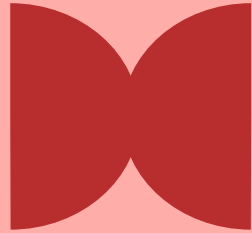


Mobile App Development with AI using Claude Code



Málaga AI



Olmo Gallegos Hernández

Structure

Part 1 Introduction - Concepts, definitions



Part 2 Basic Claude Code setup for Mobile App development



Part 3 Improvements and refinements

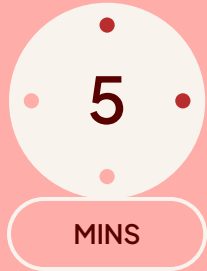


Part 4 Credits and references



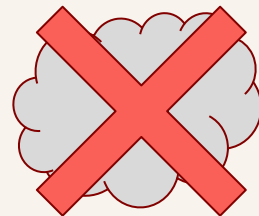
Part 1

Introduction - Definitions



“Claude”

Disclaimer



CLI - Command Line Interface

- GitHub CLI (gh)
- GitLab CLI (glab)
- Claude Code
- git
- Tip: Install as many as possible

Anthropic - The company behind Claude

The logo for Anthropic, featuring the word "ANTHROPIC" in a bold, white, sans-serif font, centered within a dark gray rectangular box.

Markdown

- Lightweight markup language
- Files with .md extension

Claude Code - Coding assistant



LLM, or simply "Model"

- Haiku 4.5
- Sonnet 3.5
- Sonnet 4
- Sonnet 4.5
- OPUS 4.6

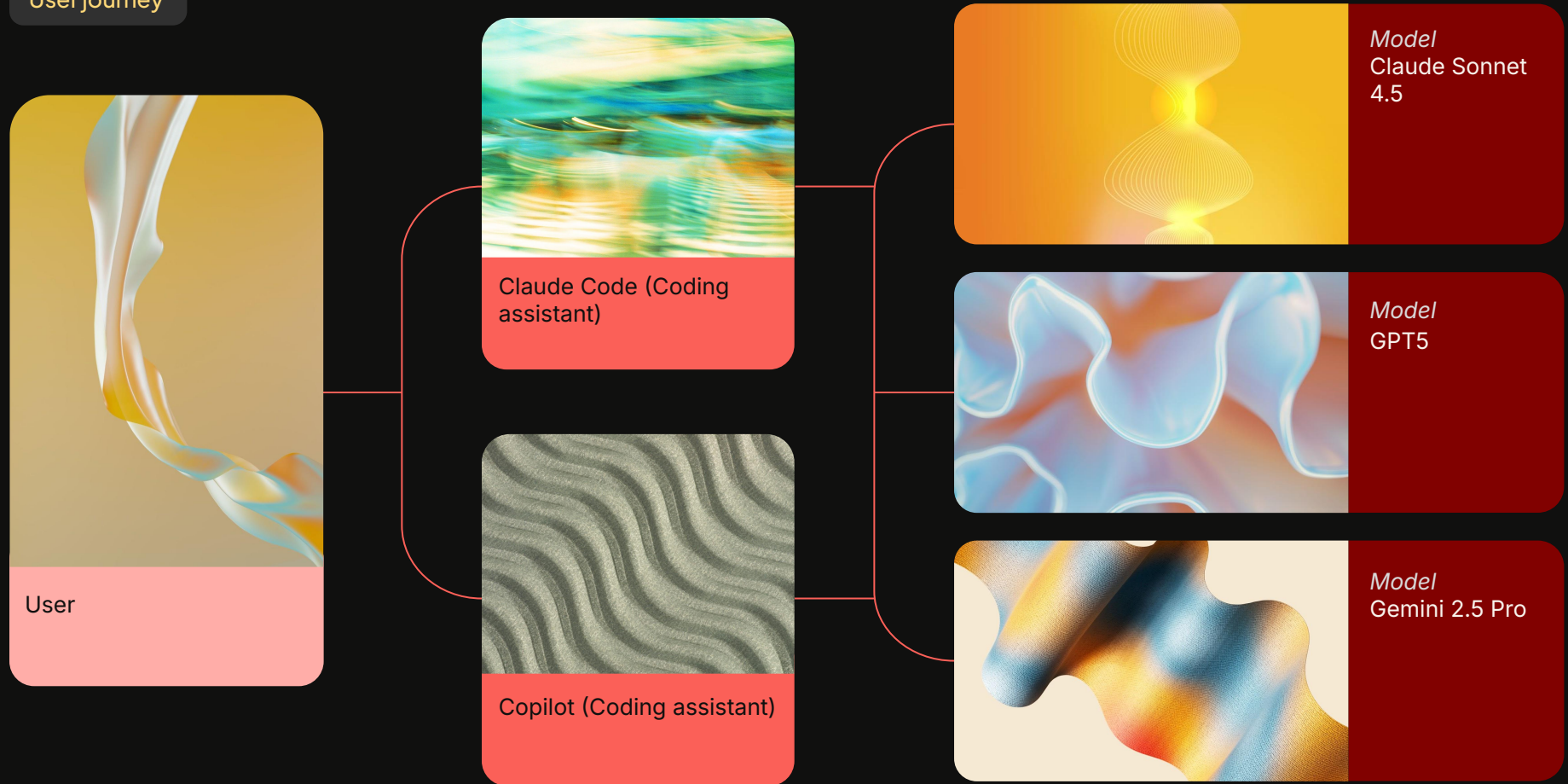
The logo for Claude Sonnet 4.5, featuring a small orange starburst icon followed by the text "Claude Sonnet 4.5" in a white, sans-serif font, all contained within a light gray rectangular box.

MCP - Model Context Protocol

- Connect LLM to APIs
- Access external resources
- Example: Shortcut, Jira, Gitlab



User journey



Part 2

Basic Claude Code setup



Installing Claude Code

In order to install Claude code, open your terminal and execute:

```
curl -fsSL https://claude.ai/install.sh | bash
```

Get your API Key from Anthropic.

Authenticate.

<https://code.claude.com/docs/en/setup>

Pro tip

Create a Shortcut to create a command for executing claude in YOLO mode:

```
alias claude-yolo='claude --dangerously-skip-permissions'
```

Add this to .zshrc or .bashrc



Example CLI usages

Opening a PR using GitHub CLI

```
gh pr create --title "Initial project and Claude Code setup" --body "..."
```

Opening an MR using glab (GitLab CLI)

```
glab mr create --source-branch bump/2.9.2.3-open --target-branch develop --title "..." --description "..."
```

Listing the available MRs (GitLab CLI)

```
glab mr list
```

Invoking Claude Code with a prompt

```
claude "find the code where we manage push notifications"
```

Starting Claude code

```
claude
```



Claude Code: Three modes

- Bypass permissions mode (YOLO)
 - Plan Mode
 - Accept Edits mode
 - Normal mode
-
- Shift+Tab to switch between them

```
* Implementing MVVM-C architecture... (5m 11s · ↓ 12.4k tokens)
```

```
└─ ■ Create branch and implement MVVM-C architecture
```

```
> █
```

```
▸▸ bypass permissions on (shift+tab to cycle) · 5 files +0 -0 · PR #2 · esc to interrupt · ctrl+t to hide tasks
```



Plan Mode

- Claude Code will do more research about our project's contents, will read more files, and will come up with a detailed plan about how to complete the task.
- We can either accept the plan or add more context and keep on planning.
- Plan mode is better for tasks that require a wider understanding of the project, or require multiple steps to complete.



CLAUDE.md

Markdown file where **local** context for this project will be provided to Claude Code

- Automatically read by Claude Code on each session
- Created manually or when executing **/init**
- Located either in **project root folder** or in **.claude/CLAUDE.md**
- Global context for all projects: **/home/user/.claude/CLAUDE.md**
- Local context, not committed: **CLAUDE.local.md**



Useful commands



`/context`

Shows current context that Claude Code has



`/clear`

Clears current context



`/compact`

Compacts all available context into one single prompt in order to reuse it. The context will be reduced to one single summarized prompt.



`/model`

Displays all available models, the one we're working with, and allows us to select a different model

Useful commands and keys



`/review`

Reviews a PR providing a link



`#`

Switches to Memory Mode



`@`

Allows us to refer files



`/fast`

Toggles Fast mode (Opus 4.6) -
Released Feb 2026

Escape key

Cancels current prompt

- Very useful to correct mistakes or avoid mistakes
- `#` enters memory mode and can add additional context
- Double Escape goes back in conversation with Claude Code



Part 2

Basic Claude Code setup: Summary

15

MINS

Basic Claude Code setup

Run Claude Code in YOLO mode

Add a CLAUDE.md file to your repository

Install all necessary MCPs and CLIs

Use Esc key, Double Esc, use Memory mode

Part 3

What refinements can we do
to this basic setup?



Subagents

.claude/agents

- android-developer.md
- issue-triager.md



Commands

Claude will include them in the available command list

`.claude/commands`

- `create-pr.md`
- `audit-code-quality.md`



Hooks

Executed pre or post:

- Opening a file
- Entering a prompt
- Starting the session with Claude Code

Example: PreToolUse, PostToolUse

- Which tools?
- “Name a list of the tools you have access to”

`.claude/hooks`

- `run-ktlint.md`



GitHub integration

An improvement over using GitHub CLI

- Allows us to mention @claude in issues or comments
- /install-github-app

```
> /install
```

```
/install-slack-app
```

```
Install the Claude Slack app
```

```
/install-github-app
```

```
Set up Claude GitHub Actions for a repository
```

```
/doctor
```

```
Diagnose and verify your Claude Code installation and settings
```



Part 4

Credits



MINS

Claude Code 101 - Course by Anthropic

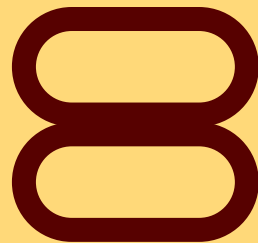
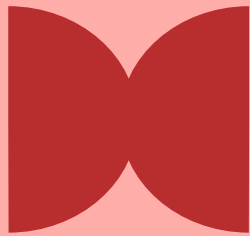
Claude Code in action - Course by Anthropic

www.github.com/pedrovg/MyClaude

www.github.com/voghDev/KataLLMAndroid

www.github.com/voghDev/KataLLMiOS

Demo Time



Málaga AI



Olmo Gallegos Hernández

Questions or feedback?

olmo.gallegos@domestika.org



Olmo Gallegos

Thank you!

