DeltaForceOS

CODEX ROUTE / BEGINNER MODULE 02

Beginner Tool Glossary: Codex Route

BUILD OUTCOME

Understand the core tool categories so the rest of the curriculum feels like a map, not alphabet soup.

BEST FOR

Use this PDF when you want exact prompts and clicks for this route instead of general theory.

MODEL TIER

Light model for glossary cards; mid model for deciding where a tool fits.

FIRST INSTRUCTION

```
Inspect this repo for the Beginner Tool Glossary build.
Explain the files a beginner needs: README.md, package.json, src, public, scripts, .env.local, and any
Supabase files.
Do not edit yet. Give me the smallest safe build plan.
```

Before using the agent, make the workspace boring and clear.

01 Open the project folder. If you do not have one yet, create a new folder on Desktop and name it after the lesson.

02 Install Node.js if npm is not available. Beginners can check by opening Terminal and typing npm -v.

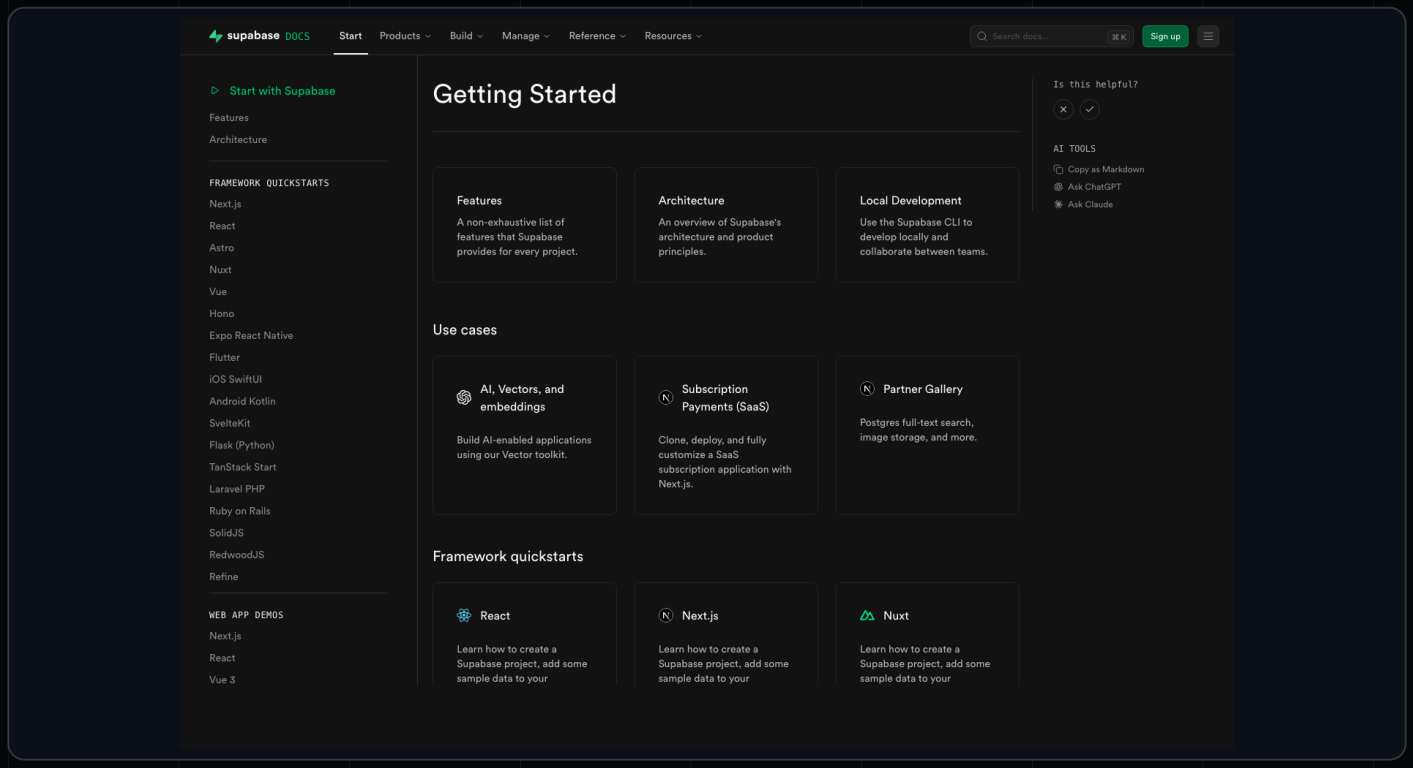
03 Create .env.local in the project root and add only the keys this lesson needs.

04 Confirm .env.local is listed in .gitignore so secrets do not go to GitHub.

05 Open the official dashboard for the first two tools in this lesson and create one tiny test project.

06 Decide where proof will live: Supabase row, screenshot, live URL, or Skool post.

Supabase: click path for this route.



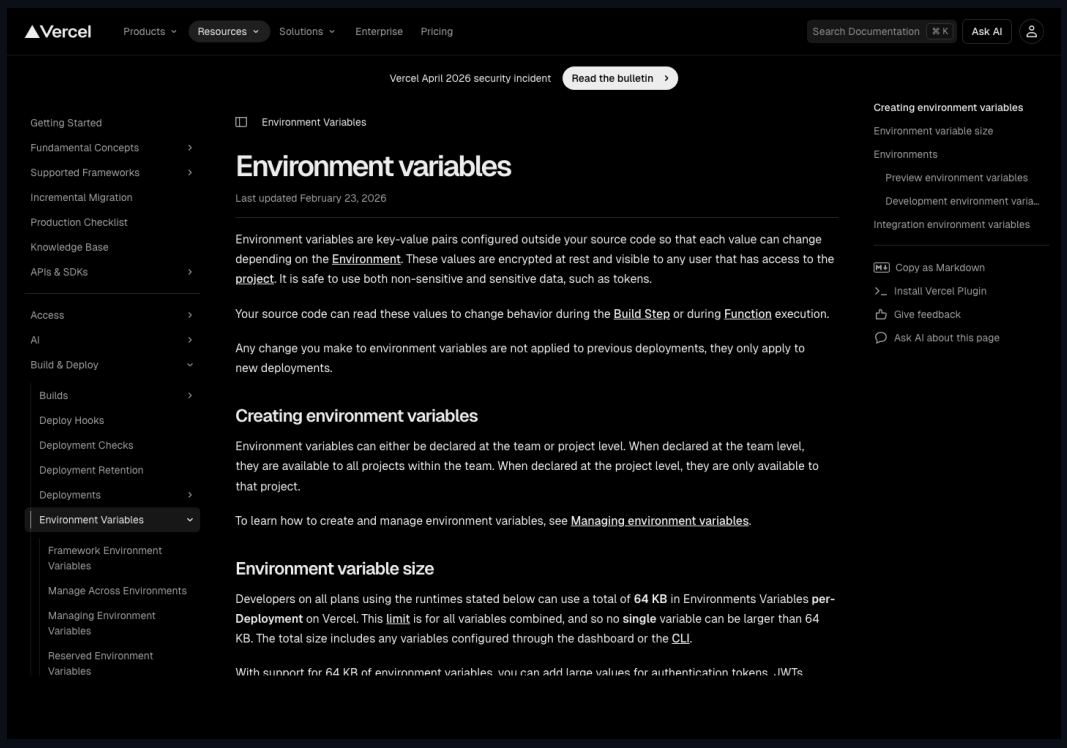
WHERE TO CLICK

Open app.supabase.com, click New project, choose org, name it, save the project URL and anon/publishable key.

ENV NAMES

`NEXT_PUBLIC_SUPABASE_URL, NEXT_PUBLIC_SUPABASE_ANON_KEY, SUPABASE_SERVICE_ROLE_KEY`

Vercel: click path for this route.



WHERE TO CLICK

Open vercel.com/new, import the GitHub repo, then go to Project Settings -> Environment Variables.

ENV NAMES

```
Copy the same server keys from .env.local into Preview and Production values.
```

Copy this box exactly, then let the agent ask clarifying questions only if needed.

INSPECT PROMPT

```
Inspect this repo for the Beginner Tool Glossary build.
Explain the files a beginner needs: README.md, package.json, src, public, scripts, .env.local, and any
Supabase files.
Do not edit yet. Give me the smallest safe build plan.
```

BEGINNER CHECK

If the agent proposes a huge rebuild, ask it to shrink the scope to the smallest artifact that proves this lesson.

DO NOT PASTE

Do not paste service-role keys, payment secrets, signing keys, or private client data into the chat.

Copy this box exactly, then let the agent ask clarifying questions only if needed.

BUILD PROMPT

```
Implement the smallest useful version of Beginner Tool Glossary.
Outcome: Understand the core tool categories so the rest of the curriculum feels like a map, not alphabet
soup.
Tools allowed: Skool, Supabase, Vercel, OpenAI, Claude, LiveKit, Mux, PostHog
Keep secrets server-side. Add code comments only where a beginner would get lost.
```

BEGINNER CHECK

If the agent proposes a huge rebuild, ask it to shrink the scope to the smallest artifact that proves this lesson.

DO NOT PASTE

Do not paste service-role keys, payment secrets, signing keys, or private client data into the chat.

TYPE THIS INTO FILES

```
# README.md lesson block
Lesson: Beginner Tool Glossary
Outcome: Understand the core tool categories so the rest of the curriculum feels like a map, not alphabet
soup.
Tools: Skool, Supabase, Vercel, OpenAI, Claude, LiveKit, Mux, PostHog

# .env.local placeholders
NEXT_PUBLIC_SUPABASE_URL=your_value_here
NEXT_PUBLIC_SUPABASE_ANON_KEY=your_value_here
OPENAI_API_KEY=server_only
ANTHROPIC_API_KEY=server_only

# Verify
npm run build
```

WHERE IT GOES

README.md gets instructions. .env.local gets real local values. Vercel gets production values. Source files get implementation.

Basic English. Click by click. Do not skip ahead.

01 Open Notion (or Apple Notes if you do not use Notion). Click + New page. Title it My Operator Stack.

02 Inside the page, type /table and press Enter to make a 3-column table. Name the columns Job, Tool, and Why I picked it.

03 Add a row: Job = Database, Tool = Supabase, Why = stores users, runs, and customer data.

04 Add a row: Job = Deploy, Tool = Vercel, Why = hosts the site and runs scheduled jobs.

05 Add a row: Job = Brain, Tool = Claude or OpenAI, Why = reasons over text and writes JSON.

06 Add a row: Job = Email, Tool = Resend, Why = sends transactional email like magic links.

Finish the build. The last step always posts proof in Skool.

01 Add a row: Job = Auth, Tool = Supabase Auth, Why = signs members in safely.

02 Add a row: Job = Analytics, Tool = PostHog, Why = tracks events and funnels.

03 Open a new browser tab and visit each tool's docs link from the Sources page of this PDF.

04 For each tool, write one sentence in your Notion table about what the tool owns and what it must NOT own.

05 Add a fourth column called Status. For each row, type learn now, learn later, or ignore so you focus.

06 Take a screenshot of the finished table and post it in the Beginner 02 Skool thread.

The build is not finished until verification is boring.

01 Can you name the database, deploy layer, model layer, and payment layer?

02 Can you explain which keys are safe in a browser?

03 Can a beginner choose the right tool for a new workflow?

04 Run npm run build or the focused test command.

05 Download or open the produced artifact and inspect it visually.

06 Write the failure and fix in README.md or the Skool proof post.

01 Commit safe files to GitHub after reviewing the diff.

02 Open Vercel and deploy a preview first.

03 Add missing env vars under Project Settings -> Environment Variables if the build fails.

04 Run the workflow with fake or test data in preview.

05 Promote or deploy production only after the smoke test passes.

06 Save the live URL and proof artifact in Skool.

AGENT DEPLOY PROMPT

```
Verify the preview build, list any missing env vars, and give me the exact smoke test before production.
```

If something goes wrong, ask for a small repair instead of a total rewrite.

01 Thinking the model is the product; the workflow is the product.

02 Using browser automation where a stable API exists.

03 Letting unused tools keep billing every month.

04 Ask the agent to explain the failing layer: local app, env var, database, external API, prompt, deploy, or auth.

05 Ask for one fix, one test, and one rollback step.

06 Do not let the agent change unrelated files to hide the failure.

SKOOL PROOF

Make your personal stack inventory and label every tool as learn now, learn later, or ignore.

WHAT TO POST

Artifact, screenshot or URL, what worked, what broke, exact prompt used, and the next target.

SOURCES

Codex: https://developers.openai.com/codex

Skool: https://help.skool.com/

Supabase: https://supabase.com/docs/guides/getting-started

Vercel: https://vercel.com/docs/environment-variables

OpenAI: https://platform.openai.com/docs/quickstart

Claude: https://docs.anthropic.com/en/docs/overview