Content Dashboard Guide

Build a Real-Time Analytics Dashboard With Claude Code

Track every Instagram and YouTube post on one screen. No coding experience required.

Section 1: Set Up Claude Code

Before you build anything, you need Claude Code installed on your computer. This takes about two minutes.

Step 1: Download VS Code Go to code.visualstudio.com in your browser. Click the big blue download button for your operating system (Mac, Windows, or Linux). Open the downloaded file and drag VS Code into your Applications folder (Mac) or run the installer (Windows).

Step 2: Install the Claude Code Extension Open VS Code. Look at the left sidebar. You will see a row of icons. Click the one that looks like four squares (this is the Extensions panel). In the search bar at the top, type "Claude Code". You will see it in the results. Click the blue "Install" button.

Step 3: Sign In After installing, you will see a Claude Code icon in your sidebar. Click it. It will ask you to sign in with your Anthropic account. If you do not have one yet, go to console.anthropic.com and create a free account. Sign in and you are ready.

Step 4: Verify It Works Open the Claude Code chat panel (click the Claude icon in the sidebar). Type "Hello" and press Enter. If Claude responds, you are good to go.

Section 2: Set Up Supabase

Your dashboard needs a database to store post data. Supabase is free for small projects and takes about five minutes to set up.

Step 1: Create a Supabase Account Go to supabase.com in your browser. Click "Start your project" and sign up with your GitHub account (or email). If you do not have GitHub, use email.

Step 2: Create a New Project Click "New Project". Give it a name like "my-content-dashboard". Choose a region close to you. Set a database password (save this somewhere safe). Click "Create new project" and wait about 30 seconds for it to spin up.

Step 3: Get Your Connection String In your Supabase dashboard, click "Project Settings" in the left sidebar. Click "Database". Scroll down to "Connection string" and click "URI". Copy the entire connection string. It looks something like:
`postgresql://postgres:yourpassword@db.abcdefgh.supabase.co:5432/postgres`

Step 4: Add to Your Environment In your project folder, create a file called `.env` (or open it if it already exists). Add this line:

```
DATABASE_URL=postgresql://postgres:yourpassword@db.abcdefgh.supabase.co:5432/postgres
```

Replace the connection string with the one you copied.

Section 3: Connect Instagram Graph API

The Instagram Graph API lets your dashboard pull in post metrics like views, likes, saves, and shares.

Step 1: Create a Meta Developer App Go to developers.facebook.com. Click "My Apps" in the top right. Click "Create App". Choose "Business" type. Give it a name like "Content Dashboard". Click "Create App".

Step 2: Add Instagram Graph API In your app dashboard, find "Add Products" and click "Set Up" on Instagram Graph API. This adds the Instagram product to your app.

Step 3: Generate an Access Token Go to Graph API Explorer (developers.facebook.com/tools/explorer/). Select your app. Click "Get Token" and choose "Get User Access Token". Check the permissions: `instagram_basic`, `instagram_manage_insights`, `pages_show_list`, `pages_read_engagement`. Click "Generate Access Token" and authorize.

Step 4: Extend to Long-Lived Token Short tokens expire in 1 hour. You need a long-lived token. Tell Claude Code:

```
I have a short-lived Instagram Graph API token. Exchange it for a long-lived token using the Meta
API. My app ID and secret are in the .env file.
```

Step 5: Add to Environment Add this to your `.env` file:

```
INSTAGRAM_ACCESS_TOKEN=your_long_lived_token_here
```

Section 4: Connect YouTube Data API

The YouTube Data API lets your dashboard pull in video metrics like views, likes, comments, and watch time.

Step 1: Go to Google Cloud Console Open console.cloud.google.com. Sign in with the Google account that owns your YouTube channel.

Step 2: Create a Project Click the project dropdown at the top. Click "New Project". Name it "Content Dashboard". Click "Create".

Step 3: Enable the YouTube Data API In the left sidebar, click "APIs & Services" then "Library". Search for "YouTube Data API v3". Click it, then click "Enable".

Step 4: Create Credentials Click "APIs & Services" then "Credentials" in the sidebar. Click "Create Credentials" then "API key". Copy the API key.

Step 5: Add to Environment Add this to your `.env` file:

```
YOUTUBE_API_KEY=your_api_key_here
```

Section 5: Build the Sync Pipeline

Your dashboard needs fresh data. A sync pipeline automatically pulls post metrics from Instagram and YouTube every 6 hours.

Tell Claude Code:

```
Build a cron job that runs every 6 hours and does the following:
1. Pulls my 25 most recent Instagram posts using the Instagram Graph API and stores their metrics
(views, likes, comments, shares, saves, reach) in a Supabase table called "content_analytics"
2. Pulls my 25 most recent YouTube videos using the YouTube Data API and stores their metrics (views,
likes, comments, watch time) in the same table
3. Each record should include: platform, post_id, title, thumbnail_url, published_at, views, likes,
comments, shares, saves, reach, engagement_rate
4. Upsert by post_id so we update existing records instead of creating duplicates
```

Claude Code will create the database table, the API integration code, and the cron job. Deploy it and your data starts flowing automatically.

Section 6: Build the Dashboard Page

Now for the fun part. Tell Claude Code exactly what you want to see.

KPI Cards:

```
Build a dashboard page with 6 KPI cards across the top:
1. Unique Viewers — total unique views across all posts
2. Reach — total reach across all posts
3. Engagement Rate — average engagement rate
4. Followers — current follower count
5. Shares — total shares
6. Saves — total saves

Each card should show: the current value, a percentage change vs the previous period (with an up or down arrow), and a small sparkline chart showing the trend over the selected time range.
```

Charts:

```
Below the KPI cards, add two charts:
1. A "Views & Reach" area chart showing views and reach over time. X-axis is dates, Y-axis is count. Two overlapping areas with different colors.
2. An "Engagement Breakdown" bar chart showing likes, comments, shares, and saves as grouped bars.
```

Top Posts Grid:

```
Below the charts, add a "Top Performing Posts" grid that shows a list of posts sorted by the selected metric. Each row shows: thumbnail, platform badge (Instagram or YouTube), title, post date, and inline metrics (views, likes, shares, engagement rate). Add a dropdown to sort by different metrics.
```

Section 7: Add Filters and Controls

Your dashboard needs controls so you can slice the data different ways.

Tell Claude Code:

```
Add the following controls to the dashboard:
1. Platform filter buttons at the top: All, Instagram, YouTube. Clicking one filters all KPI cards,
charts, and the posts grid to that platform only.
2. Time range buttons: 7d, 14d, 30d, 90d. Clicking one changes the time range for all KPI cards
(including the percentage change calculation) and charts.
3. On the top posts grid, add a sort dropdown with options: Views, Likes, Shares, Engagement Rate.
Default to Views.
```

Section 8: Deploy to Vercel

Time to make your dashboard live so you can access it from anywhere.

Step 1: Push to GitHub If your project is not already on GitHub, tell Claude Code:

```
Initialize a git repository, create a .gitignore that excludes .env and node_modules, and push to a
new GitHub repository called "content-dashboard".
```

Step 2: Connect Vercel Go to vercel.com and sign in with your GitHub account. Click "Add New Project". Select your content-dashboard repository. Vercel auto-detects the framework. Click "Deploy".

Step 3: Add Environment Variables In your Vercel project settings, go to "Environment Variables". Add all the variables from your `.env` file: DATABASE_URL, INSTAGRAM_ACCESS_TOKEN, YOUTUBE_API_KEY.

Step 4: Verify Open your Vercel URL. Your dashboard should load with real data. The cron job will keep it updated every 6 hours automatically.

Section 9: Set Up ManyChat Automation

This section walks you through setting up the ManyChat Comment Automation so people can comment "CONTENT DASHBOARD" on your posts and automatically receive this guide in their DMs.

Step 1: Install the ManyChat Automation Tool Open your terminal in VS Code (Terminal menu, then New Terminal). Run:

```
node tools/manychat-automation.cjs --login
```

This opens a browser window. Sign in to ManyChat with your Google account. Complete 2FA if prompted. The session saves automatically.

Step 2: Create the JSON Config Create a file called `content-dashboard-manychat.json` with this content:

```
{
  "name": "CONTENT DASHBOARD DM",
  "keyword": "CONTENT DASHBOARD",
  "commentReplies": [
    "Sending the dashboard guide to your DMs now!",
    "Sent you a message with the guide, check it out!",
    "Check your DMs for the dashboard guide!"
  ],
  "openingDM": "Hey, I have got the Content Dashboard Guide ready for you. It walks through every step to build a real-time analytics dashboard with Claude Code from scratch. Tap the button below and I will send it right over.",
  "buttonLabel": "Send me the link",
  "followGateText": "Nearly there! The Content Dashboard Guide is especially for my followers. Right after you follow me, I will send you the link so you can dive straight in!",
  "deliveryDM": "Here is the Content Dashboard Guide. It covers everything from installing Claude Code to connecting Instagram and YouTube APIs to deploying your live dashboard. Ten sections, step by step, starting from zero. No coding experience needed.",
  "linkLabel": "Content Dashboard Guide",
  "linkUrl": "YOUR_DRIVE_LINK_HERE",
  "followUpDM": "Hey, just checking in. Did you grab the Content Dashboard Guide? The link is still ready for you whenever you are."
}
```

Replace `YOUR_DRIVE_LINK_HERE` with the Google Drive link to the PDF.

Step 3: Run the Automation

```
node tools/manychat-automation.cjs --config content-dashboard-manychat.json
```

This creates the Comment Automation in ManyChat. It goes live immediately. Anyone who comments "CONTENT DASHBOARD" on any of your posts will receive the guide via DM.

Section 10: What to Track and Why

Now that your dashboard is live, here is how to use it.

KPI Cards to Watch:

- **Engagement Rate** is the most important number. It tells you what percentage of viewers are interacting with your content. Higher engagement = more reach from the algorithm.
- **Saves** are the strongest signal on Instagram. A save means someone wants to come back to your content. High save rate = reference-worthy content.
- **Shares** are the fastest path to viral reach. Each share puts your content in front of a new audience.

Trend Arrows: The up/down arrows compare your current period to the previous period. If you selected "30d", the arrow compares the last 30 days to the 30 days before that. A red down arrow on engagement rate means your content quality may be slipping. A green up arrow on shares means your content is getting more amplified.

Top Posts Leaderboard: Sort by different metrics to learn different things:

- Sort by **Views** to find your highest-reach content
- Sort by **Saves** to find your most valuable content
- Sort by **Engagement Rate** to find your most engaging content
- Sort by **Shares** to find your most viral content

The pattern in your top posts tells you what to make more of.

What is Next?

You now have a real-time content dashboard that tracks every post across Instagram and YouTube. No more checking three different apps. No more guessing what is working.

For more Claude Code skills and build guides, follow @KyleWhitrow on Instagram.

Comment **CONTENT DASHBOARD** on any of my posts to get this guide sent to your DMs.

Send an email to **kyle@nustimulus.com** for inquiries.

nustimulus.com