Procedural website generation via WFC noise generation, deterministic layout denoising and LLM text synthesis

ZhongJun Qian 301541827

Project Idea & Use-Case

Generate synthetic, brandless webpage layouts for design exploration and phishing-detection research.

NLP is used here for text region fills

Dataset and models

For data wrangling YOLOv8 and my own dataset is used to train for semantic band detection

For NLP text gen Qwen/Qwen2.5-7B-Instruct:together via API call to Hugging Face.

Local LLM hosting with 6 GB is not ideal, colab integration into pipeline would slow down debugging

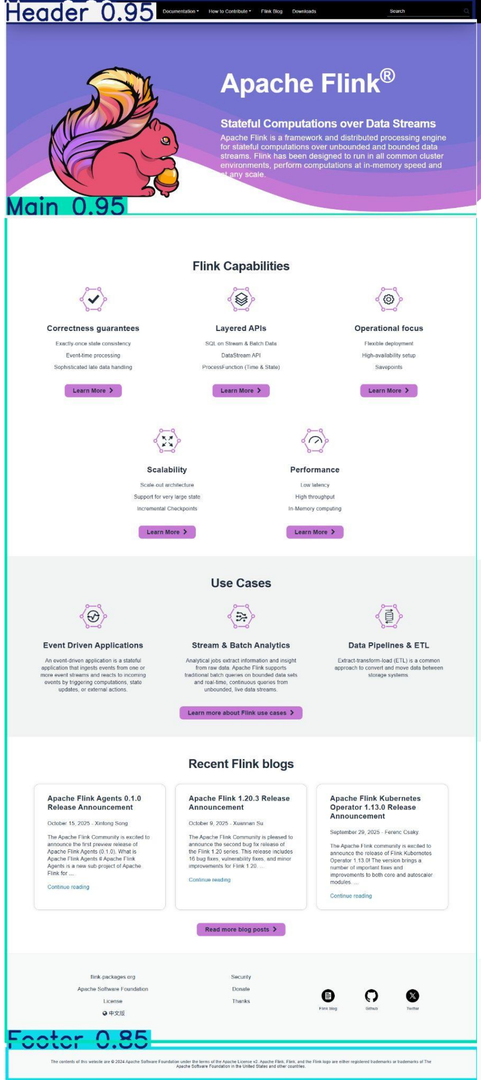
Bias/Balance

P1 data wrangling was mostly tech adjacent english websites, P2 RAG is based off of 50 of P1’s 200 site’s texts

Pipeline P1-overview

For data-wrangling:

Website semantic band is detected with CV, individual elements with RG based encoding



Apache Flink®

Stateful Computations over Data Streams

Apache Flink is a framework and distributed processing engine for stateful computations over unbounded and bounded data streams. Flink has been designed to run in all common cluster environments, perform computations at in-memory speed and at any scale.

Flink Capabilities

Correctness guarantees
Exactly-once state consistency
Event-time processing
Sophisticated late data handling
Learn More

Layered APIs
SQL on Stream & Batch Data
DataStream API
ProcessFunction (Time & State)
Learn More

Operational focus
Flexible deployment
High-availability setup
Savepoints
Learn More

Scalability
Scale-out architecture
Support for very large state
Incremental Checkpoints
Learn More

Performance
Low latency
High throughput
In-Memory computing
Learn More

Use Cases

Event Driven Applications
An event-driven application is a stateful application that ingests events from one or more event streams and reacts to incoming events by triggering computations, state updates, or external actions.

Stream & Batch Analytics
Analytical jobs extract information and insight from raw data. Apache Flink supports traditional batch queries on bounded data sets and real-time, continuous queries from unbounded, live data streams.

Data Pipelines & ETL
Extract-transform-load (ETL) is a common approach to convert and move data between storage systems.

Learn more about Flink use cases

Recent Flink blogs

Apache Flink Agents 0.1.0 Release Announcement
October 15, 2025 - Xintong Song
The Apache Flink Community is excited to announce the first preview release of Apache Flink Agents (0.1.0). What is Apache Flink Agents # Apache Flink Agents is a new sub-project of Apache Flink for ...
Continue reading

Apache Flink 1.20.3 Release Announcement
October 9, 2025 - Xuannan Su
The Apache Flink Community is pleased to announce the second bug fix release of the Flink 1.20 series. This release includes 16 bug fixes, vulnerability fixes, and minor improvements for Flink 1.20. ...
Continue reading

Apache Flink Kubernetes Operator 1.13.0 Release Announcement
September 29, 2025 - Ferenc Csaky
The Apache Flink community is excited to announce the release of Flink Kubernetes Operator 1.13.0! The version brings a number of important fixes and improvements to both core and autoscaler modules. ...
Continue reading

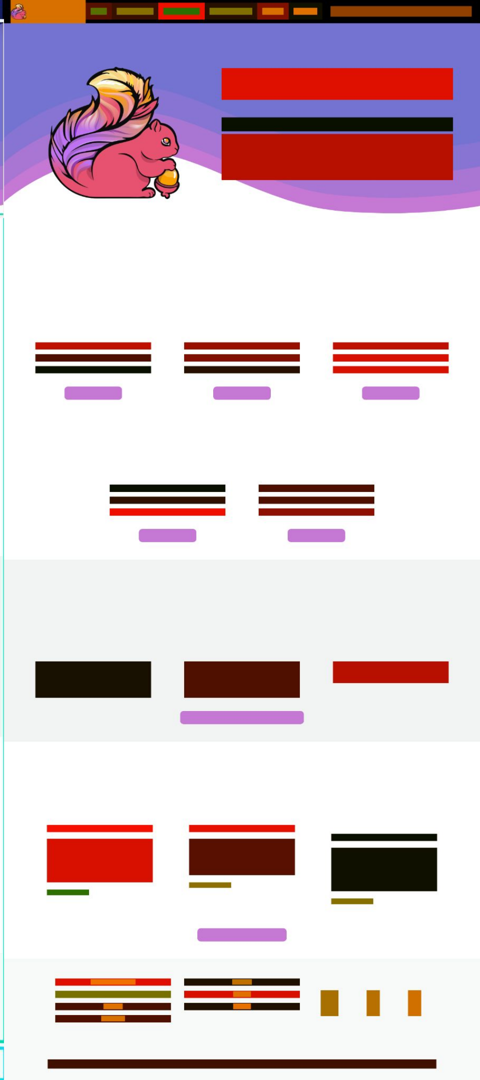
Read more blog posts

flink-packages.org
Apache Software Foundation
License
中文版

Security
Donate
Thanks

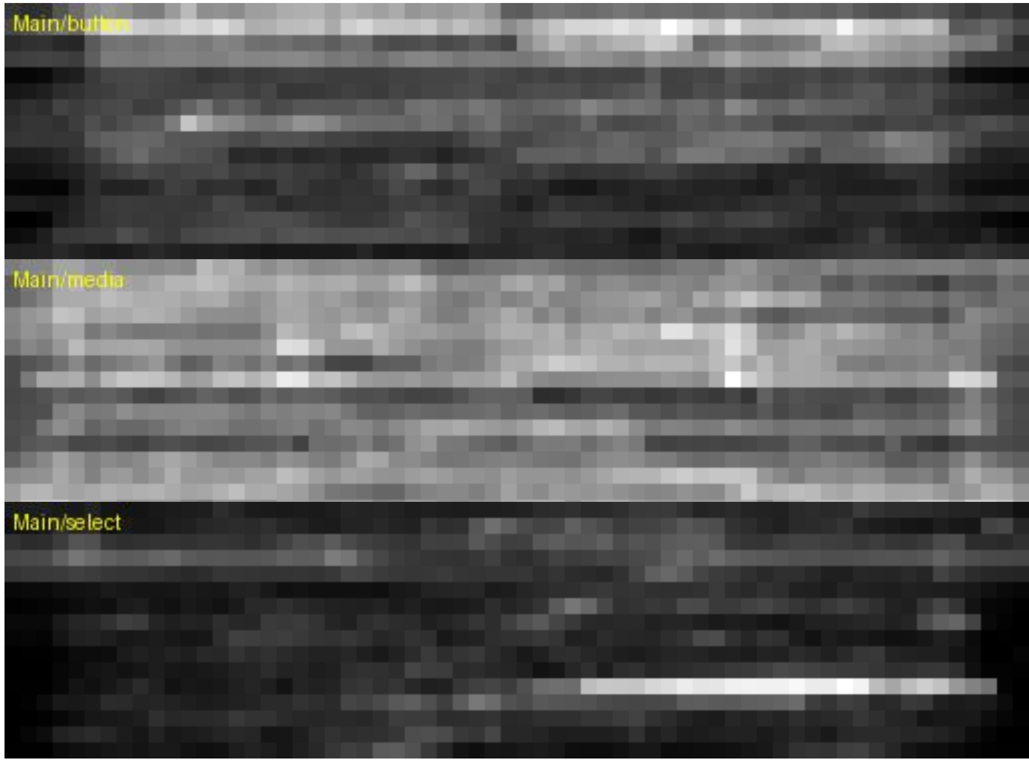
Flink blog
Github
Twitter

The contents of this website are © 2024 Apache Software Foundation under the terms of the Apache License v2. Apache Flink, Flink, and the Flink logo are either registered trademarks or trademarks of The Apache Software Foundation in the United States and other countries.



Pipeline P1-overview

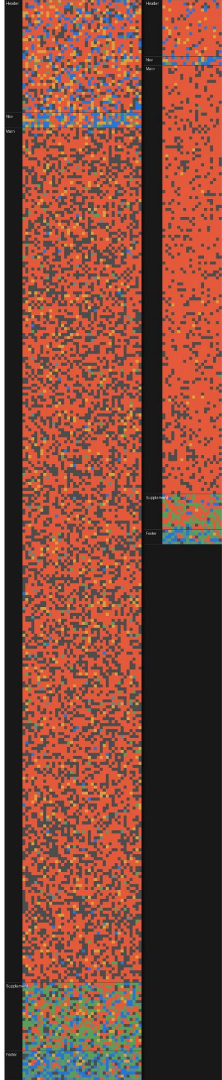
By aggregating the data I was able to procure element type density per band, as well as adjacency rules for WFC



P2-Generation

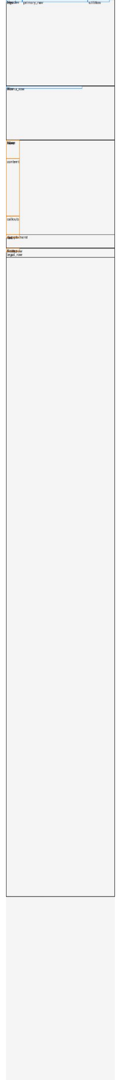
Wave Function Collapse based off of data from normalized wrangled data produced unusable noise on the left

Taking inspiration from Stable Diffusion, I used a 2×2 pooling step to smooth the WFC noise by collapsing tiny one-pixel fluctuations into more stable regions before semantic denoising



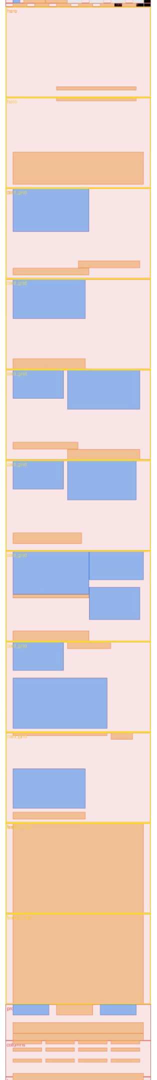
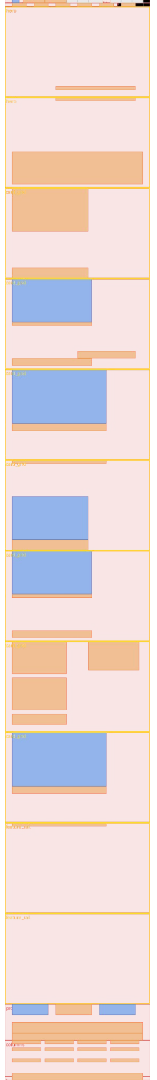
P2-Generation

The zone system detects large-scale patterns in the pooled WFC grid and assigns each region a semantic role like hero, card grid, or footer column based on dataset derived rules.



P2-Generation

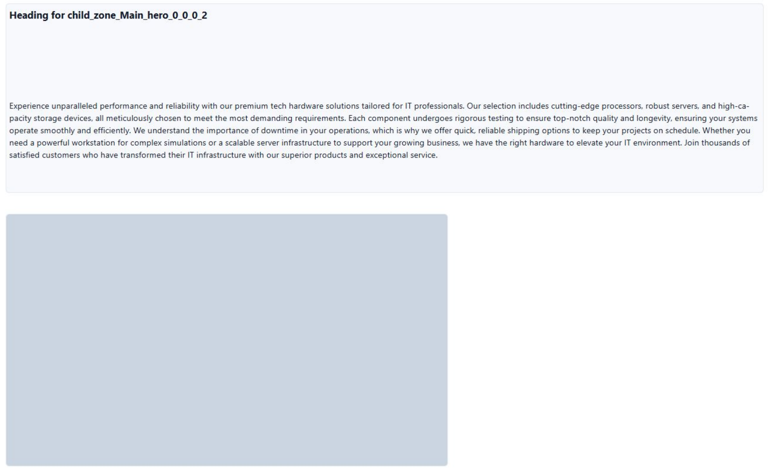
By applying design rules such as grids and view port restraints, and cross referencing the previously mentioned zones map I was able to achieve a somewhat realistic wireframe

P2-Generation

The character budget is calculated from grid area, not from browser font metrics.

Since text wraps naturally in CSS grid, area-based budgets are sufficient to keep the text visually contained



Heading for child_zone_Main_hero_0_0_0_2

Experience unparalleled performance and reliability with our premium tech hardware solutions tailored for IT professionals. Our selection includes cutting-edge processors, robust servers, and high-capacity storage devices, all meticulously chosen to meet the most demanding requirements. Each component undergoes rigorous testing to ensure top-notch quality and longevity, ensuring your systems operate smoothly and efficiently. We understand the importance of downtime in your operations, which is why we offer quick, reliable shipping options to keep your projects on schedule. Whether you need a powerful workstation for complex simulations or a scalable server infrastructure to support your growing business, we have the right hardware to elevate your IT environment. Join thousands of satisfied customers who have transformed their IT infrastructure with our superior products and exceptional service.

Prompt Engineering

A staged LLM pipeline:

1. Generate site name
2. Use site name as context to generate title and subtitle
3. Using site name and hero output to generate section plan and text take away
4. For each section I pass the previous section takeaway section num and intent and some RAG snippets
5. Return paragraph and split locally

Problem and solutions

Problem: Inconsistent, Nonsensical Text
Early runs with the LLM produced disconnected sections

Attempted Solution: Recursive, Context-Passing Generation
Recursively pass the previous context to the next generation to keep consistency with staged chains

New Problem: API Cost & Compounded Hallucination
Every stage is a new LLM call and feeding LLM calls into each other is worsening entropy

Stop gap solution:
Switch to an offline, neutral text corpus for most elements.
Use the staged LLM generator only as a design prototype / corpus builder, not as a per-element live dependency.

Final Discussion

Overall Project: To augment mass watering hole / phishing campaigns, to assist in cybersecurity research

NLP Model specific: Generate placeholder text for vertical slice/ MVP websites for fast demos and iterations

Data collected from both P1 and P2 as well as ROG corpus are tech hardware / software vendors, pipeline performs only on imitating said websites.