ChatGPT

Project Architecture and Technology Stack

We will build a newsletter web app that scrapes Twitter (X) content, stores it in Supabase, sends emails via Beehiiv, and tracks user stats with Google Analytics. The site will be hosted on Vercel. Below is a comprehensive integration plan covering each component.

Hosting on Vercel & Supabase Setup

- **Vercel Deployment:** Use a Next.js or static site hosted on Vercel. Vercel easily integrates with Supabase. For example, you can start with the official Supabase Next.js template (`npx create-next-app --example with-supabase`) which is pre-configured for Vercel [1].
- **Supabase Project:** Create a Supabase (Postgres) project via the Supabase dashboard (free tier available). Supabase provides a hosted database, authentication, and real-time API [2].
- **Environment Variables:** In your Vercel project settings (or local `.env`), set the Supabase URL and anon key obtained from your project's API settings. For example, rename `.env.example` to `.env.local` and fill in `NEXT_PUBLIC_SUPABASE_URL` and `NEXT_PUBLIC_SUPABASE_ANON_KEY` [3]. This lets your front-end code securely connect to Supabase.
- **Supabase Tables:** Define tables such as `subscribers` (to track user info and subscription status) and any content tables (e.g. `tweets` or `articles`). Use Supabase's SQL editor or CLI to create these. Supabase supports real-time updates and serverless functions (Edge Functions) if needed [2].

Google Analytics Integration

- **GA4 Property:** Create a Google Analytics 4 property for your site. In the GA dashboard, choose "Web" as the platform and follow the setup to get your Measurement ID (format `G-XXXXXXX`) [4].
- **Add Tracking Code:** Integrate the GA tracking code into your site. For a Next.js app, you might use a library like `react-ga` or `next-ga`. For example, in a custom component you can initialize it with your Measurement ID and log pageviews on route changes [5]. Alternatively, insert the GA `<script>` snippet in a global `<head>` component.
- **Verify Tracking:** Deploy your changes to Vercel. Visit your site in a browser and then check the Google Analytics **Realtime** dashboard – you should see your visit logged [6]. Google Analytics will then gather detailed traffic statistics for your site. (Note: Vercel also has built-in analytics for performance, but GA provides deeper user behavior insights [7].)

Twitter Scraping with Apify

- **Apify Tweet Scraper V2:** Use the **Apify Tweet Scraper V2** (X/Twitter Scraper) to fetch tweets. This is a ready-made actor that can perform searches, fetch a user's tweets, replies, etc., at about \$0.40 per 1,000 tweets [8].
- **Apify Account & Token:** Sign up for an Apify account and obtain an API token from the Apify Console. You will use this token to authenticate API requests [9].

- **Configure Inputs:** Decide what tweets to scrape. For example, set `searchTerms` for keyword searches or `startUrls` for specific profiles or tweet URLs. The Apify docs show input examples using JSON (e.g. `"searchTerms": ["from:NASA since:2024-01-01 ..."]`) [10].
- **Run the Scraper via API:** Use the Apify Client library in your code. For instance, in Python:

```
from apify_client import ApifyClient
client = ApifyClient("<YOUR_API_TOKEN>")
run_input = {
  "searchTerms": ["some query"],
  "startUrls": ["https://twitter.com/someuser"]
}
run = client.actor("apidojo/tweet-scraper").call(run_input=run_input)
```

  This calls the Apify actor programmatically [11].
- **Store Results:** The actor returns structured tweet data (text, metadata, etc.). Save these results into Supabase (e.g. insert into a `tweets` table) for further processing or display.

Email Newsletter with Beehiiv

- **Beehiiv Platform:** Use Beehiiv as the newsletter platform. Beehiiv provides tools to create and send newsletters, manage subscriber lists, and analyze campaign performance. Its API "lets you seamlessly manage your email newsletter platform, enabling tasks such as subscriber management, campaign analytics, and content automation" [12].
- **Subscriber Management:** Keep subscriber data in Supabase so you are not locked to one provider. You can use Beehiiv's API to add/update subscribers (or segments) from your application. For example, whenever a new user pays, your backend can call Beehiiv's **Create Subscription** endpoint to add them to the newsletter list. If you ever switch providers, having the list in your own DB means you can export it easily.
- **Beehiiv API & Webhooks:** Beehiiv v2 API supports operations like adding/updating subscriptions and fetching posts. You can also use webhooks to react to events (e.g. subscriber joins) if needed [12]. This allows tight integration: e.g., when a user signs up on your site, use Beehiiv's API to subscribe them automatically.
- **Content Delivery:** Compose your newsletter posts in Beehiiv's editor, or use its API to generate campaign content. Beehiiv will handle emailing to your subscriber list on each send. You can segment or tag subscribers as needed for targeted sends.

Paid Subscription Flow (Stripe Integration)

- **Stripe for Payments:** For paid newsletter subscriptions, integrate Stripe. You have two main approaches:

- *Option A – Beehiiv Built-in:* Beehiiv offers a paid subscription feature (on higher plans) that directly connects to Stripe [13]. By toggling on paid subscriptions and linking your Stripe account, Beehiiv handles the checkout flow and subscriber tagging automatically. After a reader pays via Stripe, Beehiiv can grant them access to paid content tiers with no extra coding.

- *Option B – Custom Integration:* Alternatively, implement Stripe Checkout yourself. Create Stripe Products/Prices and use Stripe's Checkout UI (hosted or embedded) for the payment. When the payment succeeds, use a backend (e.g. a Supabase Edge Function) to handle the `checkout.session.completed` webhook. Supabase Edge Functions are ideal for this – they can verify Stripe's signature and then update your database [14]. After verifying the payment, the function would mark the user as subscribed in your Supabase table and call Beehiiv's API to add the subscriber (simulating Beehiiv's flow).

- **Automatic Subscription:** In either case, ensure **payment precedes subscription**. For custom flow, your webhook handler should only create the subscription after successful payment. For example, upon receiving Stripe's event, add the user's email to your `subscribers` table and call:

```
POST https://api.beehiiv.com/v2/publications/[PUB_ID]/subscriptions
Authorization: Bearer <API_KEY>
Content-Type: application/json

{ "email": "user@example.com", "firstName": "User", ... }
```

  This adds them as a Beehiiv subscriber. (Beehiiv's docs show how to add subscriptions via API.)

- **Summary of Flow:** User clicks "Subscribe," is taken through Stripe checkout, and on success our system automatically subscribes them. This can be done with minimal delay: Stripe sends a webhook, our function confirms the payment, and the user is added to Beehiiv and our DB. This fully automates the pay-then-subscribe logic.

Analytics & Monitoring

- **Google Analytics on Vercel:** With GA4 set up as above, you can monitor site traffic, pageviews, and conversions (e.g. how many reach the payment page). Embed GA code on all pages (for Next.js, wrap `ReactGA.pageview` in a `useEffect` on route change). Check GA's real-time view after deployment to ensure data flows [6].
- **Other Metrics:** Beehiiv provides newsletter open and click rates. Vercel's own analytics can show performance metrics. Together with GA, you'll have a full picture of both site usage and newsletter engagement.

Data Flow Summary

1. **Content Ingestion:** Run the Apify Tweet Scraper on a schedule (e.g. via CRON or serverless function). Store scraped tweets in Supabase for content.
2. **Website Content:** Build your site pages (news lists, article detail, etc.) fetching data from Supabase. Host it on Vercel.
3. **User Sign-Up:** Site has a subscription signup form. Upon submission and successful Stripe payment, a webhook/Edge Function adds the user to Supabase and calls the Beehiiv API to subscribe them.
4. **Newsletter Sending:** Use Beehiiv's interface or API to compose and send newsletters to your `subscribers` list. Beehiiv will send emails and track opens/clicks.
5. **Analytics:** Google Analytics (via the tracking code) and Beehiiv's analytics let you monitor traffic and subscriber metrics in real time.

By following this plan, you'll have a cohesive stack: **Vercel** for fast hosting, **Supabase** for flexible backend storage, **Google Analytics** for stats [6], **Apify** for Twitter scraping [8] [11], **Beehiiv** for newsletters [12], and **Stripe** for payments [13] [14]. This architecture ensures a smooth user experience where payment automatically triggers newsletter subscription.

Sources: Vercel/Supabase starter docs [1] [3]; Google Analytics integration guide [4] [6]; Apify Tweet Scraper documentation [8] [11]; Beehiiv API/Integration info [12] [13]; Supabase Edge Functions guide [14].

[1] [3] Supabase Starter
https://vercel.com/templates/authentication/supabase

[2] [12] Integrate the Beehiiv API with the Supabase API - Pipedream
https://pipedream.com/apps/beehiiv/integrations/supabase

[4] [5] [6] [7] Add Google Analytics to Vercel
https://www.simpleanalytics.com/glossary/google-analytics/add-google-analytics-to-vercel

[8] [10] Tweet Scraper V2 - X / Twitter Scraper · Apify
https://apify.com/apidojo/tweet-scraper

[9] [11] Tweet Scraper V2 - X / Twitter Scraper API in Python · Apify
https://apify.com/apidojo/tweet-scraper/api/python

[13] How to set up a Stripe account for paid subscriptions - Knowledge Base - beehiiv
https://www.beehiiv.com/support/article/12643795789335-how-to-set-up-a-stripe-account-for-paid-subscriptions?srsltid=AfmBOoppzrHJXhb7Anlwh2FzeJrGES952Li_toi5tmAavKhXf6cL60Xd

[14] Edge Functions | Supabase Docs
https://supabase.com/docs/guides/functions