# Cursor vs GitHub Copilot vs Claude Code for Android — What Actually Works

By **Bhavik Sadiwala** · Staff Engineer Android | AI Agents & Automation · Procore Technologies

I have used all three daily for over a year on a large production Android codebase — not toy projects, not tutorials, real Kotlin with Clean Architecture written by 20 engineers over five years. Here is what I actually think.

> **Use all three. They are not competitors — they solve different problems at different points in your workflow.**

## Cursor — Best for Deep Codebase Work

Cursor's biggest advantage is codebase indexing. When you ask it to add error handling that follows the same pattern as an existing UseCase, it actually reads that UseCase and replicates the pattern. Generic AI tools invent a pattern. Cursor finds yours.

### Where it shines for Android

- Refactoring across multiple files (ViewModel, UseCase, Repository in one operation)
- Understanding and extending existing architecture patterns in a large codebase
- Writing Jetpack Compose UI that matches your existing component library

### My usage pattern

I use Cursor for anything that requires understanding the existing codebase. New features, refactors, architecture changes — anything where context matters.

## GitHub Copilot — Best for In-Flow Completion

Copilot lives in Android Studio. No context switching. You are writing Kotlin, it completes your Kotlin.

### Where it shines for Android

- Boilerplate that follows predictable patterns (Room DAOs, Retrofit interfaces, Hilt modules)
- Test writing — once you write the first test, Copilot generates the rest in the same structure
- Documentation comments — it reads the function signature and writes the KDoc

### My usage pattern

Copilot is always on. It handles the mechanical parts of coding. I accept roughly 40% of its suggestions without modification.

## Claude Code — Best for Agentic Tasks

Claude Code runs in the terminal and can read files, write files, run commands, and chain operations. It is not a code completion tool — it is an agent that executes multi-step tasks.

### Where it shines

- "Find all places we are using deprecated API X and update them to API Y" across the entire codebase
- "Set up the boilerplate for a new feature module following our existing module structure"
- Generating migration scripts, updating dependencies, bulk refactors

## Real Numbers From the Team

- Boilerplate writing time: down ~60% (Copilot)
- Cross-file refactoring time: down ~70% (Cursor)
- Bulk codebase updates: down ~80% (Claude Code)
- Overall PR cycle time: down ~40%

## Where to Start

Start with Copilot — lowest friction, installs in Android Studio, immediate value. Then add Cursor for engineers working on complex features or refactors. Add Claude Code last for specific agentic tasks. Do not try to adopt all three at once — the learning curve for each is real.

*Bhavik Sadiwala* is a Staff Engineer Android at Procore Technologies, specialising in AI-powered mobile features and automation. bhaviksadiwala.com | LinkedIn