

BasedShell Project Overview

Detailed technical and product summary generated from repository state on main.

Generated: 2026-02-12 10:18

| | |
|--------------------------------|-----------------------------------------------------------|
| Project | BasedShell |
| Version | 1.0.0 |
| Platform Focus | macOS desktop terminal application |
| Core Stack | Electron 31, node-pty 1.0, xterm.js 5.5, TypeScript, Vite |
| Source Footprint | 20 source files, about 7,123 lines in src/ |
| Packaging | electron-builder targets: dmg, zip |
| Primary Runtime Windows | Main terminal window + standalone settings window |

Executive Summary

BasedShell is a production-grade, keyboard-first terminal built as a native-feeling macOS app. The project has moved beyond a basic terminal container and now includes multi-tab orchestration, runtime Git telemetry, command palette workflows, inline search, toast notifications, persistent preferences, and a dedicated settings window with sectioned navigation. The current implementation emphasizes reliability and architectural guardrails, including DOM contract validation, typed cross-process APIs, settings schema versioning, and explicit theme metadata.

Current Product Surface

- Terminal sessions backed by node-pty with login-shell profile support.
- Multi-tab UX with activity states (active, unread output, exited) and keyboard shortcuts.
- Status HUD with shell, cwd, Git branch/dirty state, command context, tab count, and theme segment.
- Command palette with fuzzy matching, pinned actions, and recents.
- Standalone settings window with section-based panels and live preview.
- Theme system spanning terminal ANSI colors and UI chrome tokens, including Catppuccin flavors.

Architecture Overview

The application uses a three-layer Electron architecture with strict type sharing and focused responsibilities per layer.

| Layer | Location | Primary Responsibilities |
|------------------|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| Main Process | src/main/ | Window lifecycle, IPC handlers, settings persistence, session manager orchestration, system appearance, Git status resolution, menu wiring |
| Preload Bridge | src/preload/preload.ts | Context-isolated, typed API surface (invoke/send/subscribe) exposed as window.terminalAPI |
| Renderer UIs | src/renderer/ | Terminal UI, tabs, search, command palette, toasts, settings window, theme application |
| Shared Contracts | src/shared/ | Cross-process types, settings schema, theme metadata, appearance resolution |

IPC Contract (Main Channels)

| Type | Channel | Purpose |
|--------|---------------------------------------------------------------|---------------------------------------------------------|
| invoke | app:get-version, app:get-home-directory | Environment and app metadata |
| invoke | system:get-appearance | Current OS dark/light state |
| invoke | git:status | Repo root, branch, dirty flag for active cwd |
| invoke | settings:get / settings:update | Read and persist typed application settings |
| invoke | settings:open-window | Open or focus the standalone settings window |
| invoke | terminal:create-session | Create PTY session from selected profile and cwd |
| send | terminal:write / terminal:resize / terminal:close-session | Session IO and lifecycle control |
| event | terminal:data / terminal:exit / terminal:context | PTY output, exit notifications, cwd/ssh context updates |
| event | settings:changed / menu:action / system:appearance-changed | Cross-window sync and command dispatch |

Session Management Notes

- PTY sessions are created through node-pty with clamped cols/rows and validated cwd.
- Runtime environment is sanitized to remove npm-injected variables that can break shell behavior (notably `npm_config_prefix` leakage).
- On macOS, spawn-helper execute bit is proactively repaired to prevent `posix_spawnp` failures.
- Session context polling updates cwd and ssh host hints to keep tab titles and telemetry accurate.

Renderer and UX Capabilities

Main Terminal Window

- Tab strip with overflow handling, density modes, enter/exit animations, unread and exited state indicators.
- Repository-aware tab labels using Git repo/branch context when available, with fallback path labels and SSH host prefixes.
- Status bar segments act as interactive controls with contextual tooltips and state coloring.
- Search UX supports case-sensitive and regex toggles, directional next/previous navigation, and result counters.
- Command palette supports fuzzy ranking, pinning, recents, and keyboard-centric action execution.
- Toast system provides non-blocking notifications and ARIA-compatible announcements.

Standalone Settings Window

- Dedicated BrowserWindow with hiddenInset title bar on macOS and titlebar-safe header spacing.
- Left-side section navigation where only one section panel is displayed at a time on the right.
- Live preview path applies theme and opacity before save; Cmd/Ctrl+S persists changes.
- Cross-window consistency via settings:changed events; main and settings windows stay synchronized.

Theme and Appearance System

| Category | Available Options |
|-----------------------|-------------------------------------------------------------------------|
| Core Themes | graphite, midnight, solarized-dark, paper, aurora, noir, fog |
| Catpuccin Flavors | catpuccin-latte, catpuccin-frappe, catpuccin-macchiato, catpuccin-mocha |
| Meta Selection | system (maps by OS appearance through shared theme metadata) |
| Appearance Preference | system, dark, light |
| Vibrancy | Optional under-window vibrancy on macOS |

Engineering Quality Guardrails

- DOM contract check script validates required assertDom selectors against renderer HTML IDs to prevent mixed-state UI crashes.
- Settings schema includes an explicit schemaVersion and sanitization logic for bounds-safe numeric values and profile fallback behavior.
- Main process enforces single-instance lock and persists window state for reliable app relaunch behavior.
- Typed shared contracts reduce IPC drift between main, preload, and renderer layers.

Build, Packaging, and Operations

Development and release workflow is script-driven and optimized for local iteration.

| Command | Purpose |
|----------------------------------|----------------------------------------------------------------------------------------------------|
| <code>npm run dev</code> | Build main once, watch main process TS, run Vite dev server, and launch Electron with renderer URL |
| <code>npm run typecheck</code> | Runs DOM contract check + TS noEmit for main and renderer configs |
| <code>npm run build</code> | Clean + compile main + production bundle renderer |
| <code>npm run start</code> | Build and launch packaged runtime locally |
| <code>npm run package:mac</code> | Generate macOS distributables (dmg and zip) through electron-builder |

Top Source Modules by Size

| File | LOC | Responsibility |
|---------------------------------|-------|-------------------------------------------------------------------------------|
| src/renderer/main.ts | 1,923 | Primary terminal UX controller, tab lifecycle, search, status HUD, shortcuts |
| src/renderer/styles.css | 1,088 | Core main-window styling and theme-token-driven visual rules |
| src/renderer/themes.ts | 830 | Theme definitions and chrome token mapping including Catppuccin flavors |
| src/renderer/command-palette.ts | 473 | Action registry, fuzzy ranking, pins, recents, keyboard behavior |
| src/main/session-manager.ts | 392 | PTY spawn, IO routing, context polling, environment sanitation |
| src/main/main.ts | 379 | Window lifecycle, IPC registration, settings sync, OS appearance integration |
| src/renderer/settings.ts | 351 | Standalone settings window state, preview, persistence, nav section switching |

Recommended Next Milestones

- Finalize and polish PR9 task presets plus smart history workflows in command palette.
- Execute PR10 release gate: accessibility pass, reduced-motion behavior validation, and cross-theme QA.
- Add automated renderer tests for critical window flows (settings save/reset, theme sync, tab lifecycle).
- Add a small docs page for IPC channel contracts and renderer lifecycle expectations for future contributors.

End of report.