

Tapflow

Technical Architecture

Version 1.0 — January 28, 2026

Final Decisions • Ship in 4 Weeks

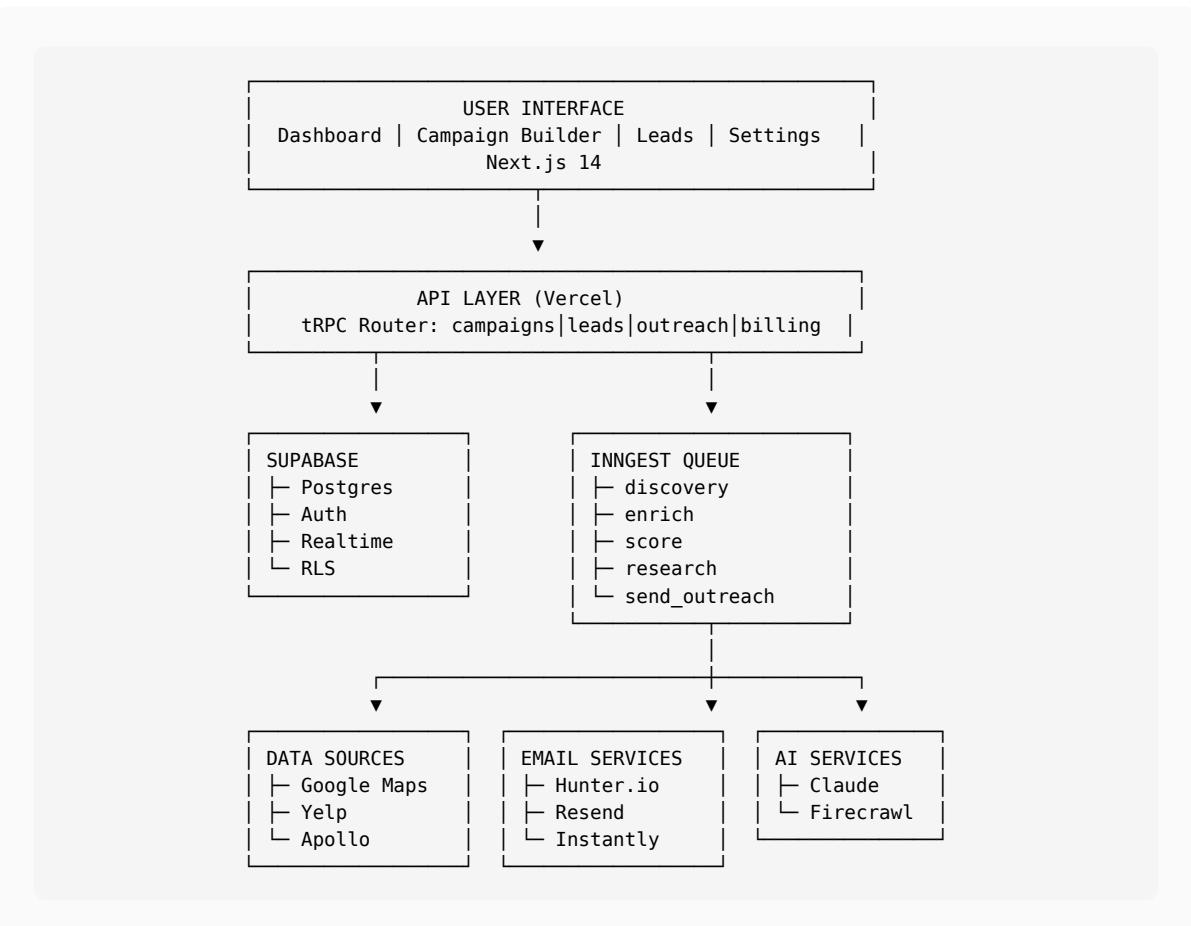
1 Executive Summary

This document defines the complete technical architecture for Tapflow, the AI-powered Lead Generation SaaS. Every decision is final—no hedging. The goal: **ship MVP in 4 weeks, scale to 10K users without re-architecture.**

1.1 Stack at a Glance

Layer	Decision	Reason
Hosting	Vercel	Zero-ops, edge functions, generous free tier
Database	Supabase	Postgres + Auth + Realtime in one
Queue	Inngest	Serverless-native, no Redis to manage
Auth	Supabase Auth	Already using Supabase, one less vendor
AI	Claude Sonnet 4	Best cost/quality for writing tasks
Email	Resend + Instantly	Modern APIs, best deliverability
Payments	Stripe	Industry standard

2 System Architecture



3 Infrastructure Decisions

3.1 Hosting: Vercel

Decision: Vercel

Rejected: Railway, AWS

Factor	Vercel	Railway	AWS
Zero-config deploy	Yes	Yes	No
Edge functions	Global	Regional	Complex
Free tier	100GB BW	5 credit	Pay-per-use
Next.js native	Yes	Good	Manual
Ops overhead	Zero	Low	High

Reasoning: We're building Next.js—Vercel made Next.js. Zero ops = ship faster.

3.2 Database: Supabase

Decision: Supabase

Rejected: PlanetScale, Neon

Factor	Supabase	PlanetScale	Neon
Real Postgres	Yes	MySQL	Yes
Built-in Auth	Yes	No	No
Realtime	WebSocket	No	No
Row Level Security	Native	No	No
File Storage	S3-compat	No	No

Reasoning: Auth + Database + Realtime + Storage = one vendor. RLS makes multi-tenancy trivial.

3.3 Queue: Inngest

Decision: Inngest

Rejected: Trigger.dev, BullMQ

Reasoning: No Redis to manage. Step functions let us break agents into resumable stages. 25K free runs covers MVP.

4 Database Schema

4.1 Core Tables

```
-- Users and Organizations
CREATE TABLE organizations (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    name TEXT NOT NULL,
    plan TEXT DEFAULT 'starter',
    stripe_customer_id TEXT,
    created_at TIMESTAMPTZ DEFAULT NOW()
);

CREATE TABLE campaigns (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    org_id UUID REFERENCES organizations(id),
    name TEXT NOT NULL,
    business_type TEXT NOT NULL,
    target_location TEXT NOT NULL,
    status TEXT DEFAULT 'active',
    created_at TIMESTAMPTZ DEFAULT NOW()
);

CREATE TABLE prospects (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    campaign_id UUID REFERENCES campaigns(id),
    company_name TEXT NOT NULL,
    website TEXT,
    phone TEXT,
    address TEXT,
    source TEXT,
    created_at TIMESTAMPTZ DEFAULT NOW()
);

CREATE TABLE contacts (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    prospect_id UUID REFERENCES prospects(id),
    name TEXT,
    title TEXT,
    email TEXT,
    email_verified BOOLEAN DEFAULT false,
    linkedin_url TEXT,
    is_primary BOOLEAN DEFAULT false
);

CREATE TABLE lead_scores (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    prospect_id UUID REFERENCES prospects(id),
    score INTEGER,
    tier TEXT,
    scoring_factors JSONB,
    scored_at TIMESTAMPTZ DEFAULT NOW()
);
```

5 Agent Orchestration

5.1 Event Flow

```
campaign.created → discovery.started → prospect.found →  
prospect.enriched → prospect.scored → outreach.generated →  
[HUMAN APPROVAL] → outreach.sent → reply.received
```

5.2 Agent Pipeline

Agent	Trigger	Output
Discovery	campaign.created	List of raw prospects
Enrichment	prospect.found	Contact info, tech stack
Research	prospect.found	Pain points, website analysis
Scoring	prospect.enriched	A/B/C tier assignment
Content	prospect.scored	Personalized email sequence
Outreach	outreach.approved	Email sent via Instantly

5.3 Concurrency Limits

- Discovery: 10 concurrent jobs
- Enrichment: 20 concurrent (parallelized)
- Research: 5 concurrent (API rate limits)
- Sending: 2 concurrent (deliverability)

6 Cost Estimates

6.1 Monthly Infrastructure

Service	100 Users	1K Users	10K Users
Vercel	Free	20/mo	150/mo
Supabase	Free	25/mo	75/mo
Inngest	Free	50/mo	250/mo
Claude API	50/mo	300/mo	2000/mo
Hunter.io	34/mo	104/mo	500/mo
Instantly	37/mo	97/mo	500/mo
Firecrawl	16/mo	83/mo	500/mo
Stripe fees	30/mo	250/mo	2500/mo
Total	167/mo	929/mo	6475/mo

6.2 Revenue vs Cost

Users	MRR	Cost	Margin
100	15K	167	98.9%
1,000	120K	929	99.2%
10,000	750K	6,475	99.1%

Break-even: 12 users at average 150/mo

7 Reusable Components from slc-lead-gen

File	Reuse	Effort
lead-discovery.js	90%	4 hours
website-scorer.js	100%	2 hours
business-scraper.js	100%	2 hours
email-generator.js	80%	6 hours
v2/agents/	100%	1 hour
twilio-client.js	100%	1 hour

Total porting effort: 26 hours (3.5 dev days)

8 Implementation Timeline

8.1 Week 1: Foundation

- Supabase project + schema migration
- Next.js + tRPC setup
- Supabase Auth flow
- Basic dashboard shell

8.2 Week 2: Core Pipeline

- Port discovery agent
- Port enrichment flow
- Lead list view
- Campaign creation wizard

8.3 Week 3: AI + Outreach

- Scoring agent
- Content generation
- Outreach approval queue
- Instantly integration

8.4 Week 4: Polish + Launch

- Stripe billing
- Usage tracking
- Error handling
- Beta launch to 10 customers

Document generated by Claw • Architecture finalized January 28, 2026