15. How to Build an Onchain App Using the Dune API

Onchain analysts aren’t just dashboard builders. They're increasingly becoming **backend architects**—powering real-time applications, alerts, and user-facing data tools.

With Dune’s public API, you can build apps that pull directly from live blockchain queries. This article walks you through how to use the Dune API to turn your dashboards into apps.

🧠 What You’ll Learn

- What the Dune API can do
- The lifecycle of a query execution
- How to build a frontend app that monitors blockchain activity
- Real-world example: **Uniswap Pool Watcher**

🔌 Dune API Basics

Dune recently opened its API to **all users**, including free-tier accounts.

The API is centered around **query execution**, not just static dashboards. That means your apps can:

- Run saved SQL queries
- Get the latest blockchain data
- Monitor changes and respond dynamically

To get started, grab your API key from Dune API Settings.

⚙️ The API Workflow

Every Dune API integration follows the same 3-step pattern:

1. **Execute a query**
2. **Check execution status**
3. **Fetch results**

Here's what each step looks like:

1. Execute

```
bash

POST https://api.dune.com/api/v1/query/<query_id>/execute
```

Pass in your API key and any required parameters.

2. Check Status

```
bash


CopyEdit
GET https://api.dune.com/api/v1/execution/<execution_id>/status
```

Poll this endpoint until status is `completed`.

3. Fetch Results

```
bash


CopyEdit
GET https://api.dune.com/api/v1/execution/<execution_id>/results
```

This returns the result set of your SQL query in JSON format.

🛠️ Example Project: Uniswap Pool Watcher

Let's say you want to build a frontend that:

- Tracks newly created Uniswap V3 pools
- Lets users pick a pool and see the latest swaps
- Sends alerts for large transactions (e.g., >$10k)

Here's the architecture we'd use:

Frontend:

- Next.js
- Tailwind
- Axios (for API requests)
- Dexie (for local storage)

Backend:

- Prisma + SQLite (to store execution IDs & cache data)

API:

- Dune's public API, with 3 queries:
  - New Pools: `https://dune.com/queries/2056212`
  - Latest Swaps: `https://dune.com/queries/2056310`
  - Large Swaps: `https://dune.com/queries/2056547`

👨‍💻 Code Snippets

Here's a basic function to execute a Dune query:

```
ts


CopyEdit
export const executeQuery = async (id: string, params: any) => {
  const res = await axios.post(
    `https://api.dune.com/api/v1/query/${id}/execute`,
    { parameters: params },
    { headers: { 'x-dune-api-key': YOUR_API_KEY } }
  );
  return res.data.execution_id;
};
```

Then check the execution status:

```
ts


CopyEdit
export const getStatus = async (execution_id: string) => {
  const res = await axios.get(
    `https://api.dune.com/api/v1/execution/${execution_id}/status`,
    { headers: { 'x-dune-api-key': YOUR_API_KEY } }
  );
  return res.data.state;
};
```

And finally, fetch the results:

```
ts


CopyEdit
export const getResults = async (execution_id: string) => {
  const res = await axios.get(
    `https://api.dune.com/api/v1/execution/${execution_id}/results`,
    { headers: { 'x-dune-api-key': YOUR_API_KEY } }
  );
  return res.data.result.rows;
};
```

🧩 Features You Can Add

- Re-run queries every 5 mins for fresh data
- Alert users with toasts or push notifications
- Save favorite pools to local storage

- Graph swap activity over time
- Add login, permissions, and dashboards

⚠️ Caching and Limits

- Execution IDs are valid for **24 hours**
- Avoid unnecessary API calls—cache recent executions
- If you rerun the same query too often, you'll hit rate limits (especially on free plans)

Use a local DB (like SQLite or Dexie) to store execution IDs and prevent duplicate runs.

🌐 Deploying Your App

The final stack can be deployed like any full-stack JS app:

- Vercel or Netlify for frontend
- Railway or Supabase for backend
- GitHub for version control

🧠 Why This Matters

Dashboards are great. But **applications drive action**.

With the Dune API, onchain analysts can:

- Build analytics tools for DAOs
- Create investor dashboards
- Send real-time DeFi alerts

- Monitor NFT mints and wash trading

You're not limited to charts. You can ship full products.

🧭 Next: ERC-4337 and the Rise of Account Abstraction

Now that you've built apps on top of blockchain data, let's dive deeper into **how new forms of user accounts are reshaping onchain behavior**—and what you need to know to analyze them effectively.

Next: 16. Account Abstraction — Why It Matters for Wallet UX and Analysts