# Umicom Studio IDE — Milestone Review

\*Created by: \*\*\*\*Umicom Foundation\*\*\*\* · Author: \*\*\*\*Sammy Hegab\*\*\*\* · Date: \*\*\*\*2025‑10‑02\*\*\*\* · License: \****MIT***

## Executive Summary

We have established the core application plumbing (actions/menu/keymap, settings bus, live theming and font updates), a modern output pipeline (ANSI console with filters), and a practical developer experience (ripgrep search, build/run/test façade, and project manager). Tooling is in place—Meson build files, CI starter, Windows/MSYS helpers, and convenience scripts. Starting with Part 26 we switched from size padding to **curated, useful‑only** drops containing real code, scripts, build configurations (Meson/Ninja), CI workflows, onboarding guides, contributor checklists, style and formatting configs, sample projects, and planning assets, documentation, wireframes, and reference PDFs.

## Delivered Components

### Core & UI Glue

* **Output console** with colour/filter chain; **adapter** to keep legacy UmiOutputPane compiling.
* **Actions / Menu / Keymap** via app‑level GSimpleActions.
* **Settings bus** with **theme\_live** and **font\_live** (runtime updates).
* **Preferences UI** bound to **json\_store**; emits live settings changes.
* **Command Palette (stub)** with UTF‑8 case‑insensitive filtering.
* **Search panel** powered by **ripgrep** (--vimgrep).
* **Build tasks** façade wiring runner → pipeline → console.
* **Project manager** (open folder, index, MRU, status line).
* **Watcher glue** to refresh the file tree on filesystem events.

### Platform & Tooling

* **Meson** root and src/meson.build (append‑safe source lists).
* **Windows/MSYS** PATH hint + launcher; **options** (GOption), **logging** (levels), **crash‑guard**, and **profiler** timers.
* **Test scaffold** for lightweight unit tests.
* **CI (Linux)** workflow; sample project samples/hello\_cgtk/.
* **Scripts** for setup/build and commit helpers.

### Documentation & Assets

* Markdown guides: **DEV\_SETUP**, **NEW\_CONTRIBUTORS**, **FEATURES**, **MILESTONES**, **RUNBOOK**, **STYLE**, **DEV\_GUIDE**.
* Reference PDFs mirrored under docs/reference/.
* Wireframe SVGs for the main window, search, preferences, build console, and grid explorations.

## Strengths

* Clear modular layering; **heavily commented** code and newcomer‑friendly APIs.
* Adapter patterns minimise breakage while we modernise subsystems.
* Cross‑platform groundwork (MSYS hints, launcher, scripts).
* Solid build story: Meson + scripts + CI starter; sample app validates the toolchain.
* From #26 onward, drops contain **useful content only**—no filler.

## Gaps & Risks

* **Palette & menus**: commands are not yet fully surfaced in the main menu/toolbar.
* **Search streaming**: current read‑all approach; should stream with back‑pressure.
* **Build UX**: expose Build/Run/Test via UI (buttons/palette/menus).
* **Windows CI**: add MSYS2 job; verify GTK/json‑glib/ripgrep availability.
* **Formatting policy**: missing repo‑level .clang-format and .editorconfig.
* **Tests**: scaffold exists, but few concrete unit tests are committed.
* **Repository weight**: heavy SVG/PDF assets—consider keeping under docs/ or using LFS later.

## Plan for Part 29

### Option A — Wire‑up & Polish *(recommended next)*

1. **Menus & Palette**
   * Add menu actions: *Preferences…*, *Command Palette…*, *Build*, *Run*, *Test*.
   * Register default palette commands (Open Folder; Build/Run/Test; Toggle Theme; Font ±; Open Settings JSON).
2. **Build UX**
   * Add a toolbar or bottom‑bar with Build/Run/Test mapped to umi\_build\_tasks\_\* and routed to the console.
3. **Streaming Search**
   * Switch umi\_search\_panel to incremental stdout reading; append rows live; hard cap for row count.
4. **Formatting & Lint**
   * Add .clang-format, .editorconfig, and scripts/format.{sh,ps1}; optional Meson format target.
5. **Windows CI**
   * MSYS2 workflow: install deps; build sample + main app.

### Option B — Testing & Robustness

* Unit tests for utf8, json\_store, options, and rg\_runner (spawn/exit handling).
* Crash‑guard toggle via environment/option; verify logs are routed to the console in dev builds.

### Option C — Packaging & Onboarding

* Meson install rules; desktop metadata; headless‑mode docs; richer **DEV\_GUIDE**.

## Acceptance Checklist for Part 29

## Next Steps

Please choose a track for Part 29:

* **A: Wire‑up & Polish** (menus/palette/build UX + search streaming + formatting + Windows CI)
* **B: Testing & Robustness** (unit tests + crash‑guard toggle)
* **C: Packaging & Onboarding** (install rules + richer docs)