

AI Tools Lab

Hands-on classes on Claude, Claude Code, Lovable, Replit, Figma, Notion, PRD tools, Zapier, Make, n8n, business chatbots and voice agents, from basic to advanced



AUTHOR

Varun Kumar

Product Management Master Handbook

v0.1 · 23 October 2026

Built from the 55-lecture course study guide and extended to the 12-module Product Core curriculum. Hypothetical examples are labelled; no real company results are invented.

Contents

Module 13: AI Tools Landscape and Your Starter Stack	6
13.1 How to use the AI Tools Lab	6
13.2 The AI tools map	7
13.3 Which tool for which job	9
13.4 What it costs: free versus paid	10
13.5 What you need before you start	12
13.6 Accounts, API keys and safety	12
13.7 A starter stack for three budgets	14
13.8 How the classes connect: one example project	14
13.9 Exercises with model answers	15
13.10 Key takeaways	15
13.11 Check your understanding	16
13.12 Recommended resources	16
Module 14: Claude for Daily PM Work: Basic to Advanced	17
14.1 What Claude is and where you can use it	17
14.2 Level 1 · Your first hour with Claude	18
14.3 Level 2 · Features working PMs use every week	20
14.4 Level 3 · Connect Claude to your apps	22
14.5 A daily routine for PMs	24
14.6 Labs	25
14.7 Prompt library for PMs	27
14.8 Common mistakes	27
14.9 Where to find official help and sources	27
14.10 Exercises with model answers	28
14.11 Key takeaways	28
14.12 Check your understanding	29
14.13 Recommended resources	30
Module 15: Claude Code: Basic to Advanced	31
15.1 What Claude Code is	31
15.2 Level 1 · Install and your first session	33
15.3 Creating a repository and pushing to GitHub	35
15.4 Level 2 · Working like a professional	36
15.5 Level 3 · Extending Claude Code	38
15.6 Costs	40
15.7 Labs	41
15.8 PM use cases	43
15.9 Common mistakes	43
15.10 Exercises with model answers	43
15.11 Key takeaways	44
15.12 Check your understanding	45
15.13 Recommended resources	46

Module 16: Lovable: Basic to Advanced	47
16.1 What Lovable is	47
16.2 The editor	48
16.3 Level 1 · Your first app in 30 minutes	48
16.4 Level 2 · Building like a pro	49
16.5 Level 3 · Database, login, AI, payments and GitHub	51
16.6 Labs	54
16.7 Prompt library	55
16.8 Saving credits	55
16.9 Common mistakes	55
16.10 Exercises with model answers	56
16.11 Key takeaways	56
16.12 Check your understanding	57
16.13 Recommended resources	57
Module 17: Replit and Other AI App Builders	58
17.1 Replit in one picture	58
17.2 Level 1 · Build your first app	59
17.3 Level 2 · Data, secrets and quality	60
17.4 Level 3 · Deploy, schedule and integrate	60
17.5 Labs	61
17.6 The other AI app builders	62
17.7 Common mistakes	63
17.8 Exercises with model answers	63
17.9 Key takeaways	64
17.10 Check your understanding	64
17.11 Recommended resources	64
Module 18: Figma and AI Design Tools	65
18.1 Figma's products	65
18.2 Level 1 · The interface and your first screens	66
18.3 Level 2 · Prototypes, reviews and handoff	67
18.4 Level 3 · Figma AI, Figma Make and MCP	68
18.5 Labs	70
18.6 PM best practices	71
18.7 Exercises with model answers	71
18.8 Key takeaways	71
18.9 Check your understanding	72
18.10 Recommended resources	72
Module 19: Notion, PRD Tools and AI Documentation	73
19.1 Notion in one page	73
19.2 Level 1 · Pages, blocks and databases	74
19.3 Level 2 · A product hub that runs itself	75
19.4 Level 3 · Notion AI and connecting to Claude	76
19.5 PRD tools compared	77

19.6 An AI-assisted PRD workflow	78
19.7 Labs	78
19.8 Common mistakes	79
19.9 Exercises with model answers	80
19.10 Key takeaways	80
19.11 Check your understanding	81
19.12 Recommended resources	81
Module 20: Automation with Zapier, Make and n8n	82
20.1 The building blocks	82
20.2 Choosing the tool	85
20.3 How to connect Gmail and Slack (step by step)	86
20.4 Zapier · Basic to advanced	87
20.5 Make · Basic to advanced	88
20.6 n8n · Basic to advanced	89
20.7 Labs	91
20.8 Reliability and safety	93
20.9 Exercises with model answers	94
20.10 Key takeaways	94
20.11 Check your understanding	95
20.12 Recommended resources	96
Module 21: AI Chatbots for Business: WhatsApp, Slack and Web	97
21.1 What a business chatbot does	97
21.2 WhatsApp: App or Platform?	98
21.3 How an AI WhatsApp bot works	100
21.4 No-code options	102
21.5 Slack, Telegram and website bots	102
21.6 Design the bot before you build it	103
21.7 Guardrails, privacy and quality	103
21.8 Costs at a glance (illustrative)	104
21.9 Labs	104
21.10 Exercises with model answers	105
21.11 Key takeaways	106
21.12 Check your understanding	107
21.13 Recommended resources	107
Module 22: Voice Agents for Businesses	108
22.1 What a voice agent does	108
22.2 How a call works	109
22.3 Platforms	110
22.4 The law and consent (summary, not legal advice)	111
22.5 Designing the conversation	112
22.6 Build it: platform + n8n tools	113
22.7 Labs	114
22.8 Metrics	115

22.9 Exercises with model answers	115
22.10 Key takeaways	116
22.11 Check your understanding	117
22.12 Recommended resources	117
Module 23: Learning Paths, Courses and Communities	118
23.1 How to choose a course	118
23.2 Claude: courses from every major provider	119
23.3 AI product management programmes (India and global)	121
23.4 Courses for the other AI tools	122
23.5 YouTube channels	123
23.6 Communities	123
23.7 A 12-week plan with one assignment a week	124
23.8 Portfolio projects that impress	125
23.9 Best practices for learning tools that change monthly	125
23.10 Key takeaways	125
23.11 Check your understanding	126
23.12 Recommended resources	126

MODULE 13

AI Tools Landscape and Your Starter Stack

Which AI tool to use for which job, what each one costs, what you need before you start, and how the AI Tools Lab classes fit together.

≈ 4 h study

Prices and features change every month

Every price, free limit and feature name in the AI Tools Lab was checked on **23 October 2026** against official pages or, where noted, several independent reviews. AI tools change quickly, so **check the official pricing page before you pay or teach from these numbers**. Screens in this lab are **illustrative wireframes** that show the layout and the order of clicks. They are not real screenshots, and button names may differ slightly in the live product.

13.1 How to use the AI Tools Lab

The AI Tools Lab is a set of hands-on classes. Each class follows the same pattern, so you can practise straight from the PDF:

1. **What the tool is and when to use it**, in plain words.
2. **What you need**: account, plan, cost, other tools.
3. **Level 1 (basic)**: your first result in under an hour.
4. **Level 2 (intermediate)**: the features working PMs use every week.
5. **Level 3 (advanced)**: connecting the tool to other apps, automating, and shipping.
6. **Labs**: numbered steps with a picture of each screen or workflow.
7. **Mistakes, exercises with model answers, a quiz and resources**.



13.2 The AI tools map

AI tools for PMs fall into eight groups. Most real projects combine two or three groups, for example **Claude** to write the spec, **Lovable** to build the app and **n8n** to connect it to Gmail and Slack.

1 · AI ASSISTANTS

- Claude, ChatGPT, Gemini, Microsoft Copilot
- Think, write, analyse files, research
- Class 14

2 · CODING AGENTS

- Claude Code, Cursor, GitHub Copilot, Codex
- Edit real code and repositories
- Class 15

3 · APP BUILDERS

- Lovable, Replit, Bolt, v0, Figma Make
- Prompt to working web app
- Classes 16 and 17

4 · DESIGN

- Figma, FigJam, Figma Make
- Screens, prototypes, design systems
- Class 18

5 · DOCS, PRDS AND PLANNING

- Notion, ChatPRD, Confluence, Linear, Jira
- Specs, wikis, roadmaps
- Class 19

6 · AUTOMATION

- Zapier, Make, n8n
- Connect Gmail, Slack, Sheets, CRMs
- Class 20

7 · CHATBOTS

- WhatsApp Business Platform, Botpress, Voiceflow, n8n
- Answer customers in chat
- Class 21

8 · VOICE AGENTS

- Vapi, Retell AI, ElevenLabs Agents, Twilio
- Answer and make phone calls
- Class 22

The eight groups of AI tools a PM uses. Pick the group by the job. Pick the tool inside the group by cost, the integrations you need and what your team already uses.

13.3 Which tool for which job

If you want to...	Start with	Free to try?	Upgrade when
Write a PRD, summarise interviews, analyse a CSV	Claude (Projects + file upload)	Yes	You hit message limits or need Claude Code
Change code in a real repository	Claude Code	Needs Pro or API	Daily use: Max or a team plan
Build a clickable app with login and a database	Lovable	Yes (5 credits a day)	You need private projects or a custom domain
Build and host a full-stack app in the browser	Replit Agent	Yes (limited)	You publish more than one app
Design screens and a click-through prototype	Figma	Yes (Starter)	You need Dev Mode or more files
Keep specs, roadmaps and research in one place	Notion	Yes	You need full Notion AI (Business plan)
Send a Slack alert when an email arrives	Zapier	Yes (100 tasks a month)	You need multi-step Zaps
Build complex automations cheaply	n8n (self-hosted)	Yes (free software)	You want managed cloud hosting
Answer customers on WhatsApp	A WhatsApp BSP + n8n or Botpress	Trial credits only	Real customer traffic (per-message fees)
Answer phone calls with AI	Vapi or Retell AI	Trial credits	Any real calls (per-minute fees)

Job-to-tool chooser. Free tiers are enough to learn every tool in this lab. Real customer traffic (messages and calls) always costs money.

13.4 What it costs: free versus paid

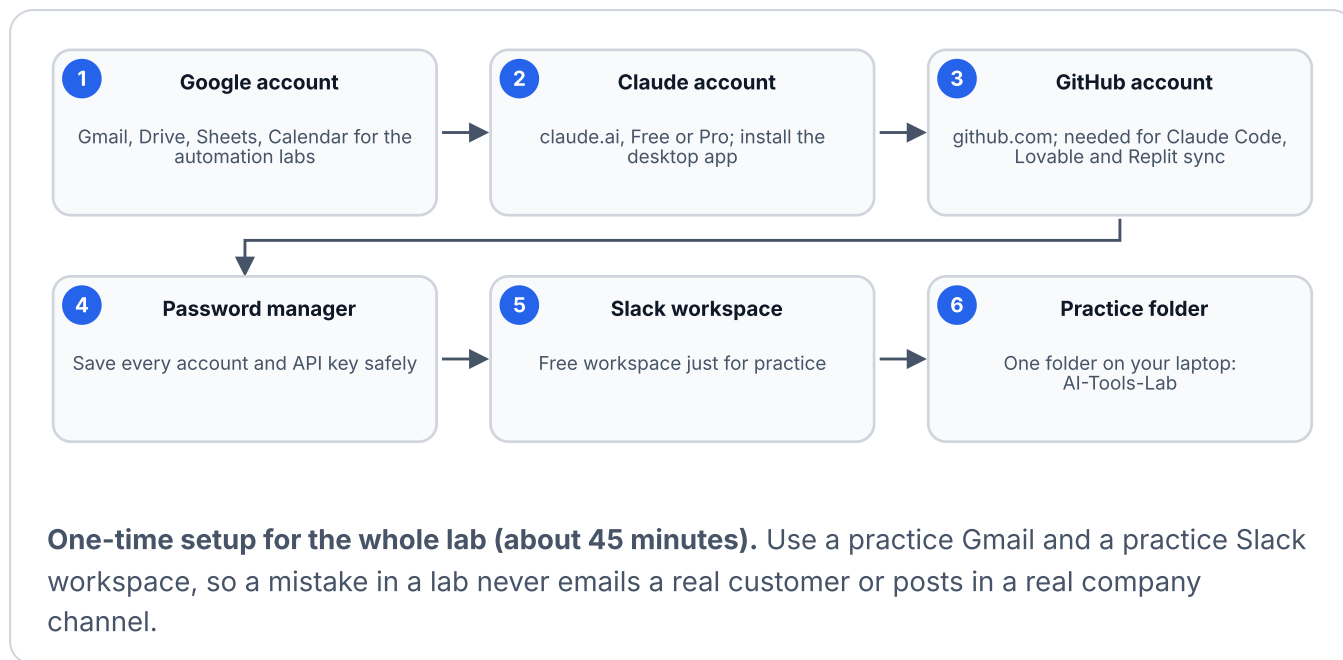
The rule for this lab

You can complete **every Level 1 lab for free**. Most Level 2 labs are free too. Paid tools appear only where real messages, phone calls, or production hosting are involved, and each class says so clearly.

Tool	Free tier	First paid tier	What you pay for
Claude	Free plan with limited messages	Pro, about \$20 a month	More usage, Claude Code, more models; Max for heavy use
Claude API	No free tier for production	Pay per token	Building your own apps and bots on Claude
Lovable	5 credits a day (max 30 a month), public projects	Pro, about \$25 a month (100 credits)	Private projects, custom domain, more credits
Replit	Starter: daily Agent credits, 1 published app	Core, about \$20 to \$25 a month	More Agent work, more apps, collaborators
Figma	Starter: limited files and AI credits	Professional, about \$16 a month per full seat	Dev Mode, MCP server, more Figma Make use
Notion	Free plan, limited AI trial	Business, about \$20 per member a month	Full Notion AI, agents, meeting notes
Zapier	100 tasks a month, two-step Zaps	Professional	Multi-step Zaps, more tasks
Make	1,000 credits a month, 2 active scenarios	Core	More credits and scenarios
n8n	Community Edition is free to self-host	Cloud Starter, about €20 to €24 a month	Managed hosting; you pay the server if self-hosting
WhatsApp Business Platform	No free tier for templates	Per delivered template message	Marketing, utility, authentication messages; check Meta's rate card
Voice agents (Vapi, Retell)	Trial credits	Usage, roughly \$0.07 to \$0.30 a minute all-in	Speech-to-text, AI model, voice, phone line

Free and paid tiers at a glance (checked 23 Oct 2026; verify before paying). Figures are rounded and come from the official pricing pages where available, otherwise from several 2026 reviews. Indian rupee prices vary by tax and billing region.

13.5 What you need before you start



Hardware and software: any laptop from the last five years works. For Claude Code you also need a terminal (built into macOS and Windows) and, for some installs, **Node.js** from nodejs.org. Nothing in this lab needs a powerful computer, because the AI runs in the cloud.

13.6 Accounts, API keys and safety

Many labs connect one app to another. There are three ways apps connect, and you will meet all three:

Easiest

Built-in connector or integration: click Connect, sign in, approve

Middle

API key or OAuth credential pasted into an automation tool

Most control

MCP server: lets an AI assistant use another app's tools

Three ways tools connect to each other. Connectors are what you use most. API keys appear in n8n, chatbots and voice agents. MCP (Model Context Protocol) is how Claude and Claude Code use tools such as Figma, Notion, GitHub and your own n8n workflows.

Five safety rules for every lab

1. **Never paste an API key into a chat, a document, a prompt, or frontend code.** Keep keys in the tool's secret store (n8n credentials, Lovable or Replit secrets, Netlify environment variables).
2. **Use test data.** Do not upload customer data, salary data or anything under NDA to a free AI plan. Check your company's AI policy first.
3. **Approve actions one by one** until a workflow has run correctly at least ten times.
4. **Turn off what you no longer use.** Disconnect connectors and delete unused API keys.
5. **Set spending limits** on every paid API (Anthropic Console, OpenAI, Twilio, Vapi) before your first test.

13.7 A starter stack for three budgets

₹0 A MONTH (LEARNING)

- Claude Free
- Lovable Free
- Figma Starter
- Notion Free
- Zapier Free or n8n on your laptop
- Telegram bot for chat practice
- Voice agent trial credits

ABOUT \$45 A MONTH (SERIOUS PM)

- Claude Pro (includes Claude Code)
- Lovable Pro
- Figma and Notion free tiers
- n8n self-hosted on a small server
- Pay-as-you-go API with a \$10 limit

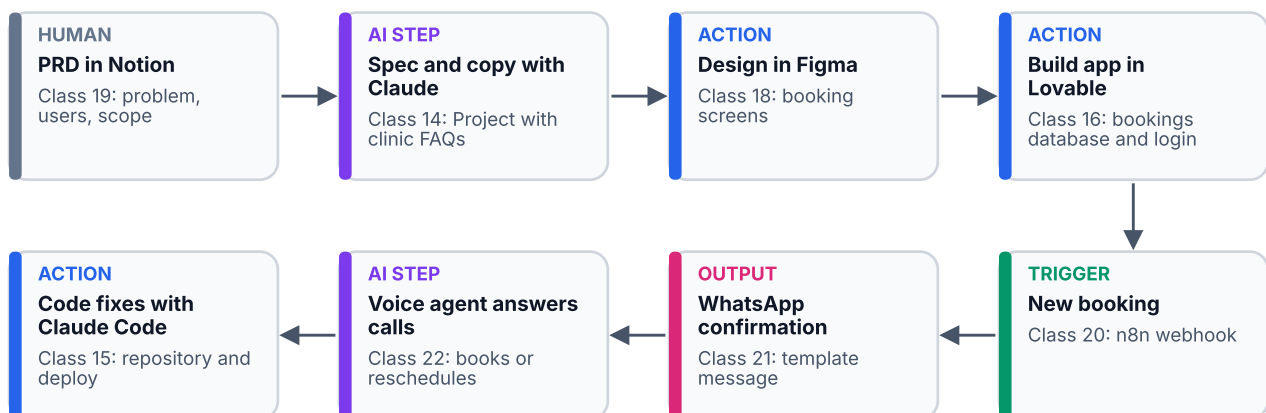
TEAM OR CLIENT WORK

- Claude Team or Max
- Lovable or Replit paid
- Figma Professional
- Notion Business
- n8n Cloud or Make Core
- WhatsApp BSP and a voice agent platform on usage billing

Starter stacks. Start at ₹0. Upgrade one tool at a time, only when a free limit is actually blocking you.

13.8 How the classes connect: one example project

The capstone that ties the lab together is a **small business assistant** for a hypothetical clinic. Each class adds one piece:



Capstone: a clinic's AI front desk, built across the lab. Hypothetical project for practice. Use test phone numbers and made-up patients only. Node types shown: Action · AI step · Human step · Output · Trigger (starts the flow).

13.9 Exercises with model answers

Exercise 1 · Pick the tools

A D2C brand wants: a landing page with a waitlist form, an email to each signup, and a weekly Slack summary of signups. Which tools would you use, free first?

Model answer

Landing page and form: **Lovable** (free tier) or **Claude artifacts** for a quick static page. Email and Slack: **Zapier** (free covers a two-step "new form row → send email" Zap) plus a second Zap for the weekly summary, or one **n8n** workflow that does both. Summary text: an **AI step** (Claude via n8n, or Zapier's AI step). Paid only when signups exceed the free task limits.

Exercise 2 · Spot the risk

A teammate pasted the company's OpenAI API key into a Lovable prompt so the app "can talk to AI". What is wrong, and what should they do?

Model answer

The key is now in the chat history and may end up in frontend code where any visitor can read it. **Revoke the key immediately**, create a new one, and store it as a **secret** (Lovable Cloud or Supabase secrets) used only by a server-side function. Set a spending limit on the new key.

13.10 Key takeaways

- Choose the tool group by the **job**, and the tool by **cost, integrations and team fit**.
- Every Level 1 lab is **free**. Messages, calls and production hosting are what cost money.
- Keep **secrets out of prompts and frontend code**, use **test data**, and set **spending limits**.
- Connectors, API keys and **MCP** are the three ways tools talk to each other.

13.11 Check your understanding

CHECK YOUR UNDERSTANDING

Which group would you use to change code inside an existing GitHub repository?

- A. AI assistants such as the Claude chat app
- B. Coding agents such as Claude Code
- C. Automation tools such as Zapier
- D. Voice agents

Why: Coding agents read, edit and commit code in a real repository.

What usually costs money even when you are only testing a WhatsApp or phone bot with real people?

- A. Writing the prompt
- B. Designing the flow in Figma
- C. Delivered messages and call minutes
- D. Creating a Google account

Why: Messaging and telephony are billed per message or per minute.

Where should an API key for an automation live?

- A. In the prompt so the AI can see it
- B. In a shared Google Doc
- C. In the tool's credential or secret store
- D. In the website's JavaScript

Why: Credential stores keep keys encrypted and out of chats and frontend code.

13.12 Recommended resources

- [Claude pricing](#) · [Lovable pricing](#) · [Replit pricing](#) · [Figma pricing](#)
- [Notion pricing](#) · [Zapier pricing](#) · [Make pricing](#) · [n8n pricing](#)
- [WhatsApp Business Platform pricing \(Meta\)](#)
- [Model Context Protocol](#) — the standard behind Claude's connectors.

MODULE 14

Claude for Daily PM Work: Basic to Advanced

Use Claude every day for writing, research, analysis and prototypes, then connect it to Gmail, Calendar, Drive, Slack, GitHub and your own tools.

≈ 8 h study

Feature names move around

Claude ships new features often, and menus get renamed. If a button in a lab has moved, search the [Claude Help Center](#) for the feature name. Screens below are **illustrative wireframes**, not screenshots.

14.1 What Claude is and where you can use it

Claude is Anthropic's AI assistant. You talk to it in plain language, give it files, and it writes, analyses, researches, codes and builds small apps. The same account works in several places:

WEB: CLAUDE.AI

- Every feature
- Best for long work sessions
- Artifacts, Projects, connectors

DESKTOP APP (MAC, WINDOWS)

- Same as web, plus local extensions
- Keyboard shortcut to open quickly
- Works with files on your laptop

MOBILE APP (IOS, ANDROID)

- Voice mode for hands-free thinking
- Photo of a whiteboard → summary
- Good for quick capture

CLAUDE CODE

- Terminal, IDE, desktop and web
- Works inside code repositories
- Class 15

INSIDE OTHER APPS

- Slack, browser extension and others where your plan offers them
- Check Settings → Connectors and the Help Center

API (CLAUDE CONSOLE)

- Build Claude into your own app or bot
- Pay per use
- Used in Classes 20 to 22

Where you can use Claude. Start on the web or desktop app. Add the others when a task needs them.

14.1.1 Plans

Plan	Who it is for	What you get (checked 23 Oct 2026)
Free	Trying Claude	Chat, file upload, limited daily messages, some connectors
Pro	Individual PMs	More usage, Projects, more models, Claude Code included
Max	Heavy daily users	5× or 20× Pro usage, priority access, early features
Team / Enterprise	Companies	Admin controls, shared Projects, single sign-on, data controls

Chat and Claude Code draw from the **same usage pool** on Pro and Max, so a long coding session leaves less chat time that day. Check claude.com/pricing for current limits.

14.2 Level 1 · Your first hour with Claude

Level 1 · Beginner

Goal: get one genuinely useful result (a meeting summary, an email, a first PRD draft) and learn the four habits that make answers better.

14.2.1 The anatomy of a good prompt

Most weak answers come from missing context, not from a weak model. Use four parts:



The four-part prompt. Example — "I'm a PM at a food-delivery app (context). Turn these interview notes into the top five problems with a quote for each (task). Notes attached (input). Use a table, plain English, max 200 words (format)."

The wireframe shows a Claude chat window titled 'Claude' with the URL 'claude.ai/new'. On the left is a sidebar with '+ New chat', 'Chats', 'Projects', 'Artifacts', and 'Settings'. The main chat area has a title 'Interview synthesis'. The chat history includes:

- Message 1 (User):** "I'm a PM at a food-delivery app. Turn the attached notes into the top 5 problems, with one quote each. Table, max 200 words." (Annotated with 1)
- Attachment:** A dashed box labeled 'interview-notes.pdf (attached)'.
- Message 2 (Claude):** "Here are the five problems... (table with Problem, Evidence, Quote, How many users)" (Annotated with 2)
- Message 3 (User):** "Reply... e.g. 'Now rank them by frequency and add a confidence column'" (Annotated with 3)

Below the chat window, three numbered instructions are provided:

- 1 Write the four parts: context, task, input, format. Attach files with the paperclip or drag and drop.
- 2 Read the answer critically. Check quotes against the source, because AI can misquote.
- 3 Improve by replying, not by starting again: ask for a change, a different format, or a shorter version.

A first chat in Claude (illustrative wireframe, not a real product screenshot). Use the pencil on your own message to edit and resend; use retry on Claude's answer to get a different version.

14.2.2 Four habits for Level 1

1. **Give the source material.** Attach the PDF, CSV or notes instead of asking from memory.
2. **Ask for a format.** Tables, bullet lists, "max 150 words", "for a VP audience".
3. **Iterate in the same chat.** "Shorter", "more specific", "add a risks section".
4. **Start a new chat for a new topic.** Long mixed chats get slower and less focused.

14.2.3 Files Claude can read

PDFs, Word documents, slides, spreadsheets and CSVs, images and screenshots, and plain text or code files. Claude can **run code to analyse data files**: upload a CSV and ask for averages, a chart or a cohort table.

Data analysis prompt

"Here is `orders.csv`. 1) Describe the columns. 2) Weekly order count and average order value. 3) A line chart of both. 4) Three observations a PM should act on. Show the method, so I can check it."

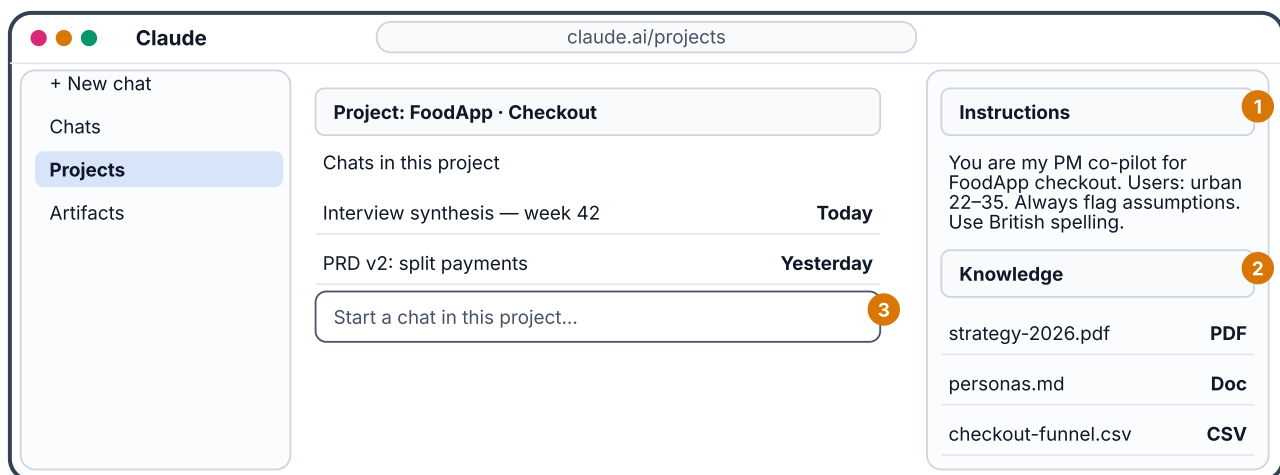
14.3 Level 2 · Features working PMs use every week

Level 2 · Working PM

Goal: stop re-explaining your product in every chat, produce documents and prototypes you can share, and research with sources.

14.3.1 Projects: your product's memory

A **Project** is a workspace with **custom instructions** and **knowledge files** that every chat inside it can use. Make one Project per product or per big initiative.



- 1 Instructions: who Claude is in this project, your users, rules (tone, spelling, 'always cite the source file').
- 2 Knowledge: upload the strategy, personas, PRDs, research and metrics. Keep files current; delete stale ones.
- 3 Every chat started here sees the instructions and knowledge, so you never paste context again.

Setting up a Project (illustrative wireframe, not a real product screenshot). On Team plans you can share a Project with colleagues so everyone uses the same context.

14.3.2 Artifacts: documents, code and clickable prototypes

When Claude produces something substantial (a document, a diagram, a web page, a small app), it appears as an **artifact** in a panel beside the chat. You can edit it by asking for changes, see versions, copy or download it, and **publish** it to get a link you can share.

Claude

Chats
Projects
Artifacts

Build a clickable mobile prototype of a split-bill screen: items list, 'split with' picker, total per person. Sample data only.

I've built the prototype. Tap a friend to add them; totals update.

Make the total sticky at the bottom and add a 'Request on UPI' button.

Preview **Code**

Live prototype: Split bill

Paneer tikka	₹320
You + Asha	₹160 each

Publish

- 1 Ask for 'sample data only' and 'mobile-first'. Claude writes HTML, CSS and JavaScript and shows a live preview.
- 2 Change it in plain words. Each change creates a new version you can go back to.
- 3 Publish gives a link for stakeholders. Check what is public before sharing, and never put real customer data in a prototype.

An artifact prototype beside the chat (illustrative wireframe, not a real product screenshot). Artifacts are ideal for clickable concepts and one-page tools. For a real product with login and a database, move to Lovable (Class 16) or Claude Code (Class 15).

14.3.3 Research with sources

Turn on **web search** (and **Research**, where your plan offers it) for questions that need current information: competitors, regulations, market sizes. Research runs many searches and returns a report with citations.

Always open the sources

Click at least three citations and confirm the claim is really there. AI research can misread a source or rely on a weak one. Mark any number you could not verify as "unverified" in your document.

14.3.4 Styles, memory and skills

- **Styles** set how Claude writes (concise, formal, explanatory, or a custom style learned from your own writing samples).
- **Memory and past-chat search** (where enabled) let Claude recall earlier work. You can view, edit or turn memory off in Settings.
- **Skills** are packaged instructions (and sometimes scripts) that teach Claude a repeatable task, such as your company's PRD format or a spreadsheet report. Claude loads a skill when the task matches. You can use Anthropic's skills and create your own.

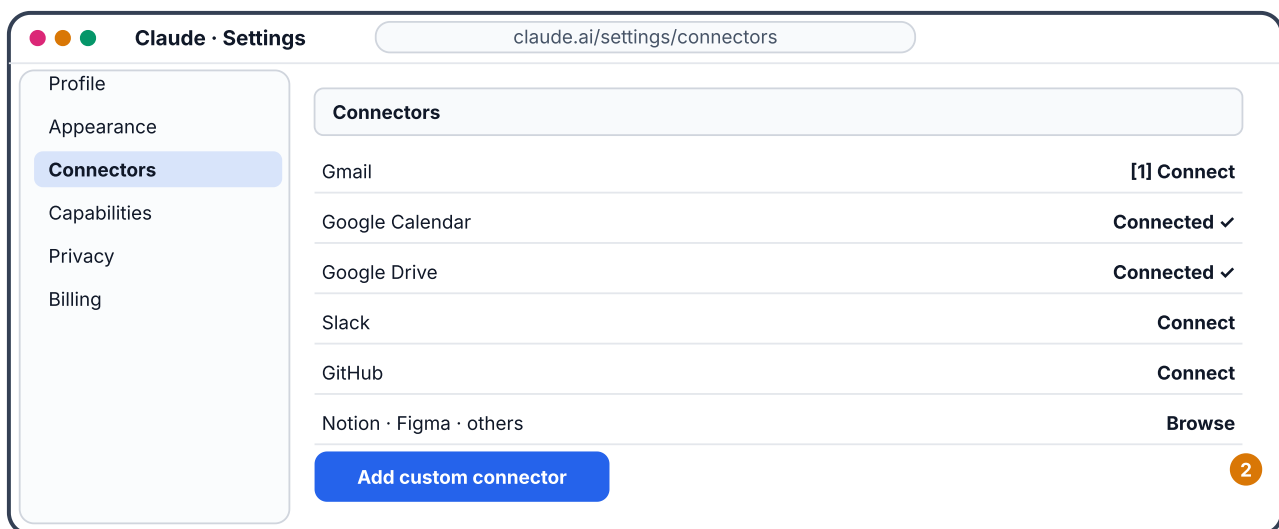
14.4 Level 3 · Connect Claude to your apps

Level 3 · Advanced

Goal: let Claude read your email, calendar, documents and Slack, act with your approval, and use your own tools through MCP.

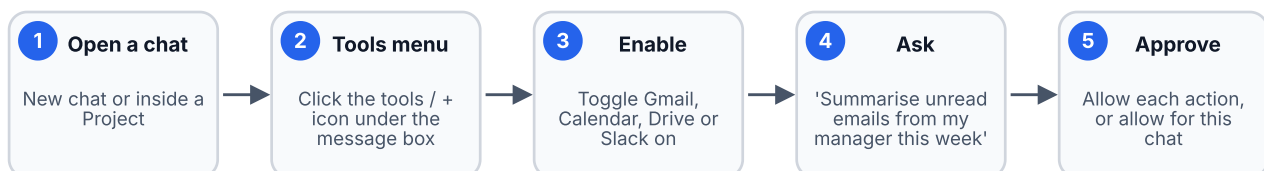
14.4.1 Connectors

Connectors give Claude permission to use another app on your behalf. Go to **Settings** → **Connectors**, click **Connect** next to a service, sign in, and approve the permissions.



- 1 Click Connect, sign in to the other app and approve. On Team and Enterprise plans an admin may need to enable a connector first.
- 2 Custom connector: paste the URL of a remote MCP server (for example your own n8n workflow, Class 20) to give Claude new tools.

Connecting Gmail, Calendar, Drive and Slack (illustrative wireframe, not a real product screenshot). Gmail: Claude can search, read and draft emails, but it does not send them for you. Each action asks for your approval unless you change that setting.



Turning a connector on for one chat. If Claude says it cannot reach a tool, the connector is usually disconnected. Reconnect it in Settings → Connectors.

14.4.2 What PMs do with connectors

Connect or	Daily use	Example prompt
Gmail	Triage, find decisions, draft replies	"Find every email about the payments launch this month. List decisions and open questions, then draft a reply to the latest thread."
Google Calendar	Prep and planning	"What meetings do I have tomorrow? For each, what should I prepare, based on related Drive docs?"
Google Drive	Search and summarise docs	"Compare the Q3 and Q4 roadmap docs and list what moved."
Slack	Catch up and draft updates	"Summarise #checkout-squad since Monday: decisions, blockers, who is waiting on me."
GitHub	Read code and issues	"Read the open issues labelled 'checkout' and group them by theme."
Notion / Figma	Specs and designs	"Pull the requirements from the Split Bill PRD in Notion and check the Figma file covers each one."

14.4.3 Custom connectors and MCP

MCP (Model Context Protocol) is an open standard that lets an AI assistant use tools from another system. A **remote MCP server** has a URL. Add it under **Settings → Connectors → Add custom connector**, and Claude gets that server's tools. In Class 20 you will build an **n8n workflow with an MCP Server Trigger** and add it to Claude, so you can say "log this bug to our tracker" and n8n does it.

14.4.4 The API and Claude Console

To put Claude **inside your own app, bot or automation**, you use the **Claude API**:

1. Sign in at console.anthropic.com (separate billing from your claude.ai plan).
2. Add a payment method and **set a monthly spend limit**.
3. Create an **API key** and store it in a secret store (never in a chat or frontend code).
4. Test prompts in the **Workbench**, then use the key in n8n, Make, Zapier, Lovable or code.

14.5 A daily routine for PMs

Time	Task	Feature	Prompt starter
09:00	Inbox and calendar brief	Gmail + Calendar connectors	'Brief me: urgent emails, today's meetings, what to prep'
10:00	Synthesise user interviews	Project + file upload	'Top themes with quotes and counts; flag contradictions'
11:30	Draft or tighten a PRD section	Project + PRD skill or template	'Critique this scope against our strategy file'
13:00	Analyse a metric drop	File upload + analysis	'Explain the conversion drop by segment; show the method'
15:00	Prototype an idea for design review	Artifact	'Clickable mobile prototype, sample data, three states'
16:30	Competitor or market question	Research with citations	'How do the top 3 rivals handle refunds? Cite sources'
17:30	Weekly or daily update	Slack connector	'Draft my update from #squad and my notes, 120 words'

A day with Claude. Keep one Project per product, so every task in this list starts with the right context.

14.6 Labs

Lab 1 · Build your product Project (20 minutes)

1. Projects → **New project** → name it after your product.
2. Write instructions: product in one line, users, goals this quarter, writing rules, "flag assumptions".
3. Upload three files: a strategy or vision doc, personas, and one metrics CSV (use `datasets/orders.csv` from this handbook if you have nothing).
4. Ask: "Using only the knowledge files, what are the three biggest risks to this quarter's goal?"
5. Check the answer cites the files. If it invents facts, add to the instructions: "Answer only from the knowledge files; say 'not in files' otherwise."

Lab 2 · Interview synthesis (30 minutes)

1. Inside your Project, attach five interview notes (real or written by you).
2. Prompt: "Affinity-map these interviews: themes, number of users per theme, one quote each, and the job-to-be-done behind each theme."
3. Follow up: "Which theme is most likely to change our roadmap? What would we need to believe?"
4. Save the result as an artifact document and publish it for your team.

Lab 3 · A clickable prototype you can share (30 minutes)

1. Prompt: "Build a mobile-first clickable prototype for [feature]: three screens, sample data only, a 'Prototype · not real' label."
2. Ask for two changes, one at a time.
3. Ask for an edge case: "Add the empty state and an error state."
4. Click **Publish**, open the link in a private window, and check it works on your phone.

Lab 4 · Morning brief from Gmail and Calendar (15 minutes)

1. Settings → Connectors → connect **Gmail** and **Google Calendar** (use a practice account first).
2. In a new chat, turn both on in the tools menu.
3. Prompt: "Brief me for today: meetings with what to prep, emails that need a reply from me, and anything with a deadline this week."
4. Ask Claude to **draft** two replies. Open Gmail and check the drafts are there. Claude drafts; you review and send.

Lab 5 · Weekly Slack update (15 minutes)

1. Connect **Slack** (practice workspace).
2. Prompt: "Read #practice-squad from the last 7 days. Draft a weekly update: shipped, in progress, blocked, decisions needed. 120 words."
3. Edit the draft yourself before posting. You own what you send.

Lab 6 · Give Claude a custom tool (advanced, after Class 20)

1. Build the n8n "log a bug to Google Sheets" workflow with an **MCP Server Trigger** (Class 20, Lab 5).
2. Copy its production URL.
3. Claude → Settings → Connectors → **Add custom connector** → paste the URL.
4. In a chat: "Log a bug: checkout button overlaps on small screens, high priority." Approve the tool call, then check the sheet.

14.7 Prompt library for PMs

Job	Prompt
PRD critique	"Act as a skeptical engineering lead. List the 10 questions you'd ask about this PRD, then rewrite the scope section to answer them."
Prioritisation	"Score these 12 ideas with RICE. Show your assumptions for reach and effort, and mark which ones are guesses."
Metrics	"Build a KPI tree for 'weekly active ordering users'. Include input metrics I can move this quarter."
Stakeholders	"Rewrite this update for (a) the CEO in 3 lines, (b) engineers with the technical detail."
Interview prep	"Ask me one PM interview question at a time on product sense. After my answer, score it and show a stronger answer."
Release notes	"Turn these merged pull request titles into customer-friendly release notes, grouped by benefit."

14.8 Common mistakes

- **No context:** asking "write a PRD" with no product, users or goal. Use a Project.
- **Trusting numbers and quotes without checking:** always verify against the source.
- **Pasting confidential data into a personal plan:** follow your company's AI policy; use Team or Enterprise for work data.
- **One endless chat:** start a new chat per task; keep long-lived context in the Project.
- **Letting Claude send on your behalf too early:** keep approvals on for email and Slack actions.

14.9 Where to find official help and sources

- **Docs:** docs.claude.com (API, prompting, tools, MCP, Claude Code).
- **Help Center:** support.claude.com (features, plans, connectors, troubleshooting).
- **Claude Academy** (free; formerly Anthropic Academy at anthropic.com/learn, which reportedly now redirects there): courses including *Claude 101*, *AI Fluency*, *Building with the Claude API*, *Introduction to MCP* and *Claude Code in Action* at academy.claude.com.
- **Prompt engineering guide:** in the docs, under "Prompt engineering".
- **Cookbook:** github.com/anthropics/anthropic-cookbook for API examples.

- **Anthropic on YouTube:** official talks and feature walkthroughs.

14.10 Exercises with model answers

Exercise 1 · Fix the prompt

"Write a PRD for a loyalty program." Rewrite it using the four-part prompt.

Model answer

"I'm the PM for a grocery delivery app in Indian metros; 60% of orders come from repeat users (context). Draft a one-page PRD for a points-based loyalty program aimed at increasing 90-day retention (task). Use our PRD template and strategy file in this Project (input). Sections: problem, goals and metrics, users, scope in and out, risks, open questions; max 600 words; flag every assumption (format)."

Exercise 2 · Which feature?

You need (a) a shareable clickable concept for tomorrow's design review, (b) Claude to remember your product's personas in every chat, (c) a summary of last week's Slack decisions. Which Claude feature solves each?

Model answer

(a) An **artifact**, published as a link. (b) A **Project** with the personas in knowledge files and the instructions. (c) The **Slack connector**, enabled in that chat.

14.11 Key takeaways

- Good results come from **context, a clear task, source material and a format**.
- **Projects** hold context; **artifacts** turn answers into shareable documents and prototypes.
- **Connectors** let Claude work with Gmail, Calendar, Drive, Slack, GitHub and more, with your approval.
- **Custom connectors (MCP)** and the **API** extend Claude to your own tools and apps.

14.12 Check your understanding

CHECK YOUR UNDERSTANDING

You keep pasting the same personas and strategy into every chat. What should you use?

- A. A longer prompt
- B. A Project with instructions and knowledge files
- C. The API
- D. A new account

Why: Projects give every chat inside them the same instructions and files.

With the Gmail connector, what does Claude do with a reply you ask for?

- A. Sends it immediately
- B. Creates a draft for you to review and send
- C. Posts it to Slack
- D. Nothing; Gmail is read-only

Why: Claude can search, read and draft in Gmail, but it does not send on your behalf.

How do you give Claude a tool from your own n8n workflow?

- A. Paste the n8n password into a chat
- B. Add the workflow's MCP server URL as a custom connector
- C. Upload the workflow as a PDF
- D. It is not possible

Why: Remote MCP servers are added as custom connectors in Settings → Connectors.

Where should your Claude API key live when you use it in an automation?

- A. In the chat
- B. In the automation tool's credential store
- C. In a public GitHub repository
- D. In the artifact code

Why: Keys belong in secret or credential stores, never in chats or shared code.

14.13 Recommended resources

- [Claude Help Center](#) · [Claude docs](#) · [Claude Academy](#)
- [Connectors: getting started](#)
- [Claude Console](#) for API keys and spend limits
- [Anthropic cookbook on GitHub](#)

MODULE 15

Claude Code: Basic to Advanced

Use Claude's coding agent to read, edit and build real projects, create and push GitHub repositories, deploy prototypes, and extend it with MCP, skills, subagents and hooks.

≈ 10 h study

Commands and menus evolve

Claude Code updates often. If a command here behaves differently, run `/help` inside Claude Code or check the [Claude Code docs](#). Terminal screens below are **illustrative**.

15.1 What Claude Code is

Claude Code is an **agentic coding tool**. You describe what you want in plain language; it reads the files in a project, plans, edits code, runs commands (tests, builds, git), shows you what it changed, and asks permission before risky steps. Module 11 introduced it for vibe coding. This class goes from first install to advanced automation.

Why a PM should learn it

You can build and change prototypes yourself, answer "how does this feature actually work?" by reading the codebase, write data scripts, fix copy and small UI bugs, update documentation, and turn a PRD into a working demo in an afternoon. Engineers still review anything that goes to production.

15.1.1 Where Claude Code runs

TERMINAL (CLI)

- Type claude in any project folder
- Most complete feature set
- macOS, Linux, Windows

IDE EXTENSIONS

- VS Code, Cursor, JetBrains
- See diffs inline
- Good once you edit code daily

DESKTOP APP

- Graphical app on Mac and Windows
- Projects on your laptop
- No terminal needed

WEB: CLAUDE.AI/CODE

- Runs in the cloud on your GitHub repositories
- Start tasks from any browser or phone
- Opens pull requests

GITHUB ACTIONS

- Mention @claude in an issue or pull request
- Claude works and replies in GitHub

AGENT SDK AND HEADLESS

- claude -p 'task' in scripts and CI
- Build your own agents on the same engine

The places you can use Claude Code. Start with the desktop app or terminal. Move to the web version to run tasks on GitHub repositories from anywhere.

15.1.2 What you need

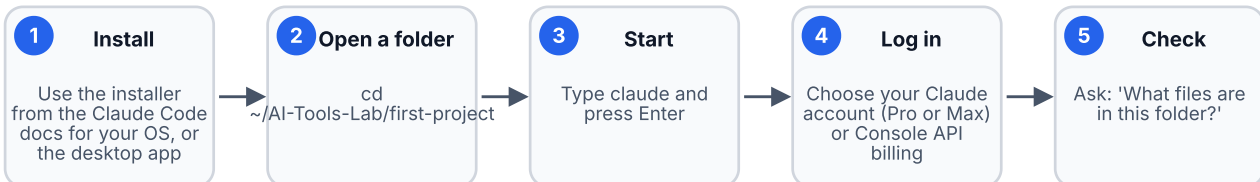
Need	Detail
Claude plan or API	Pro, Max, Team or Enterprise includes Claude Code. Or pay per use with a Claude Console API account.
Git	Version control; installed on macOS, install "Git for Windows" on Windows.
GitHub account	To store and share repositories, and for the web version.
A terminal	Terminal (macOS), PowerShell or Windows Terminal (Windows). Not needed for the desktop app.
Optional	Node.js (some installs and projects), the GitHub CLI <code>gh</code> , a Netlify account for deploying.

15.2 Level 1 · Install and your first session

Level 1 · Beginner

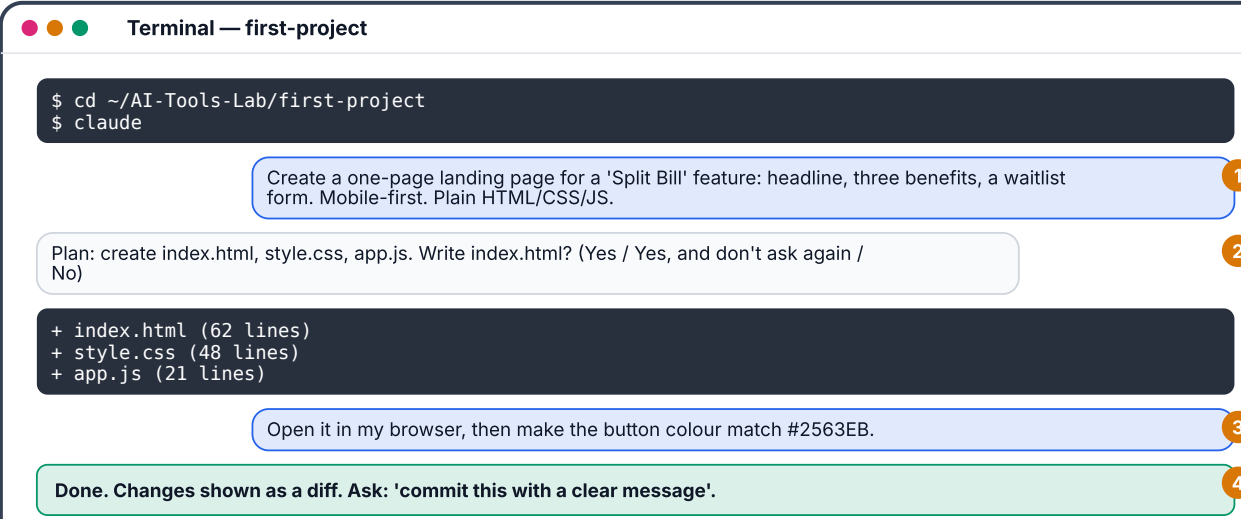
Goal: install Claude Code, make one change to a small project, review it and save it with git.

15.2.1 Install



Install and log in (10 minutes). Typical installer commands are a one-line script for macOS and Linux, a PowerShell line for Windows, or `npm install -g @anthropic-ai/claude-code`. Copy the current command from the docs rather than from a blog.

15.2.2 Your first session



The terminal window, titled "Terminal — first-project", shows a session with Claude Code. It starts with the user navigating to the project directory and running the Claude command. A request is made to create a landing page. Claude asks for confirmation to write the files. The files are created, and the user is asked to open them in a browser and make a color change. Finally, the changes are committed to the repository.

```
$ cd ~/AI-Tools-Lab/first-project
$ claude
```

Create a one-page landing page for a 'Split Bill' feature: headline, three benefits, a waitlist form. Mobile-first. Plain HTML/CSS/JS.

Plan: create index.html, style.css, app.js. Write index.html? (Yes / Yes, and don't ask again / No)

```
+ index.html (62 lines)
+ style.css (48 lines)
+ app.js (21 lines)
```

Open it in my browser, then make the button colour match #2563EB.

Done. Changes shown as a diff. Ask: 'commit this with a clear message'.

- 1 Describe the outcome, the constraints (mobile-first, plain HTML) and any data rules (sample data only).
- 2 Claude asks before writing files or running commands. Read the request, then approve once, or approve for the session.
- 3 Iterate in small steps. One change per message is easiest to check.
- 4 Review the diff, then ask Claude to commit. Small commits make mistakes cheap to undo.

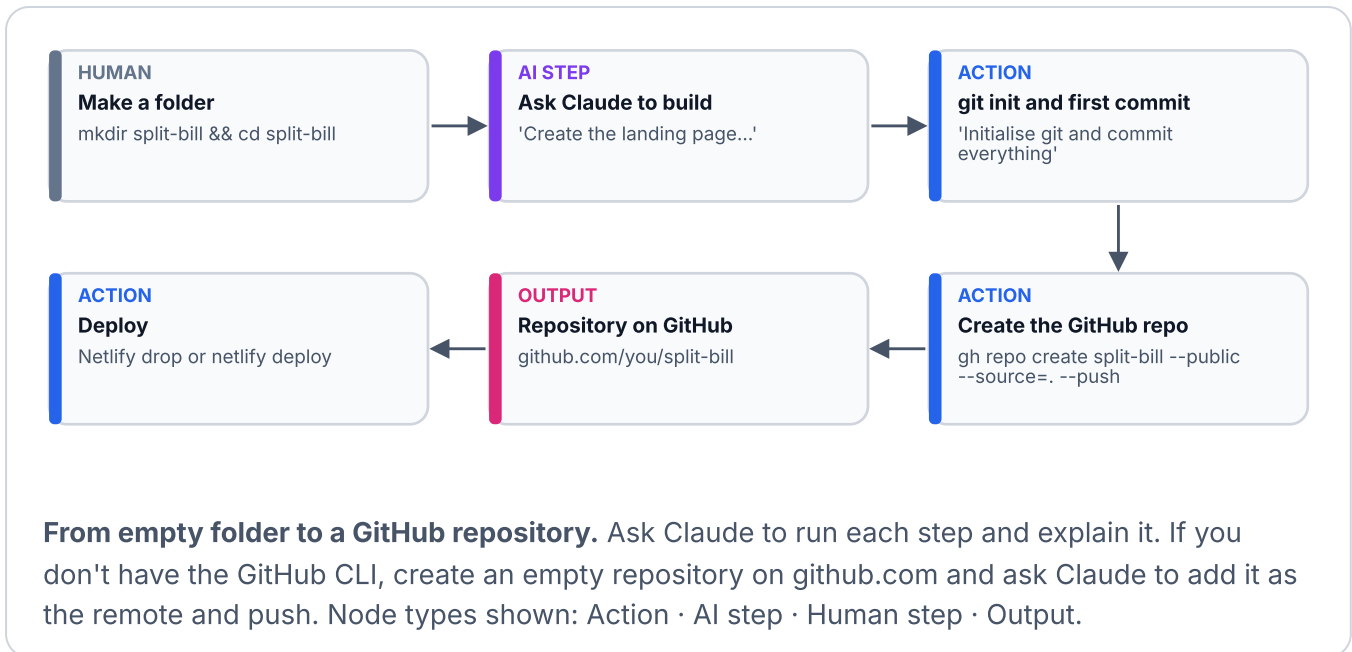
A first Claude Code session in the terminal (illustrative wireframe, not a real product screenshot). Press Esc to stop Claude mid-step. Press Esc twice to rewind to an earlier point in the conversation and undo its changes (checkpoints).

15.2.3 Essential commands

Command or key	What it does
<code>claude</code>	Start a session in the current folder
<code>claude -c</code>	Continue your most recent session here
<code>claude --resume</code>	Pick an older session to resume
<code>/help</code>	List commands available in your version
<code>/init</code>	Create a <code>CLAUDE.md</code> file that describes the project for future sessions
<code>/clear</code>	Start fresh in the same window (clears the conversation)
<code>/compact</code>	Summarise a long conversation to free up context
<code>/model</code>	Switch model (for example a faster or a more capable one)
<code>/permissions</code>	See and change what Claude may do without asking
<code>/mcp</code>	Manage connected MCP servers
Shift+Tab	Cycle modes: normal → auto-accept edits → plan mode
Esc, Esc Esc	Stop the current step; rewind to an earlier checkpoint

15.3 Creating a repository and pushing to GitHub

Every project you build should live in a **git repository**, so you can undo mistakes and share the work.



Prompt that does it all

"Initialise a git repository here, add a sensible .gitignore, commit everything with a clear message, then create a **public** GitHub repository called split-bill with the GitHub CLI and push. Explain each command before you run it."

Two other routes to a repository:

- **Start on GitHub:** create the repository at github.com/new, then ask Claude: "Clone github.com/you/split-bill here and open it."
- **Start in Lovable or Replit:** connect the project to GitHub (Classes 16 and 17), then clone that repository and improve it with Claude Code.

15.4 Level 2 · Working like a professional

Level 2 · Working PM

Goal: make Claude reliable on real projects with project memory, planning before coding, tests, and the right permissions.

15.4.1 CLAUDE.md: the project's instructions

`CLAUDE.md` sits in the project folder. Claude reads it at the start of every session. Run `/init` to create a first version, then edit it.

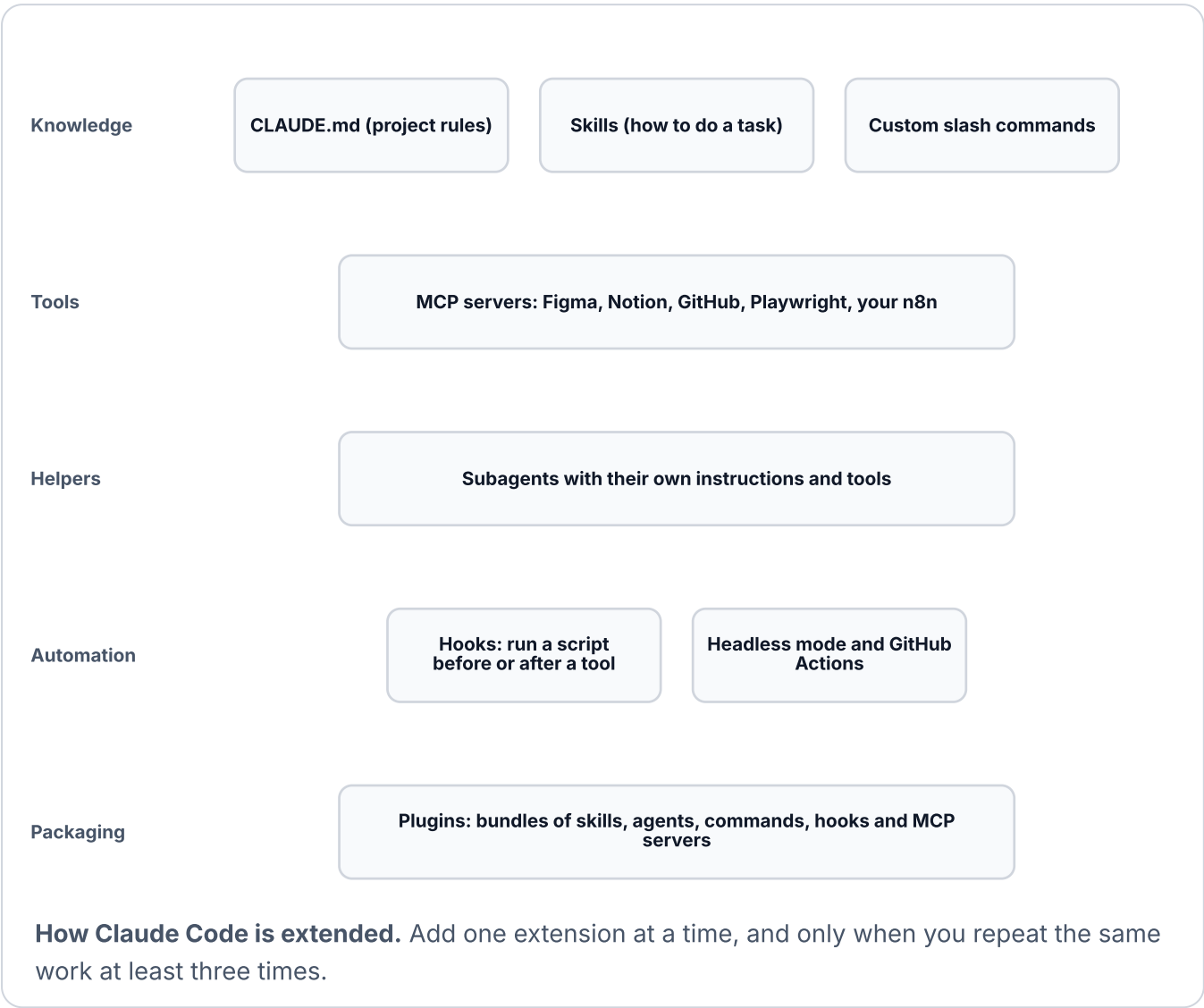
15.4.4 Permissions and safety

- Claude asks before editing files and running commands. Approve carefully, especially for commands that delete, deploy or spend money.
- Use `/permissions` to pre-approve safe, repeated commands (for example `npm test`) and keep risky ones on "ask".
- Never paste API keys into the chat. Put them in a `.env` file that is listed in `.gitignore`.
- Review the diff before every commit. You are responsible for what you push.

15.5 Level 3 · Extending Claude Code

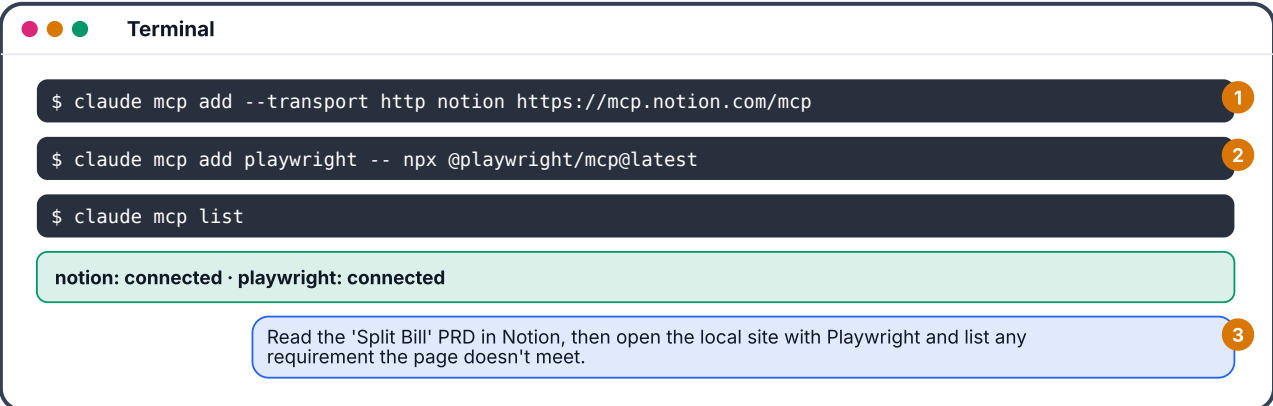
Level 3 · Advanced

Goal: give Claude new tools (MCP), reusable know-how (skills and commands), helpers (subagents), automatic checks (hooks), and run it without you (headless, GitHub, web).



15.5.1 MCP servers: new tools for Claude

Add an MCP server once, and Claude can use its tools in every session. Use `claude mcp add`, or `/mcp` inside a session.



The terminal window shows the following commands and output:

```
$ claude mcp add --transport http notion https://mcp.notion.com/mcp
$ claude mcp add playwright -- npx @playwright/mcp@latest
$ claude mcp list
```

Output: `notion: connected · playwright: connected`

Read the 'Split Bill' PRD in Notion, then open the local site with Playwright and list any requirement the page doesn't meet.

- 1 Remote servers: `--transport http` plus the service's MCP URL. The first use opens a browser to sign in. Check each vendor's docs for the current URL.
- 2 Local servers run a command on your machine. Playwright lets Claude open and click through web pages to test them.
- 3 Combine tools in one request: read the spec, test the page, report gaps.

Adding MCP servers (illustrative wireframe, not a real product screenshot). Popular MCP servers for PMs include Figma (build from a design), Notion and Linear or Jira (specs and tickets), GitHub, Playwright (browser testing), and your own n8n workflows (Class 20).

15.5.2 Skills and custom commands

- A **skill** is a folder with a `SKILL.md` file (instructions, examples, optional scripts) inside `.claude/skills/` in your project or your user folder. Claude uses it automatically when a task matches its description. Example: a "release-notes" skill that turns merged pull requests into customer-friendly notes in your house style.
- A **custom slash command** is a Markdown prompt in `.claude/commands/`, run by typing `/its-name`. Example: `/prd-check` asks Claude to compare the code with `docs/prd.md`.

15.5.3 Subagents

A **subagent** is a helper with its own instructions, tools and separate context. Create one with `/agents`. Good PM examples: a **reviewer** that checks a change against the PRD, a **tester** that runs Playwright checks, a **copy editor** that enforces your tone guide. Subagents keep long, noisy work out of your main conversation.

15.5.4 Hooks

Hooks run your own script automatically at set moments, for example **before** a tool runs (block edits to `.env`), or **after** a file is edited (run the formatter). Manage them with `/hooks`. Hooks are rules that always run, unlike instructions, which Claude may occasionally miss.

15.5.5 Plugins

A **plugin** bundles skills, subagents, commands, hooks and MCP servers so a team can install the same setup with one `/plugin` command. Use them to share your team's standard Claude Code setup.

15.5.6 Running without you

Mode	How	PM use
Headless	<code>claude -p "Summarise the changes since last Friday"</code> in a script	Scheduled reports, release notes
GitHub Actions	Run <code>/install-github-app</code> , then write <code>@claude</code> in an issue or pull request	"@claude fix this typo and open a PR"
Claude Code on the web	claude.ai/code → pick a GitHub repository → describe the task	Start a task from your phone; review the pull request later

15.6 Costs

- **Pro and Max plans** include Claude Code; usage is shared with chat, and heavy agentic work uses more.
- **API billing** charges per token. Run `/cost` to see what an API session has spent, and set a spend limit in the Console.
- Cheaper habits: one clear task per session, `/clear` between unrelated tasks, `/compact` in long sessions, plan mode before big changes.

15.7 Labs

Lab 1 • Install and first change (20 minutes)

1. Install Claude Code (desktop app or CLI) and log in.
2. Make a folder `first-project` and start `claude` in it.
3. Ask for a one-page landing page with sample data. Approve the file writes.
4. Ask for one design change. Review the diff.
5. Press **Esc Esc** and rewind the last change, to see that undo works.

Lab 2 • Create and push a GitHub repository (20 minutes)

1. In the same folder: "Initialise git, add a .gitignore, and commit."
2. "Create a public GitHub repository called first-project and push." (Install `gh` and run `gh auth login` first, or create the repository on github.com and ask Claude to push to it.)
3. Open the repository on GitHub and check your files are there.

Lab 3 • From PRD to deployed prototype (60 minutes)

1. Copy `templates/01-prd-template.md` from this handbook into the project as `docs/prd.md` and fill in a small feature.
2. Shift+Tab to **plan mode**: "Read docs/prd.md and plan a clickable prototype that covers every requirement."
3. Approve the plan, leave plan mode, and let Claude build.
4. "Check the page at 375 px width; fix horizontal scroll; label it as a prototype."
5. Deploy: drag the folder onto app.netlify.com/drop, or ask Claude to use the Netlify CLI. Share the link.

Lab 4 • CLAUDE.md and a repeat mistake (15 minutes)

1. Run `/init`. Edit CLAUDE.md using the five-part structure above.
2. Notice one thing Claude got wrong in Lab 3. Add a rule for it.
3. Start a new session and repeat the task. The mistake should not return.

Lab 5 · Test your page with Playwright MCP (20 minutes)

1. `claude mcp add playwright -- npx @playwright/mcp@latest`
2. "Open site/index.html with Playwright, submit the waitlist form empty and with a valid email, and report what happens."
3. Ask Claude to fix any bug it found, then re-run the test.

Lab 6 · A release-notes skill (30 minutes)

1. "Create a skill in .claude/skills/release-notes that turns git commit messages from the last 7 days into customer-friendly release notes, grouped by benefit, max 150 words."
2. Make three small commits.
3. Ask: "Write this week's release notes." Check Claude used the skill.

Lab 7 · Claude Code on the web (15 minutes)

1. Open claude.ai/code and connect GitHub when asked.
2. Choose your `first-project` repository.
3. Task: "Add a FAQ section with three questions and open a pull request."
4. Review the pull request on GitHub and merge it yourself.

15.8 PM use cases

Task	Prompt
Understand a feature	"Explain how refunds work in this codebase, with file names. Don't change anything."
Size a request	"What would it take to add a coupon field at checkout? List files to touch and risks."
Data question	"Write a Python script that reads data/orders.csv and prints weekly retention by cohort."
Copy fix	"Change every 'Checkout' button label to 'Pay now' and show me the diff."
Docs	"Update README.md so a new teammate can run this project in five minutes."
Issue triage	"Read the open GitHub issues and group them into themes with counts."

15.9 Common mistakes

- **Huge first prompts:** build in small steps and commit after each working step.
- **Approving everything without reading:** especially deletes, deploys and package installs.
- **No CLAUDE.md:** Claude rediscovers the project every session and repeats mistakes.
- **Secrets in code:** keys belong in `.env` (git-ignored) or the host's environment variables.
- **Treating a prototype as production:** production code needs engineering review, tests and security checks (Module 11).

15.10 Exercises with model answers

Exercise 1 · Plan or just do it?

Which of these should start in plan mode? (a) Fix a typo in the footer. (b) Add login with email OTP to the prototype. (c) Rename a CSS class.

Model answer

Only **(b)**. It touches several files, adds data storage and has security decisions, so agree the approach first. (a) and (c) are small, contained edits you can review in one diff.

Exercise 2 · Which extension?

Your team keeps asking Claude to check new screens against the Figma design, and someone keeps accidentally committing `.env`. What do you add?

Model answer

Add the **Figma MCP server** (so Claude can read the design) and a short **skill** or **subagent** for "compare the implementation with the Figma frame". For `.env`, add it to `.gitignore` **and** a **PreToolUse hook** that blocks edits or commits touching `.env`. A hook always runs, unlike an instruction.

15.11 Key takeaways

- Claude Code works on **real files and repositories** and asks before risky actions.
- Use **git from the first minute**: commit small, working steps and push to GitHub.
- **CLAUDE.md** and **plan mode** are the two habits that most improve results.
- **MCP, skills, subagents, hooks and plugins** extend Claude Code once you repeat a task.
- Headless mode, **GitHub Actions** and **Claude Code on the web** let it work while you do something else.

15.12 Check your understanding

CHECK YOUR UNDERSTANDING

Which key cycles into plan mode?

- A. Ctrl+C
- B. Shift+Tab
- C. Esc
- D. F5

Why: Shift+Tab cycles normal, auto-accept edits and plan mode.

What does CLAUDE.md do?

- A. Stores your API key
- B. Gives Claude project instructions at the start of every session
- C. Deploys the site
- D. Lists your GitHub followers

Why: Claude reads CLAUDE.md to learn the project's purpose, stack, rules and commands.

You want an automatic check that blocks edits to .env every time. Use...

- A. A line in the prompt
- B. A hook
- C. Plan mode
- D. The /model command

Why: Hooks run your script at fixed points, so the rule is enforced every time.

How do you let Claude read a spec in Notion?

- A. Copy the Notion password into the chat
- B. Add Notion's MCP server with `claude mcp add`
- C. Export Notion as a ZIP
- D. It's not possible

Why: MCP servers give Claude Code tools for other apps; Notion offers a remote MCP server.

15.13 Recommended resources

- [Claude Code docs](#) · [Claude Code best practices](#)
- [Claude Academy: Claude Code in Action](#) (free; formerly Anthropic Academy)
- [DeepLearning.AI: Claude Code: A Highly Agentic Coding Assistant](#) (free, about 2 hours)
- [Coursera: Claude Code: Software Engineering with Generative AI Agents \(Vanderbilt\)](#)
- [GitHub CLI](#) · [Netlify Drop](#) · [Playwright MCP](#)

MODULE 16

Lovable: Basic to Advanced

Go from a prompt to a published web app with a database, login, AI features and payments, then sync it to GitHub and keep improving it with Claude Code.

≈ 8 h study

Lovable changes fast

Lovable adds and renames features often. If a button below has moved, search the [Lovable docs](#). Screens are **illustrative wireframes**.

16.1 What Lovable is

Lovable builds web apps from a conversation. You describe the app, and Lovable writes a real React application, shows a live preview, and can add a **database, user login, file storage and server functions** for you. You can publish with one click, connect your own domain, and sync the code to GitHub.

When to use Lovable

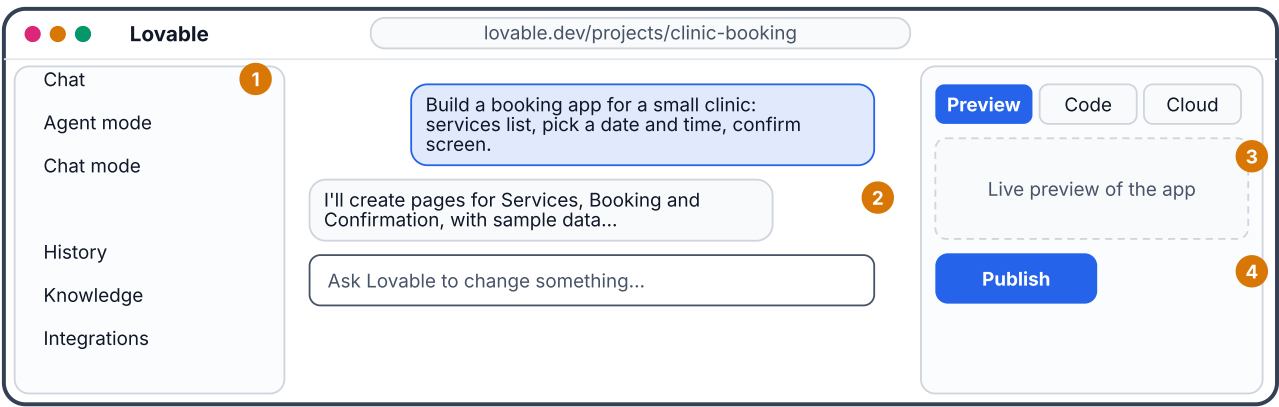
Clickable MVPs, internal tools, landing pages with forms, dashboards, booking and directory apps, and portfolio projects. For deep changes to a large existing codebase, use Claude Code (Class 15). For production at scale, an engineering team reviews security, performance and data.

16.1.1 What it costs (checked 23 Oct 2026)

Plan	Price	What you get
Free	\$0	5 credits a day (up to 30 a month), public projects on a Lovable address
Pro	about \$25 a month	100 monthly credits (more tiers available), private projects, custom domains, remove the badge, credit top-ups
Business	about \$50 a month	Team features such as single sign-on and security controls

A **credit** is used each time Lovable does work for you. Bigger changes use more. Backend services (database, AI calls, email) may have their own usage limits. Check lovable.dev/pricing.

16.2 The editor



1 Agent mode builds and edits. Chat mode discusses and plans without changing code, so use it to think before spending build credits.

2 Each message becomes a version in History. You can restore any version.

3 The preview updates live. Use Visual edits (click an element) for small text, colour and spacing changes.

4 Publish puts the app on a public link. Paid plans add custom domains.

The Lovable editor (illustrative wireframe, not a real product screenshot). Knowledge holds your project instructions, and Integrations is where you connect a database, GitHub, payments and email.

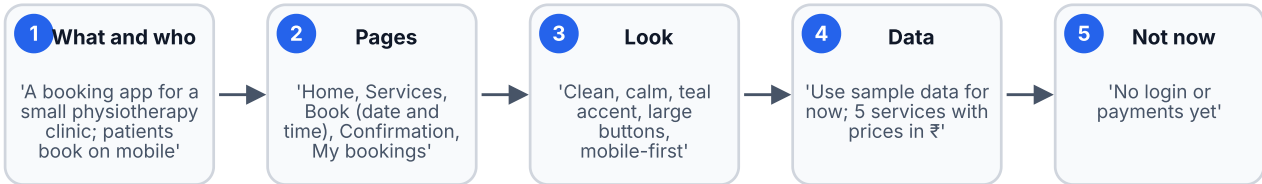
16.3 Level 1 · Your first app in 30 minutes

Level 1 · Beginner

Goal: build and publish a small, good-looking app, and learn the prompt pattern that saves credits.

16.3.1 The first-prompt template

A strong first prompt saves many fixes later. Include five things:



The Lovable first prompt. Saying what not to build yet keeps the first version small, cheap and easy to check.

16.3.2 Then iterate in small steps

1. Look at the preview on desktop **and** in the phone view.
2. Ask for **one change per message**: "Make the service cards show duration and price."
3. Use **Visual edits** for tiny text or colour changes, which is faster and cheaper than a prompt.
4. If something breaks, click **Try to fix**, or restore the last good version from **History**.
5. Click **Publish** and open the link on your phone.

16.4 Level 2 · Building like a pro

Level 2 · Working PM

Goal: plan before you build, keep the app consistent, debug calmly, and avoid wasting credits.

16.4.1 Plan in chat mode, then build

The wireframe shows a chat window titled "Lovable · Chat mode". It contains three numbered steps: 1. A user prompt: "I want patients to reschedule a booking. Don't build yet. What's the simplest approach, and what could go wrong?". 2. An AI response: "Option A: a Reschedule button on My bookings that reopens the date picker... Risks: double bookings, time zones...". 3. A user confirmation: "Go with option A. Prevent double bookings. Build it.". Below the chat window, a green bar indicates "Switching to agent mode: building the reschedule flow".

- 1 Ask for options and risks first. Chat mode doesn't change code.
- 2 Read the plan. Remove anything you don't need.
- 3 Approve one clear plan, then let agent mode build it.

Planning a feature in chat mode (illustrative wireframe, not a real product screenshot).
Planning first is the biggest credit saver: one good build beats five corrections.

16.4.2 Knowledge: rules for every prompt

Open **Knowledge** (project instructions) and write rules Lovable follows on every change:

- What the app is, who uses it, and the tone of the copy.
- Design rules: colours, fonts, spacing, "mobile-first", "large tap targets".
- Data rules: "sample data only", "prices in ₹", "dates as 23 Oct 2026".
- Constraints: "don't change the booking flow unless asked".

16.4.3 Debugging without panic

Symptom	What to do
Red error in the preview	Click Try to fix once. If it fails twice, restore the last working version and ask in smaller steps.
A change broke something else	Restore from History , then say exactly what must not change.
It keeps repeating a mistake	Add a rule to Knowledge ; describe the bug with a screenshot.
Layout wrong on mobile	Switch the preview to phone size and name the element and the problem.
You are lost	Chat mode: "Explain how this page works and which files control it."

16.4.4 Reusing and sharing

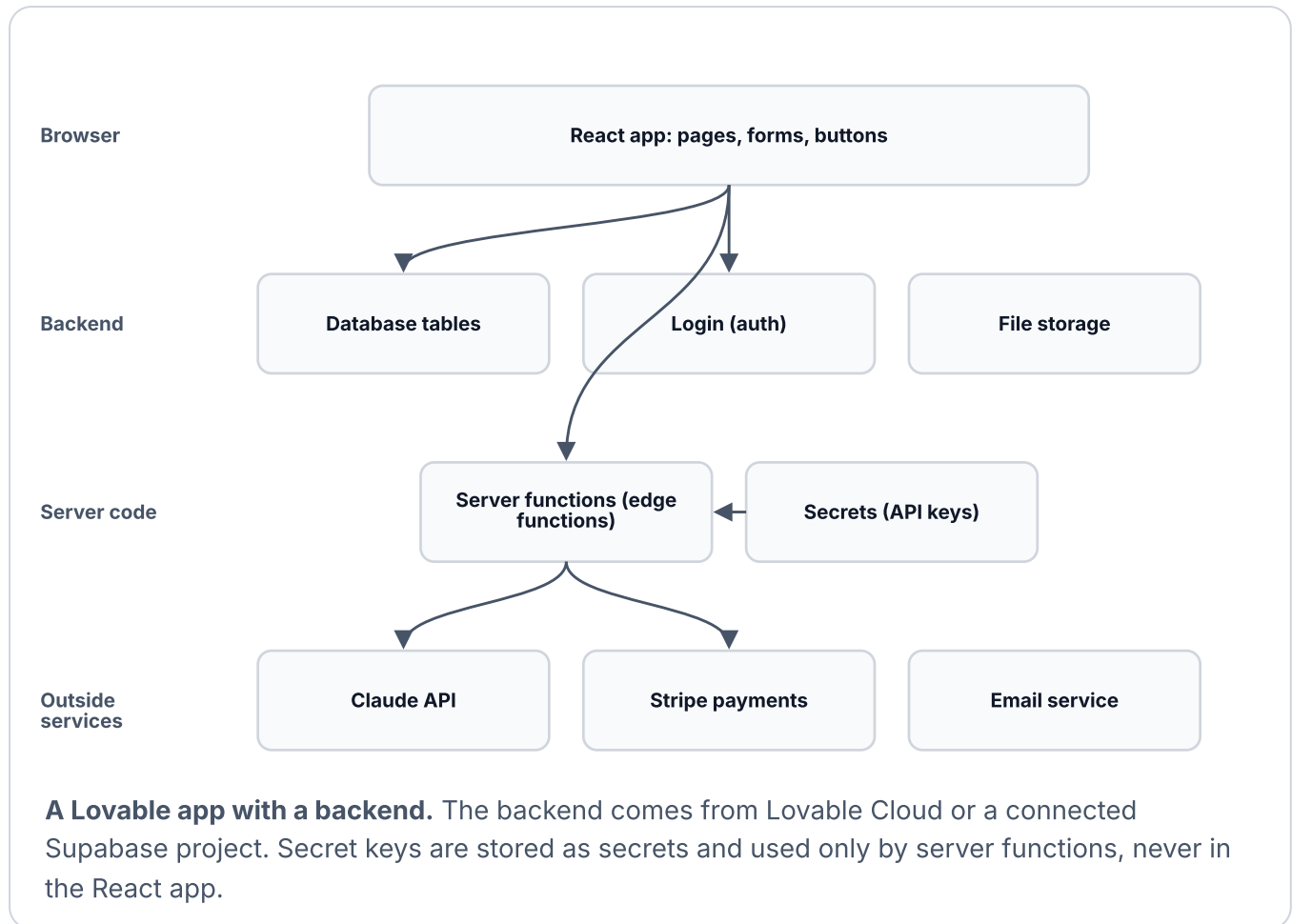
- **Remix** copies a project so you can try a risky idea without touching the original.
- **Share** a preview link for feedback before you publish.
- Upload images, logos or a screenshot of a design, and say "match this".

16.5 Level 3 · Database, login, AI, payments and GitHub

Level 3 · Advanced

Goal: turn the prototype into a working app with real data and accounts, safely.

16.5.1 How a Lovable app with a backend fits together



16.5.2 Database and login

Ask in plain words: "Add a backend. Store bookings in a database with patient name, phone, service, date and time. Add email login so patients see only their own bookings." Lovable creates the tables and the login screens.

Row-level security is not optional

Row-level security (RLS) rules decide which rows each user can read or change. Without them, a logged-in user might read everyone's bookings. Ask: "Turn on row-level security so each user can only read and edit their own bookings; staff can read all." Then **test with two accounts**. Run Lovable's security check before you publish.

16.5.3 AI features with a secret key

To add an "Ask the clinic" chatbot powered by Claude:

1. Get an API key from console.anthropic.com and set a spend limit.

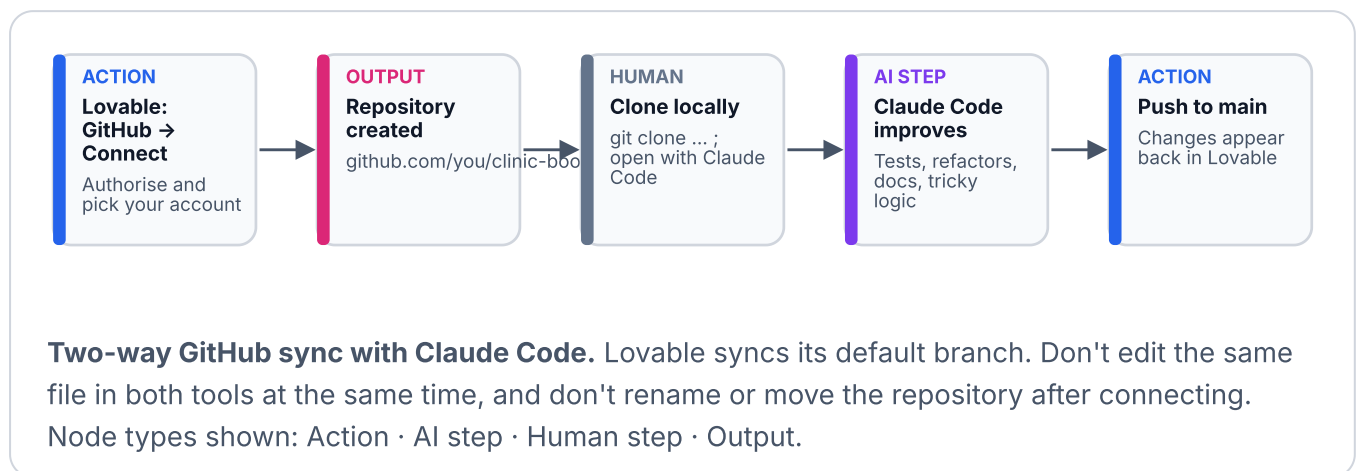
2. In Lovable, add it as a **secret** named `ANTHROPIC_API_KEY`. Never paste it in the chat or the code.
3. Prompt: "Create a server function that calls the Anthropic Messages API with the `ANTHROPIC_API_KEY` secret. Answer only from the clinic FAQ text in the database. Connect it to an 'Ask us' chat button."
4. Test it with five questions, including one it should refuse ("Can you diagnose me?").

Lovable may also offer **built-in AI** that needs no key of your own. It is the quickest option for a demo, with usage limits.

16.5.4 Payments and email

- **Stripe**: "Add a Pay deposit button using Stripe Checkout in test mode." Use Stripe's test cards and switch to live keys only after review.
- **Email**: connect an email service (for example Resend) and "send a confirmation email after a booking".

16.5.5 GitHub sync: work in Lovable and Claude Code



16.5.6 Publish and custom domain

1. Click **Publish**. Check the app in a private browser window.
2. On a paid plan: Settings → **Domains** → add your domain and follow the DNS steps shown.
3. Re-publish after changes. The live site updates only when you publish.

16.6 Labs

Lab 1 • Landing page with waitlist (Free, 20 minutes)

1. Prompt with the five-part template for a waitlist landing page.
2. Make two improvements, one prompt each. Use Visual edits for one text change.
3. Publish and open the link on your phone.

Lab 2 • Clinic booking app with a database (40 minutes)

1. Build the booking app with sample services.
2. Chat mode: "How should we store bookings?" Then approve the plan.
3. "Add a backend and save bookings to the database."
4. Make three bookings, refresh, and confirm they are still there.

Lab 3 • Login and row-level security (30 minutes)

1. "Add email login. Patients see only their own bookings."
2. "Turn on row-level security for bookings: owners read and edit their own; staff role reads all."
3. Create two test accounts. Check account A cannot see B's bookings.
4. Run the security check and fix anything it reports.

Lab 4 • 'Ask us' AI assistant (Pro or API, 30 minutes)

1. Add the clinic FAQ as a database table or Knowledge text.
2. Store `ANTHROPIC_API_KEY` as a secret.
3. Ask for the server function and chat button as described above.
4. Test five normal questions and one it must refuse.

Lab 5 • GitHub sync and Claude Code (30 minutes)

1. Connect GitHub and let Lovable create the repository.
2. Clone it and run Claude Code: "Write a README explaining how to run and deploy this app."
3. Commit and push to main. Check the README appears in Lovable's code view.

Lab 6 · Publish with a custom domain (paid, 20 minutes)

1. Publish the app.
2. Add your domain under Domains and follow the DNS instructions.
3. Wait for the certificate, then test on mobile.

16.7 Prompt library

Goal	Prompt
Polish	"Make this feel premium: consistent spacing, larger headings, subtle shadows, no new features."
Empty and error states	"Add empty, loading and error states to My bookings."
Accessibility	"Check colour contrast and keyboard navigation; fix issues without changing the layout."
Admin view	"Add a staff-only page listing today's bookings with a status dropdown."
Analytics	"Track button clicks for Book and Pay deposit; show counts on the admin page."
Explain	"In chat mode: explain how login works in this app, in plain language."

16.8 Saving credits

- Plan in **chat mode**, then build once.
- Use **Visual edits** for small text, colour and spacing changes.
- One change per prompt, and be specific about the element.
- Restore a version instead of asking Lovable to "undo everything".
- Move heavy refactoring and documentation to **Claude Code** via GitHub.

16.9 Common mistakes

- **Building everything in the first prompt:** start small and add features one at a time.
- **No row-level security:** test with two accounts before anyone real signs up.

- **API keys in prompts or code:** always use secrets and server functions.
- **Publishing real data in a public free project:** free projects are public.
- **Fighting the same bug for many credits:** restore, simplify the request, and add a Knowledge rule.

16.10 Exercises with model answers

Exercise 1 · Write the first prompt

Write a first Lovable prompt for an internal tool that tracks customer feedback by theme.

Model answer

"Build an internal feedback tracker for a SaaS product team (what and who). Pages: Feedback list with filters, Add feedback form, Themes view with counts, Feedback detail (pages). Clean, neutral, compact tables, desktop-first (look). Use 20 sample feedback items across 5 themes and 3 customer segments (data). No login, no integrations yet (not now)."

Exercise 2 · What's the security bug?

A booking app shows "My bookings", but when you change the booking ID in the address bar you can open someone else's booking. What is missing and how do you fix it?

Model answer

Row-level security, or the server check behind it. The app filters bookings in the page but the database still returns any row. Ask Lovable to enable RLS so users can only read rows where they are the owner, then retest with two accounts and the edited address.

16.11 Key takeaways

- Use the **five-part first prompt**, then change one thing at a time.
- **Chat mode** to plan, **agent mode** to build, **History** to undo.
- Real apps need a **backend, login and row-level security**. Test with two accounts.
- Keep keys in **secrets** used by **server functions**.
- **GitHub sync** lets you combine Lovable's speed with Claude Code's depth.

16.12 Check your understanding

CHECK YOUR UNDERSTANDING

Which Lovable mode lets you discuss and plan without changing the code?

- A. Agent mode
- B. Chat mode
- C. Publish
- D. Remix

Why: Chat mode plans and explains; agent mode makes changes.

What stops a logged-in user from reading other users' rows?

- A. A nicer login page
- B. Row-level security rules
- C. A custom domain
- D. More credits

Why: RLS rules decide which rows each user can access.

Where should a Claude API key go in a Lovable app?

- A. In the chat prompt
- B. In the React code
- C. In a secret used by a server function
- D. In the page title

Why: Secrets used only by server-side functions keep keys away from visitors.

16.13 Recommended resources

- [Lovable docs](#) · [Lovable pricing](#)
- [Supabase docs: row-level security](#)
- [Stripe test cards](#)
- Lovable's official YouTube channel and community Discord, for feature walkthroughs and launch-week tutorials

MODULE 17

Replit and Other AI App Builders

Build and host full-stack apps in the browser with Replit Agent, and know when to choose Bolt, v0, Cursor, GitHub Copilot or Figma Make instead.

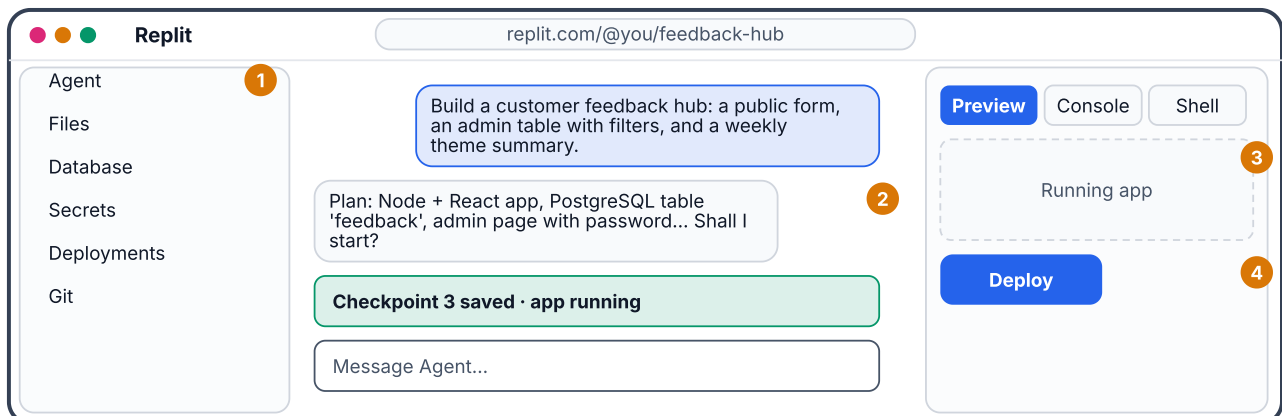
≈ 5 h study

Fast-moving tools

App builders change pricing and features monthly. Treat the comparisons as a starting point and check each vendor's pricing page. Screens are **illustrative wireframes**.

17.1 Replit in one picture

Replit is a complete coding environment in your browser: editor, terminal, database, hosting and an AI **Agent** that can build a full-stack app (front end, back end and database) from a description. Nothing to install, and it works on a phone too.



- 1 Agent builds whole features; the Assistant makes quick edits and answers questions about the code.
- 2 Agent proposes a plan first. Read it and remove what you don't need before it starts.
- 3 Every Agent step saves a checkpoint you can roll back to, including the database.
- 4 Deploy publishes the app on a replit.app address or your own domain.

The Replit workspace (illustrative wireframe, not a real product screenshot). Secrets hold API keys; the Database tab shows a PostgreSQL database you can query.

17.1.1 What it costs (checked 23 Oct 2026)

Plan	Price	Notes
Starter	Free	Daily Agent credits, one published app, limited Agent capability
Core	about \$20 a month billed yearly (about \$25 monthly)	Monthly usage credits, more apps, collaborators, custom domains
Pro / Teams	about \$100 a month	For teams with heavier use

Replit uses **effort-based pricing**: a bigger Agent task costs more than a small one, and usage beyond your plan's credits is billed. Set a **usage alert** in billing settings. Check replit.com/pricing.

17.2 Level 1 · Build your first app

Level 1 · Beginner

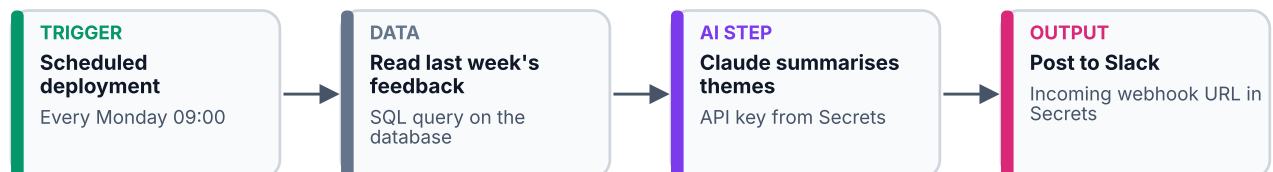
1. Sign up at replit.com, click **Create**, and describe the app to Agent (use the five-part prompt from Class 16: what and who, pages, look, data, not now).
2. Review Agent's plan, remove extras, and approve.
3. Test in the preview. Ask for one change per message.
4. If a change goes wrong, **roll back** to the previous checkpoint.

17.3 Level 2 · Data, secrets and quality

Level 2 · Working PM

- **Database:** ask "Store feedback in the database with date, customer segment, text and theme." Use the Database tab to view rows and run SQL such as `SELECT theme, COUNT(*) FROM feedback GROUP BY theme;`.
- **Secrets:** add `ANTHROPIC_API_KEY` under Secrets. Ask Agent to read it from the environment in server code only.
- **Ask for tests:** "Add tests for the form validation and run them."
- **Admin protection:** "Protect the admin page with login; never show admin data on public pages."
- **Explain the code:** ask Assistant "Which file handles the form? Explain in plain language."

17.4 Level 3 · Deploy, schedule and integrate



Weekly AI theme digest on Replit. Replit's scheduled deployments run a script on a timetable. The same digest can also be built in n8n (Class 20). Node types shown: AI step · Data · Output · Trigger (starts the flow).

Level 3 · Advanced

- **Deployments:** choose the type the app needs: static site, autoscale web app (scales with traffic), reserved machine (always on) or scheduled job.
- **Custom domain:** Deployments → Settings → link your domain and add the DNS records shown.
- **GitHub:** connect Git to push the code to a GitHub repository, then work on it with Claude Code (Class 15).
- **Integrations:** Slack incoming webhooks, Stripe in test mode, the Claude API through Secrets.

17.5 Labs

Lab 1 • Feedback hub (Free, 30 minutes)

Build the feedback form and admin table from the picture above with sample data. Roll back once on purpose, to learn checkpoints.

Lab 2 • Database and SQL (20 minutes)

Store submissions in the database. Run a `GROUP BY theme` query in the Database tab and ask Agent to show the counts on the admin page as a bar chart.

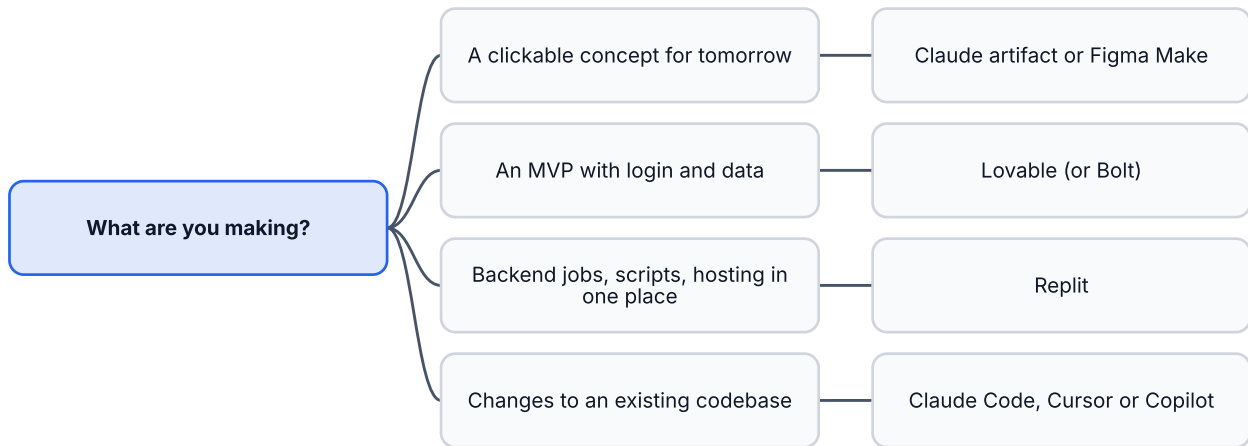
Lab 3 • Weekly digest to Slack (Core or API, 40 minutes)

Create a Slack incoming webhook in your practice workspace. Add it and your Claude key as Secrets. Ask Agent for the scheduled job in the workflow above. Run it manually once and check the Slack message.

17.6 The other AI app builders

Tool	Best for	Backend and data	Free to start	Get the code out
Lovable	Polished MVPs with login and database	Lovable Cloud or Supabase	Yes, 5 credits a day	GitHub sync
Replit	Full-stack apps, scripts, scheduled jobs, hosting	Built-in PostgreSQL	Yes, daily credits	Git / GitHub
Bolt.new	Fast in-browser apps, many frameworks	Supabase and others	Yes, token-limited	Download or GitHub
v0 (Vercel)	Beautiful UI and Next.js apps	Vercel integrations	Yes, limited credits	Deploy to Vercel or copy code
Figma Make	Turning designs into working prototypes	Light; prototype focus	Starter plan limits	Copy code
Claude artifacts	One-page tools and clickable concepts	None (front end only)	Yes	Copy or download
Cursor / Windsurf	AI code editors for real codebases	Whatever your code uses	Free tiers	Your own repository
GitHub Copilot	Coding help and agents inside GitHub and editors	Whatever your code uses	Free tier	Your own repository
Firebase Studio	Google stack apps with Firebase backend	Firebase	Yes	Your own repository
Claude Code	Building and changing real repositories	Whatever your code uses	With Pro or API	Your own repository

Choosing an AI app builder (checked 23 Oct 2026). For a PM, the usual path is Claude artifacts or Lovable for the first concept, then Claude Code on the GitHub repository once the project gets serious.



Which builder should I open?. You can switch later. Builders that sync to GitHub keep your options open.

17.7 Common mistakes

- **No usage alerts:** effort-based pricing can surprise you. Set alerts on day one.
- **Secrets in code:** use the Secrets tab or environment variables.
- **Public admin pages:** protect anything with customer data behind login.
- **Tool-hopping:** pick one builder per project and finish the MVP first.

17.8 Exercises with model answers

Exercise · Pick the builder

(a) A pricing-page concept for a design review in two hours. (b) A nightly script that emails a sales summary. (c) Add a "save for later" feature to your company's existing web app.

Model answer

(a) **Claude artifact** or **Figma Make**: fast, no backend needed. (b) **Replit** scheduled deployment or an **n8n** workflow (Class 20). (c) **Claude Code**, Cursor or Copilot on the existing repository, with engineering review, because it changes production code.

17.9 Key takeaways

- **Replit** gives you editor, AI Agent, database, secrets and hosting in one browser tab.
 - **Checkpoints** make experiments safe; **usage alerts** keep costs safe.
 - Choose a builder by **what you are making**; prefer tools that **export to GitHub**.
-

17.10 Check your understanding

CHECK YOUR UNDERSTANDING

In Replit, what lets you undo an Agent step, including database changes?

- A. Secrets
- B. Checkpoints and rollback
- C. Deployments
- D. The Shell

Why: Agent saves checkpoints you can roll back to.

Which tool is designed to change an existing production codebase?

- A. Claude artifacts
- B. Figma Make
- C. Claude Code or Cursor
- D. A landing page builder

Why: Coding agents and AI editors work inside your existing repository.

17.11 Recommended resources

- [Replit docs](#) · [Replit pricing](#)
- [Bolt.new](#) · [v0 by Vercel](#) · [Cursor docs](#) · [GitHub Copilot docs](#)
- Replit's official YouTube channel for Agent walkthroughs

MODULE 18

Figma and AI Design Tools

Design screens, build click-through prototypes, run workshops in FigJam, generate working prototypes with Figma Make, and hand designs to Claude Code through Figma's MCP server.

≈ 6 h study

Illustrative screens

Figma's interface is updated often. Screens below are **illustrative wireframes**; search the [Figma Help Center](#) if a menu has moved.

18.1 Figma's products

FIGMA DESIGN

- Screens, components, design systems
- Click-through prototypes

FIGJAM

- Whiteboard for workshops
- Journey maps, affinity maps, retros

DEV MODE

- Specs, measurements and code hints for engineers
- Includes the MCP server

FIGMA MAKE

- Prompt or design → working prototype
- AI writes the code

FIGMA SLIDES

- Presentations built from design files

COMMUNITY

- Free templates, UI kits, plugins
- Start from a kit instead of from zero

The Figma family. PMs mostly use FigJam for workshops, Figma Design to review and sketch, and Figma Make for quick working prototypes.

18.1.1 What it costs (checked 23 Oct 2026)

Plan	Price	Notes
Starter	Free	Limited files, a daily AI credit allowance, 3 Figma Make files
Professional	about \$16 a month (full seat); about \$12 (Dev seat); about \$3 (Collab seat)	Unlimited files, Dev Mode, MCP server, monthly AI credits
Organization / Enterprise	from about \$55 a month per full seat	Design systems at scale, admin controls

Viewing and commenting are free. A PM who only reviews and comments can use a free or Collab seat. Check figma.com/pricing.

18.2 Level 1 · The interface and your first screens



The screenshot shows the Figma Design editor interface. On the left is the 'Layers' panel (callout 1) listing 'Frame: Home', 'Header', 'Item list', 'Total bar', 'Frame: Split', and 'Assets'. At the top center are tool buttons: 'Move', 'Frame (F)', 'Shape (R)', 'Text (T)', and 'Comment'. The central 'Canvas' (callout 2) displays 'three phone frames side by side' and includes a tip: 'Tip: press F, then pick a phone size from the right panel to create a frame.' On the right is the 'Properties' panel (callout 3) with 'Design' and 'Prototype' tabs. Under 'Auto layout', 'Direction' is set to 'Vertical', 'Gap' is '16', and 'Padding' is '24'. A 'Fill · Stroke · Effects' section is at the bottom of the panel.

- 1 Layers panel: every frame, group and element. Name layers clearly, for example 'Total bar'.
- 2 Canvas: your screens. Use frames sized to a phone or desktop.
- 3 Properties panel: size, auto layout, colours, text styles. The Prototype tab adds clicks between screens.

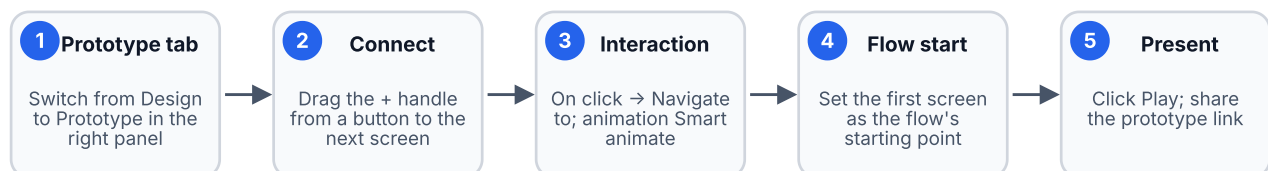
The Figma Design editor (illustrative wireframe, not a real product screenshot). Core shortcuts include F (frame), R (rectangle), T (text), Shift+A (add auto layout).

Level 1 · Beginner

- **Frames** are screens. Start with a phone frame (for example 390 × 844).
- **Auto layout** (Shift+A) stacks items with consistent gaps and padding, and resizes when content changes. Use it for buttons, cards and lists.
- **Components** are reusable elements (a button, a card). Change the main component and every copy updates.
- **Styles and variables** keep colours, text sizes and spacing consistent.
- **Start from a UI kit** in Figma Community, so you are not drawing buttons from scratch.

18.3 Level 2 · Prototypes, reviews and handoff

18.3.1 Click-through prototypes



Make a clickable prototype. Test the prototype with five users before anyone writes code (Module 3: usability testing).

Level 2 · Working PM

- **Variants** group versions of a component (button: default, hover, disabled), so states are designed, not forgotten.
- **Comments**: press C and click to leave feedback on an exact spot; mention people with @.
- **Dev Mode** shows engineers measurements, colours, assets and code hints. Mark frames **Ready for dev**.
- **Branching** (paid plans) lets designers explore without breaking the main file.
- **PM review checklist**: every state designed (empty, loading, error, success)? Copy final? Accessible contrast and tap sizes? Edge cases from the PRD covered?

18.3.2 FigJam for workshops

AFFINITY MAP

- Paste interview quotes as stickies
- Group into themes
- Vote with stamps

USER JOURNEY MAP

- Stages across the top
- Actions, thoughts, pain points, opportunities

PRIORITISATION 2x2

- Impact versus effort
- Move stickies live with the team

RETRO

- Went well, didn't, try next
- Timer and voting

FLOW DIAGRAM

- Shapes and connectors for user or system flows

AI IN FIGJAM

- Summarise or sort stickies; generate a first board from a prompt (where available)

FigJam boards PMs run. Templates for all of these are in FigJam's template picker and Figma Community.

18.4 Level 3 · Figma AI, Figma Make and MCP

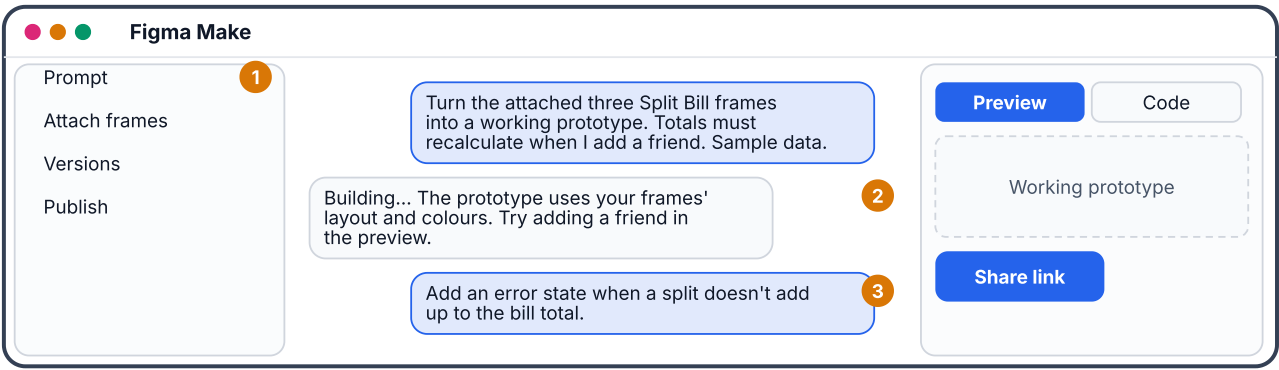
Level 3 · Advanced

Goal: go from idea or design to a working prototype, and let Claude Code build directly from your Figma frames.

18.4.1 Figma AI

Figma's AI features (they vary by plan and change often) help with tasks such as generating a first draft of a layout, renaming layers, rewriting or translating text, removing image backgrounds and searching for similar designs. Each use spends **AI credits** from your plan.

18.4.2 Figma Make: prompt to working prototype



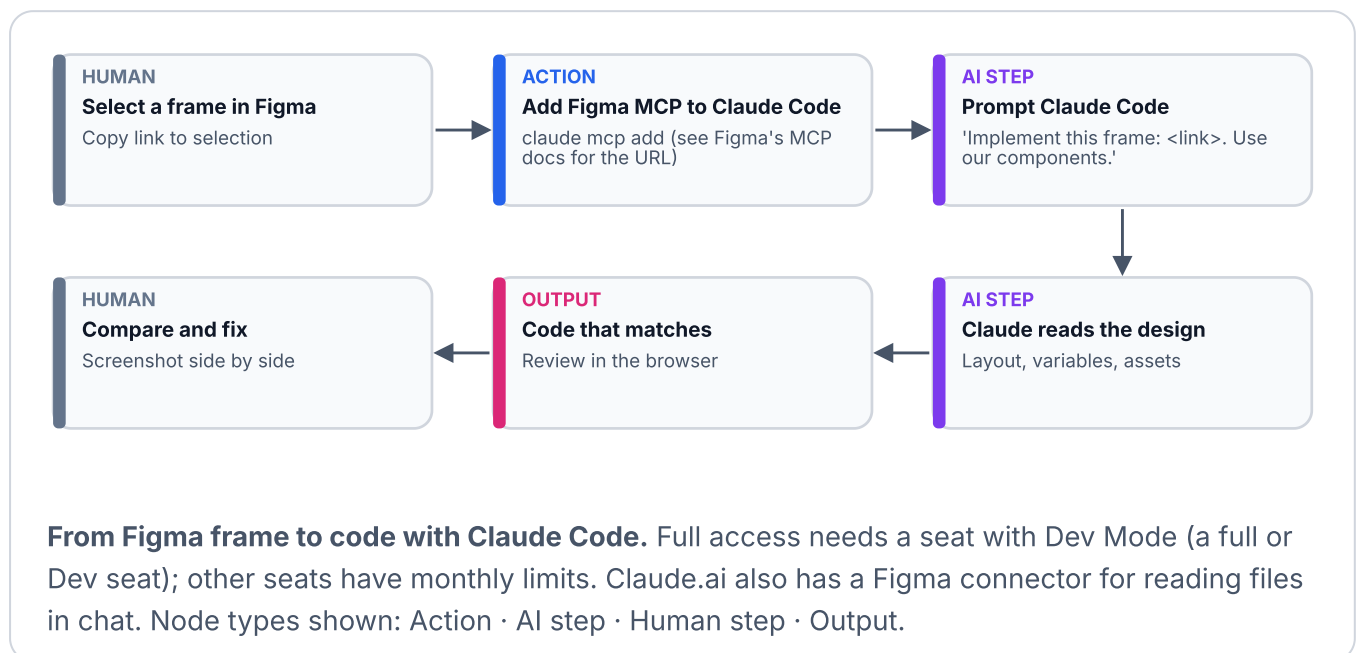
The Figma Make interface is shown with a sidebar on the left containing 'Prompt', 'Attach frames', 'Versions', and 'Publish'. The main area has three numbered steps: 1. 'Prompt' (Turn the attached three Split Bill frames into a working prototype. Totals must recalculate when I add a friend. Sample data.), 2. 'Building...' (The prototype uses your frames' layout and colours. Try adding a friend in the preview.), and 3. 'Add an error state when a split doesn't add up to the bill total.' On the right, there are buttons for 'Preview', 'Code', 'Share link', and a 'Working prototype' placeholder.

- 1 Start from a prompt, or attach existing frames so the result matches your design.
- 2 Make writes real code. Test interactions in the preview.
- 3 Ask for edge cases and states, one per message.

Figma Make (illustrative wireframe, not a real product screenshot). Make is ideal for testing behaviour (calculations, flows) that a click-through prototype can't show.

18.4.3 Figma MCP server + Claude Code

The **Figma MCP server** lets Claude Code (and other AI tools) read your design: layout, components, variables and assets. Claude can then build the screen in code to match.



18.5 Labs

Lab 1 · Three screens with auto layout (Free, 40 minutes)

1. Duplicate a free mobile UI kit from Figma Community.
2. Create three phone frames: Bill, Split with friends, Request payment.
3. Build the list items and buttons with auto layout and one Button component with three variants.

Lab 2 · Clickable prototype and usability test (30 minutes)

1. Connect the three screens in the Prototype tab.
2. Present and share the link.
3. Test with two people using the think-aloud method. Note where they hesitate.

Lab 3 · FigJam affinity map (30 minutes)

1. Open a FigJam affinity-map template.
2. Paste 20 interview quotes as stickies (use Class 14 Lab 2 output).
3. Group them into themes, name each theme, and vote on the top two.

Lab 4 · Figma Make prototype (Free or paid, 30 minutes)

1. Open Figma Make, attach your three frames.
2. Ask for a working prototype with live total recalculation.
3. Add one edge case, then share the link.

Lab 5 · Claude Code builds from Figma (Professional, 40 minutes)

1. Follow Figma's MCP guide to add the server to Claude Code.
2. Copy the link to your Bill frame.
3. Ask Claude Code: "Implement this frame as a responsive HTML page."
4. Compare with the design and ask for fixes until it matches.

18.6 PM best practices

- **Low fidelity first:** boxes and real copy before colour and polish.
- **Design the states,** not just the happy path.
- **Name layers and frames** so engineers and AI tools understand them.
- **Comment, don't redesign:** describe the problem; let the designer solve it.
- **Keep one source of truth:** link the Figma file in the PRD, and the PRD in the Figma file.

18.7 Exercises with model answers

Exercise · What to use

(a) Run a 45-minute remote workshop to sort 60 interview quotes. (b) Show a stakeholder how the split-bill calculation behaves with three people. (c) Hand a finished screen to engineering.

Model answer

(a) **FigJam** affinity map. (b) **Figma Make** (a click-through prototype can't calculate). (c) Mark the frame **Ready for dev** in **Dev Mode**, and optionally let engineers use the **MCP server** with their coding agent.

18.8 Key takeaways

- **Frames, auto layout and components** are the three skills that make Figma easy.
- **Prototype and test** before building. **FigJam** is your workshop room.
- **Figma Make** turns designs into working prototypes; the **MCP server** lets Claude Code build from your frames.

18.9 Check your understanding

CHECK YOUR UNDERSTANDING

Which feature makes a list of cards keep consistent spacing as items change?

- A. Comments
- B. Auto layout
- C. Dev Mode
- D. FigJam stamps

Why: Auto layout stacks items with fixed gaps and padding and resizes with content.

You need to show a live calculation in a prototype. Use...

- A. A click-through prototype
- B. Figma Make
- C. A FigJam board
- D. A PDF export

Why: Figma Make writes working code, so values can be calculated.

What gives Claude Code direct access to a Figma frame's layout and variables?

- A. Exporting a PNG
- B. The Figma MCP server
- C. A Slack message
- D. Figma Slides

Why: The MCP server exposes design context to AI coding tools.

18.10 Recommended resources

- [Figma Help Center](#) · [Figma pricing](#)
- [Figma MCP server collection](#)
- [Figma's official YouTube channel](#) and Figma Community UI kits
- Module 5 (UX and design collaboration) for the theory behind these labs

MODULE 19

Notion, PRD Tools and AI Documentation

Build a product workspace in Notion, write and critique PRDs with AI, turn specs into tickets, and connect your docs to Claude.

≈ 6 h study

Plans and AI features change

Notion's AI packaging changed several times in 2026. Check notion.com/pricing before buying. Screens are **illustrative wireframes**.

19.1 Notion in one page

Notion is a workspace for documents and databases. Everything is a **page** made of **blocks** (text, headings, checklists, tables, images, embeds). A **database** is a collection of pages with properties, shown in different **views**. PMs use Notion for PRDs, roadmaps, research repositories, meeting notes and decision logs.

19.1.1 Plans (checked 23 Oct 2026)

Plan	Price	AI
Free	\$0	A limited AI trial
Plus	per member, monthly	A limited AI trial
Business	about \$20 per member a month	Full Notion AI: AI writing and Q&A, AI meeting notes, enterprise search, research mode, agents
Enterprise	custom	Business features plus admin and security controls

Custom AI **agents** can use a separate credit meter on top of the plan. If your company already pays for Claude, you can do much of the AI work with the **Claude connector** instead (Lab 4).

19.2 Level 1 · Pages, blocks and databases

Product hub

Roadmap

PRDs

Research

Decisions

Meeting notes

Board Table Timeline Calendar

Now · Split payments (RICE 42) Asha

Now · Saved addresses (RICE 35) Ravi

Next · Loyalty points (RICE 28) Meena

Later · Group ordering (RICE 19) —

+ New

Properties

Status Now

Quarter Q4 2026

Owner Asha

RICE score 42

PRD ↗ Split payments PRD

1 Each card is a page. Click + New and choose a template so every item starts with the same structure.

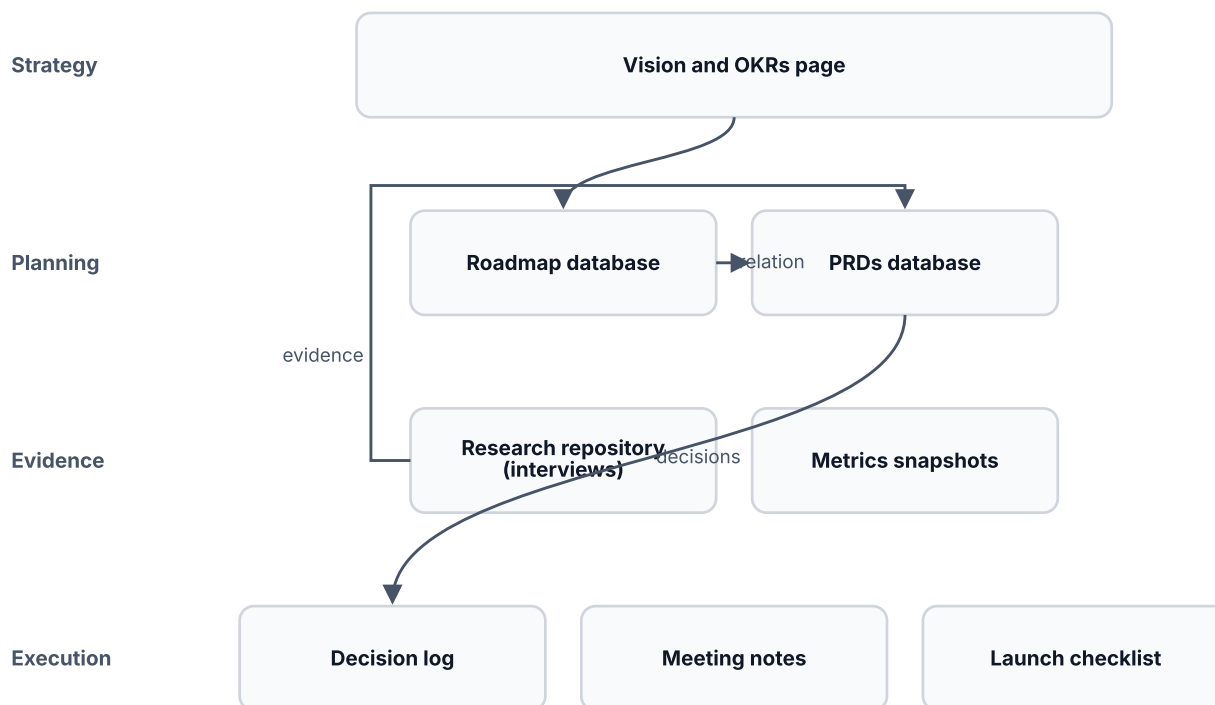
2 Properties hold the data you filter and sort by. 'PRD' is a relation that links to the PRDs database.

A Notion roadmap database (board view) (illustrative wireframe, not a real product screenshot). Same data, four views: board for stand-ups, table for scoring, timeline for planning, calendar for launches. The roadmap items are hypothetical.

Level 1 · Beginner

- Type **/** to insert any block (heading, to-do, toggle, table, callout, database).
- Drag the **⋮** handle to move blocks; turn any block into another type.
- Create a database with **/database**. Add properties: Status (select), Owner (person), Quarter (select), Score (number).
- Add **views** with **+** next to the view tabs, each with its own filter and sort.
- Share a page with **Share**; choose who can view, comment or edit.

19.3 Level 2 · A product hub that runs itself



A PM product hub in Notion. Relations connect the databases, so a PRD shows the roadmap item it belongs to and the interviews that support it.

Level 2 · Working PM

- **Templates:** create a PRD template inside the PRDs database (use `templates/01-prd-template.md` from this handbook). Every new PRD starts complete.
- **Relations and rollups:** link PRDs to roadmap items and interviews; roll up "number of supporting interviews" onto each PRD.
- **Linked views:** show "My open PRDs" or "This quarter's roadmap" on your personal home page.
- **Buttons and automations:** a button that creates this week's meeting note; a database automation that notifies Slack when Status changes to "Launched".
- **Integrations:** embed Figma frames, link GitHub pull requests, sync Jira or Linear issues.

19.4 Level 3 · Notion AI and connecting to Claude

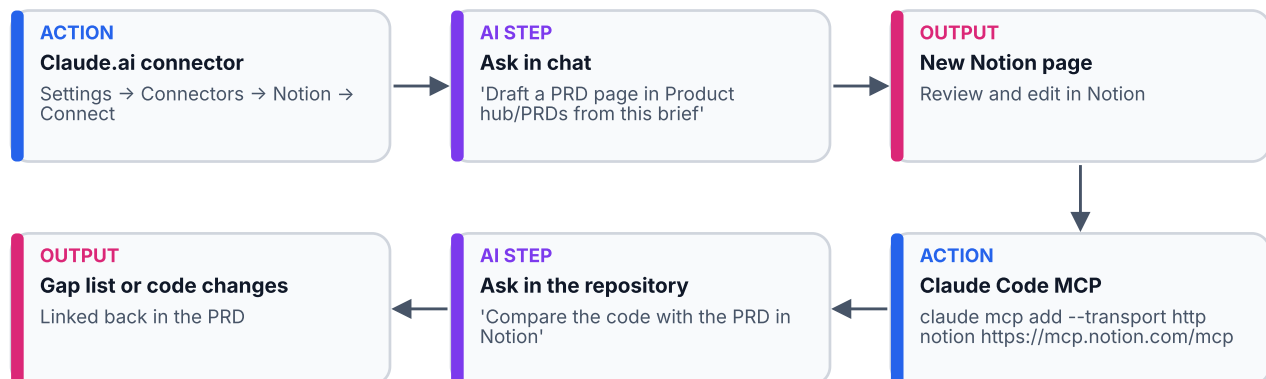
Level 3 · Advanced

Goal: let AI fill properties, summarise research, draft and critique PRDs, and work with your Notion pages from Claude and Claude Code.

19.4.1 Notion AI (Business plan)

- **Ask** questions across your workspace: "What did we decide about refunds?"
- **Write and edit** inside a page: draft, shorten, change tone, translate.
- **AI autofill** database properties, for example a "Theme" and "Sentiment" for every interview note.
- **AI meeting notes**: transcribe and summarise meetings into action items.
- **Research mode** and **agents** for multi-step tasks across pages.

19.4.2 Claude with Notion



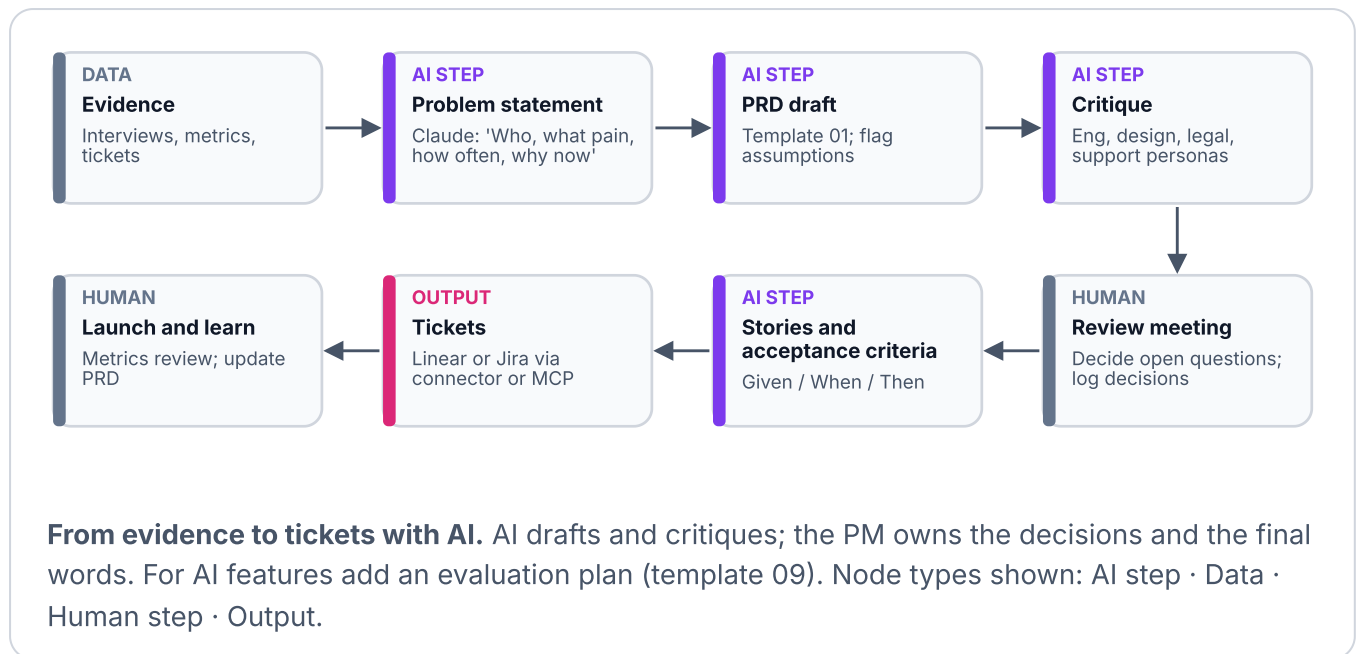
Two ways to connect Claude and Notion. You choose which pages the connection can access when you sign in. Start with a practice teamspace. Node types shown: Action · AI step · Output.

19.5 PRD tools compared

Tool	Strength	AI help	Best when
Notion + Notion AI	Docs and databases in one place	Draft, autofill, Q&A, meeting notes	Your team already lives in Notion
Claude Projects + a PRD skill	Strongest reasoning and critique	Draft, critique, rewrite, tickets	You want quality writing and review; export to any tool
ChatPRD	Purpose-built AI PRD assistant	PRD drafts, coaching, templates	You want a guided PRD writer
Confluence + Atlassian Rovo	Enterprise wiki linked to Jira	Draft, summarise, search	Your company runs on Jira
Coda	Docs with formulas and tables	AI columns and writing	PRDs that need live calculations
Linear (docs and projects)	Fast issue tracking with project docs	Summaries and drafting	Engineering uses Linear
Jira Product Discovery	Ideas, insights and prioritisation	Summaries via Rovo	Discovery feeding a Jira delivery team
Productboard	Feedback to roadmap	Feedback insights	High volume of customer feedback
Google Docs + Gemini / Word + Copilot	Familiar documents	Drafting inside the doc	Company uses Google or Microsoft suites

Tools for writing and managing PRDs (checked 23 Oct 2026). The tool matters less than the habit: one template, linked evidence, explicit decisions, and a review loop.

19.6 An AI-assisted PRD workflow



The critique prompt that finds gaps

"Review this PRD as four people: a skeptical engineering lead, a designer, a lawyer and a support lead. For each, list the top three questions and risks. Then list missing edge cases and any requirement that isn't testable."

19.7 Labs

Lab 1 · Product hub in Notion (Free, 40 minutes)

1. Create a page "Product hub" with sub-pages for Roadmap, PRDs, Research and Decisions.
2. Make Roadmap a database with Status, Owner, Quarter and RICE score; add board, table and timeline views.
3. Add five sample items.

Lab 2 · PRD template (20 minutes)

1. In the PRDs database, create a template from `templates/01-prd-template.md`.
2. Relate PRDs to Roadmap items.
3. Create one PRD from the template for "Split payments".

Lab 3 · Research repository with AI tagging (30 minutes)

1. Create an Interviews database: Date, Segment, Notes, Theme, Sentiment.
2. Paste five interview notes.
3. With Notion AI (Business): set Theme and Sentiment to AI autofill. Without it: ask Claude to tag the notes and paste the results back.

Lab 4 · Claude writes into Notion (15 minutes)

1. In Claude: Settings → Connectors → Notion → Connect; allow the Product hub pages.
2. "Create a PRD page in Product hub/PRDs for 'Saved addresses' using the template structure. Mark every assumption."
3. Open Notion, review and edit the page.

Lab 5 · PRD to tickets (20 minutes)

1. Ask Claude: "Turn the Split payments PRD into user stories with Given/When/Then acceptance criteria, grouped into three releases."
2. If you use Linear or Jira, connect it and create the tickets in a test project; otherwise paste into a table.

Lab 6 · Compare two PRD assistants (30 minutes)

Write the same one-paragraph brief into Claude (with Template 01) and into ChatPRD or Notion AI. Compare structure, accuracy, assumptions flagged and edits needed. Note which you'd use for what.

19.8 Common mistakes

- **A PRD graveyard:** no template, no status, no owner. Use a database with Status and Owner.
- **AI-written PRDs nobody reviewed:** always run the critique and a human review meeting.
- **Unlinked evidence:** relate PRDs to interviews and metrics, so decisions can be traced.
- **Sharing too widely:** check page permissions before connecting AI tools.

19.9 Exercises with model answers

Exercise · Make it testable

Rewrite this requirement with acceptance criteria: "Split bill should be easy."

Model answer

"As a diner, I can split a bill equally or by item among up to 10 people." **Given** a bill of ₹1,200 and three people splitting equally, **when** I confirm, **then** each person is asked for ₹400. **Given** item-wise split, **when** an item is unassigned, **then** I see 'Assign all items' and cannot confirm. **Given** amounts that don't divide evenly, **then** the remainder is added to the payer's share and shown clearly.

19.10 Key takeaways

- Notion = **pages + databases + views**. Relations turn separate lists into a connected product hub.
- Use **one PRD template**, link **evidence**, log **decisions**.
- AI drafts and critiques; **you decide**. Connect Claude through the **connector** or **MCP**.

19.11 Check your understanding

CHECK YOUR UNDERSTANDING

Which Notion feature links a PRD to its roadmap item?

- A. A toggle block
- B. A relation property
- C. A callout
- D. Page cover

Why: Relations connect pages across databases.

What makes a requirement testable?

- A. Using the word 'easy'
- B. Acceptance criteria such as Given / When / Then
- C. A longer paragraph
- D. Adding an emoji

Why: Acceptance criteria state observable conditions and outcomes.

19.12 Recommended resources

- [Notion Help Center](#) · [Notion pricing](#) · [Notion MCP](#)
- [ChatPRD](#) · [Atlassian Rovo](#) · [Linear docs](#)
- Handbook templates: 01 PRD, 02 one-pager, 09 AI PRD and evaluation

MODULE 20

Automation with Zapier, Make and n8n

Connect Gmail, Slack, Google Sheets, forms and AI so routine work runs by itself, and learn to choose between Zapier, Make and n8n.

≈ 10 h study

Use practice accounts

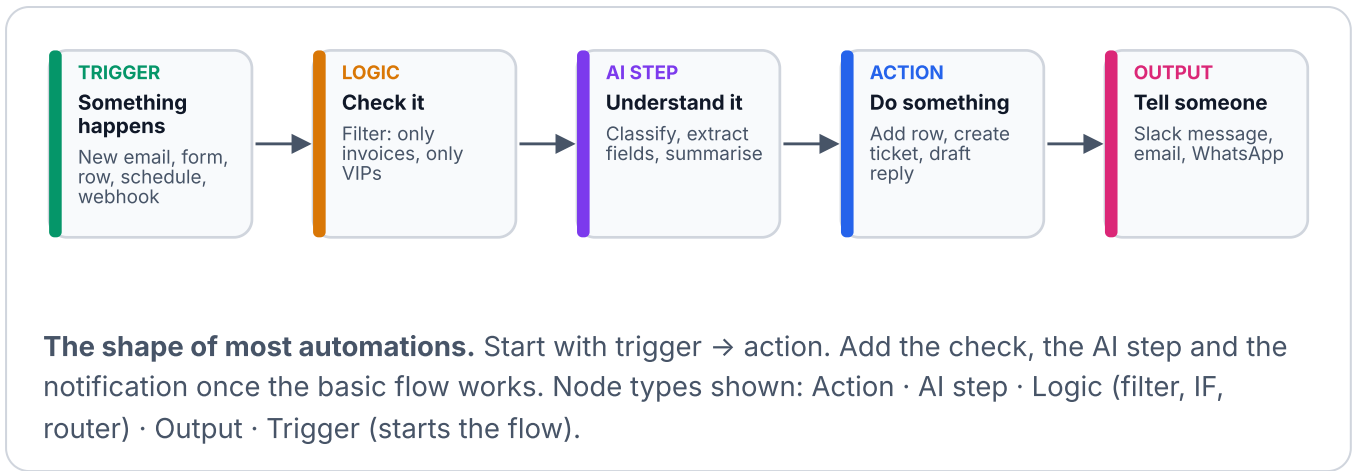
Automations act without asking. Build every lab with a **practice Gmail** and a **practice Slack workspace**, and never point a test at real customers. Screens are **illustrative wireframes**.

20.1 The building blocks

Every automation tool uses the same ideas with different names.

Idea	What it means	Zapier	Make	n8n
Automation	The whole flow	Zap	Scenario	Workflow
Trigger	What starts it	Trigger	Trigger module (first)	Trigger node
Action	A step that does something	Action	Module	Node
Condition	Only continue if...	Filter	Filter on a route	IF / Filter node
Branch	Different paths for different cases	Paths (paid)	Router	Switch / IF node
Loop	Repeat for each item	Looping (paid)	Iterator + Aggregator	Loop Over Items
Webhook	Start from another app's HTTP call	Webhooks by Zapier (paid)	Custom webhook	Webhook node
Schedule	Run at set times	Schedule by Zapier	Scheduling	Schedule Trigger
AI step	Ask a model to classify, summarise or decide	AI by Zapier / Agents	AI modules	AI Agent and chat model nodes
Billing unit	What you pay for	Tasks (successful actions)	Credits	Executions (cloud); free self-hosted

Same idea, three names. Learn the ideas once, and you can use any of the three tools.



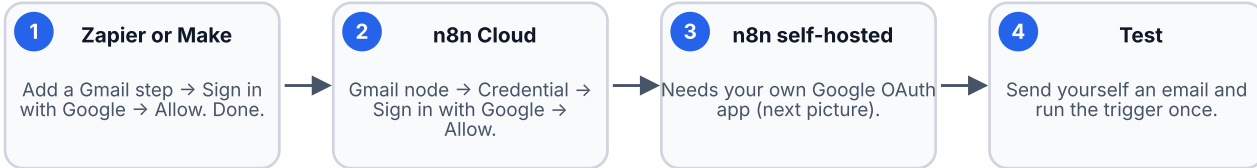
20.2 Choosing the tool

	Zapier	Make	n8n
Ease for beginners	Easiest	Medium (visual canvas)	Medium to hard
Free tier	100 tasks a month, two-step Zaps	1,000 credits a month, 2 active scenarios, 15-minute minimum interval	Community Edition free to self-host (you pay the server)
First paid tier	Professional	Core	Cloud Starter, about €20 to €24 a month
App integrations	8,000+	2,000+	Hundreds of nodes plus any API via HTTP Request
Complex logic	Paths, loops on paid plans	Strong: routers, iterators, data stores	Strongest: code, branches, sub-workflows
AI agents	Zapier Agents, AI steps	AI modules and agents	AI Agent node with memory and tools; MCP
Data stays with you	No (cloud)	No (cloud)	Yes, if self-hosted
Best for	Quick wins for non-technical teams	Visual, multi-branch flows on a budget	Power users, AI agents, high volume, privacy

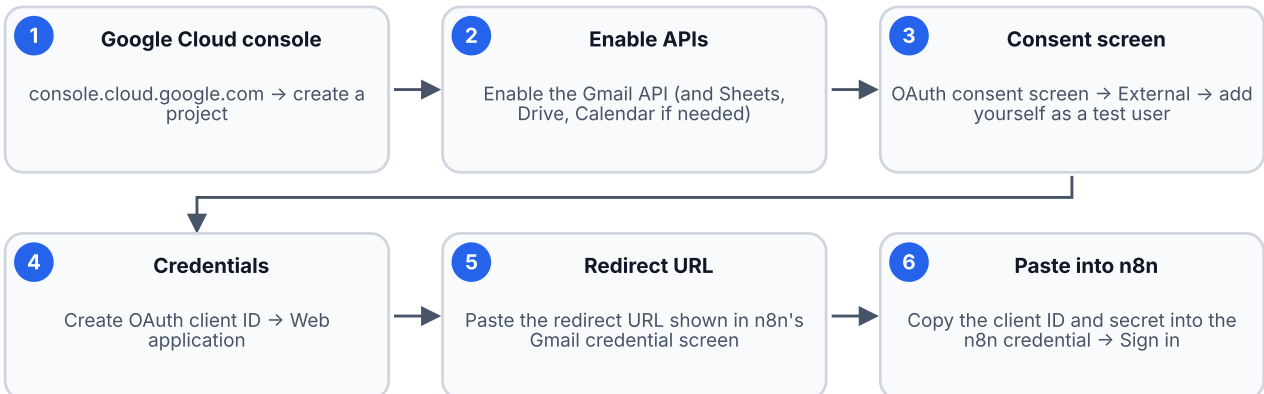
Zapier vs Make vs n8n (checked 23 Oct 2026). A good path is Zapier for your first two automations, then n8n for anything with AI agents or high volume. Check each pricing page before paying.

20.3 How to connect Gmail and Slack (step by step)

20.3.1 Gmail

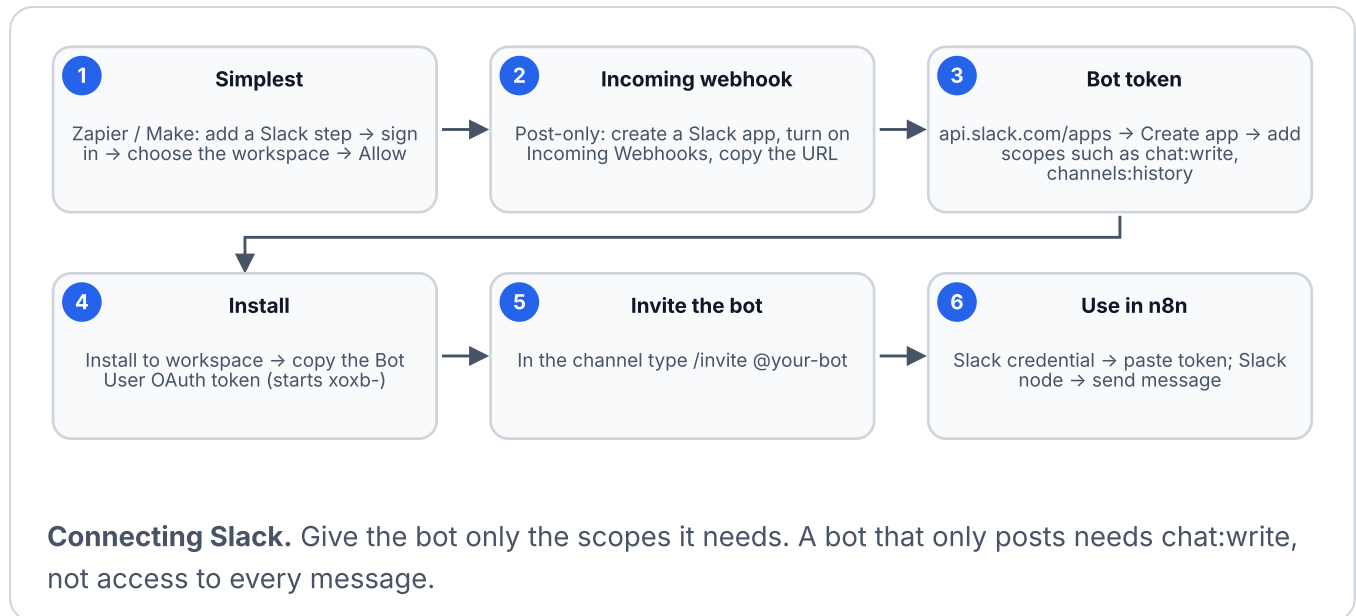


Connecting Gmail. Use a practice Gmail first. Google may show an 'unverified app' warning for self-made OAuth apps; that's expected for personal testing.



Google OAuth app for self-hosted n8n (one time, 15 minutes). The exact menu names in Google Cloud change occasionally. n8n's docs page for Google credentials has the current steps.

20.3.2 Slack



20.4 Zapier · Basic to advanced

Zapier | zapier.com/editor

1. Trigger · Gmail · New email matching search

Search string: from:boss@example.com is:important

2. Action · Slack · Send channel message

Channel: #practice-alerts

Message: Important email from {{From}}: {{Subject}}

Test step

Publish Zap

- 1 Choose the app and the event that starts the Zap, sign in, and set options.
- 2 Map fields from the trigger into the action. The {{...}} pills are data from step 1.
- 3 Test each step with real sample data before publishing.
- 4 Publish turns it on. Check Zap history for runs and errors.

A two-step Zap: Gmail to Slack (illustrative wireframe, not a real product screenshot). Free plans allow two-step Zaps like this one. Multi-step Zaps, filters on paths and webhooks need a paid plan.

Zapier Level 1

Build the Gmail → Slack Zap above. Then a second: **Google Forms → Google Sheets** (new response → add row).

Zapier Level 2 (paid)

- Add a **Filter** (only continue if Subject contains "invoice").
- Use **Formatter** to clean text, dates and numbers.
- Add **Paths** for different cases (VIP vs normal).
- Add an **AI step** (AI by Zapier) to summarise or classify before posting.

Zapier Level 3

- **Webhooks by Zapier** to receive data from your Lovable or Replit app.
- **Zapier Agents** for multi-step tasks described in plain language.
- **Zapier MCP** gives Claude access to Zapier's app actions as tools.
- **Tables and Interfaces** for small internal tools without a separate database.

20.5 Make • Basic to advanced

The wireframe shows a 'Make' scenario titled 'make.com/scenarios/lead-router'. It features a canvas with modules connected by lines. The flow starts with 'Google Forms · Watch responses' (Trigger), followed by a 'Router' module which 'splits into 2 routes'. Route A has a filter 'budget > ₹50,000' and leads to 'Slack #sales-vip'. Route B leads to 'Google Sheets add row'. A 'Run once' button is present. On the right, a 'Module settings' panel shows a map: '{{1.email}}', '{{1.budget}}' and 'Operations used: 3 credits'. A scheduling toggle is set to 'every 15 minutes'.

Canvas: modules are circles joined by lines; data flows left to right.

Google Forms · Watch responses **Trigger**

Router **splits into 2 routes**

Route A (filter: budget > ₹50,000) → Slack #sales-vip

Route B (all others) → Google Sheets add row

Run once

Scheduling: every 15 minutes

Module settings

Map: {{1.email}}, {{1.budget}}

Operations used: 3 credits

- 1 Each module is one app action. Click between modules to add a filter.
- 2 Run once with a test response and inspect the bubbles showing data in and out of each module.
- 3 Turn on scheduling. On the free plan the shortest interval is 15 minutes.

A Make scenario with a router (illustrative wireframe, not a real product screenshot). Iterators split a list into items; aggregators combine items back into one bundle. Data stores keep small tables between runs.

Make essentials

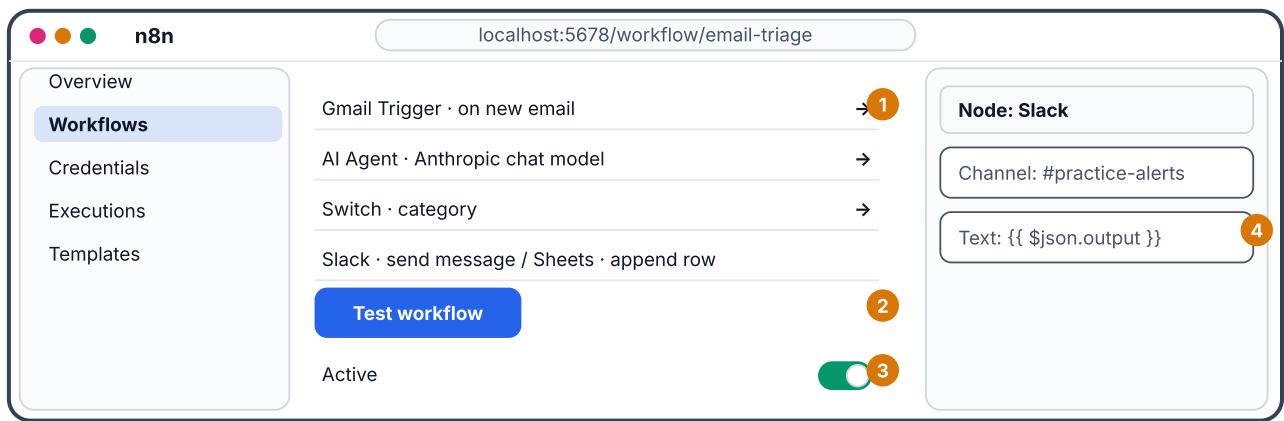
Router for branches, **filters** on each route, **iterator + aggregator** for lists (for example every attachment in an email), **error handlers** (resume, ignore, rollback) on risky modules, and **data stores** for memory between runs. Watch your **credit** use per run on the scenario's history page.

20.6 n8n · Basic to advanced

20.6.1 Getting n8n

Option	How	Cost
On your laptop	Install Node.js, then run <code>npx n8n</code> and open <code>http://localhost:5678</code>	Free
Docker	<code>docker run -it --rm -p 5678:5678 n8nio/n8n</code>	Free
Small cloud server	Docker on a small VPS or a free-tier VM (runs 24/7)	Server cost only
n8n Cloud	Sign up at <code>n8n.io</code>	Paid plans after a trial

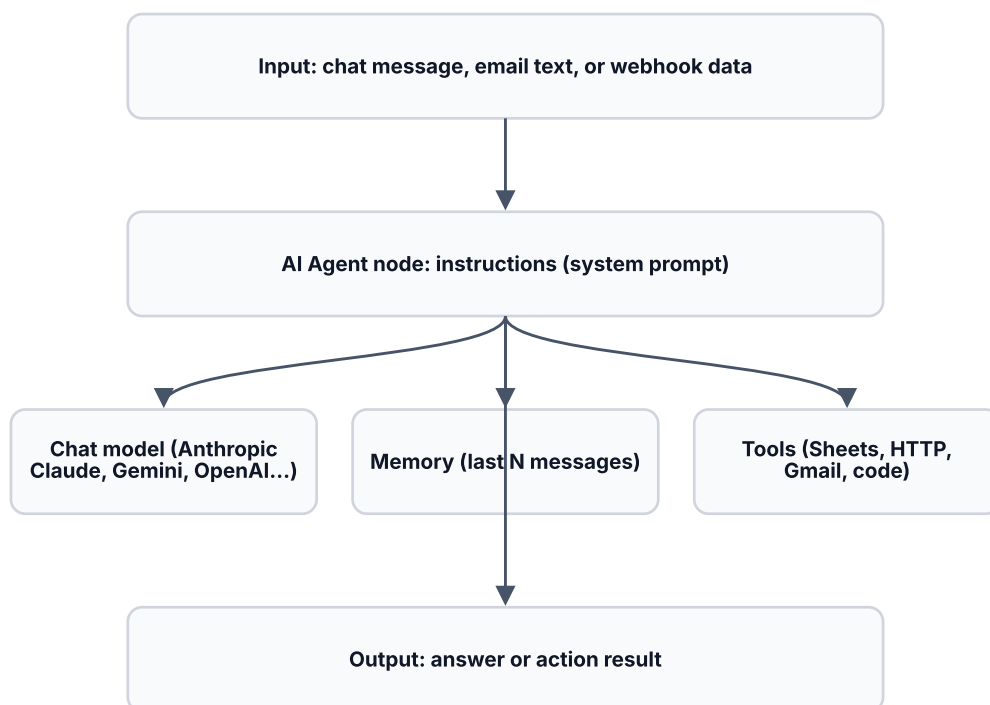
Claude.ai cannot reach `localhost`. For the MCP lab use n8n Cloud, a server, or a tunnel (for example Cloudflare Tunnel or ngrok's free tier).



- 1 Nodes run left to right. Click + to add a node; pick apps, logic or AI nodes.
- 2 Test workflow runs it once. Each node shows its input and output data, which is the best way to debug.
- 3 Active turns on the triggers so it runs automatically.
- 4 Expressions in {{ }} insert data from earlier nodes. \$json is the current item's data.

The n8n editor (illustrative wireframe, not a real product screenshot). Executions shows every past run with its data, so you can see exactly where and why something failed.

20.6.2 AI agents in n8n



The n8n AI Agent node. Add an Anthropic credential with your Claude API key to use Claude as the chat model. Free alternatives for practice include Gemini's free tier and local models through Ollama.

20.6.3 n8n and MCP

- **MCP Server Trigger** node: turns a workflow into an MCP server that exposes the **tool nodes attached to it** (for example a Custom n8n Workflow Tool that runs another workflow). Add its production URL to Claude (Settings → Connectors → Add custom connector) or to Claude Code (`claude mcp add --transport http ...`). Claude can then call your workflow as a tool.
- **MCP Client Tool** node: lets an n8n AI agent use tools from other MCP servers.
- Protect MCP and webhook URLs with **authentication** (header token), so only you can trigger them.

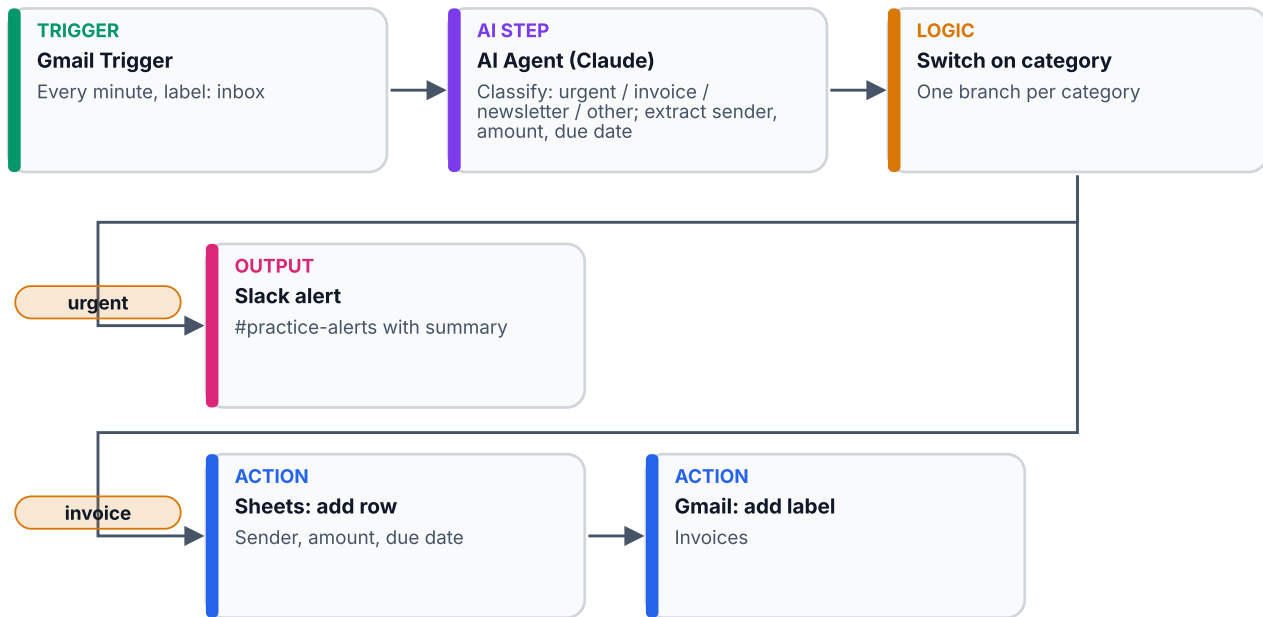
20.7 Labs

Lab 1 · Gmail → Slack in Zapier (Free, 15 minutes)

Build the two-step Zap shown above with a practice Gmail and Slack. Send yourself a matching email and check Slack within a few minutes.

Lab 2 · Form → Sheet → route in Make (Free, 30 minutes)

1. Create a Google Form with Name, Email and Budget.
2. Make: Watch responses → Router → Route A filter Budget > 50000 → Slack; Route B → Sheets add row.
3. Submit two test responses and check both routes.



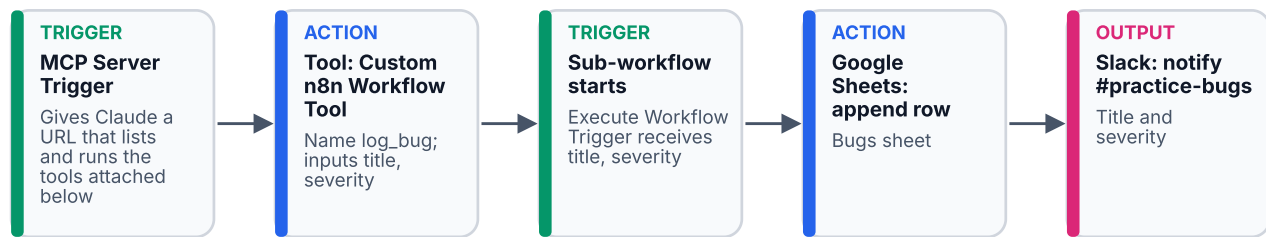
Lab 3 · AI email triage in n8n. Ask the AI step to reply in JSON with fixed fields, so the Switch node can read the category reliably. Node types shown: Action · AI step · Logic (filter, IF, router) · Output · Trigger (starts the flow).

Lab 3 · AI email triage (n8n, 45 minutes)

1. Gmail Trigger (practice account) → AI Agent with the Anthropic chat model.
2. Instructions: "Classify the email as urgent, invoice, newsletter or other. Reply only with JSON: {category, sender, amount, due_date, summary}." Turn on structured output or add a Structured Output Parser.
3. Switch on `{{ $json.output.category }}` → Slack for urgent; Sheets + Gmail label for invoices.
4. Send four test emails, one per category. Check each lands correctly in Executions.

Lab 4 · Morning digest (n8n, 30 minutes)

Schedule Trigger 08:30 → Google Calendar (today's events) + Gmail (unread from the last 12 hours) → AI Agent "Write a 120-word brief with what to prepare" → Slack DM to yourself.



Lab 5 · A tool Claude can call. The MCP Server Trigger doesn't pass data to a next node like other triggers; it exposes the tool nodes connected to it. The workflow tool calls a small sub-workflow that does the work. Add the trigger's production URL to Claude as a custom connector. Node types shown: Action · Output · Trigger (starts the flow).

Lab 5 · Give Claude a tool (n8n Cloud or server, 30 minutes)

1. Build a small sub-workflow: **Execute Workflow Trigger** (inputs `title`, `severity`) → Google Sheets append row → Slack message.
2. In a second workflow add an **MCP Server Trigger** and attach a **Custom n8n Workflow Tool** that calls the sub-workflow. Name it `log_bug` and describe it and its two inputs clearly, because Claude reads that description.
3. Turn on authentication in the trigger, activate the workflow and copy the production URL.
4. Claude → Settings → Connectors → Add custom connector → paste the URL (add the token if asked).
5. In a chat: "Log a bug: checkout button overlaps on small screens, severity high." Approve the tool call and check the sheet and Slack.

Lab 6 · Leads from your Lovable app (45 minutes)

Webhook node (copy its URL) → in your Lovable app (Class 16), ask Lovable to POST each new signup to that URL from a server function → n8n adds the lead to Sheets, asks Claude to score it, and alerts Slack for high scores. Class 21 adds a WhatsApp welcome message.

20.8 Reliability and safety

- **Test with sample data**, then turn on. Keep the first week on "notify me" rather than "act".
- **Error alerts**: in n8n set an **error workflow** that posts failures to Slack; in Make add error handlers; in Zapier turn on error notifications.
- **No duplicates**: store an ID (email ID, form response ID) and skip items you've already processed.
- **Rate limits and costs**: add waits for big batches; set API spend limits; watch task, credit and execution counts.

- **Privacy:** only send the fields the AI step needs. Don't put customer personal data into tools your company hasn't approved.
- **Document** each automation: what it does, owner, accounts used, how to switch it off.

20.9 Exercises with model answers

Exercise 1 · Design the flow

Every support email mentioning "refund" should create a row in a Refunds sheet and alert #support-leads if the amount is over ₹5,000. Sketch the nodes.

Model answer

Gmail Trigger (search "refund") → AI step extracts order ID and amount as JSON → Sheets append row → IF amount > 5000 → Slack alert to #support-leads. Add duplicate protection on the email ID and an error workflow.

Exercise 2 · Pick the tool

A non-technical ops team wants three simple automations this week. A growth team wants an AI agent processing 50,000 events a month and keeping data on company servers. Which tools?

Model answer

Ops team: **Zapier** (fastest to learn; free or Professional depending on steps). Growth team: **self-hosted n8n** (no per-task fees at high volume, AI Agent node, data stays on company infrastructure), with an engineer to run the server.

20.10 Key takeaways

- Every automation is **trigger → (check) → (AI) → action → notify**.
- **Zapier** for quick wins, **Make** for visual branching on a budget, **n8n** for AI agents, volume and privacy.
- Connect Gmail and Slack with **sign-in connectors** in Zapier and Make; n8n self-hosted needs a **Google OAuth app** and a **Slack bot token**.
- **MCP Server Trigger** turns an n8n workflow into a tool Claude can call.

20.11 Check your understanding

CHECK YOUR UNDERSTANDING

What do you need to connect Gmail to self-hosted n8n?

- A. Only your Gmail password
- B. A Google OAuth client ID and secret from a Google Cloud project
- C. A Slack bot token
- D. Nothing

Why: Self-hosted n8n needs your own OAuth app; n8n Cloud, Zapier and Make provide one.

Which Slack scope does a bot need just to post messages?

- A. admin
- B. chat:write
- C. users:read.email
- D. files:write

Why: chat:write is enough to post; give bots the least access they need.

How can Claude call one of your n8n workflows?

- A. Email it the workflow file
- B. Use an MCP Server Trigger and add its URL as a custom connector
- C. Share your n8n password
- D. It can't

Why: The MCP Server Trigger exposes the workflow as a tool over MCP.

Why ask the AI step to answer in JSON?

- A. It's shorter
- B. Later nodes can reliably read fixed fields such as category
- C. Slack requires JSON
- D. It's cheaper

Why: Structured output makes branching and mapping predictable.

20.12 Recommended resources

- [Zapier help](#) · [Make help](#) · [n8n docs](#) · [n8n workflow templates](#)
- [n8n Google credentials guide](#) · [Slack: creating an app](#)
- Official YouTube channels of n8n, Make and Zapier for feature walkthroughs; see Class 23 for creator channels

MODULE 21

AI Chatbots for Business: WhatsApp, Slack and Web

Design, build, test and price AI chatbots that answer customers on WhatsApp, help employees in Slack, and sit on a website, with safe handoff to people.

≈ 8 h study

Messaging rules and prices change

Meta updates WhatsApp pricing and policies several times a year. Always check the [official pricing page](#) and your provider's rates before launch. Examples use **hypothetical businesses**; screens are **illustrative wireframes**.

21.1 What a business chatbot does

A business chatbot answers questions and completes simple tasks in a chat channel: order status, booking an appointment, store hours, return policy, lead capture. An **AI chatbot** uses a language model (such as Claude) plus your business's **knowledge** and **tools**, so it can understand free-text questions instead of only menu buttons.

WHATSAPP

- Where Indian customers already are
- Orders, bookings, reminders, support
- Needs the Business Platform for automation

WEBSITE CHAT WIDGET

- Pre-sales questions, lead capture
- Fully under your control

INSTAGRAM / MESSENGER

- D2C brands, creators
- Same Meta platform family

SLACK OR MICROSOFT TEAMS

- Internal bots: HR policy, IT help, sales enablement

TELEGRAM

- Free and quick to set up
- Perfect for practice and communities

VOICE (PHONE)

- Calls instead of chat
- See Class 22

Channels and when to use them. The AI part is the same across channels. What changes is the channel's rules, costs and setup.

21.2 WhatsApp: App or Platform?

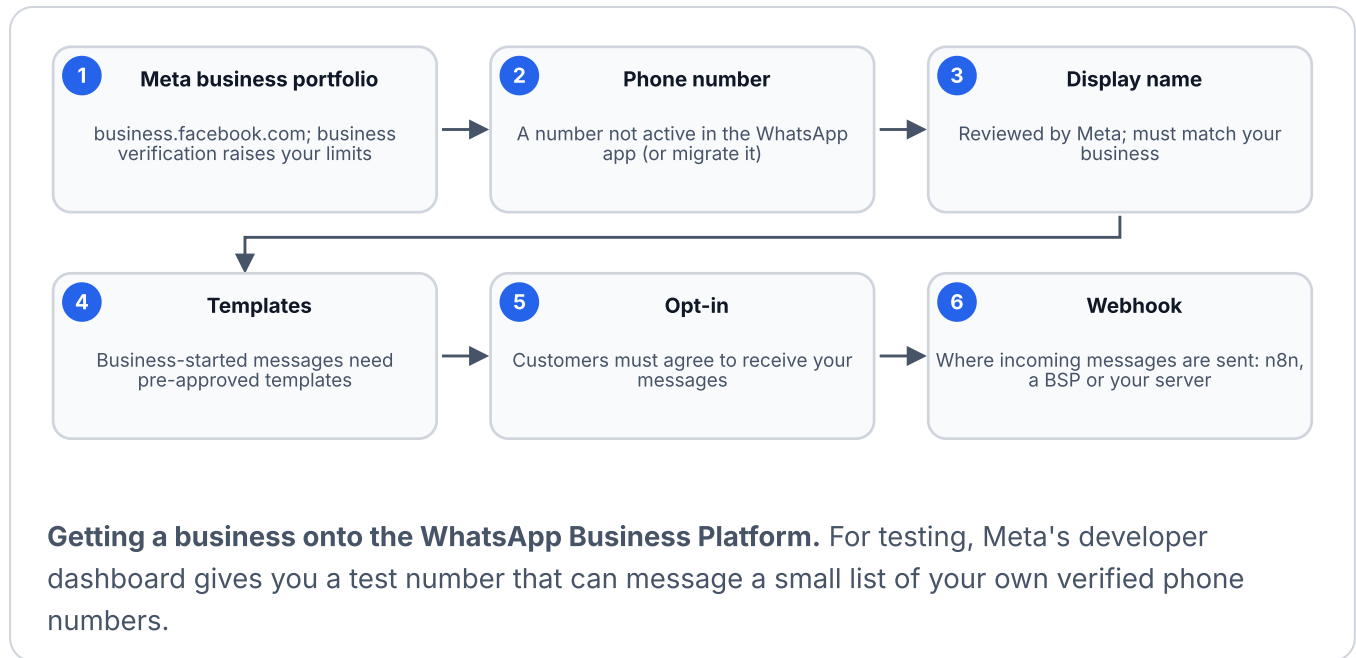
	WhatsApp Business App	WhatsApp Business Platform (API)
Who it is for	Small shops replying by hand	Businesses automating or scaling
Cost	Free app	Per delivered template message, plus any provider fee
Automation	Greeting, away message, quick replies only	Full bots, AI, integrations
Users	Phone plus a few linked devices	Unlimited agents through a provider's inbox
Setup	Download and verify the number	Meta business portfolio, number, display name, templates
AI chatbot	Not supported directly	Yes, via webhooks to n8n, a provider or your code

WhatsApp Business App vs WhatsApp Business Platform. For an AI chatbot you need the Business Platform, used directly through Meta's Cloud API or through a Business Solution Provider.

21.2.1 Direct Cloud API or a Business Solution Provider (BSP)?

- **Meta Cloud API directly:** no middleman fee; you connect it to n8n or your own code. Best when someone technical owns it.
- **A BSP** (for example Twilio, Gupshup, Interakt, WATI, AiSensy, 360dialog): onboarding help, a shared team inbox, broadcast tools and no-code flow builders; they add a monthly fee or a markup per message.

21.2.2 What Meta requires

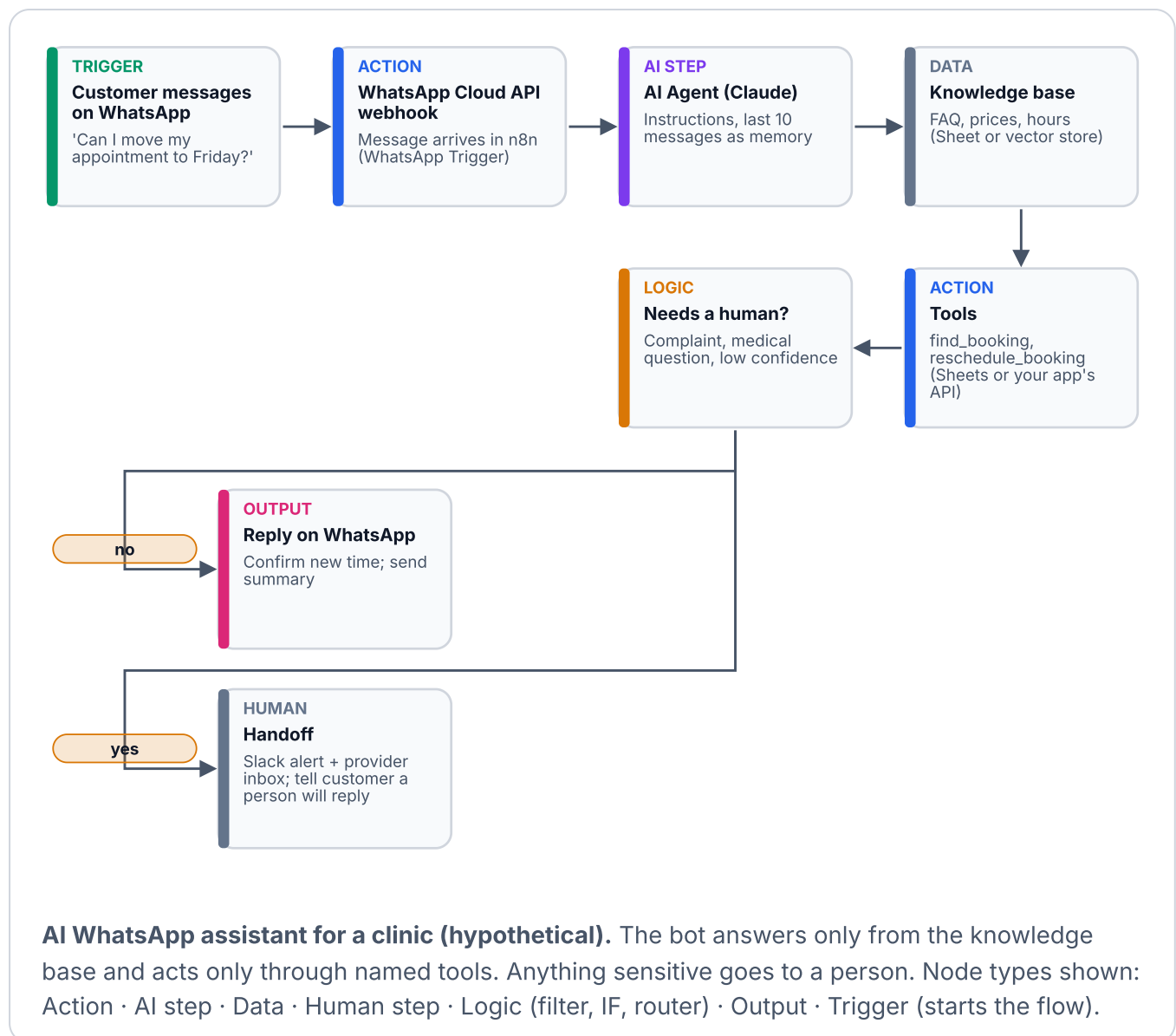


The 24-hour customer service window. When a customer messages you, a 24-hour window opens in which you can reply freely. Outside that window, a business can only start the conversation with an **approved template**.

21.2.3 Pricing basics (checked 23 Oct 2026)

- Since July 2025 Meta charges **per delivered template message**. The rate depends on the template **category** (marketing, utility, authentication) and the recipient's **country**. Marketing costs the most; authentication (OTP) the least.
- Rates differ by country. Several 2026 guides quote India authentication messages at a fraction of a rupee and marketing messages at a few times that. Use Meta's current rate card, not a blog.
- Independent guides report that from **1 October 2026** Meta began charging for some messages inside the 24-hour window that used to be free (service and utility replies). **Confirm this on Meta's pricing page** before you forecast costs.
- Add the **BSP's fee** and the **AI model's cost** (a few paise to a few rupees per conversation, depending on length and model).

21.3 How an AI WhatsApp bot works



Reschedule

Sunrise Physio

You: Can I move my appointment to Friday?

Bot: I found your booking for Wed 23 Oct, 10:00. Friday has 11:00 or 16:30. Which works?

11:00

16:30

1 Bot finds the booking with a tool

2 Quick-reply buttons reduce typing

Handoff

Sunrise Physio

You: My knee is much worse after the session.

Connecting you to our team

Bot: I'm sorry to hear that. A physiotherapist will reply here within 30 minutes. If it's an emergency, call 112.

1 Health concerns always go to a person

2 Say what happens next and when

The customer's view (illustrative wireframe, not a real product screenshot). Hypothetical clinic and messages, for practice only.

21.4 No-code options

Tool	What it is	Good for
BSP flow builders (Interakt, WATI, AiSensy, Gupshup)	Menus, forms and AI replies inside the provider	Fast WhatsApp launch for SMBs
Botpress	Visual bot builder with AI and many channels	Multi-channel bots with knowledge bases
Voiceflow	Conversation design and AI agents, chat and voice	Teams designing complex flows
Chatbase	Train a bot on your website or documents	Website FAQ bot in minutes
Intercom Fin, Zendesk AI	AI agents inside support suites	Companies already on those helpdesks
n8n + Claude	Build exactly what you need	Custom tools, privacy, low running cost

21.5 Slack, Telegram and website bots

- **Slack internal bot:** create a Slack app with the Events API and bot scopes (`app_mentions:read`, `chat:write`), connect it to n8n's Slack Trigger, and answer from your HR or IT policy documents. Restrict it to the channels it's invited to.
- **Telegram practice bot:** message **@BotFather**, send `/newbot`, copy the token, add it to n8n's Telegram Trigger. It's free and takes ten minutes, ideal for learning before WhatsApp.
- **Website widget:** the "Ask us" button from Class 16 (Lovable server function + Claude), or a Botpress or Chatbase embed code pasted into your site.

21.6 Design the bot before you build it

The one-page bot spec

1. **Job:** the top 10 questions and 3 tasks the bot handles (from real support tickets).
2. **Not allowed:** medical, legal or financial advice; refunds above a limit; anything about other customers.
3. **Knowledge:** the exact documents and their owner; how often they're updated.
4. **Tools:** each action with inputs, and whether it needs confirmation.
5. **Handoff:** when, to whom, within what time, and what the customer is told.
6. **Tone and language:** for example English and Hindi, friendly and short.
7. **Privacy:** what data is stored, for how long, and the opt-out keyword ("STOP").
8. **Success metrics:** containment rate, resolution rate, handoff rate, CSAT, cost per conversation.

21.7 Guardrails, privacy and quality

Rules that protect customers and the business

- **Answer only from the knowledge base;** say "I'm not sure, let me connect you" otherwise.
- **Confirm before acting** ("Shall I move it to Friday 11:00?").
- **Never collect** card numbers, passwords or OTPs in chat.
- **Get opt-in** before marketing messages and honour opt-outs immediately.
- Follow data-protection law (in India, the **DPDP Act**): collect the minimum, explain the purpose, delete on request.
- **Log conversations** for quality review, with access limited to staff who need it.

21.7.1 Evaluate before launch

Use `datasets/support_eval_set.csv` from this handbook (or 50 real anonymised questions): run every question through the bot, then score **correct, safe, handed off when it should, tone**. Fix the knowledge base or instructions, and re-run until it passes your bar (see template 09 for the evaluation plan).

21.8 Costs at a glance (illustrative)

Item	Typical cost driver
WhatsApp template messages	Per delivered message by category and country
BSP	Monthly plan or per-message markup
AI model	Per token; short FAQ chats are cheap, long ones cost more
n8n	Free self-hosted (server cost) or a cloud plan
People	Staff to handle handoffs and keep knowledge current

21.9 Labs

Lab 1 • Telegram FAQ bot (free, 40 minutes)

1. Create a bot with @BotFather and copy the token.
2. n8n: Telegram Trigger → AI Agent (Claude, or Gemini free tier) with instructions and a 15-line FAQ pasted into the system prompt → Telegram "send message".
3. Test ten questions, including two it should refuse.

Lab 2 • WhatsApp test number (free test, 45 minutes)

1. Create an app at developers.facebook.com, add the **WhatsApp** product, and use the provided **test number**. Add your own phone as a recipient.
2. n8n: add a WhatsApp credential (access token and phone number ID) and a WhatsApp Trigger; register its webhook URL in the Meta dashboard.
3. Reuse the Lab 1 AI Agent and reply with the WhatsApp node.
4. Message the test number from your phone and check the reply.

Lab 3 · Knowledge base and a booking tool (45 minutes)

1. Put FAQs in a Google Sheet (question, answer, category). Give the agent a **Sheets tool** to look them up.
2. Add a Bookings sheet and a tool `find_booking(phone)`; then `reschedule_booking(id, new_time)` with confirmation required.
3. Test the reschedule conversation from the phone picture above.

Lab 4 · Human handoff (20 minutes)

Add an IF node: if the agent sets `handoff: true` (complaint, health issue, low confidence), post the conversation to Slack #practice-support and send the customer the handoff message.

Lab 5 · Slack HR policy bot (30 minutes)

Create a Slack app with Events and the scopes above, connect n8n's Slack Trigger, and answer from a 2-page policy document. Mention the bot in a channel and ask three questions.

Lab 6 · Evaluate (30 minutes)

Run 30 questions from `support_eval_set.csv` through your bot with a loop in n8n, write answers to a sheet, and score them. Fix the top two failure patterns and re-run.

21.10 Exercises with model answers

Exercise · Template or free reply?

(a) A customer asked about delivery 2 hours ago; you reply with tracking. (b) You want to tell all past customers about a Diwali sale. (c) You send an OTP to log in.

Model answer

(a) Inside the 24-hour window, so a normal reply (check whether your country's current rules charge for it). (b) A **marketing template**, approved by Meta, only to opted-in customers. (c) An **authentication template**.

21.11 Key takeaways

- AI chatbots need **knowledge, tools, guardrails and a handoff to people**.
- WhatsApp automation needs the **Business Platform** (direct Cloud API or a **BSP**), approved **templates**, the **24-hour window** and **opt-in**.
- Pricing is **per delivered template message** by category and country, plus provider and AI costs.
- Practise on **Telegram** for free, then use Meta's **test number** before going live.
- **Evaluate** with a question set and track containment, handoffs and CSAT.

21.12 Check your understanding

CHECK YOUR UNDERSTANDING

Which WhatsApp option supports an AI chatbot connected to n8n?

- A. WhatsApp Business App
- B. WhatsApp Business Platform (Cloud API or BSP)
- C. Personal WhatsApp
- D. WhatsApp Web

Why: Automation and webhooks require the Business Platform.

A business wants to message a customer who last wrote 3 days ago. What's needed?

- A. Nothing
- B. An approved template message
- C. A voice call
- D. A new phone number

Why: Outside the 24-hour window, businesses can only start with an approved template.

A customer describes a worsening injury to a clinic bot. The bot should...

- A. Give treatment advice
- B. Hand off to a person and give emergency guidance
- C. Ignore it
- D. Offer a discount

Why: Health concerns are out of scope for the bot and must reach a qualified person.

21.13 Recommended resources

- [WhatsApp Business Platform docs](#) · [Pricing](#)
- [n8n WhatsApp node](#) · [n8n Telegram node](#)
- [Botpress](#) · [Voiceflow](#) · [Slack Events API](#)
- Module 10 (RAG) for knowledge bases; Module 12 (agents) for tool safety

MODULE 22

Voice Agents for Businesses

Build AI phone agents that answer and make calls for clinics, salons, shops and service companies, with the right platform, phone number, tools, costs and legal care.

≈ 7 h study

Costs and rules vary by country

Per-minute prices change often and depend on the voice, AI model and phone carrier. Telecom and consent rules differ by country. Check each vendor's pricing page and get legal advice before calling real customers. Examples are **hypothetical**; screens are **illustrative wireframes**.

22.1 What a voice agent does

A **voice agent** is an AI that talks on the phone (or in a browser or app). It listens, understands, answers, and can take actions such as booking an appointment, then hands the call to a person when needed.

RECEPTIONIST

- Answer after-hours calls
- Hours, prices, directions
- Book or reschedule

REMINDERS

- Appointment and delivery reminders
- Confirm or reschedule in the call

LEAD QUALIFICATION

- Call back web leads within minutes
- Ask 3–5 questions, book a sales call

ORDER AND STATUS LINES

- Where is my order?
- Return policy, store stock

SURVEYS AND FEEDBACK

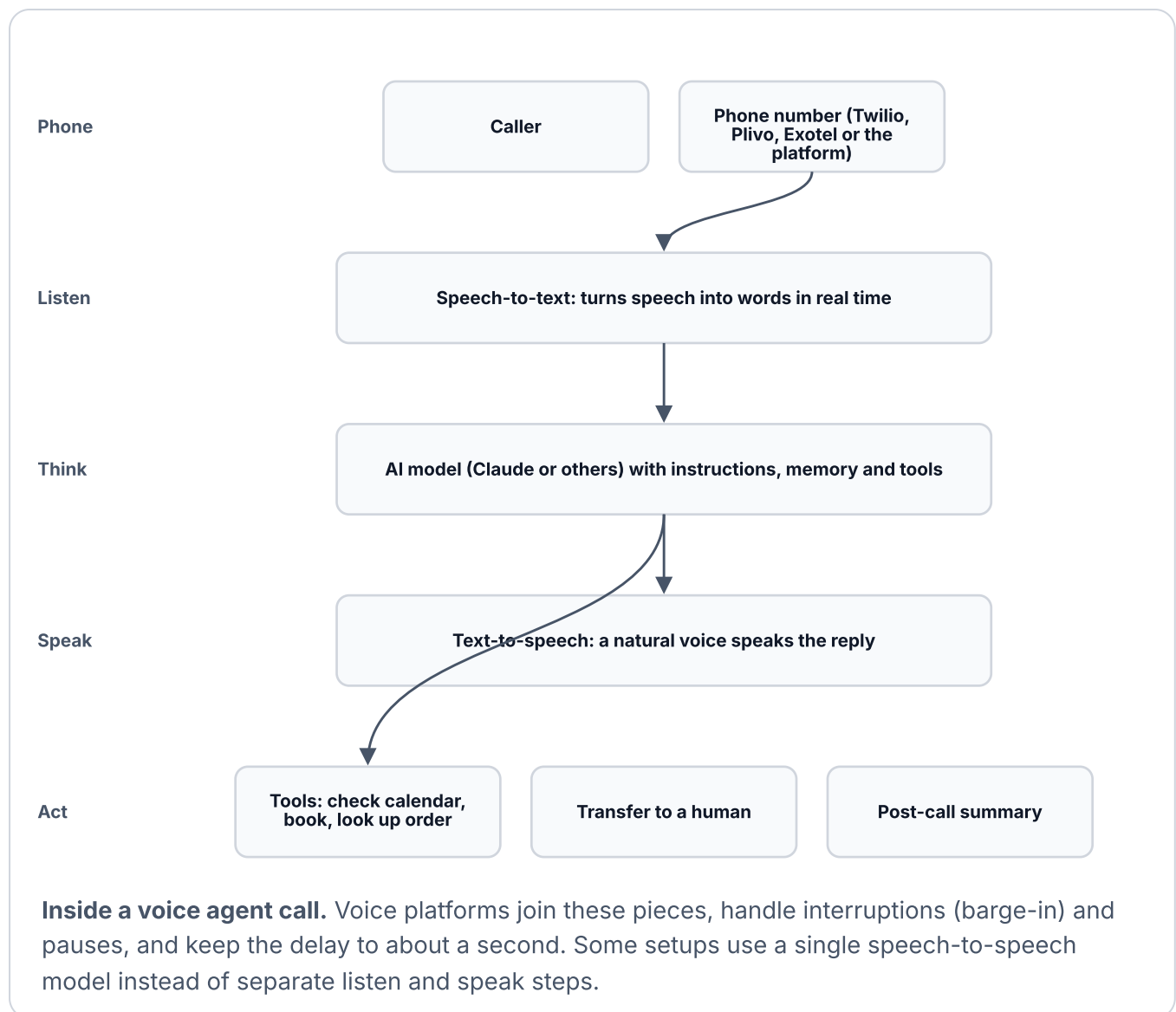
- Short NPS or post-visit calls
- Summaries to a sheet

NOT A GOOD FIRST USE

- Medical or legal advice
- Debt collection without legal review
- Anything needing judgement or empathy at length

Good first use cases. Start with short, repetitive calls where the caller's goal is clear and mistakes are easy to fix.

22.2 How a call works



22.3 Platforms

Platform	Style	Pricing model	Typical all-in cost per minute	Good for
Vapi	Developer platform, very flexible	Platform fee (about \$0.05/min) plus your chosen speech, model, voice and phone providers	about \$0.13 to \$0.32	Custom builds, choosing each component
Retell AI	Developer-friendly with a strong dashboard	Usage per minute, phone line included	about \$0.07 to \$0.31	Fast setup, receptionists and outbound
ElevenLabs Agents	Best-known for natural voices	Plans with included minutes, extra minutes billed; phone carrier separate	about \$0.08 and up	Voice quality, many languages
Bland AI	Enterprise phone agents	Per minute	varies	High-volume calling
Synthflow	No-code builder	Plans with minutes	varies	Non-technical teams
Twilio (Conversations, Relay and others)	Build-your-own on a phone platform	Telephony plus your AI costs	varies	Engineering teams wanting full control

Voice agent platforms (checked 23 Oct 2026; verify pricing). All-in cost includes speech-to-text, the AI model, the voice and the phone line. Figures are ranges from several 2026 comparisons, not quotes.

Phone numbers in India. Providers such as **Exotel**, **Plivo** and **Knowlarity** offer Indian virtual numbers and connect to voice platforms (often via SIP). Indian numbers require KYC documents. International

platforms may offer Indian numbers only through a partner, so check availability before choosing a platform.

Indian languages. Check each platform's supported languages and test Hindi and regional-language calls with real speakers. Accents, code-mixing (Hinglish) and background noise affect accuracy.

22.3.1 What it costs: a worked example

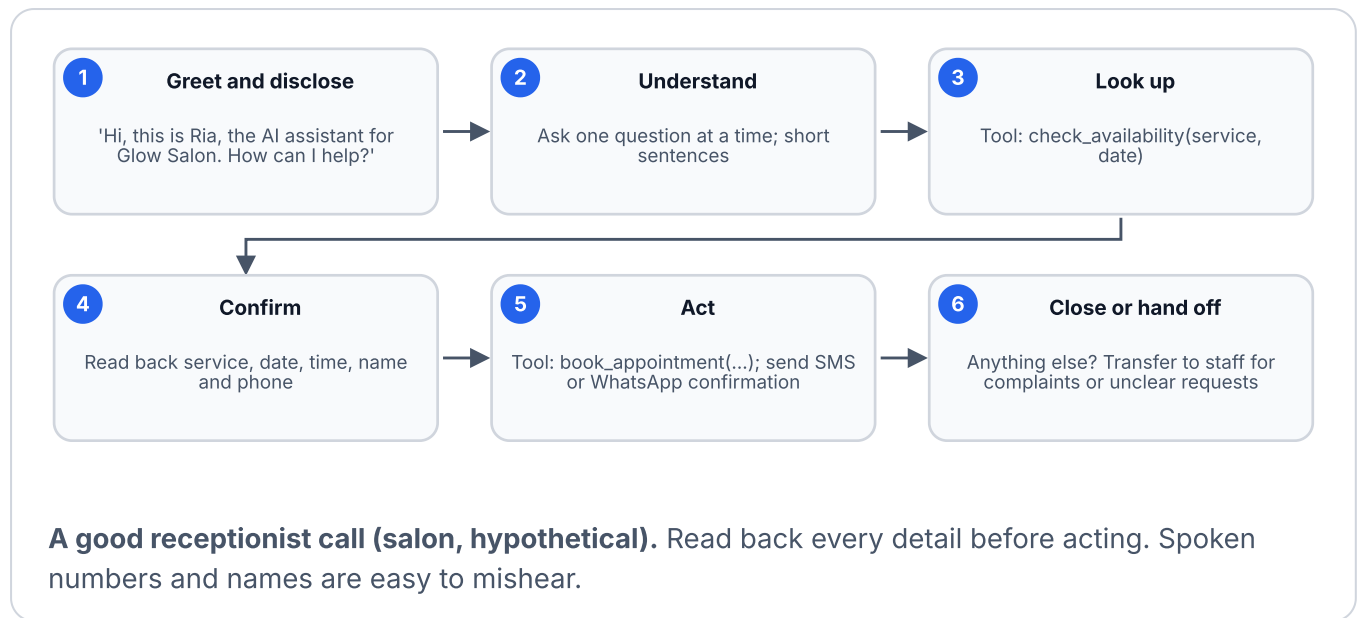
A clinic receives **1,000 calls a month, averaging 3 minutes**. At **\$0.15 per minute** all-in, that is $1,000 \times 3 \times \$0.15 = \text{\$450 a month}$ (about ₹38,000 at ₹84 to the dollar), plus a phone number fee, the platform's plan, and staff time for handoffs. Compare it with the cost of missed calls and of staff answering them.

22.4 The law and consent (summary, not legal advice)

Before any real call

- **Tell callers it's an AI** at the start, and say if the call is recorded.
- **Outbound calls** need prior consent. In India, commercial calls fall under **TRAI's commercial communication rules** (registration, DND preferences, consent). In the US, the FCC treats AI-generated voices in robocalls as "artificial" under the **TCPA**, so prior express consent is required.
- **Personal data**: collect the minimum, store recordings securely and delete them on schedule (India's **DPDP Act**, GDPR in Europe).
- Offer a **human option** ("press 0 or say 'agent'") and honour do-not-call requests.

22.5 Designing the conversation



Voice prompt rules

- Replies under **two sentences**; no lists, tables or URLs (they don't work in speech).
- Say numbers the way people speak them ("four-thirty in the afternoon").
- Handle silence ("Are you still there?") and interruptions gracefully.
- Give a clear way out: "If you'd like to talk to someone, just say 'staff'."
- Never guess. "I'm not sure, let me get a colleague to call you back."

22.6 Build it: platform + n8n tools

Agents

Phone numbers

Tools

Calls & logs

Billing

Agent: Glow Salon receptionist

Instructions: You are Ria, the AI receptionist for Glow Salon...

1

ModelClaude (or another LLM)

VoiceWarm female, Indian English

2

TranscriberReal-time, en-IN + hi-IN

Toolscheck_availability, book_appointment

3

Transfer number+91 ... staff line

Talk to agent (test in browser)

4

First message

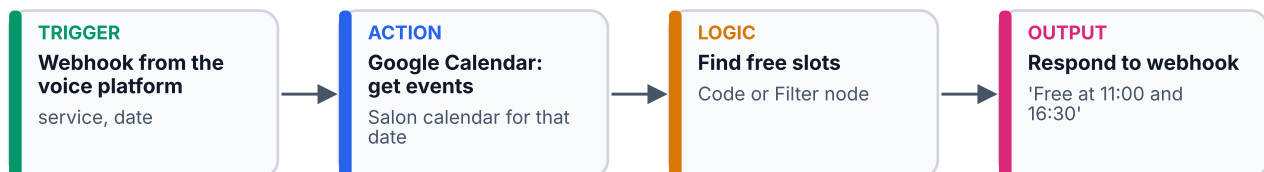
Hi, this is Ria, the AI assistant for Glow Salon. How can I help today?

Post-call webhook

https://your-n8n/webhook/call-summ...

- 1 Instructions are the system prompt: role, scope, tone, rules and when to transfer.
- 2 Pick a voice and transcription language your callers use; test with real speakers.
- 3 Tools call your n8n webhooks, which read or write Google Calendar, Sheets or your app.
- 4 Test in the browser for free before you buy a phone number.

A voice platform's agent setup (illustrative wireframe, not a real product screenshot). Names and layout differ between Vapi, Retell AI and ElevenLabs, but every platform has these settings.



The tool behind 'check_availability' (n8n). Keep tool responses short and plain, because the agent reads them aloud. Respond within a couple of seconds or callers hear silence. Node types shown: Action · Logic (filter, IF, router) · Output · Trigger (starts the flow).

22.7 Labs

Lab 1 • Browser test call (free trial credits, 30 minutes)

1. Sign up for Vapi, Retell AI or ElevenLabs Agents and create an agent from a receptionist template.
2. Write the instructions and first message for Glow Salon (hypothetical), with hours, prices and three services.
3. Click **talk/test** and make five test calls, including one where you interrupt and one where you ask for a human.

Lab 2 • Availability and booking tools (60 minutes)

1. Build the n8n `check_availability` workflow above and a `book_appointment` workflow that creates a Google Calendar event.
2. Add both as tools in the voice platform, with clear descriptions and parameters.
3. Test a full booking. Check the calendar event appears.

Lab 3 • Post-call summary (20 minutes)

Set the platform's post-call webhook to an n8n workflow that writes caller, intent, outcome and summary to a Sheet and posts transfers to Slack.

Lab 4 • Inbound phone number (paid, 20 minutes)

Buy or import a phone number (platform or Twilio; in India via a provider such as Exotel or Plivo, which needs KYC). Attach it to the agent and call it from your phone. Add the AI disclosure and recording notice to the first message.

Lab 5 • Test plan (30 minutes)

Write 20 test scenarios (booking, rescheduling, unclear request, background noise, Hindi caller, angry caller, wrong number). Run them, score each (resolved, transferred correctly, mistakes), fix the instructions and re-test.

22.8 Metrics

Metric	What it tells you
Containment / resolution rate	Share of calls fully handled by the agent
Correct transfer rate	It handed off the calls it should have
Booking conversion	Calls that ended in a booking
Response latency	Time from caller stopping to agent speaking; aim for about a second
Transcription accuracy	Mis-heard names and numbers
Average handle time and cost per call	Efficiency and budget
CSAT or complaints	Caller experience

22.9 Exercises with model answers

Exercise · Price it

A car service centre gets 600 calls a month, 4 minutes each. Platform quotes \$0.12 a minute all-in. Staff handle 20% of calls after transfer. Monthly AI cost?

Model answer

$600 \times 4 \times \$0.12 = \text{\$288}$ a month for agent minutes (assuming the transfer happens after most of the call; transferred minutes may continue to be billed until the agent leaves the call). Add the number rental and platform plan, and staff time for the 120 transferred calls.

Exercise · Fix the script

The agent says: "Our services include haircut ₹500, colour ₹2,500, spa ₹1,800, visit glowsalon.com/book or press 1 for..." What's wrong?

Model answer

Too long, a list and a URL in speech. Better: "We do haircuts, colour and spa treatments. Which one are you interested in?" Then give the one price asked for, and offer to book or send a WhatsApp link.

22.10 Key takeaways

- A voice agent = **phone + speech-to-text + AI model + text-to-speech + tools + handoff**.
- Choose **Retell AI or ElevenLabs** for speed, **Vapi or Twilio** for control, a **no-code** builder for non-technical teams.
- Budget **all-in per-minute** cost, not the headline platform fee.
- **Disclose AI**, get **consent** for outbound calls, and follow local telecom and privacy rules.
- **Read back details**, keep replies short, and test with real speakers before going live.

22.11 Check your understanding

CHECK YOUR UNDERSTANDING

Why can the all-in cost be several times the platform's headline per-minute fee?

- A. Taxes only
- B. Speech-to-text, the AI model, the voice and the phone line are billed too
- C. Because of the logo
- D. It can't

Why: Each component has its own per-minute or per-token price.

What should an outbound sales voice agent have before calling?

- A. A funny greeting
- B. Prior consent from the person and compliance with telecom rules
- C. A longer script
- D. A website

Why: Consent and telecom regulations apply to automated and AI calls.

Which reply style works best on a phone call?

- A. A bulleted list with links
- B. Short sentences, one question at a time
- C. A long paragraph
- D. A table

Why: Listeners can't scan; short, single questions are easiest to follow.

22.12 Recommended resources

- [Vapi docs](#) · [Retell AI docs](#) · [ElevenLabs Agents docs](#) · [Twilio Voice](#)
- [TRAI](#) for Indian commercial-communication rules · [FCC on AI-generated voices in robocalls](#)
- Module 12 (agentic AI) for tool safety and evaluation

MODULE 23

Learning Paths, Courses and Communities

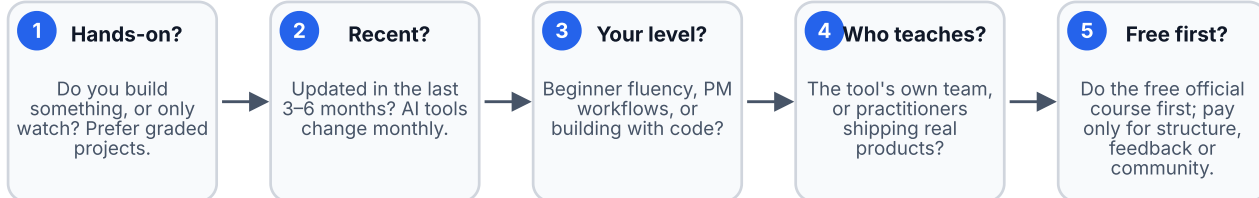
The best free and paid courses for Claude and AI tools in 2026, what each institute teaches, YouTube channels worth following, communities, a 90-day plan with weekly assignments, and portfolio projects.

≈ 3 h study

Course catalogues change

Course titles, prices, ratings and enrolment numbers below come from public listings and reviews found on **23 October 2026**. Catalogues change often, so check each course page before enrolling. Ratings with few reviews are a weak signal. Nothing here is a paid recommendation.

23.1 How to choose a course



Five questions before you pay for a course. The best learning loop is: short official course → build one small project → share it → get feedback → repeat.

23.2 Claude: courses from every major provider

Provider	Course	Level and length	Cost	Notes
Claude Academy	Claude 101	Beginner, about 2.5 h	Free	Everyday tasks and core features; the best starting point
Claude Academy	AI Fluency: Framework and Foundations	Beginner	Free	How to work with AI well; tracks for students, educators and nonprofits
Claude Academy	Claude Code in Action	Intermediate	Free	Using Claude Code in a real development workflow
Claude Academy	Building with the Claude API	Intermediate to advanced, 8+ h	Free	API, tools, RAG, agents
Claude Academy	Introduction to MCP; MCP Advanced Topics	Intermediate to advanced	Free	Building MCP servers and clients
DeepLearning.AI (with Anthropic)	Claude Code: A Highly Agentic Coding Assistant	Intermediate, about 1 h 50 min	Free	Three projects incl. building a web app from a Figma mockup with MCP
Coursera (Vanderbilt)	Claude Code: Software Engineering with Generative AI Agents	Beginner, about 5 h	Coursera Plus or paid	Strong ratings; some reviewers want more hands-on assignments
Coursera (Packt)	Claude AI for Product Managers: From Strategy to Execution	Beginner, about 5 h	Coursera Plus	PM-specific workflows

Provider	Course	Level and length	Cost	Notes
Udemy	Claude AI for Product Management (and similar PM titles)	Beginner to intermediate	Paid (frequent discounts)	Several new courses; few reviews yet, so preview lectures first
Udemy	Claude Code for Product Managers	Beginner to intermediate	Paid	Build and deploy a small product; custom Skills
LinkedIn Learning	Getting Started with Claude	Beginner, 36 min	Included with LinkedIn Learning	Quick overview with a certificate
Product Compass / cohort	Claude Code for PMs (cohort)	Intermediate	About \$2,000+	Live, structured, expensive; new
Frontend Masters	Claude Code	Intermediate (developers)	Subscription	Plan mode, skills, hooks

Claude courses (checked 23 Oct 2026). Anthropic's course site has reportedly been rebuilt as Claude Academy (academy.claude.com), with anthropic.com/learn redirecting there and a larger catalogue; check both addresses for new courses. Suggested order for PMs: Claude 101 → AI Fluency → the DeepLearning.AI Claude Code course → this handbook's Classes 14 and 15 → one paid PM-focused course only if you want structure or feedback.

23.3 AI product management programmes (India and global)

Programme	What it emphasises (from public pages and alumni reviews)	Notes
NextLeap PM Fellowship	PM foundations, user research, PRDs and wireframes, AI applications, milestones, a capstone, interview prep	Strong alumni reviews; check the current syllabus for which AI tools are taught
Airtribe AI-First Product Management	Modules from fundamentals to LLMs, RAG, vibe coding and agentic AI; weekly assignments, capstone, leaderboard	The structure this handbook's 12 core modules follow
HelloPM	Cohort-based PM programmes with mentors and case studies	Compare cohort dates, mentors and AI coverage
Maven cohorts (various instructors)	Short, intensive AI PM and AI building courses	Quality varies by instructor; read reviews
Reforge	Senior PM and growth programmes, AI modules	Best for experienced PMs

23.4 Courses for the other AI tools

Tool	Free official learning	Other good options
Lovable	Lovable docs , official YouTube tutorials	Community Discord challenges; build-along videos
Replit	Replit docs and Replit's YouTube	Udemy Replit Agent courses (check recency)
Figma	Figma Learn and Help Center , Figma's YouTube	Coursera Google UX Design Certificate (uses Figma)
Notion	Notion Help and Academy	Template creators' tutorials
Zapier	Zapier Learn	Zapier's own courses and webinars
Make	Make Academy (free, with badges)	
n8n	n8n docs and courses (Level 1 and 2, free)	YouTube creators below
Voice agents	Vapi, Retell AI and ElevenLabs docs and YouTube channels	Build-along videos for receptionist agents
WhatsApp bots	Meta's WhatsApp Business Platform docs	Your BSP's academy (Interakt, WATI, Gupshup)

23.5 YouTube channels

OFFICIAL CHANNELS

- Anthropic
- n8n
- Lovable
- Replit
- Figma
- Notion
- Zapier
- Make
- Vapi and Retell AI

CLAUDE CODE AND AI BUILDING

- Nick Saraev (Claude Code, automation)
- Riley Brown (vibe coding)
- Official 'Claude Code best practices' talks

N8N AND AUTOMATION

- Nate Herk (n8n AI agents)
- Nick Saraev
- Channels tracked as fast-growing in 2026, e.g. Emma Explains AI, Jimi Barkway

PRODUCT MANAGEMENT AND AI

- Lenny's Podcast
- Peter Yang (Behind the Craft)
- Aakash Gupta (Product Growth)

STARTUP IDEAS AND AI TOOLS

- Greg Isenberg
- Matt Wolfe (AI tool news)

HOW TO WATCH

- Code along, pause often
- Prefer videos from the last 3 months
- Check the tool's docs when a video looks outdated

Channels worth following (check recency before you binge). Creator channels are a mix of tutorials and promotion. Use them for ideas and walkthroughs, and the official docs for the facts.

23.6 Communities

- **Anthropic / Claude:** the official Claude Discord and r/ClaudeAI for tips and help.
- **n8n:** the [n8n community forum](#) (very active, with shared workflows) and r/n8n.
- **Lovable and Replit:** official Discord servers with build challenges.
- **Product management:** Lenny's community, Mind the Product, local PM meetups, and your course cohort's Slack or Discord.

23.7 A 12-week plan with one assignment a week

We ek	Learn	Assignment (portfolio-ready)
1	Class 13 + Claude 101	Set up all accounts; write a one-page AI tools policy for yourself
2	Class 14 Level 1–2	Project for a product you know; interview synthesis artifact
3	Class 14 Level 3	Gmail + Calendar morning brief; Slack weekly update
4	Class 15 Level 1–2 + DeepLearning.AI course	Landing page in Claude Code, pushed to GitHub, deployed
5	Class 18	Three-screen Figma prototype tested with two users
6	Class 16	Lovable MVP with database and login; RLS tested
7	Class 19	Notion product hub; one PRD with critique and tickets
8	Class 20 Level 1–2	Zapier Gmail → Slack; Make form router
9	Class 20 Level 3	n8n AI email triage; MCP tool Claude can call
10	Class 21	Telegram bot, then WhatsApp test-number bot with handoff
11	Class 22	Voice receptionist with booking tool (browser test)
12	Capstone	Clinic AI front desk (Class 13) end to end; case study write-up and demo video

12-week AI tools plan for PMs (about 5 hours a week). Post each assignment publicly (LinkedIn, GitHub, your portfolio site) with a short write-up: problem, what you built, what you learned, what you'd do next.

23.8 Portfolio projects that impress

1. **AI front desk for a small business** (capstone): WhatsApp bot + voice agent + booking app + automation.
2. **Inbox triage agent** for a team: n8n + Claude + Slack, with measured time saved.
3. **PRD-to-prototype in a day**: Notion PRD → Figma → Lovable or Claude Code → usability test.
4. **Feedback intelligence dashboard**: Replit or Lovable app that tags feedback with AI and charts themes.
5. **Your own MCP tool** that Claude uses at work (for example "create a Jira ticket from this chat").

For each, include the problem, users, metrics, architecture picture, costs, risks and guardrails, and what you'd do with an engineering team.

23.9 Best practices for learning tools that change monthly

- **Learn concepts, not buttons**: triggers, tools, memory, evaluation, permissions. Buttons move; concepts last.
 - **Read release notes** of your main tools monthly.
 - **Keep a build log**: a Notion page listing what you built, prompts that worked, and mistakes.
 - **Teach it**: a 5-minute Loom or LinkedIn post locks in what you learned.
 - **Respect costs and privacy** from day one: spend limits, test data, approved tools only for work data.
-

23.10 Key takeaways

- Start with **free official courses** (Claude Academy, DeepLearning.AI, tool academies), then build.
- Pay for a course only for **structure, feedback or community**, and check how recent it is.
- Follow a **weekly assignment** rhythm and publish what you build.
- Combine **official docs** for facts with **creators** for walkthroughs and ideas.

23.11 Check your understanding

CHECK YOUR UNDERSTANDING

What's the best first Claude course for most PMs?

- A. An expensive cohort
- B. Anthropic Academy's free Claude 101
- C. A random video from 2023
- D. None; just guess

Why: It's free, official and covers everyday features.

Which signal matters most when picking an AI tools course in 2026?

- A. The thumbnail
- B. How recently it was updated and whether you build projects
- C. The number of slides
- D. The course's length alone

Why: Tools change monthly; hands-on, recent courses transfer best.

23.12 Recommended resources

- [Claude Academy](#) (formerly [anthropic.com/learn](#)) · [DeepLearning.AI short courses](#)
- [Coursera: Claude Code \(Vanderbilt\)](#) · [LinkedIn Learning](#)
- [Make Academy](#) · [n8n courses](#) · [Zapier Learn](#)
- [Product Compass: best AI PM courses \(2026 review\)](#)