

A Zero-Cost Freight Flow Explorer for Utah

FAF origin-destination flows, DuckDB-WASM, and deck.gl — no backend required

Pukar Bhandari

pukar.bhandari@outlook.com

2026-10-19

Table of contents

1 Try it	1
2 What it shows	1
3 How it works	2
4 Stack	3

The Freight Analysis Framework (FAF) is the go-to public source for how much freight moves between U.S. regions, by commodity and mode. It is also awkward to use: several releases with different zone systems, benchmark years and dollar bases, plus a forecast that skips 2025 to 2029. Answering “what moves in and out of Utah?” usually means downloading gigabytes and wrangling them yourself.

So I built an explorer that does the wrangling once and ships the result as a static site. Utah is the configured state; adding another is a one-file change.

1 Try it

The live app is embedded below. Click a mode, commodity, partner state or year and every chart and the map follow.

[Open app GitHub](#)

2 What it shows

- **Origin-destination flows** in tons, value and ton-miles for every record that touches the state, by commodity, mode and year. FAF5.7.1 (2017 to 2050) and FAF6.0 (2022) are kept apart, never combined on one axis.
- **Forecast scenarios:** the mid-range forecast plus FAF5’s published High and Low scenarios, with forecast and preliminary years marked and the unpublished 2025 to 2029 shown as a gap.

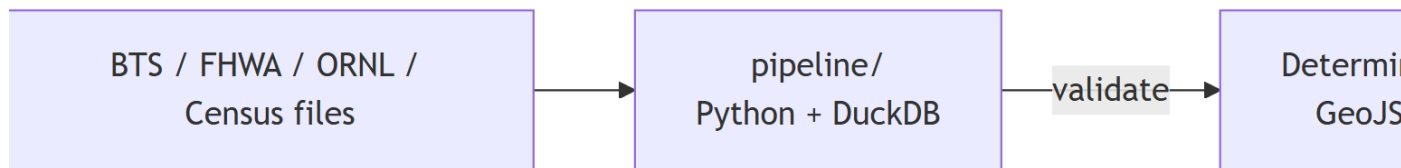
- **Linked charts instead of tables:** direction, mode and commodity (treemap), the largest partner states, trends with the scenario band, a commodity-by-year heat map, and a two-year comparison. Clicking any element filters everything else.
- **The FAF5 truck network** with modeled link volumes, and clearly labeled **experimental county-level estimates** for 2022.

! Data and caveats only

The app does not rank places, score performance, or present modeled values as measured counts. FAF flows are modeled estimates, and the app says so wherever they appear.

3 How it works

Nothing runs on a server. A local pipeline turns the raw federal files into Parquet, and the browser queries those files directly over HTTP range requests.



The pipeline is a small command-line tool run with uv:

```

cd pipeline
uv run python -m faf_pipeline update --state utah
uv run python -m faf_pipeline fetch --state utah --check-only
uv run python -m faf_pipeline validate --state utah
uv run pytest
  
```

Line 2

Fetch, build, zones, county estimates, network and validation in one go. It prints a changelog of what changed.

Line 3

Report changed files and new FAF releases without downloading anything.

Line 4

Run the checks that must pass before processed data is committed.

Line 5

Unit tests run on tiny fixture files for a fictional state.

Three design choices made the rest easy:

- **Deterministic output.** Re-running `update` on unchanged inputs produces byte-identical files, so a second run leaves no git diff and every data change is reviewable.
- **A schema guard.** Every downloaded file's columns are checked against a column map, so a new release that renames or drops a column fails loudly instead of quietly corrupting results.
- **No state literals in the code.** Utah appears only in `config/states/utah.yml`; validation fails if a state name, abbreviation or zone id shows up as a literal elsewhere.

4 Stack

Layer	Tools
Pipeline	Python, DuckDB, uv, pytest
App	Vite, Svelte 5, TypeScript
Maps	MapLibre, deck.gl, flowmap.gl, PMTiles
Queries	DuckDB-WASM over Parquet
Charts	Observable Plot

The methodology page lists every source file, release and caveat. The code is on GitHub — issues and PRs welcome.