app in which a multi-section project report can be written in Markdown, previewed live with tables, code blocks and quotes rendering correctly, and exported to PDF, HTML and DOCX. Export fidelity is verified against the sample documents shipped with the project; rendering targets are stated as design targets.

09. Applications

- Writing final-year project reports and synopses
- Maintaining weekly project logs and viva notes
- Authoring README files and software documentation
- Classroom demonstration of parsing pipelines

10. Future Scope

- Split-pane scroll synchronization
- Math (LaTeX) and diagram (Mermaid) rendering
- Git integration with diff view for .md files
- Distraction-free typewriter mode