DeltaForceOS

CODEX ROUTE / BEGINNER MODULE 07

LinkedIn Outreach Agent Basics: Codex Route

BUILD OUTCOME

Understand the outreach loop before scaling it with Apollo, Clay, Smartlead, and CRM automation.

BEST FOR

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

MODEL TIER

Light for cleanup; mid for personalized copy; heavy is unnecessary at this stage.

FIRST INSTRUCTION

```
Inspect this repo for the LinkedIn Outreach Agent Basics 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.

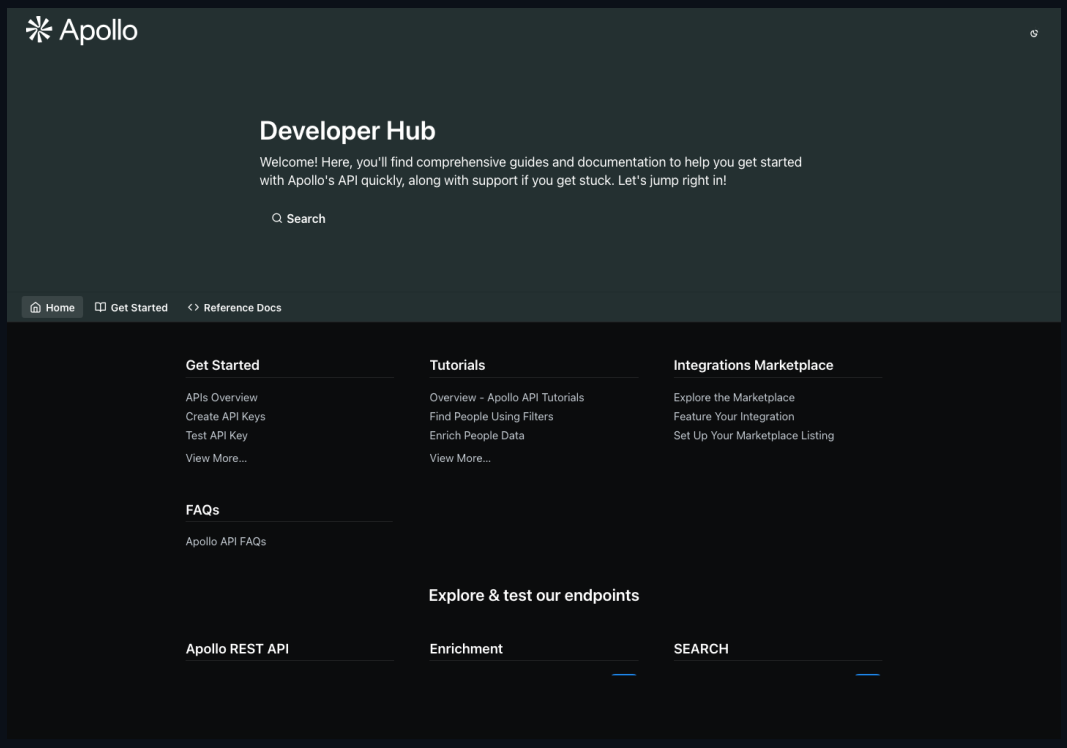
03 Create .env.local in the project root and add only the keys this lesson needs.

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.

Apollo: click path for this route.



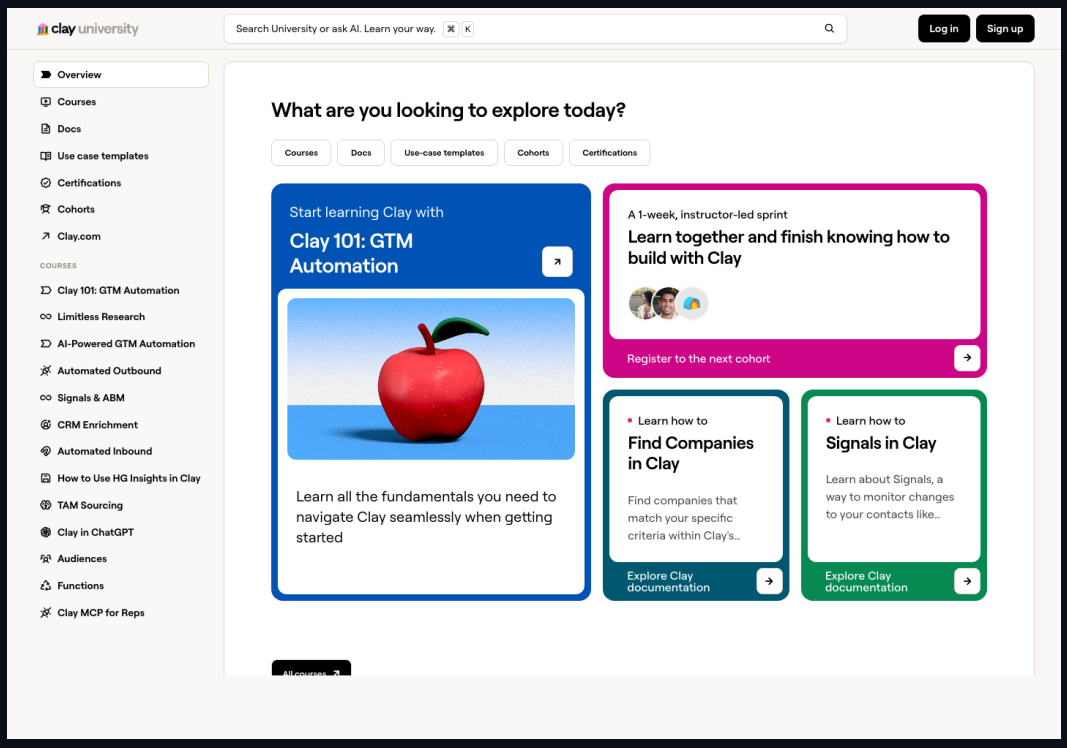
WHERE TO CLICK

Create a search from ICP filters, export a small sample, and validate the data before scaling.

ENV NAMES

`APOLLO_API_KEY if using the API.`

Clay: click path for this route.



WHERE TO CLICK

Create a table, add enrichment columns, run on a small batch, and export approved rows only.

ENV NAMES

```
Use connected accounts or API keys only inside Clay/workers.
```

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

INSPECT PROMPT

```
Inspect this repo for the LinkedIn Outreach Agent Basics 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 LinkedIn Outreach Agent Basics.
Outcome: Understand the outreach loop before scaling it with Apollo, Clay, Smartlead, and CRM automation.
Tools allowed: Apollo, Clay, LinkedIn, HubSpot, OpenAI, Claude, Supabase, Smartlead
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: LinkedIn Outreach Agent Basics
Outcome: Understand the outreach loop before scaling it with Apollo, Clay, Smartlead, and CRM automation.
Tools: Apollo, Clay, LinkedIn, HubSpot, OpenAI, Claude, Supabase, Smartlead

# .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 chrome and go to apollo.io. Sign in. Click Search in the top nav.

02 Filter by Title (Founder, CEO, Owner) and Company size 1 to 10 and your target industry. Click Search.

03 Pick five leads that fit your offer. Click each one and click Save to a new list named Outreach v1.

04 Click Export on the list. Download the CSV. Open it in Numbers or Excel to confirm the columns.

05 Open Supabase Table editor. Create a leads table with columns: id (uuid), name (text), title (text), company (text), linkedin_url (text), status (text default 'new'), reason_for_contact (text), opener (text), approved (bool default false).

06 In Supabase Table editor, click Insert and choose Import data from CSV. Upload your Apollo CSV. Confirm five rows imported.

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

01 For each lead, open their LinkedIn URL. Read their last three posts. Pick one real signal: a post they wrote, a job change, a podcast they were on, or a feature shipped.

02 Open Claude or ChatGPT. Paste this prompt: Write a 3-sentence personalized LinkedIn message for [NAME], [TITLE] at [COMPANY]. Real reason to reach out: [SIGNAL]. Sound human and brief. No fake compliments.

03 Generate openers for all five leads. Paste each opener back into the leads.opener column in Supabase.

04 Read each opener out loud. If it sounds like a template, regenerate that one with a more specific signal.

05 Mark approved=true on the three best openers. Leave the other two false so you remember to rewrite them.

06 Send the three approved openers MANUALLY through LinkedIn. Wait 48 hours. Add a replied_at column to leads and log the responses.

The build is not finished until verification is boring.

01 Can every sentence be traced to source data?

02 Does the opener sound human when read out loud?

03 Did the lead opt out or already receive a message?

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 Automating DMs before you know the message is good.

02 Personalization that invents facts.

03 Sending to stale or unverified contact data.

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

Draft ten approved outreach openers from real lead evidence, but send none until reviewed.

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

Apollo: https://docs.apollo.io/

Clay: https://docs.clay.com/

LinkedIn: Use the official documentation or dashboard for this tool.

HubSpot: https://developers.hubspot.com/docs

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