PROJECT ABSTRACT

Online Code Editor with Live Preview

A browser-based code editor with file tabs, syntax highlighting, an execution console, a live HTML preview pane, a snippet library and run history. Delivered built-to-order with working demo, source code, report, PPT and viva kit.

01. Title

Online Code Editor with Live Preview

02. Introduction / Background

Online editors and playgrounds are how most students first encounter a new language, yet building one is an excellent systems exercise: tokenizing source for highlighting, sandboxing untrusted preview output, and managing editor state. This project implements CodeForge, a browser-based code editor with file tabs (HTML/CSS/JS), regex-based syntax highlighting for JavaScript, a simulated execution console with realistic output, a live preview pane that renders HTML in a sandboxed iframe, a snippet library for inserting common patterns, keyboard shortcuts, and a run-history table. The preview pane genuinely executes the embedded demo page, so the edit-to-preview loop is real, not mocked.

03. Problem Statement

Students use online editors daily but rarely understand what happens between typing code and seeing output. This project makes that pipeline explicit and demonstrable: source text is tokenized into highlighted spans, execution results are captured into a console model, and preview HTML is isolated in an iframe sandbox — each stage visible in the UI and explainable at the viva.

04. Objectives

- Implement multi-file tab editing with per-file language labels.
- Highlight JavaScript syntax via a tokenizer (keywords, strings, numbers, comments).
- Render a live preview of HTML output in a sandboxed iframe.
- Provide a snippet library, shortcut reference and run-history views.
- Keep the editor fully client-side in one file with no backend.

05. Proposed Methodology

The build has three parts. First, the editor pane: a pre-formatted code region fed by a small tokenizer that wraps keywords, strings, numbers, comments and function names in styled spans. Second, the execution model: a console panel showing realistic run output (program results, timing, exit codes) for the demo program, plus a run-history table. Third, the preview system: an iframe with a srcdoc document that genuinely runs an interactive Fibonacci visualizer, alongside the snippet library and shortcuts views.

06. System Architecture / Working

- Editor layer: tokenizer producing highlighted spans from raw source text.
- Console layer: execution result model rendered as terminal-style output.
- Preview layer: sandboxed iframe running the demo page independently.
- Library layer: snippet cards and shortcut tables as static content views.
- Tab bar switches files; view tabs switch between editor, snippets and history.

07. Software Requirements

- Any modern web browser; no server, build step or dependencies.
- Static hosting for deployment.
- Text editor for source study.
- Optional: Node.js if the student extends the demo toward real server-side execution.

08. Expected Outcomes

- A working editor with highlighted JS source and a genuine live preview.
- Console output and run history that mirror real tooling behavior.
- Snippet library and shortcut reference views.
- Single-file, dependency-free source a student can walk through in a viva.

09. Applications

- Teaching aid for how editors, highlighting and sandboxes work.
- Starter template for college coding-practice portals.
- Portfolio project showing developer-tool UI design.
- Base for a real playground with server-side execution.

10. Future Scope

- Real multi-language execution via a sandboxed backend API.
- User accounts with saved files and shareable links.
- Collaborative editing with operational transforms.
- Integrated debugger and linting.