

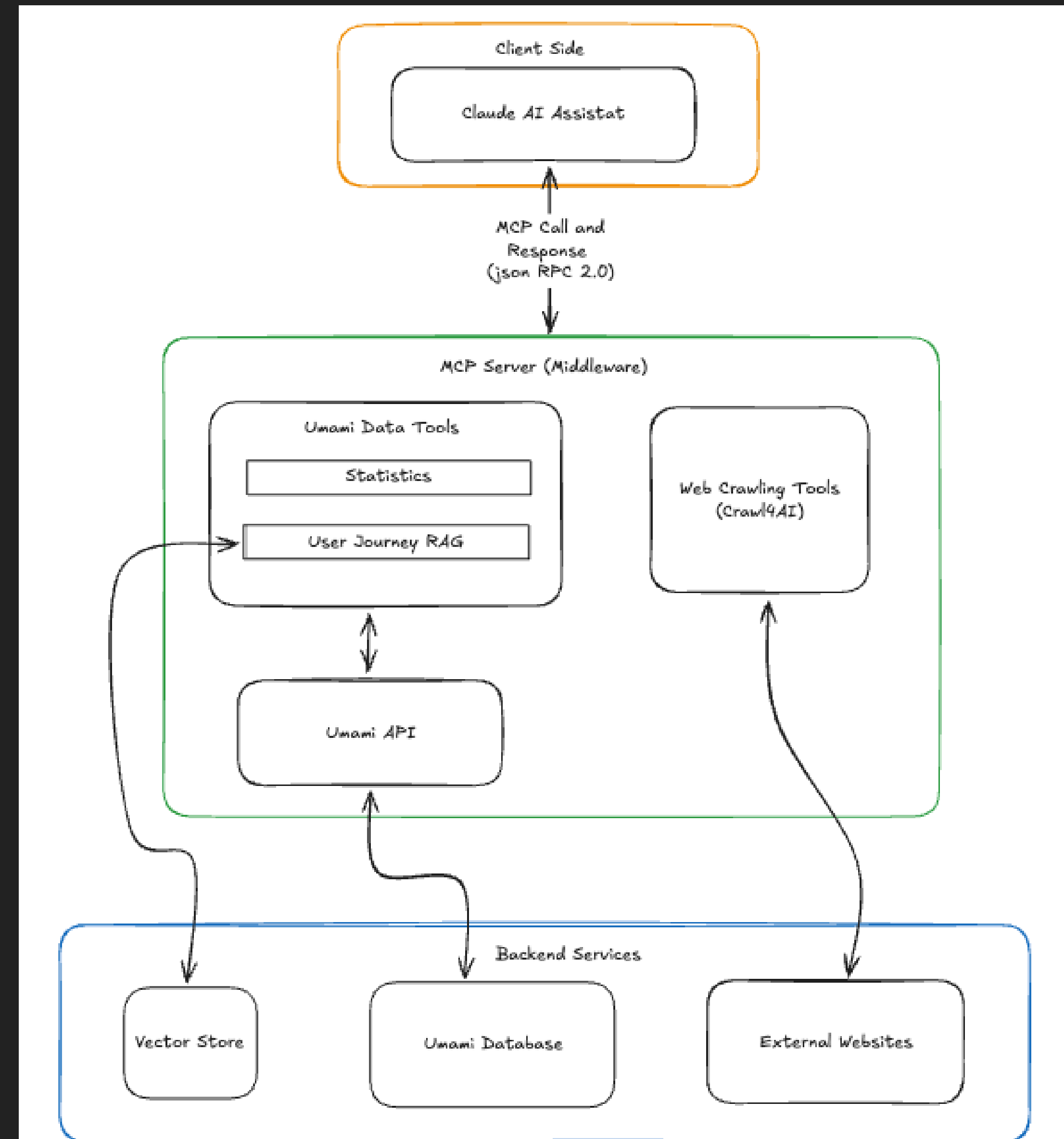
**MODEL CONTEXT PROTOCOL AND
CLAUDE**

**HARNESSING AI FOR
ADVANCED WEB
ANALYTICS GENERATION**

MCP ARCHITECTURE

Key Features

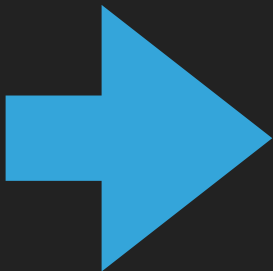
- ▶ Standardised method for LLMs to interact with external apps
- ▶ Allows for creation of middleware servers to enhance LLM capabilities
- ▶ Connect new data sources



DEVELOPMENT PROCESS WITH CLAUDE AND CURSOR

Existing MCP server code (Context)

```
weather_service > src > weather_service > server.py > ...
1 import os
2 import json
3 import logging
4 from datetime import datetime, timedelta
5 from collections.abc import Sequence
6 from functools import lru_cache
7 from typing import Any
8
9 import httpx
10 import asyncio
11 from dotenv import load_dotenv
12 from mcp.server import Server
13 from mcp.types import (
14     Resource,
15     Tool,
16     TextContent,
17     ImageContent,
18     EmbeddedResource,
19     LoggingLevel
20 )
21 from pydantic import AnyUrl
22
23 # Load environment variables
24 load_dotenv()
25
26 # Configure logging
27 logging.basicConfig(level=logging.INFO)
28 logger = logging.getLogger("weather-server")
29
30 # API configuration
31 API_KEY = os.getenv("OPENWEATHER_API_KEY")
32 if not API_KEY:
33     raise ValueError("OPENWEATHER_API_KEY environment variable required")
34
35 API_BASE_URL = "http://api.openweathermap.org/data/2.5"
36 DEFAULT_CITY = "London"
37 CURRENT_WEATHER_ENDPOINT = "weather"
38 FORECAST_ENDPOINT = "forecast"
39
40 # The rest of our server implementation will go here
41 # Create reusable params
42 http_params = {
43     "appid": API_KEY,
44     "units": "metric"
45 }
46
47 async def fetch_weather(city: str) -> dict[str, Any]:
48     async with httpx.AsyncClient() as client:
49         response = await client.get(
50             f"{API_BASE_URL}/{CURRENT_WEATHER_ENDPOINT}",
51             params={"q": city, **http_params}
52         )
53         response.raise_for_status()
54         data = response.json()
```



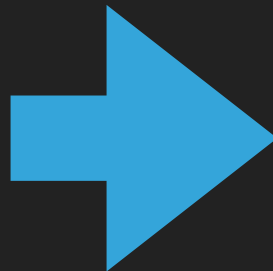
Cursor and Claude

- ▶ Analyse existing server implementation
- ▶ Understand syntax and packages
- ▶ Generate new code consistent with example
- ▶ Maintain best practices from example

+ weather_service Folder x

Using @weather_service as a reference for how to build an MCP server, create me a new mcp server called analytics_service. The purpose of this server will be to send various website stats and metrics to claude to assist it in alaysing website performance and build dashboards. Lets begin by building out the main structure for the server. Then once this done we will start to build all of the tools I want

claude-3.5-sonnet Image normal / agent submit



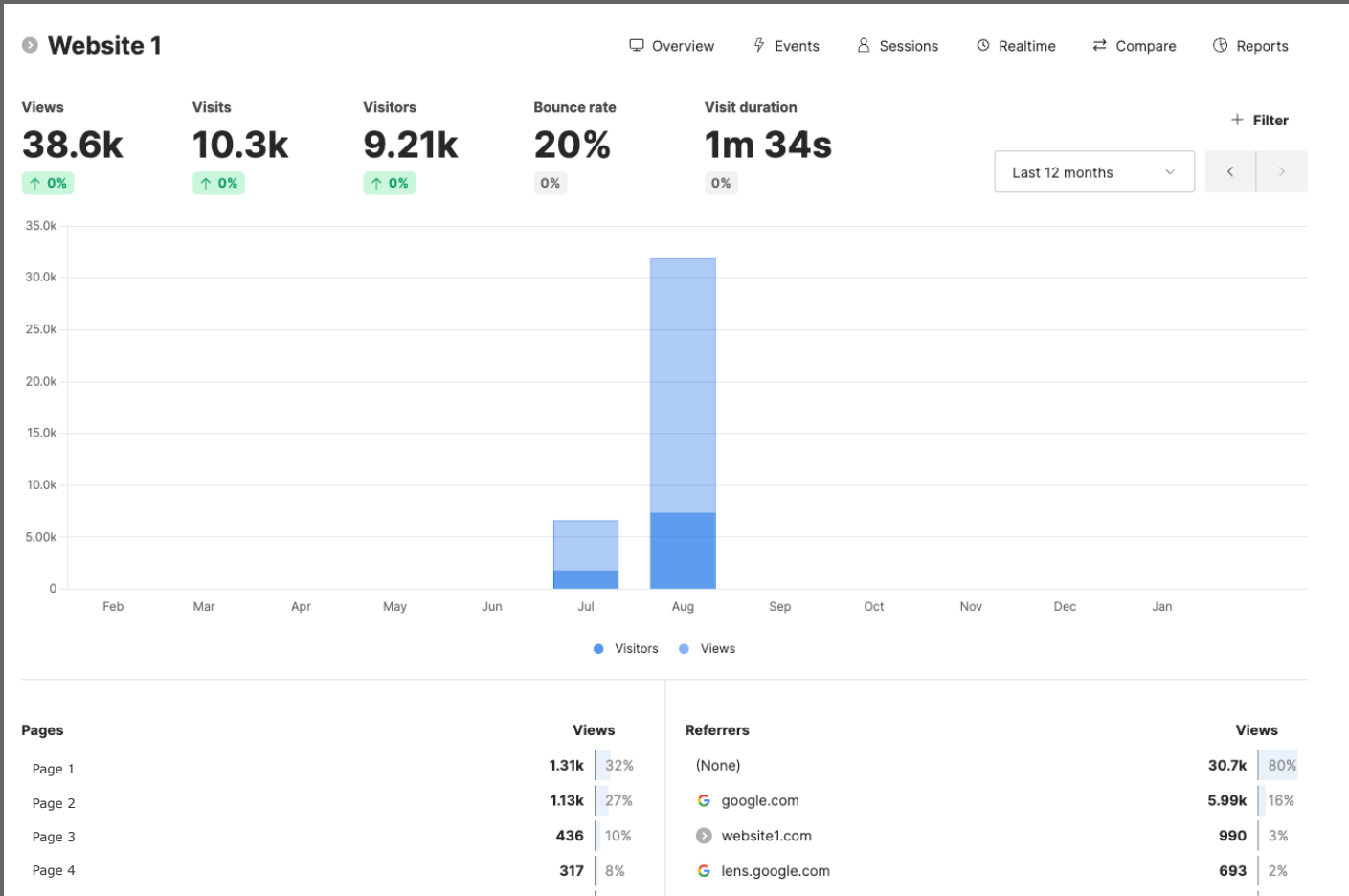
Result: Production Ready Analytics MCP Server

```
tracking_service > src > tracking_service > server.py > call_tool
1 import os
2 import json
3 import logging
4 from typing import Any, Sequence
5 from datetime import datetime
6
7 import httpx
8 from dotenv import load_dotenv
9 from mcp.server import Server
10 from mcp.types import Resource, Tool, TextContent, AnyUrl, ImageContent, EmbeddedResource
11
12 from .api import UmamiClient
13
14 # Load environment variables
15 load_dotenv()
16
17 # Configure logging
18 logging.basicConfig(level=logging.INFO)
19 logger = logging.getLogger("tracking-server")
20
21 # API configuration
22 API_BASE_URL = os.getenv("UMAMI_API_URL")
23 API_USERNAME = os.getenv("UMAMI_USERNAME")
24 API_PASSWORD = os.getenv("UMAMI_PASSWORD")
25 TEAM_ID = os.getenv("UMAMI_TEAM_ID")
26
27 if not all([API_BASE_URL, API_USERNAME, API_PASSWORD, TEAM_ID]):
28     raise ValueError("Missing required environment variables")
29
30 # Initialize API client
31 client = UmamiClient(API_BASE_URL)
32
33 # Ensure client is logged in at startup
34 if not client.login(API_USERNAME, API_PASSWORD):
35     raise RuntimeError("Failed to login to Umami API")
36 if not client.verify_token():
37     raise RuntimeError("Failed to verify Umami API token")
38
39 def get_session_ids(website_id, event_name, start_at, end_at):
40     """
41     Retrieve session IDs for a specific event on a website.
42
43     Args:
44         website_id (str): ID of the website
45         event_name (str): Name of the event to filter by
46
47     Returns:
48         list: Unique session IDs associated with the event
49     """
50     ids = []
51     page = 1
52     while True:
53         events_where = client.get_events_where(
```

AI POWERED DATA ANALYSIS UNLOCKED

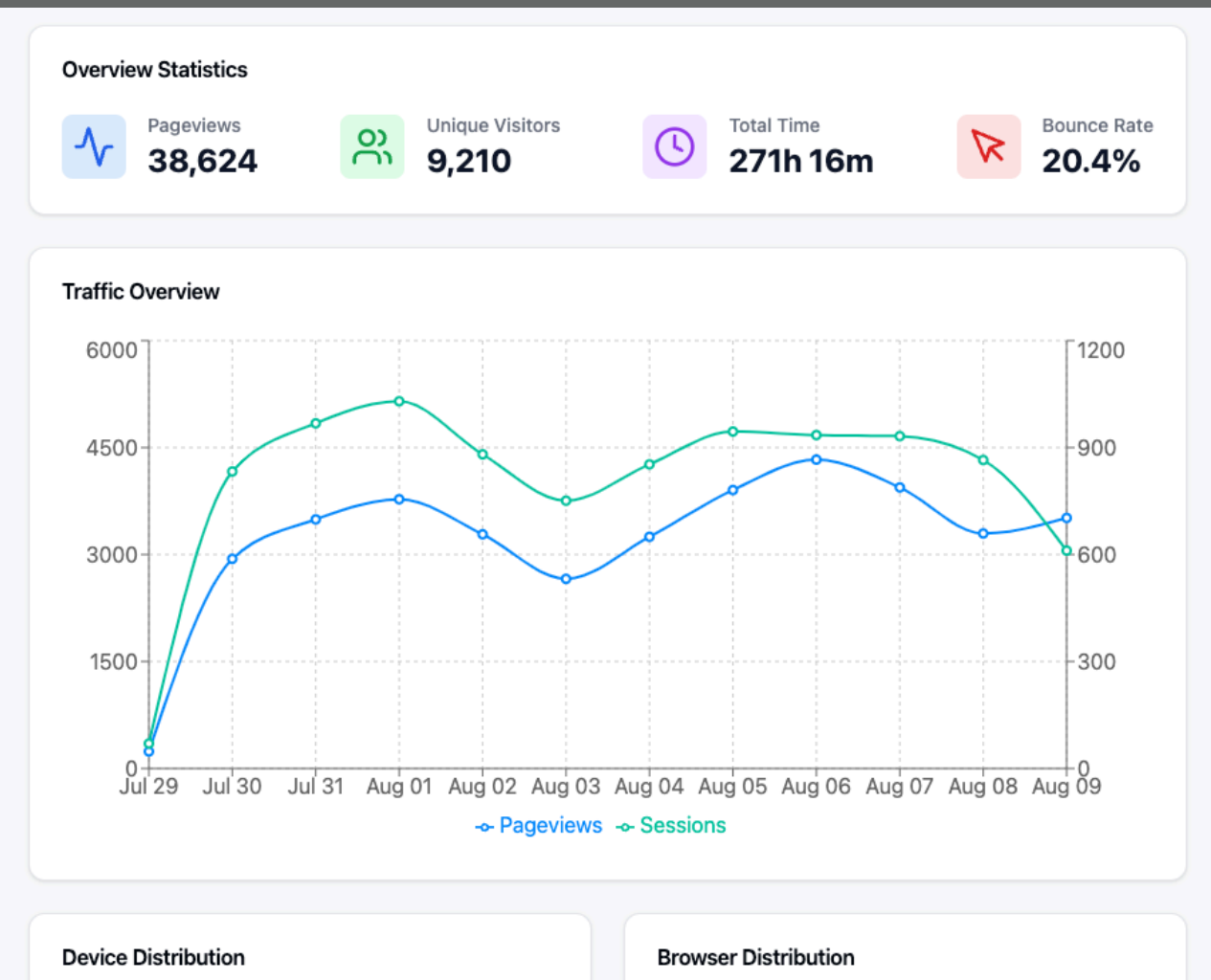
BEFORE MCP

- ▶ Static dashboard
- ▶ Limited presets and filtering
- ▶ Manual data analysis and insight recognition
- ▶ Fixed reporting views



AFTER MCP

- ▶ Customisable dashboards using natural language
- ▶ AI-powered analysis
- ▶ Dynamic insights and recommendations
- ▶ Complex user journey analysis



USER JOURNEY ANALYSIS DEMO

(1) Radom user journey visualisation

Claude

I've created a visualization of this user's journey through the website. Here's a summary of their behavior:

- Entry Point:** The user entered the site via Google Hong Kong at 13:12, initially interested in a Fujifilm Instax Pal digital camera.
- Initial Product Interest:** They spent several minutes examining the camera in different colors (pink and white), showing strong interest in this product.
- Category Exploration:** Around 13:25, they moved to explore the Lifestyle category, particularly focusing on stationery items.
- Product Browsing Pattern:** The user showed a pattern of thorough product investigation:
 - Viewed multiple color variants of each product
 - Spent significant time on product detail pages
 - Frequently returned to category pages to explore more options
- Category Transitions:** Throughout their session, they moved through several categories:
 - Technology/Cameras → Lifestyle/Stationery → Beauty → Women's Fashion
 - Each transition showing a logical flow of interest
- Engagement Level:** The user showed high engagement:
 - Long session duration (2h 39m)
 - Multiple product detail views
 - Active interaction with product variants
 - Several category explorations
- Final Interaction:** Their session ended at 15:51 while viewing a Stanley Legendary Classic Canteen, showing their interests evolved from electronics to lifestyle products.

Would you like me to modify the visualization or provide more detailed analysis of any particular part of their journey?

JB

Ok, do the same but this time find a user that made it all the way to checkout

JB

Reply to Claude...

Claude 3.5 Sonnet

Choose style

2

13

Claude

List of Available Websites

User Journey Visualization

Preview

Code

13:12

1

Entered site

Came from Google HK to view Fujifilm camera

/product/fujifilm-instax-pal-digital-camera

>

pageview

13:13

2

Product browsing

Viewed multiple colors (pink & white) of Fujifilm Instax

/product/fujifilm-instax-pal-digital-camera

>

product_details_viewed

13:25

3

Category exploration

Moved to Lifestyle category, browsed notebooks and stationery

/lifestyle

>

pageview

13:28

4

Product interaction

Viewed multiple stationery items including Pocket Notebook and Crazy B Pencil Set

/product/pocket-notebook-3

>

product_details_viewed

13:42

5

Beauty category

Explored nail art products and accessories

/beauty

>

pageview

13:57

6

Women's fashion

Browsed women's category, sorted by best sellers

/womens

>

product_clicked

15:51

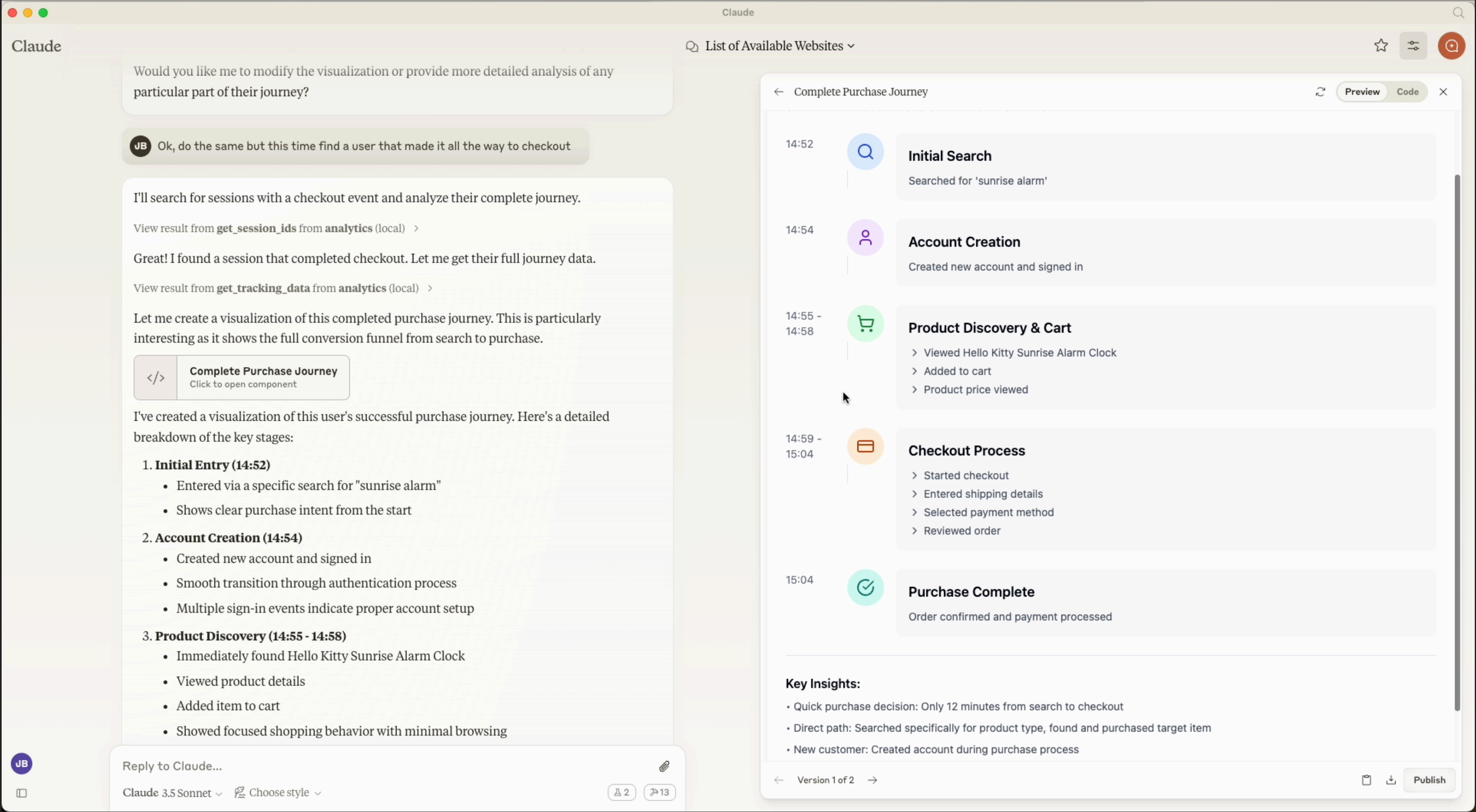
7

Last edited 33 minutes ago

Publish

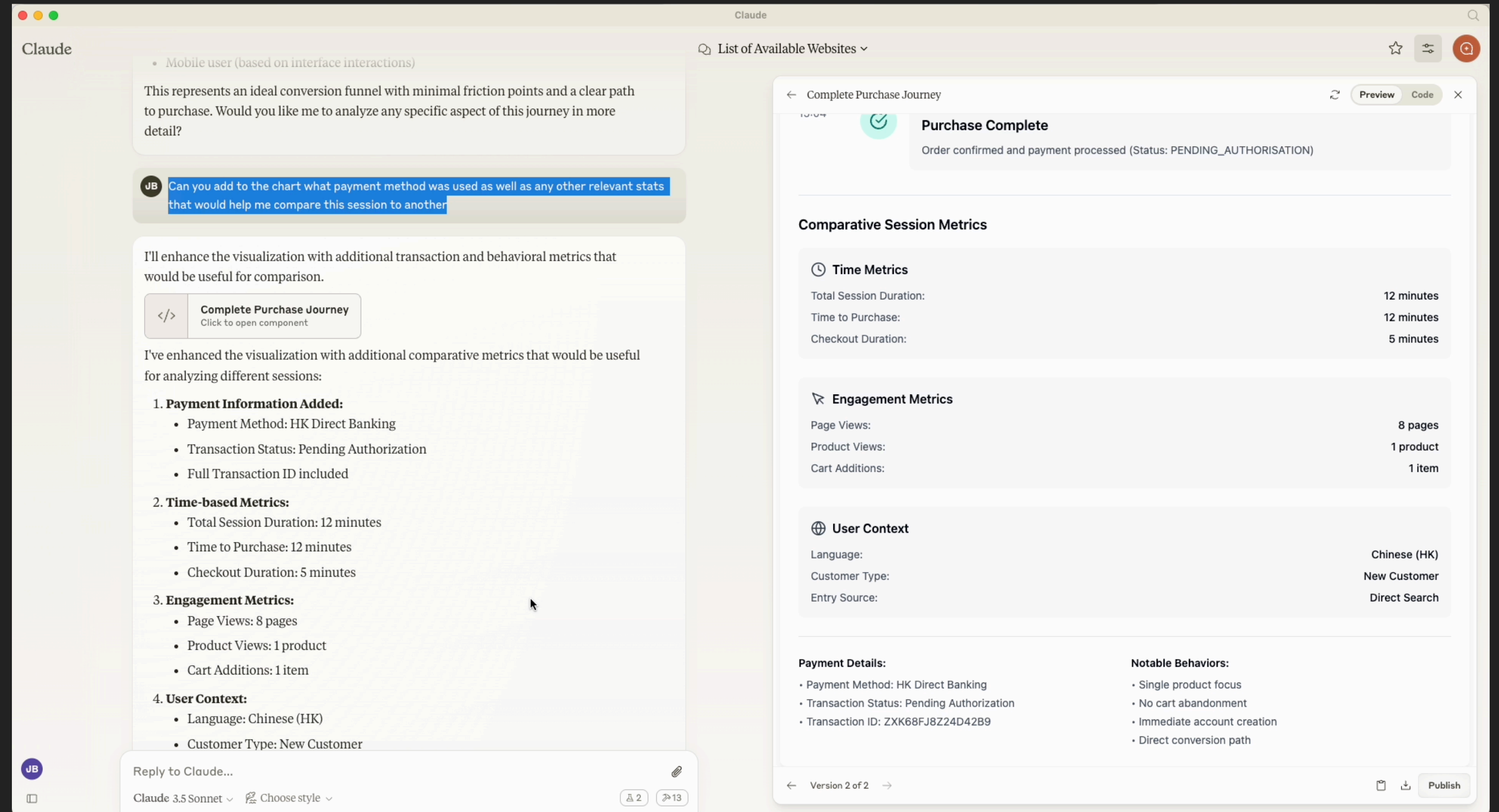
USER JOURNEY ANALYSIS DEMO

(2) Checkout complete journey visualisation



USER JOURNEY ANALYSIS DEMO

(3) Adding payment information + additional details



USER JOURNEY ANALYSIS DEMO

(4) Dynamic problem solve + analysis of two contrasting journeys

Claude

- Track new vs returning customer behaviors

Would you like me to add any other specific metrics or modify the visualization further?

JB

Ok, so now I want you to give me another random user journey who made it all the way through the checkout process. I'd like you to provide me with analysis of that user journey, including minutes taken between each of the key parts of that journey

I'll find another completed checkout session and analyze it with timing details.

View result from `get_session_ids` from `analytics` (local) >

I see we only have one completed checkout session in our dataset. Let me look instead for users who reached the "checkout_started" event, which might give us more comparison data.

View result from `get_session_ids` from `analytics` (local) >

Great! Let me get the journey data for one of these users who started checkout.

View result from `get_tracking_data` from `analytics` (local) >

Let me create a visualization for this second user's journey and compare it to our previous completed purchase.

</>

User Journey Comparison
Click to open component

I've created a detailed comparison between these two user journeys. Here are the key timing and behavioral differences:

Journey B (Abandoned Checkout):

- Entry to Product (0:00 - 0:10)**
 - Searched directly for "keshi"
 - Found Limited Edition LP immediately
 - Much faster than Journey A (10 seconds vs 1 minute)
- Product Interaction (0:10 - 0:25)**

Claude

List of Available Websites >

User Journey Comparison

Journey A (Completed)

🕒 14:52 → 15:04

Hello Kitty Sunrise Alarm Clock

Search to Product

Direct search for 'sunrise alarm'

1 min

Product to Cart

Immediate add to cart

3 min

Cart to Checkout

Started checkout quickly

1 min

Checkout Process

Completed full checkout

5 min

Total Duration

Completed Purchase

12 min

Journey B (Abandoned)

🕒 00:44 → 00:45

Keshi Requiem Limited LP

Search to Product

Direct search for 'keshi'

10 sec

Product to Cart

Quick add to cart

15 sec

Last edited 19 minutes ago

📄

📥

Publish

KEY TAKEAWAYS

**Middleware Revolution:
Simple MCP unless AI
power**

**AI can generate
answers and graphs
without you having to
create anything more
than the link**

**Codebase created start
to finish using Claude
Sonnet 3.5 and Cursor**

**Middleware can deal
with customer auth,
providing data and
much more**

**Dashboard creation no
longer requires
technical knowledge -
only natural language**