

Working with AI, Data and Pattern

Janet Huang

2023.05.11

Outline

- Session I: Working with Data
 - What is data, personal data?
 - “Dear Data” exercise
 - “Data Digitalization” exercise
- Break
- Session II: Working with AI, Data and Pattern
 - OpenAI api on Data Foundry
 - Starboard as data pattern exploration tool
 - “Working with AI” exercise

Working with Data

personal data and your story

**What is data?
What can be data?
And what data can be?**

Data is not just numbers

```
10100000000011100101010010101010100100011111  
11110100101010010001001100010100100011001010  
10101001101111010000001010101010010101100101  
010101001010101000000011111111111111111111100  
1111111111111111111100000000000000001010101010  
11110100101010100100101101001011001010101000  
000001111010101001010101010001001001100100101
```

because numbers are always a placeholder for something else

What is data?

What is Data?

“**Data** are factual information (such as measurements or statistics) used as a basic for reasoning, discussion, or calculation.”

(from Merriam-Webster)

“**Data** are units of information that are collected through **observation**.

In a more technical sense, data are a set of values of qualitative or quantitative variables about one or more persons or objects.”

(from Wikipedia)

Data is not just numbers

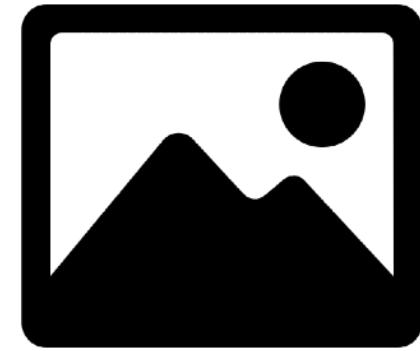
Data is about people;

Data is about story;

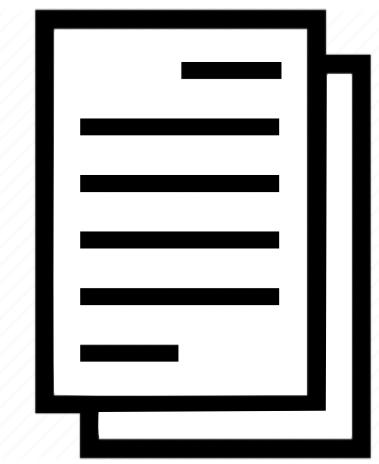
Data is the way we perceive the world.

because numbers are always a placeholder for something else

Data can be represented as many forms



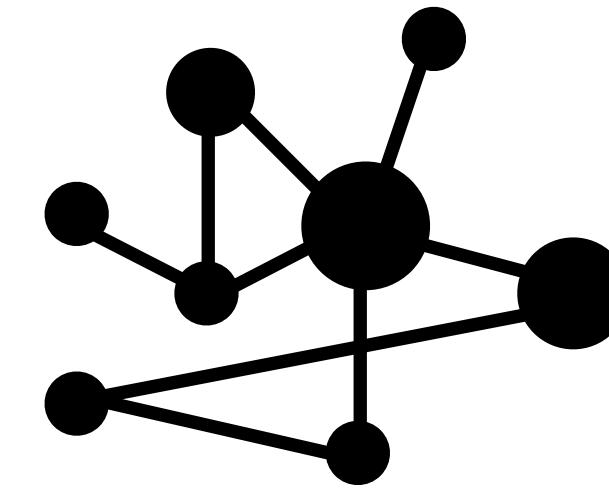
images



texts

1010000111001
010101010010100101
10001111111110100
0100100010011000101
01000110010101010

numbers



network



any forms

Personal Data

Smartphone app usage

Personality & Traits

Interests & Preferences

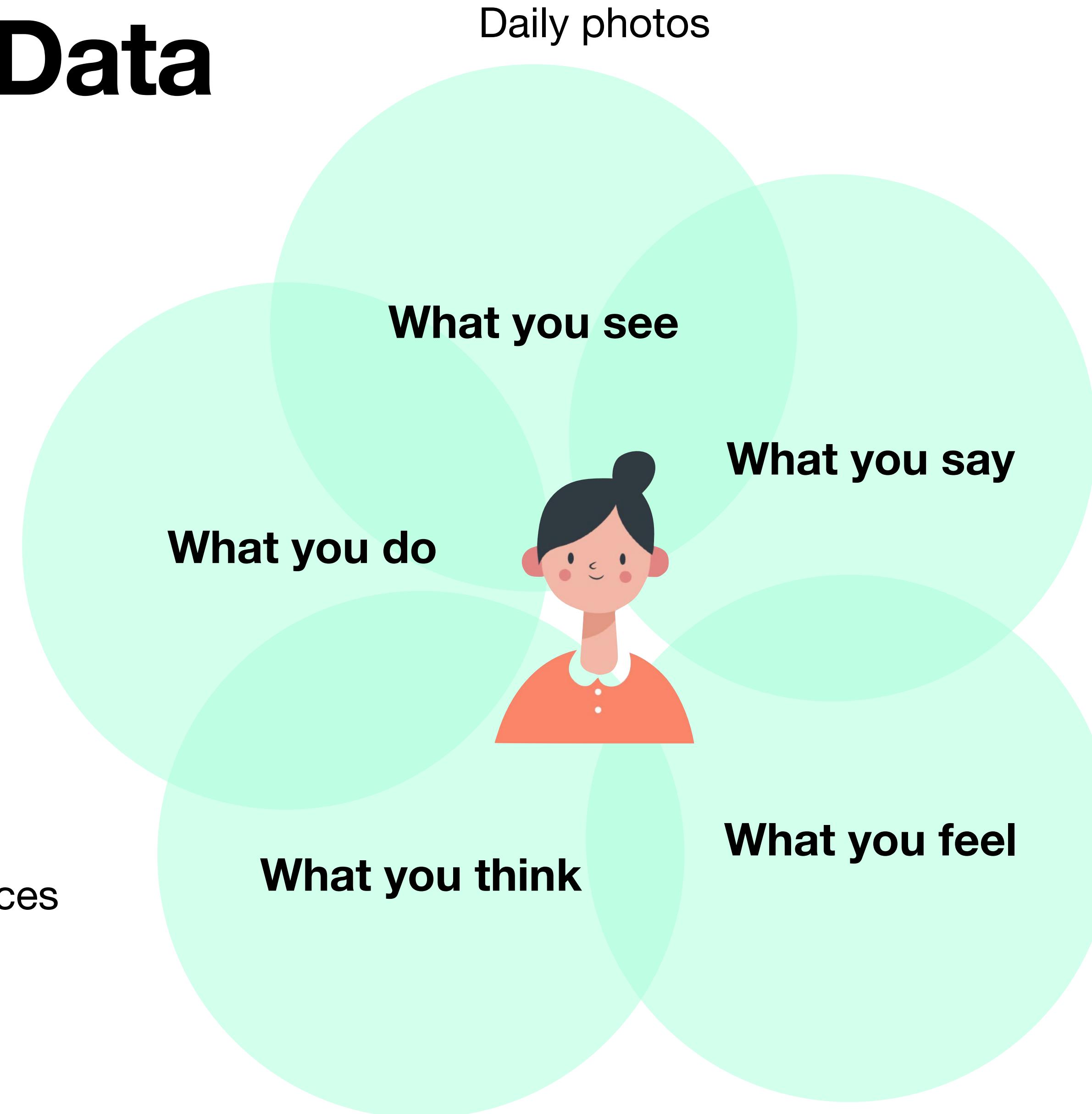
Daily photos

Todo list

Posts on social media

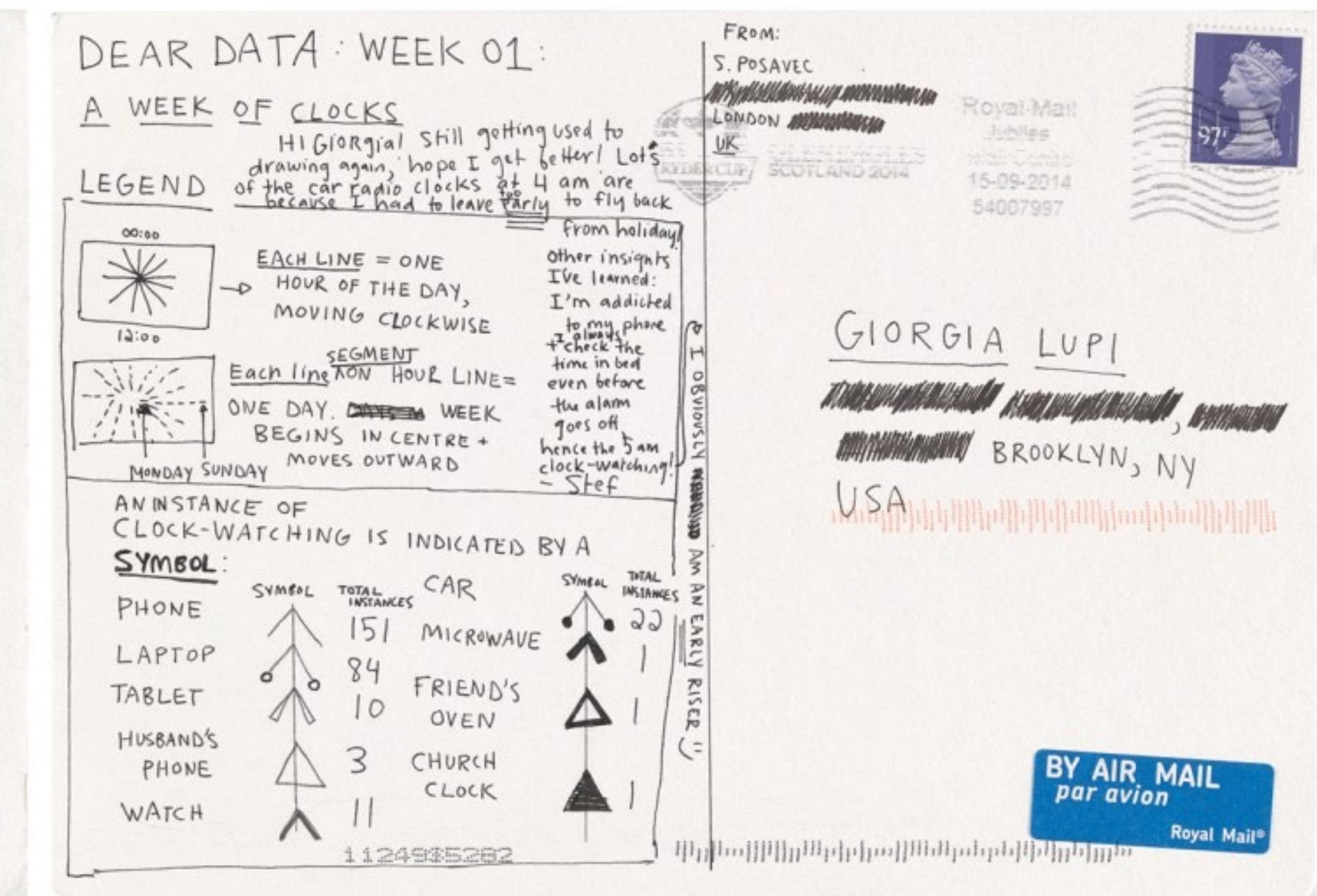
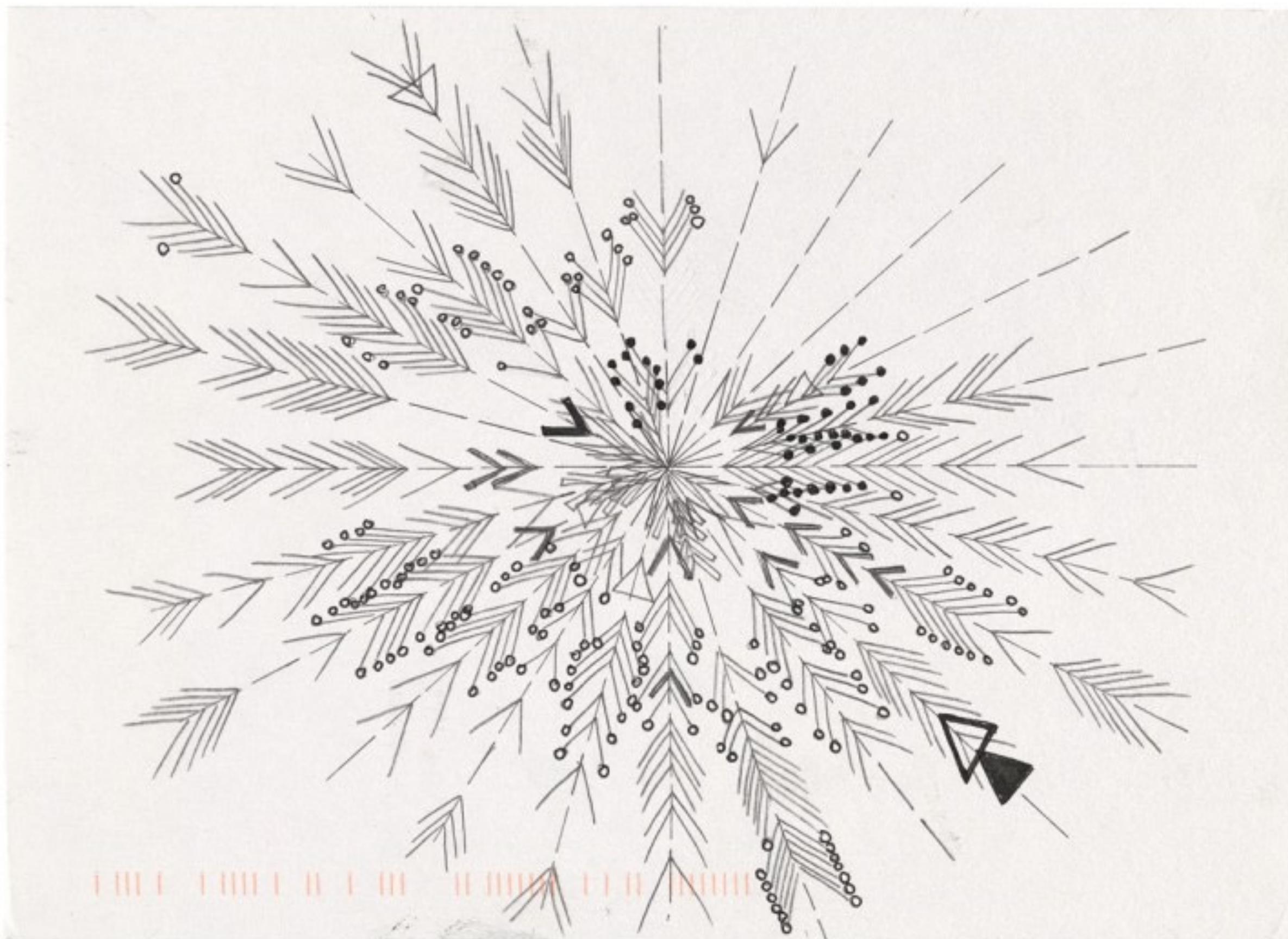
Emotion & Desires

Values & Beliefs

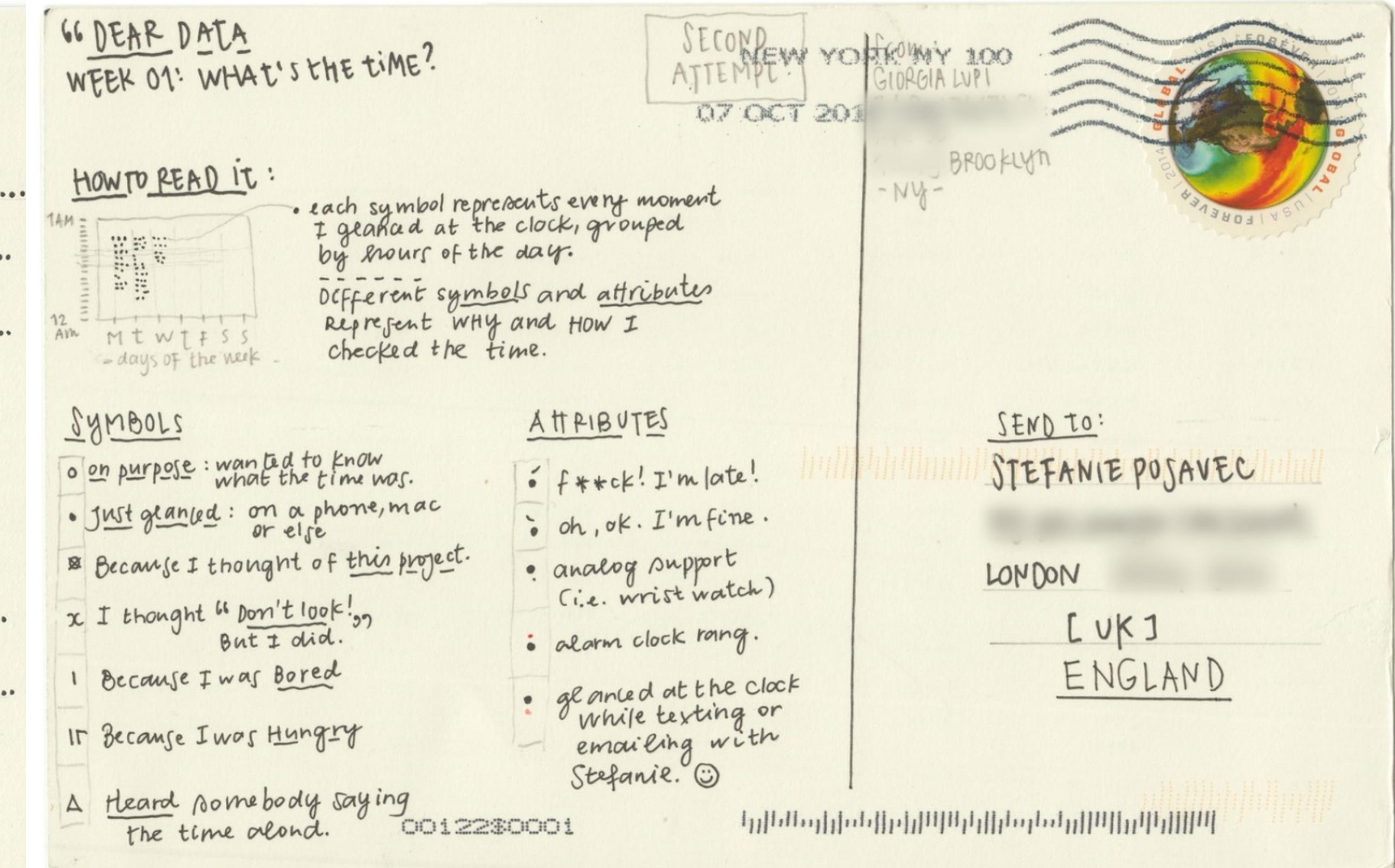
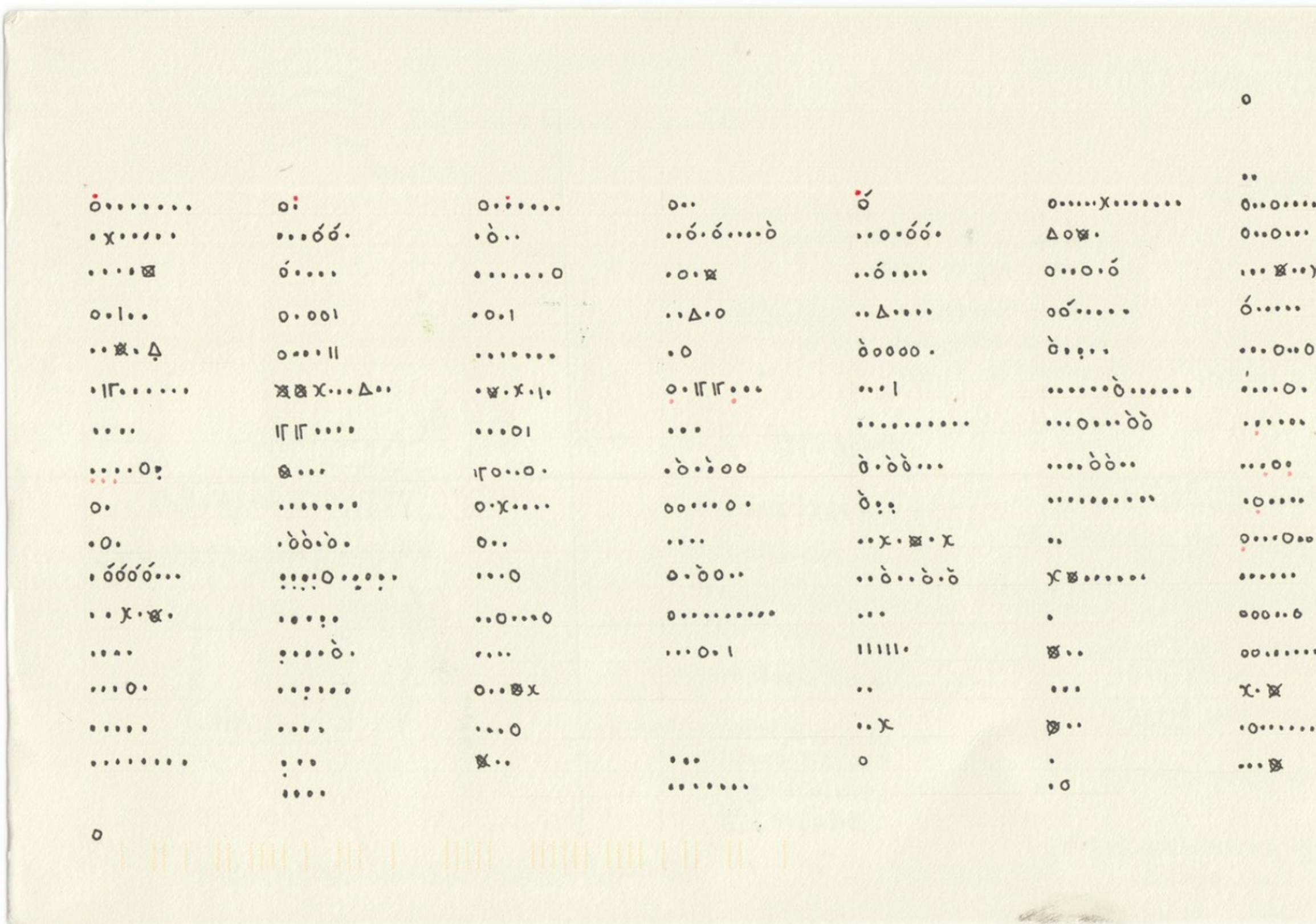


Dear Data

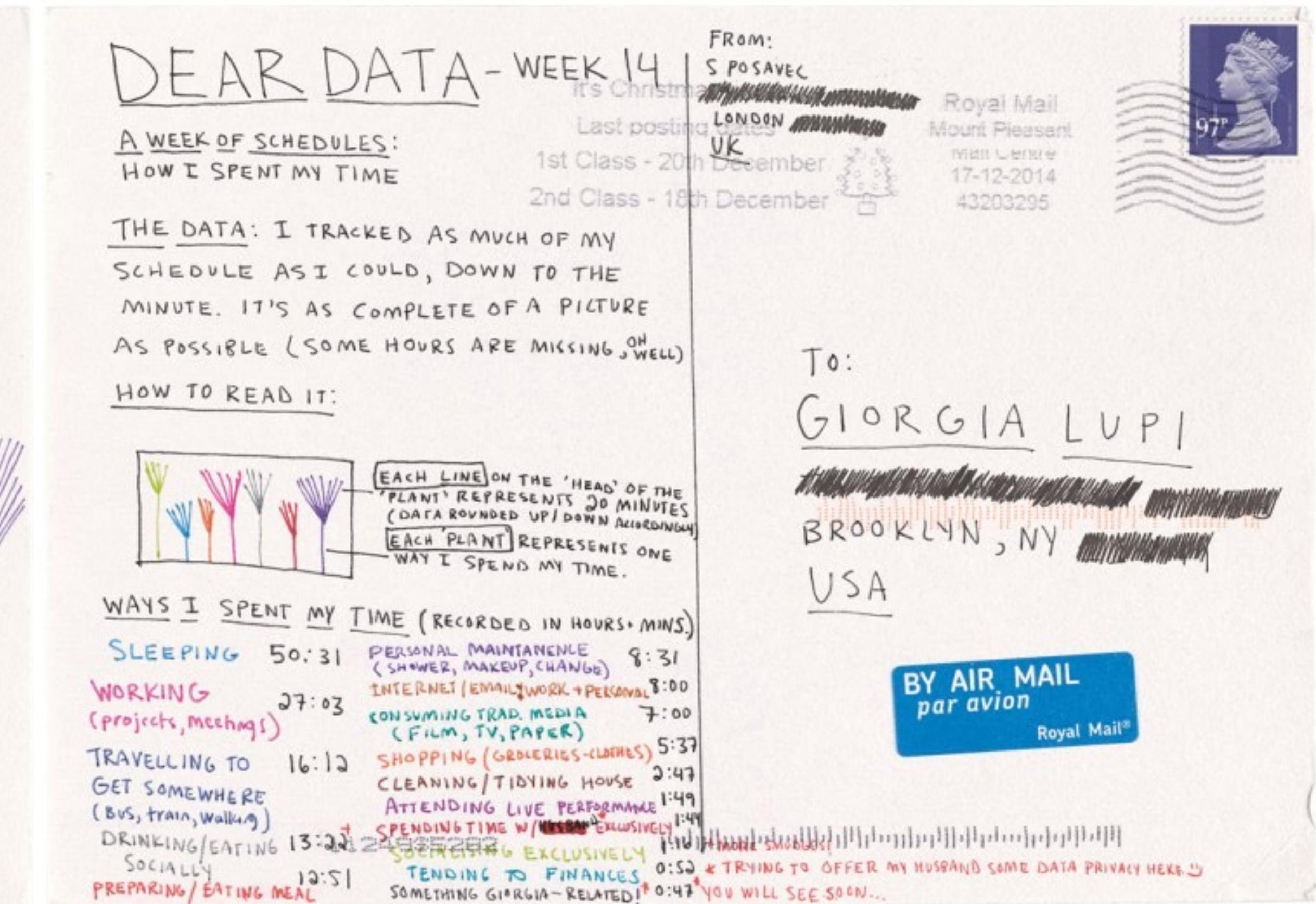
Week 1: A Week of Clocks (Stefanie)



Week 1: A Week of Clocks (Giorgia)



Week 14: A Week of Schedules (Stefanie)



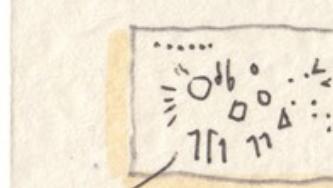
Week 14: A Week of Schedules (Giorgia)



66 DEAR DATA

WEEK 14: (ABSTRACT) PRODUCTIVITY!

HOW TO READ IT:



Every single element is a TASK performed during the week. the POSITION and ROTATION of the elements ARE ABSOLUTELY RANDOM and function of the aesthetic! 😊!

SYMBOLS = different tasks

- ✓ = line = Email sent (length = apx. email length)
- = skypecall done (dimension = apx. duration)
- △ = meetings! (dimens = duration)
- ▼ = solved a problem
- = planned something
- ↗ = delivered a talk
- ◎ = produced a document (→ ◎ → n. of edit sessions)
-][= came up with i ideas
- ◎ = reviewed a project
- || = tweeted s.thing
- = saved Adobe files
- X = bought something

- COLORS = with whom / to who
 - Stef! or Deardata related tasks
 - boyfriend (often work related)
 - partners at - - - - -
 - team at - - - - -
 - client / potential client
 - other (coworkers-friends)
- produced a design
- drawn / stitched useful things
- created a to do
- scanned a document
- ↑ helped some body
- ◎ administrative stuff
- home related things
- other

FROM:
TRIBORO NY
BKLYN-QNS/STA
16 DECEMBER 2013

11249 Brooklyn
NY - USA



SEND TO:

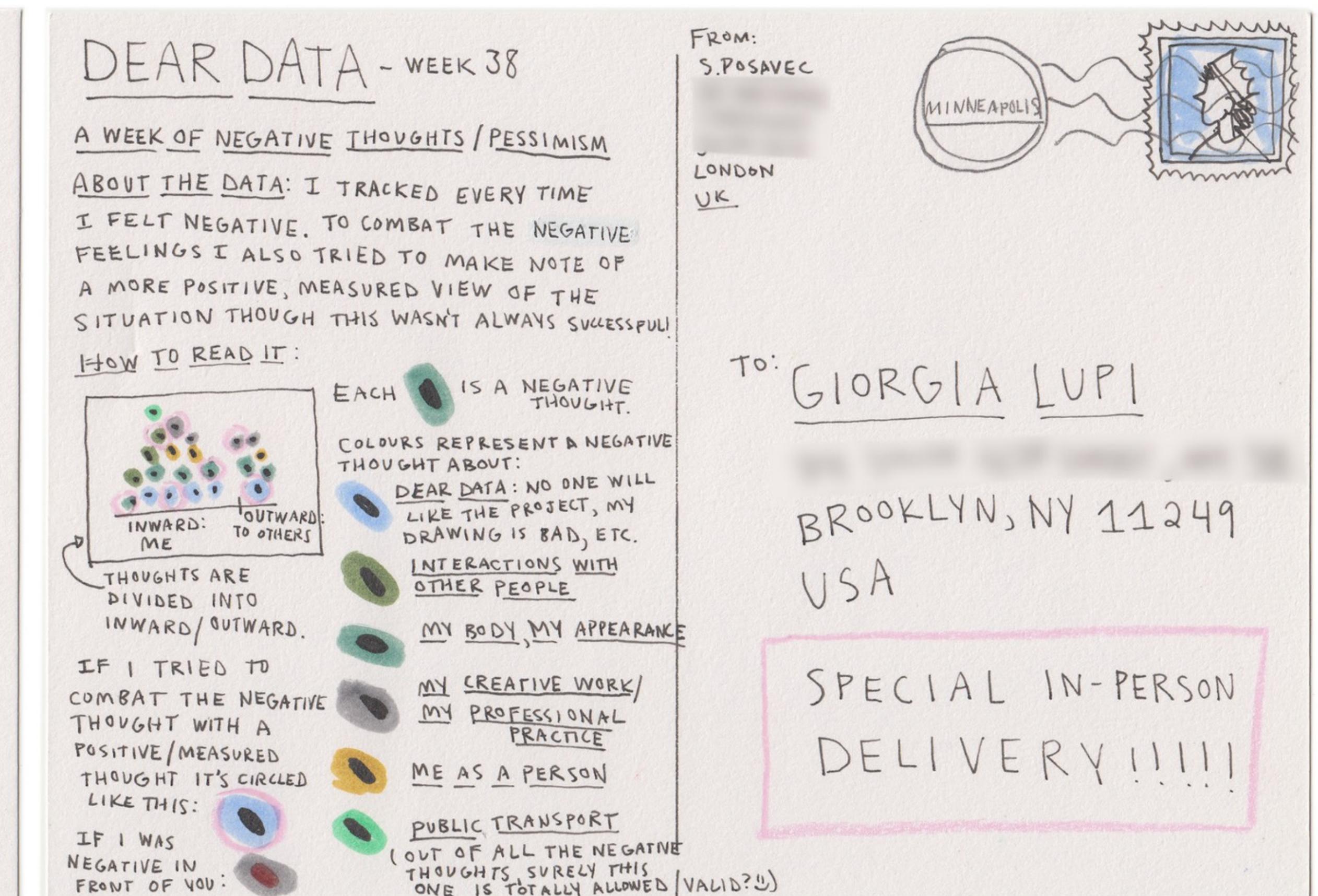
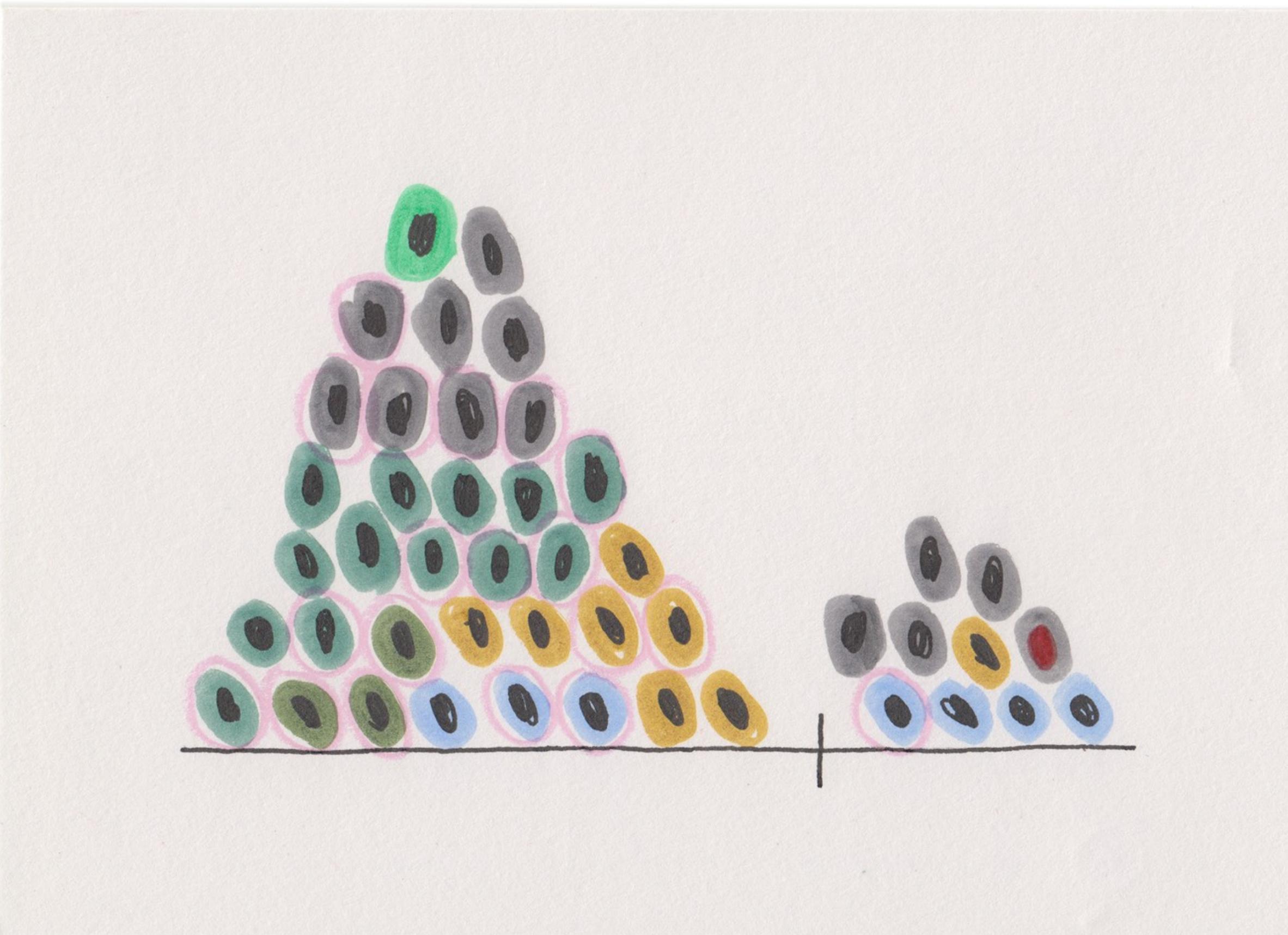
STEFANIE POSAVEC

LONDON

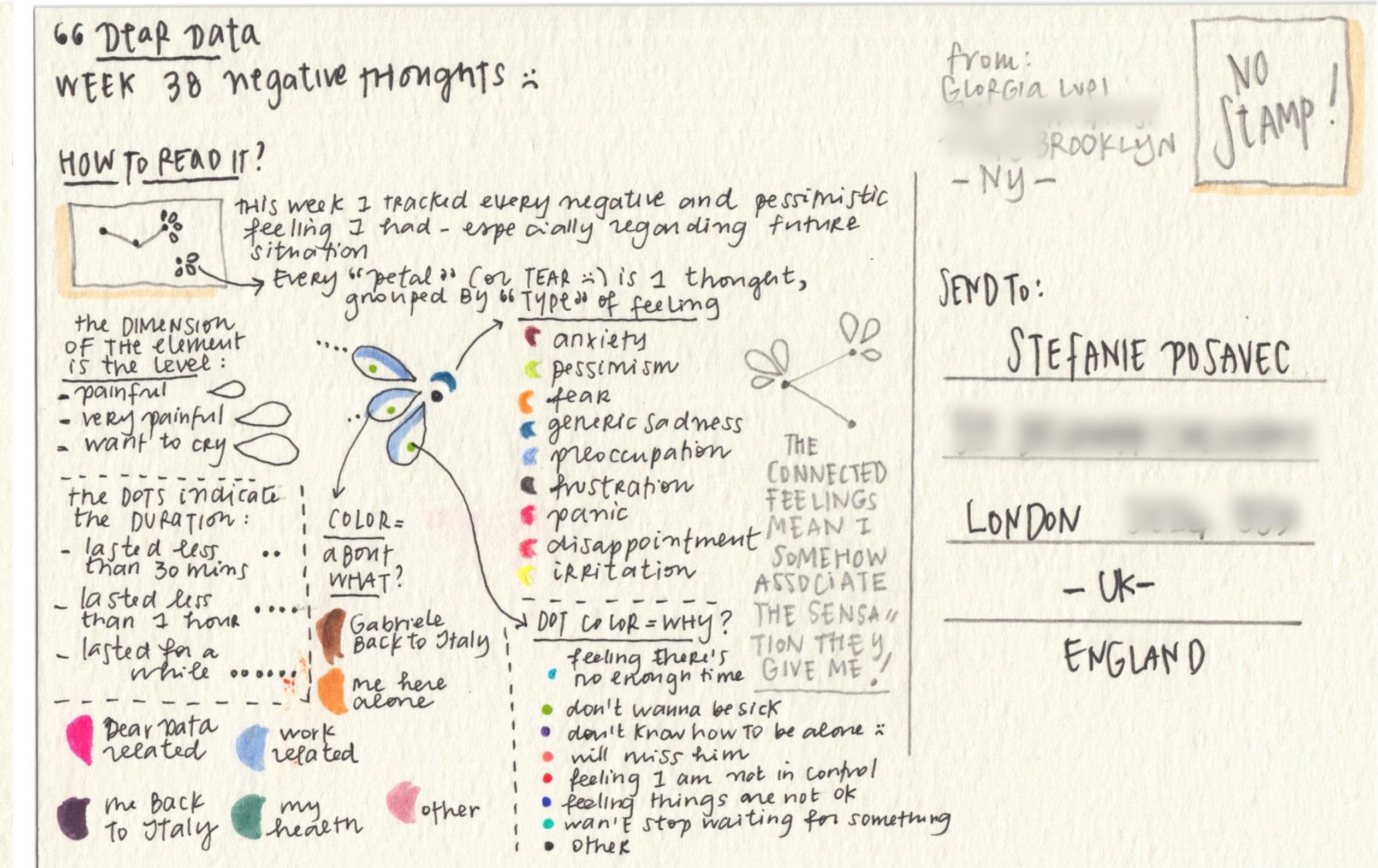
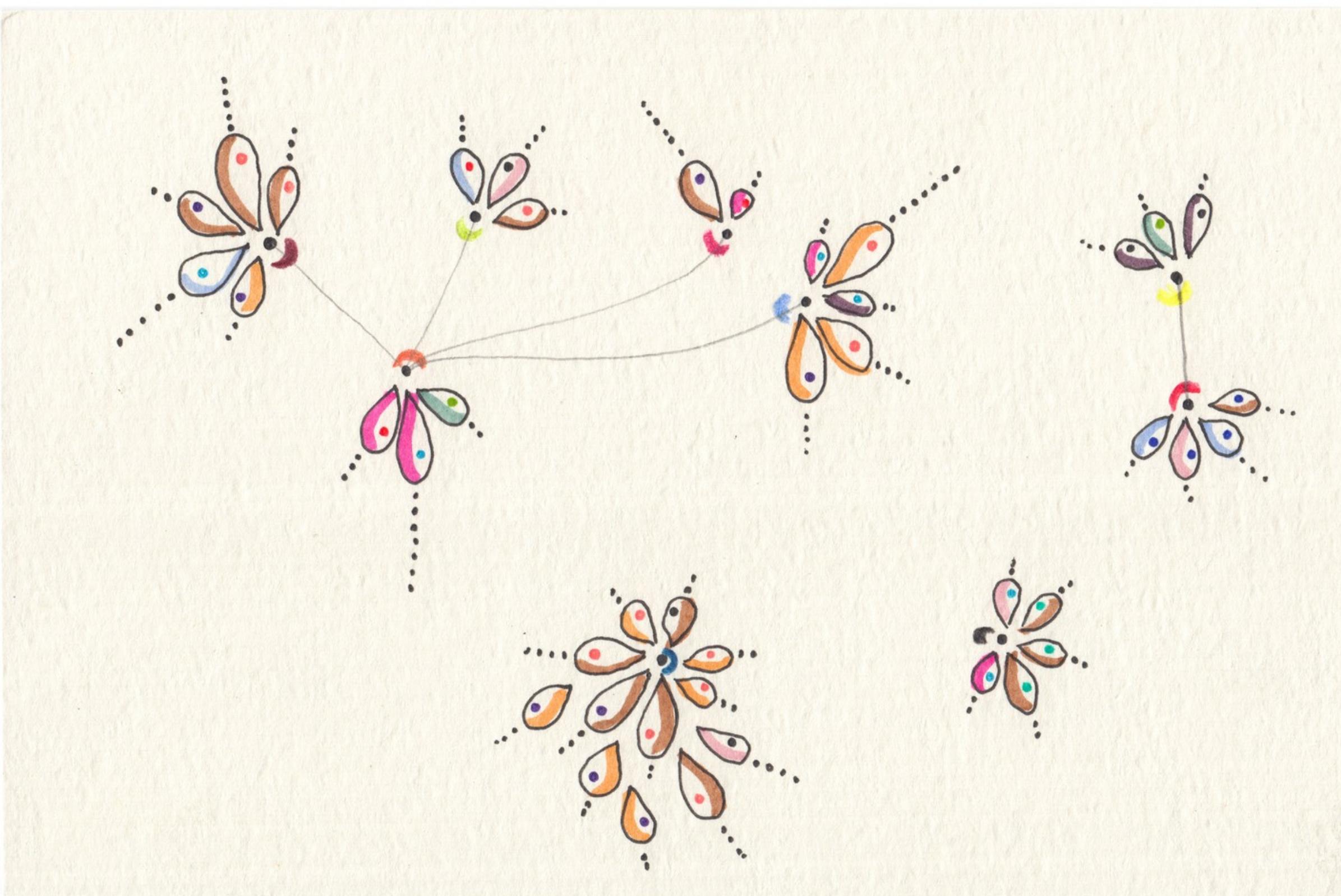
- UK -

ENGLAND

Week 38: A Week of Negative Thoughts (Stefanie)



Week 38: A Week of Negative Thoughts (Giorgia)



“Dear Data” Exercise

- Step 1. Choose a topic: emotions, daily activity, todo-lists, complaints, drinks, etc
- Step 2. Reflect on your experiences last week and record the data for the chosen topic
 - Topics: feelings, activities, time, duration, thoughts, observations, etc.
- Step 3. Define your visual vocabulary
- Step 4. Draw your personal data!!
- Step 5. Share it to your group members

Topics

- to-do lists
- daily activities
- sounds
- books
- our cities
- transportation
- envy
- desires
- emotions
- complaints
- compliments
- thank-yous
- apologies
- food preferences
- drinks
- coffees
- doors/spaces/kitchen
- our time alone
- (...)

“Dear Data” Exercise (30 mins)

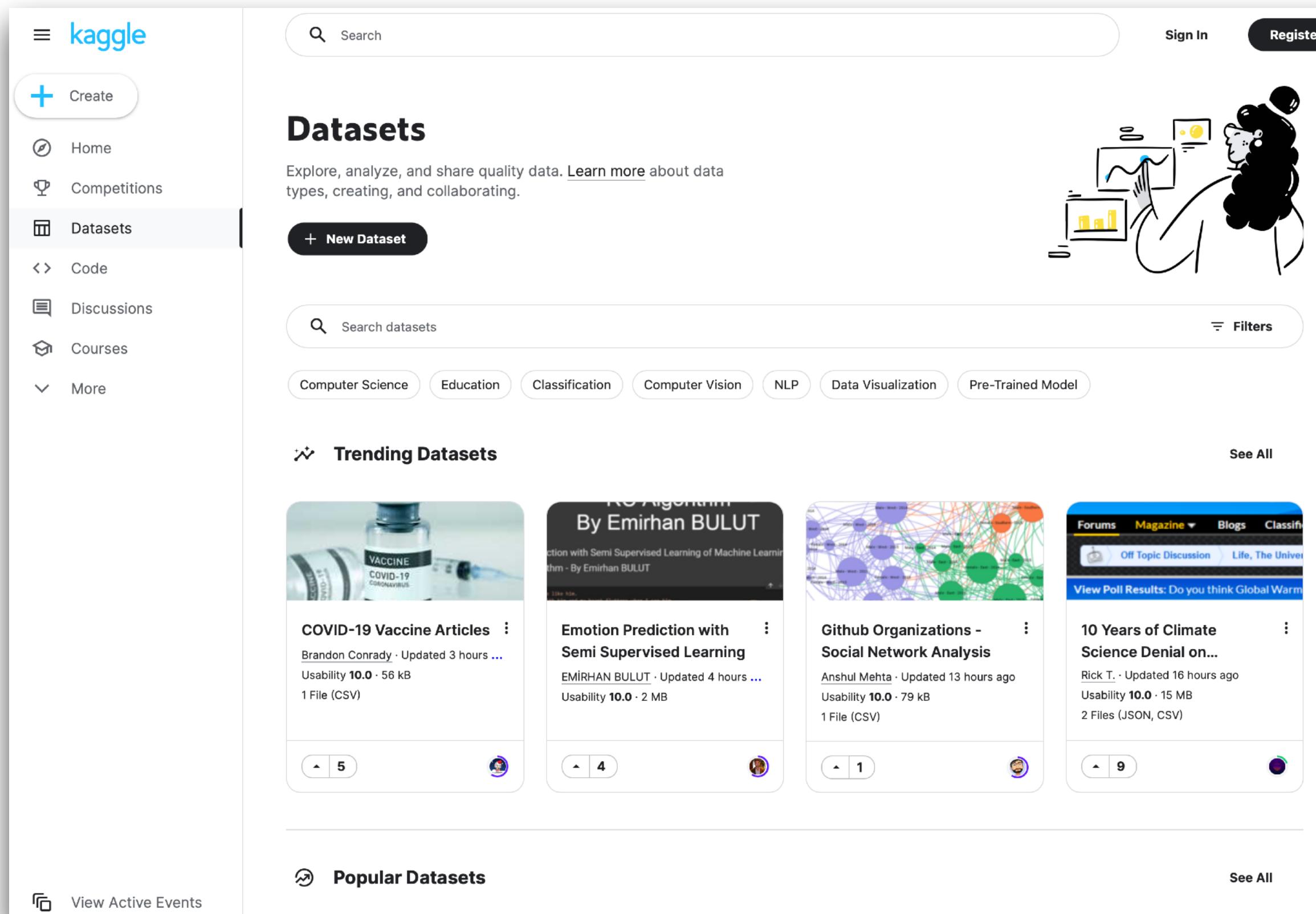
- Step 1. Choose a topic
- Step 2. Reflect on your experiences last week and record the data (e.g., feelings, activities, time, duration, thoughts, observations, etc) for the chosen topic
- Step 3. Define your visual vocabulary
- Step 4. Draw your personal data!!
- Step 5. Share it to your group members

Topics

- to-do lists
- daily activities
- sounds
- books
- our cities
- transportation
- envy
- desires
- emotions
- complaints
- compliments
- thank-yous
- apologies
- food preferences
- drinks
- coffees
- doors/spaces/kitchen
- our time alone
- (...)

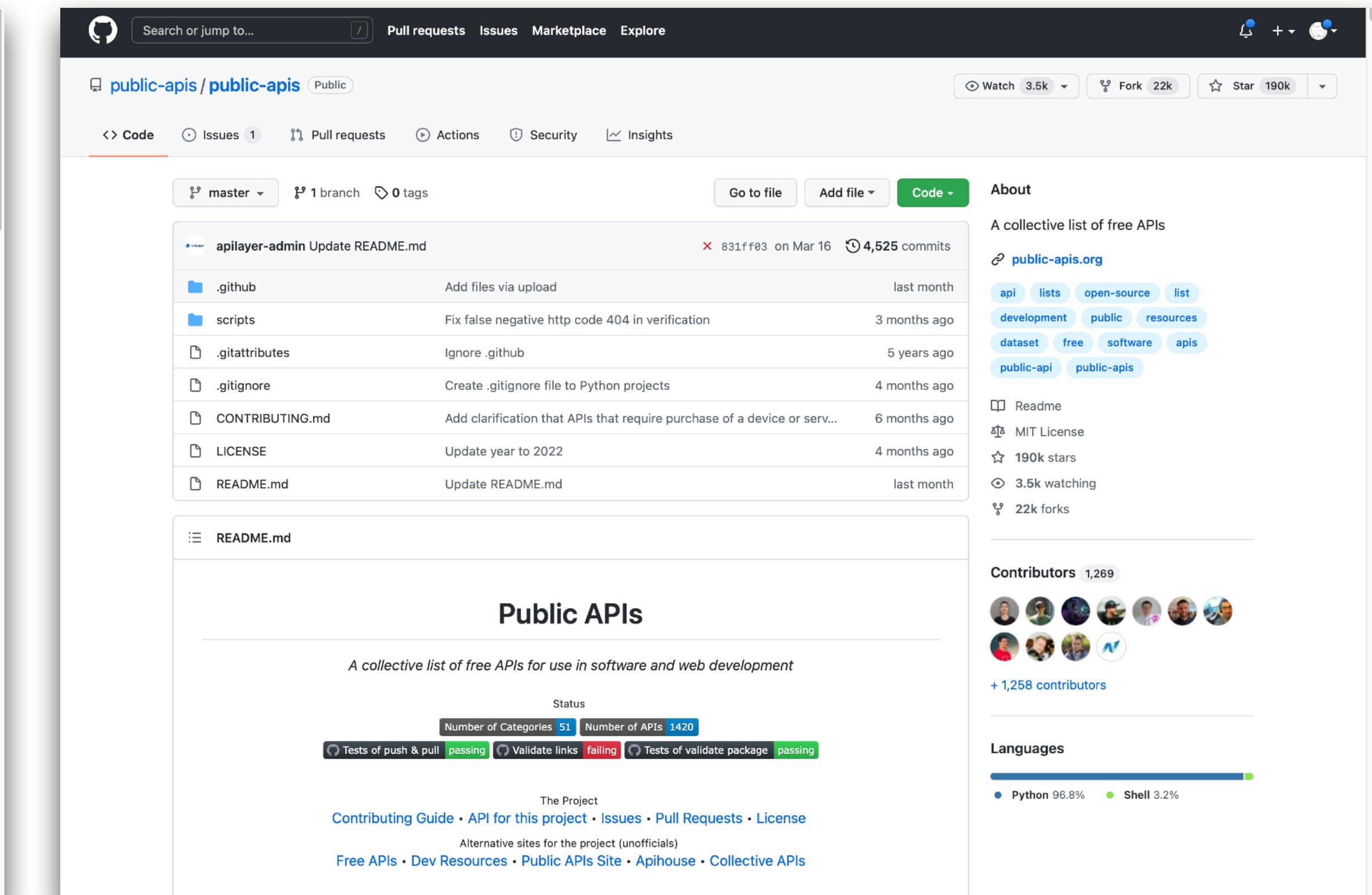
Inspired by Big Data

use existing datasets or public api to get big data



The screenshot shows the Kaggle Datasets homepage. On the left, there's a sidebar with navigation links: Home, Competitions, Datasets (which is selected), Code, Discussions, Courses, and More. The main area has a search bar at the top. Below it, a section titled "Datasets" encourages users to explore, analyze, and share quality data. It features a "New Dataset" button and a "Trending Datasets" section with five cards. The cards include: "COVID-19 Vaccine Articles" by Brandon Conrady, "Kaggle Algorithm" by Emirhan BULUT, "Emotion Prediction with Semi Supervised Learning" by EMIRHAN BULUT, "Github Organizations - Social Network Analysis" by Anshul Mehta, and "10 Years of Climate Science Denial on..." by Rick T. The footer includes a "Popular Datasets" section and a "View Active Events" link.

<https://www.kaggle.com/datasets>



The screenshot shows the GitHub repository page for "public-apis/public-apis". The repository has 3.5k watchers, 22k forks, and 190k stars. The code tab is selected, showing a list of files: README.md, .github, scripts, .gitattributes, .gitignore, CONTRIBUTING.md, LICENSE, and README.md. The README.md file is expanded, showing its content. Below the code, there's a "Public APIs" section with a subtitle "A collective list of free APIs for use in software and web development". It includes a status summary: Number of Categories 51, Number of APIs 1420, Tests of push & pull passing, Validate links failing, and Tests of validate package passing. The footer contains links to the contributing guide, API, issues, pull requests, license, and other related sites.

<https://github.com/public-apis/public-apis>

Example Datasets

Weather Data

The screenshot shows the MetaWeather API documentation. At the top, there's a navigation bar with links for Home, Language, Map, API, and About. Below it, a search bar is followed by a section titled "API". A note says "MetaWeather provides an API that delivers JSON over HTTPS for access to our data. Drop me an email if you're going to make more than maybe a request a minute to this. We also ask that you link back to [MetaWeather.com](#) where appropriate, in a sensible way that's useful to the user." Under "API", there's a "Weather States" table with columns for Name, Abbreviation, and Icon. The table lists various weather conditions like Snow, Sleet, Hail, Thunderstorm, Heavy Rain, Light Rain, Showers, Heavy Cloud, Light Cloud, and Clear, each with its corresponding icon.

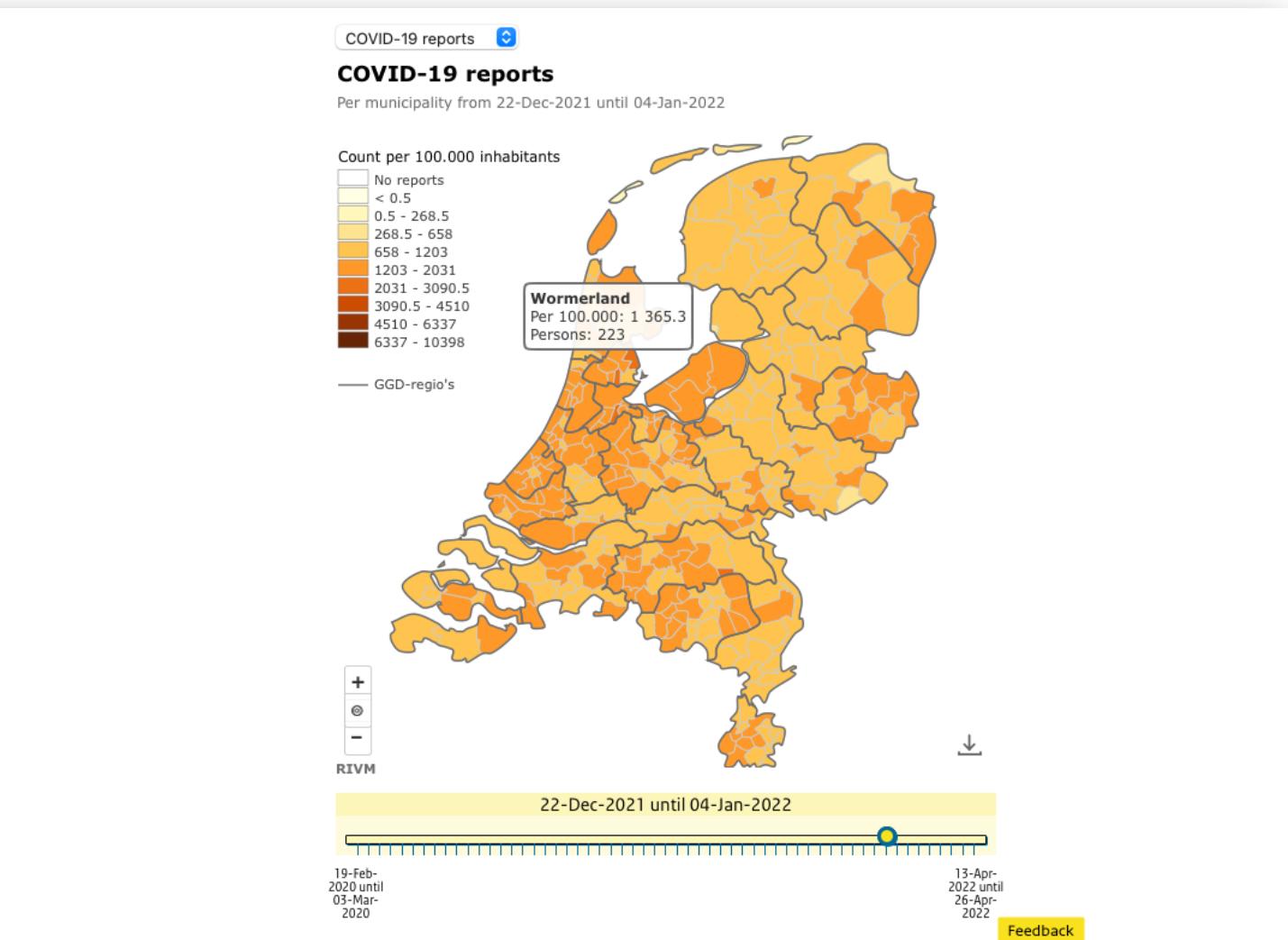
<https://www.metaweather.com/api/>

Air Quality Data

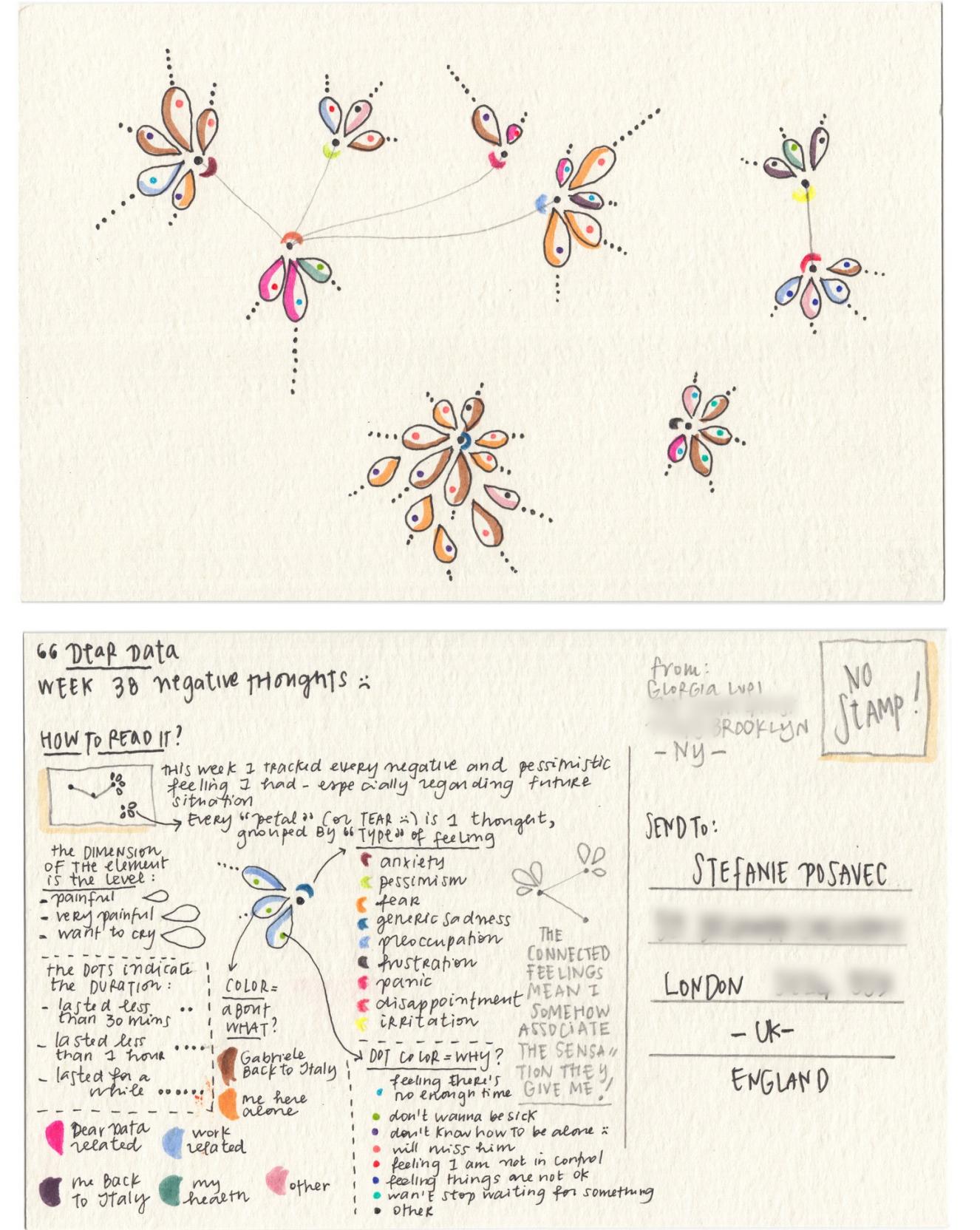
The screenshot shows the OpenAQ API documentation. It features a header with "OpenAQ 2.0.0 OAS3" and a link to "/openapi.json". Below the header, there are two sections: "default" and "v2". The "default" section contains endpoints for "/ping" (method: GET, response: Pong) and "/favicon.ico" (method: GET, response: Favico). The "v2" section contains several endpoints under sub-sections: "measurements", "averages", "locations", "latest", and "cities". Each endpoint is listed with its method (e.g., GET), path, and description.

<https://docs.openaq.org>

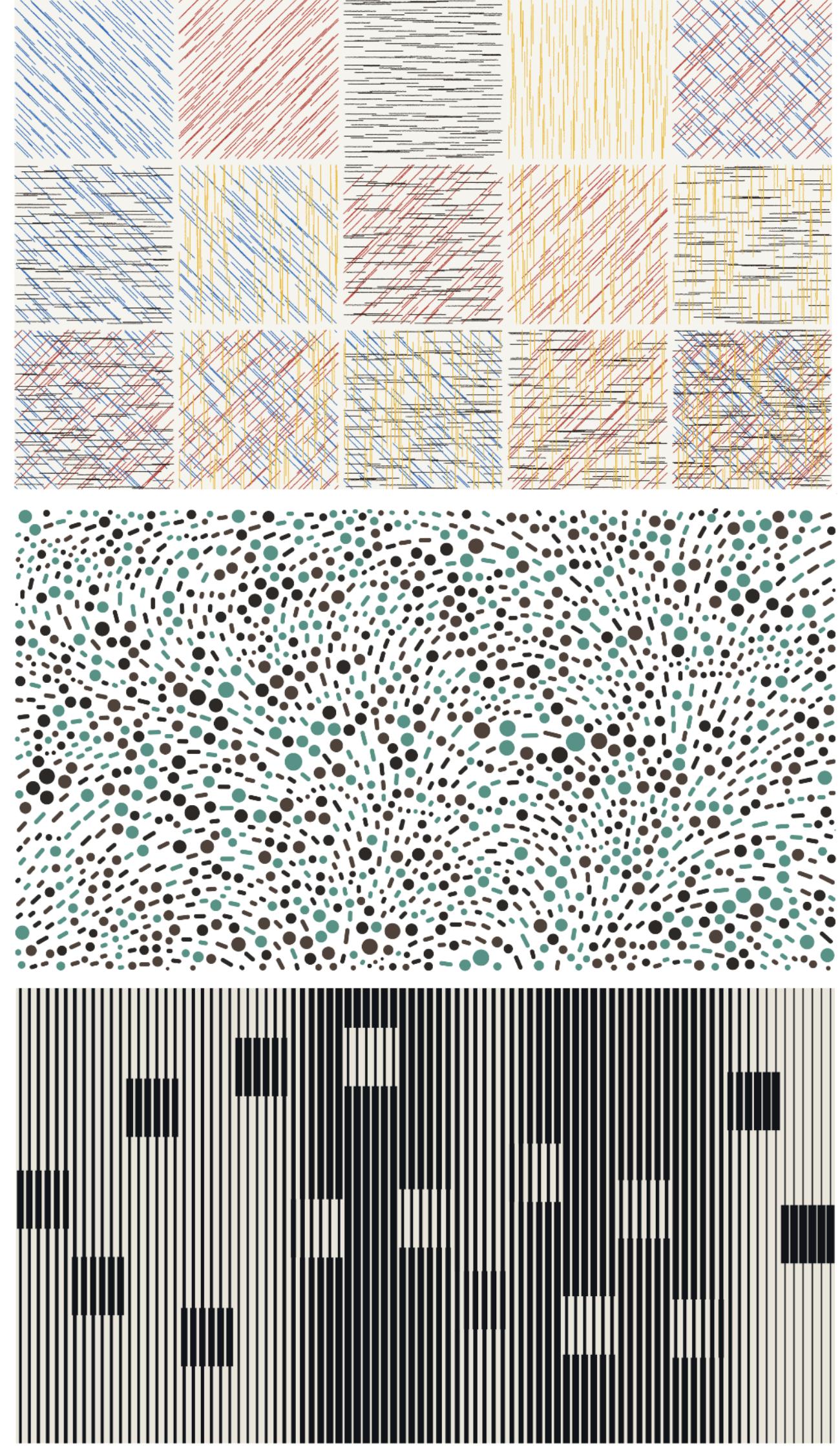
COVID-19 Data



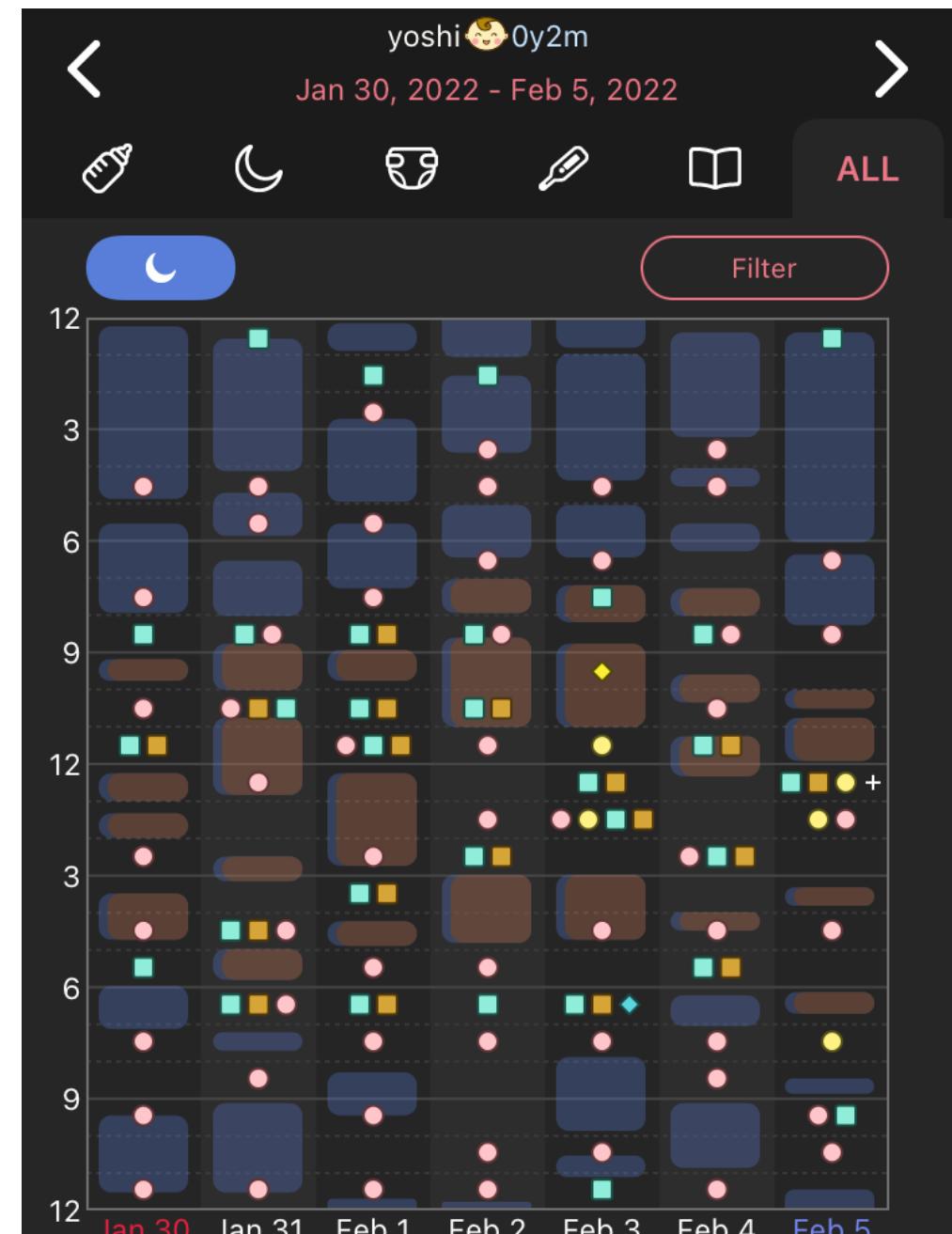
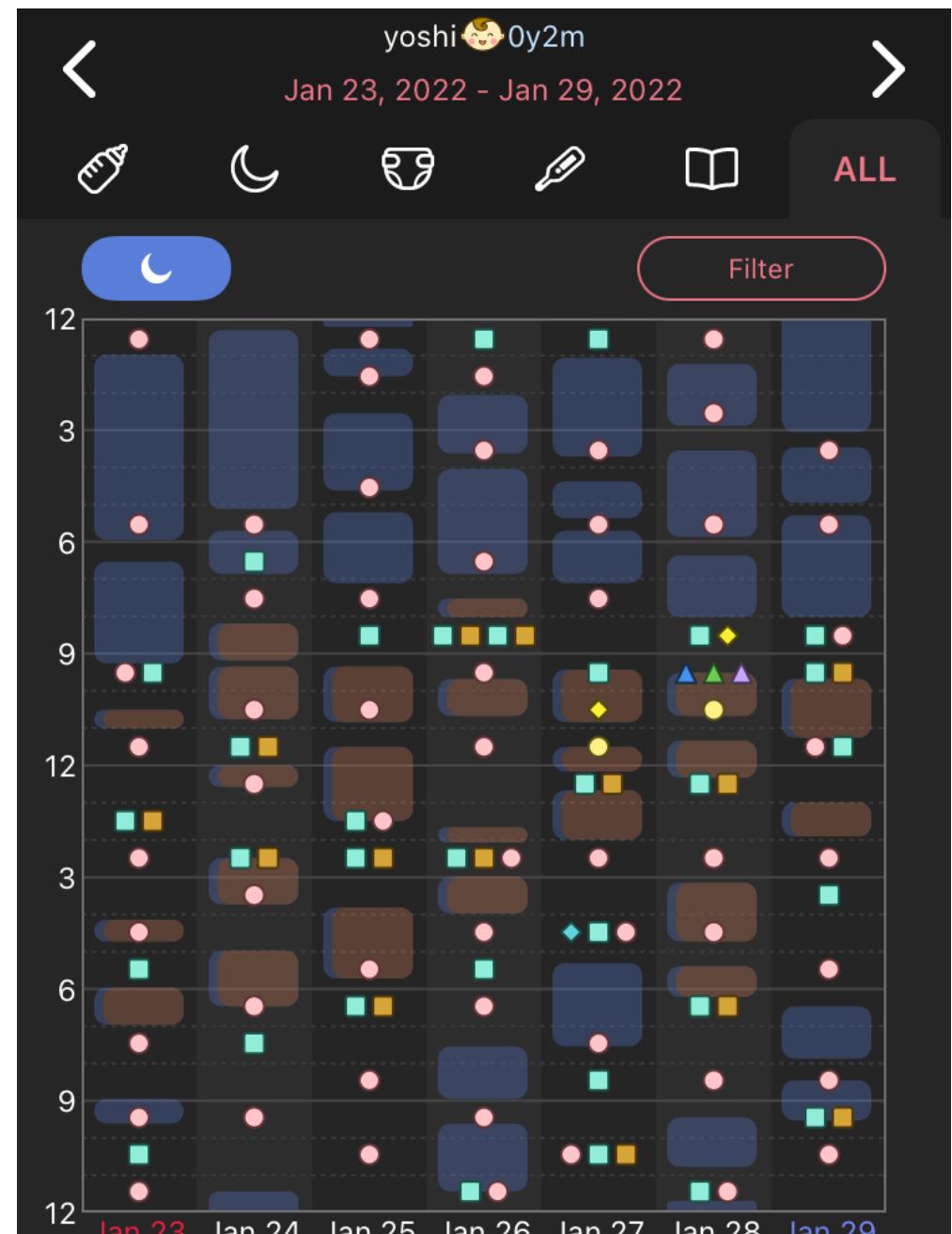
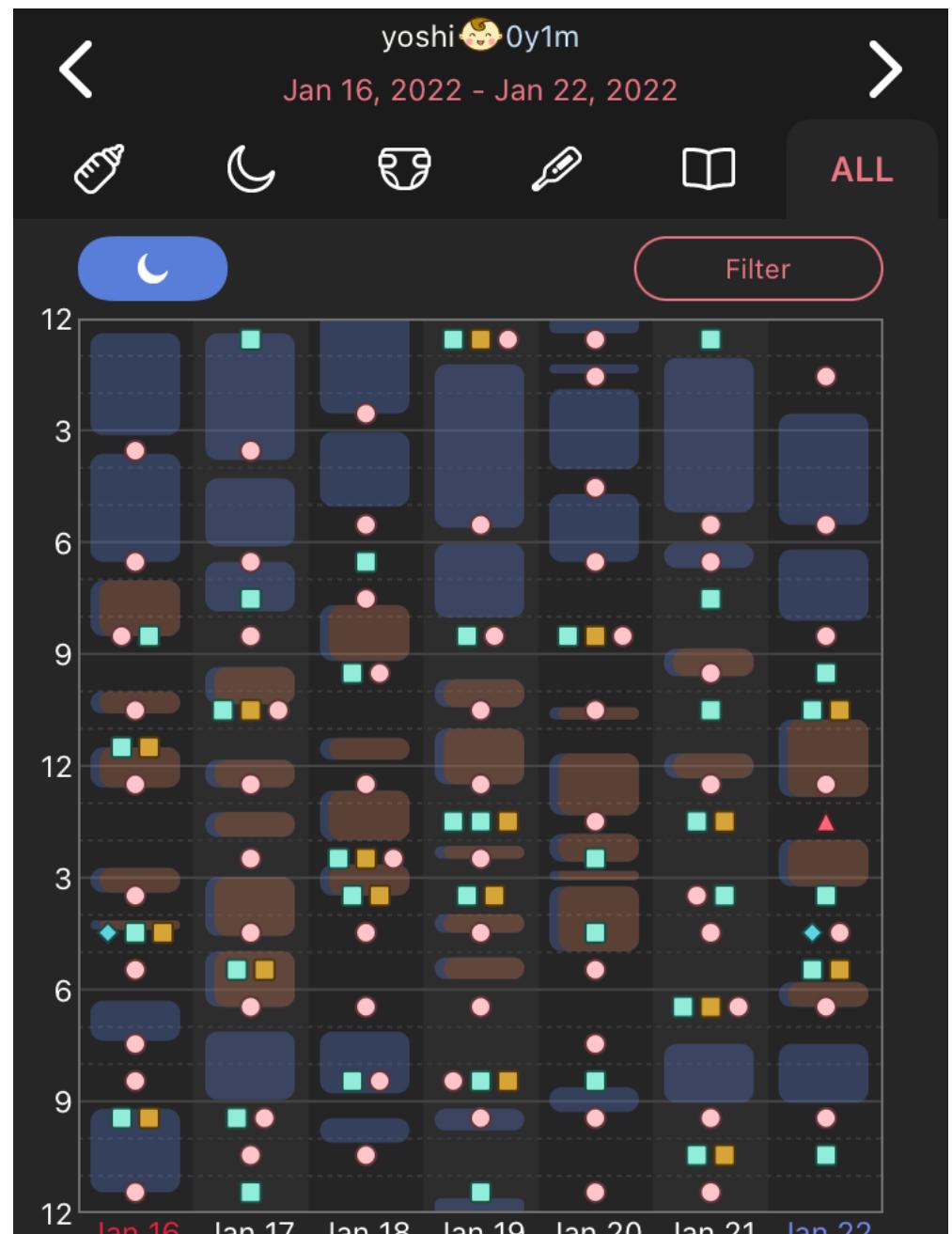
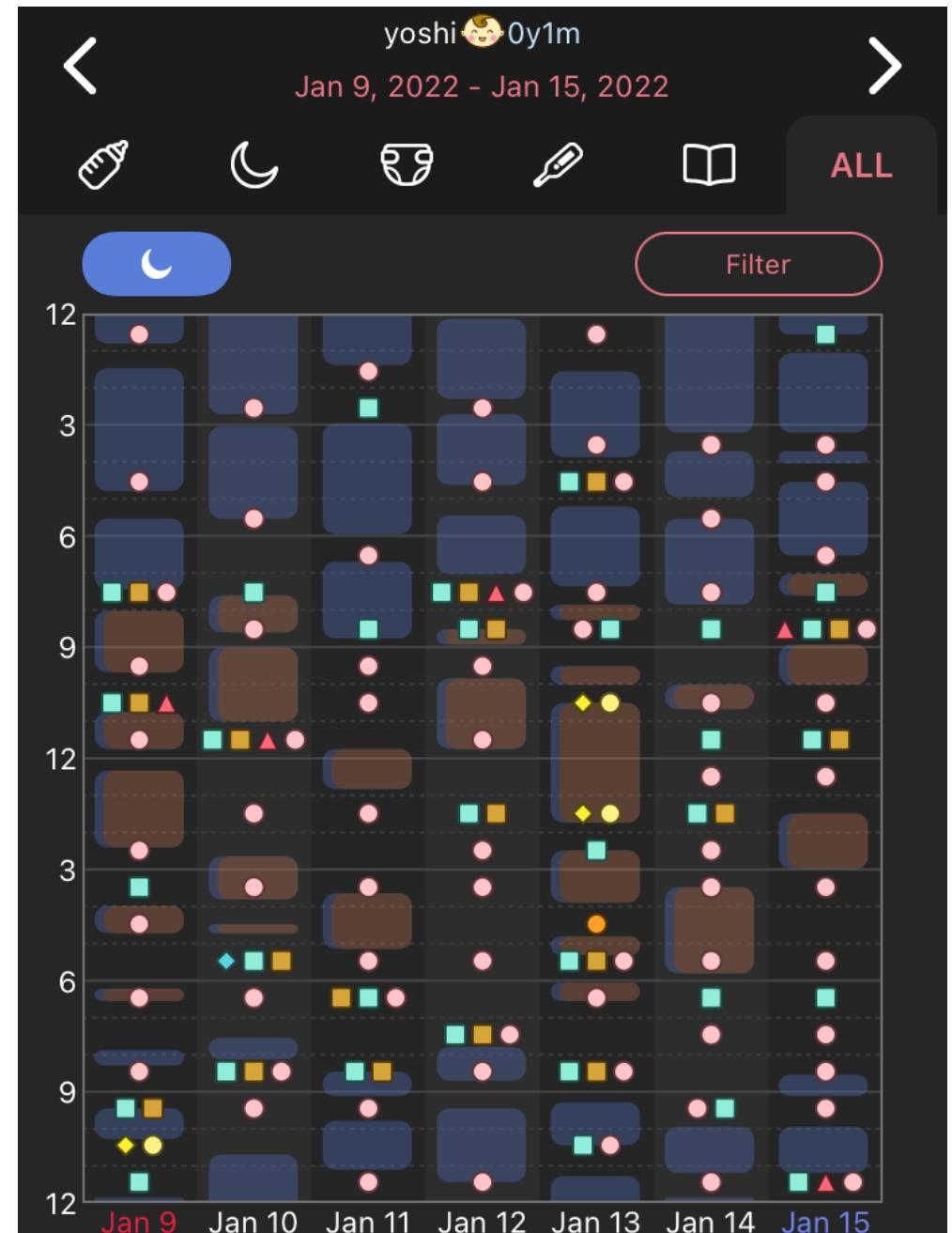
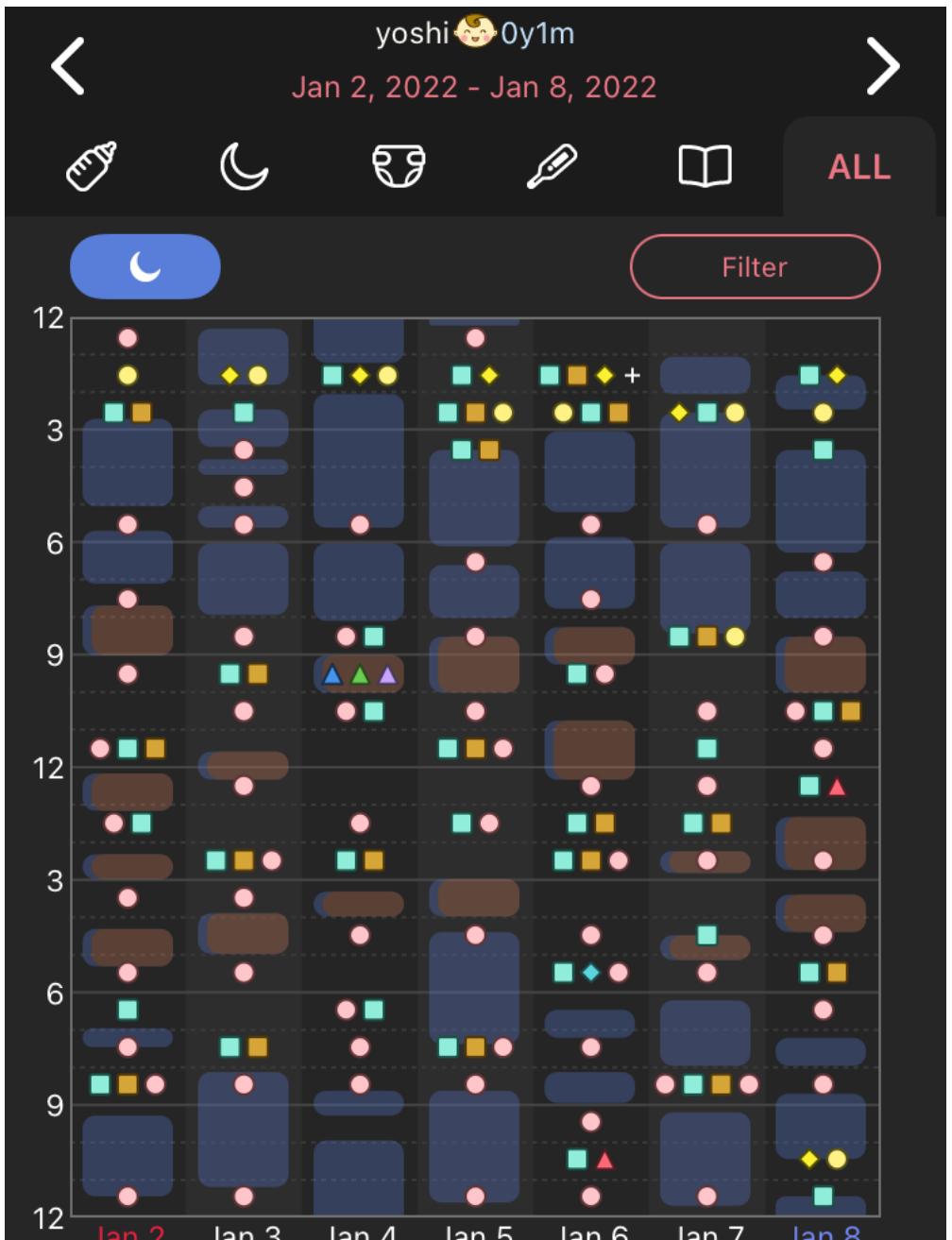
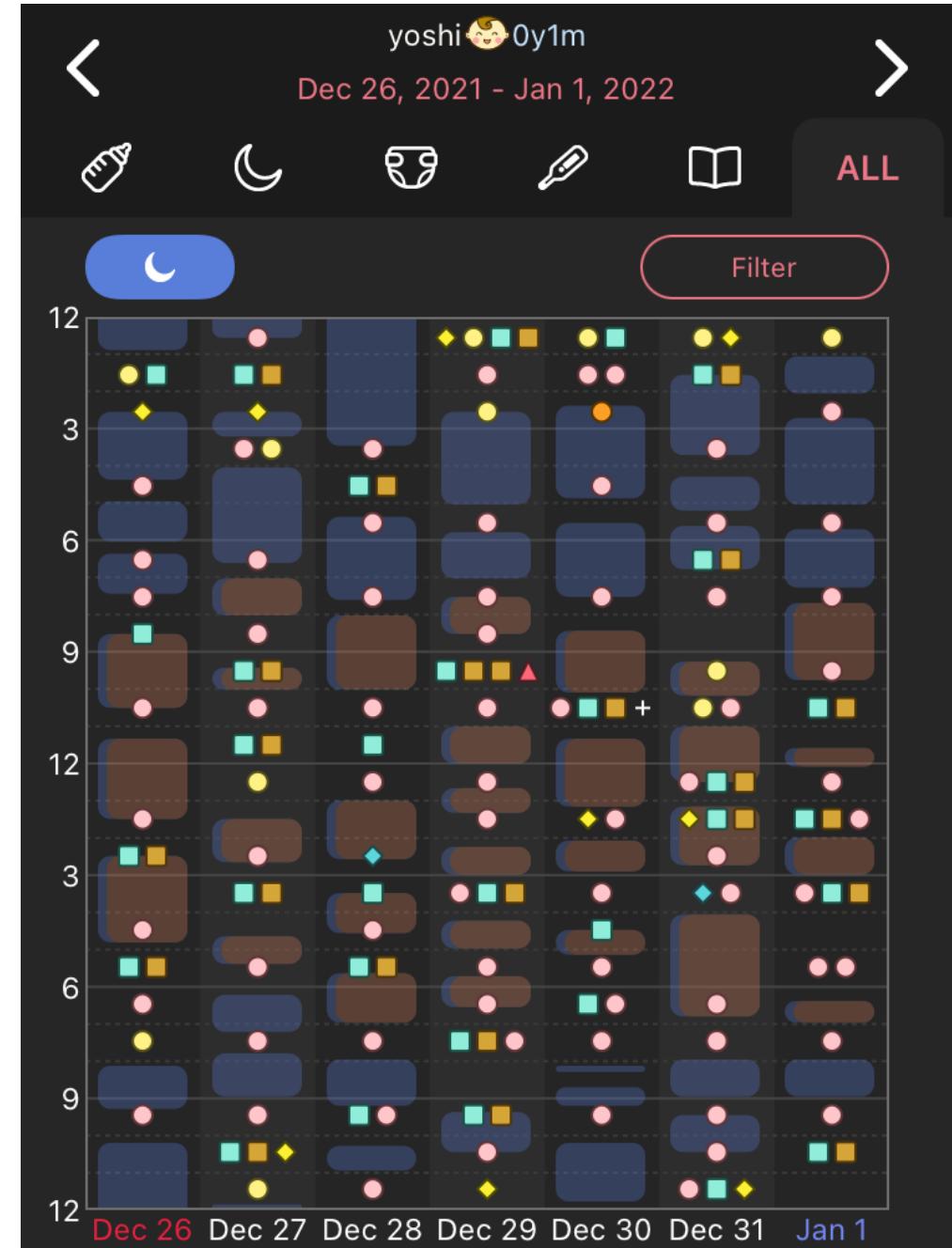
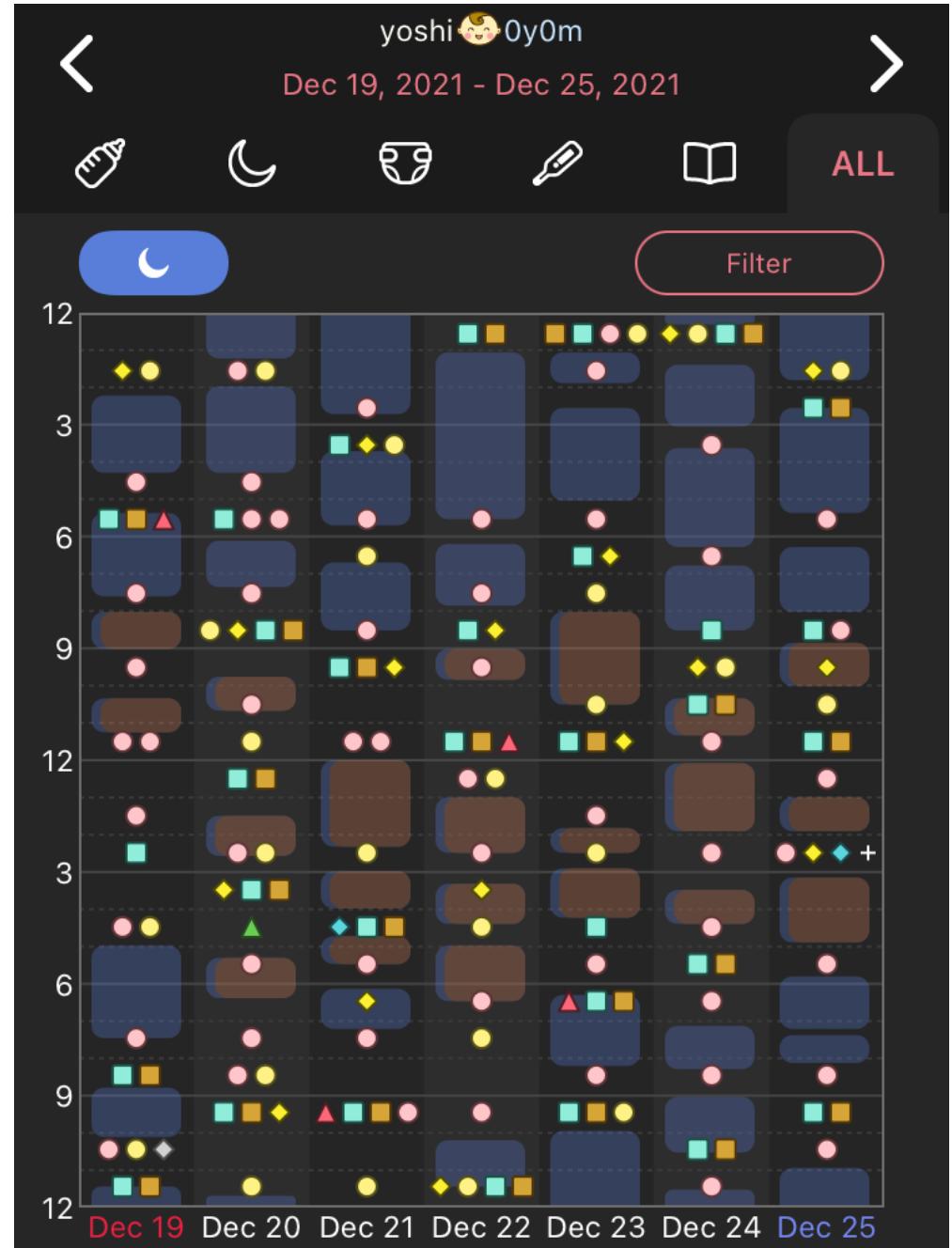
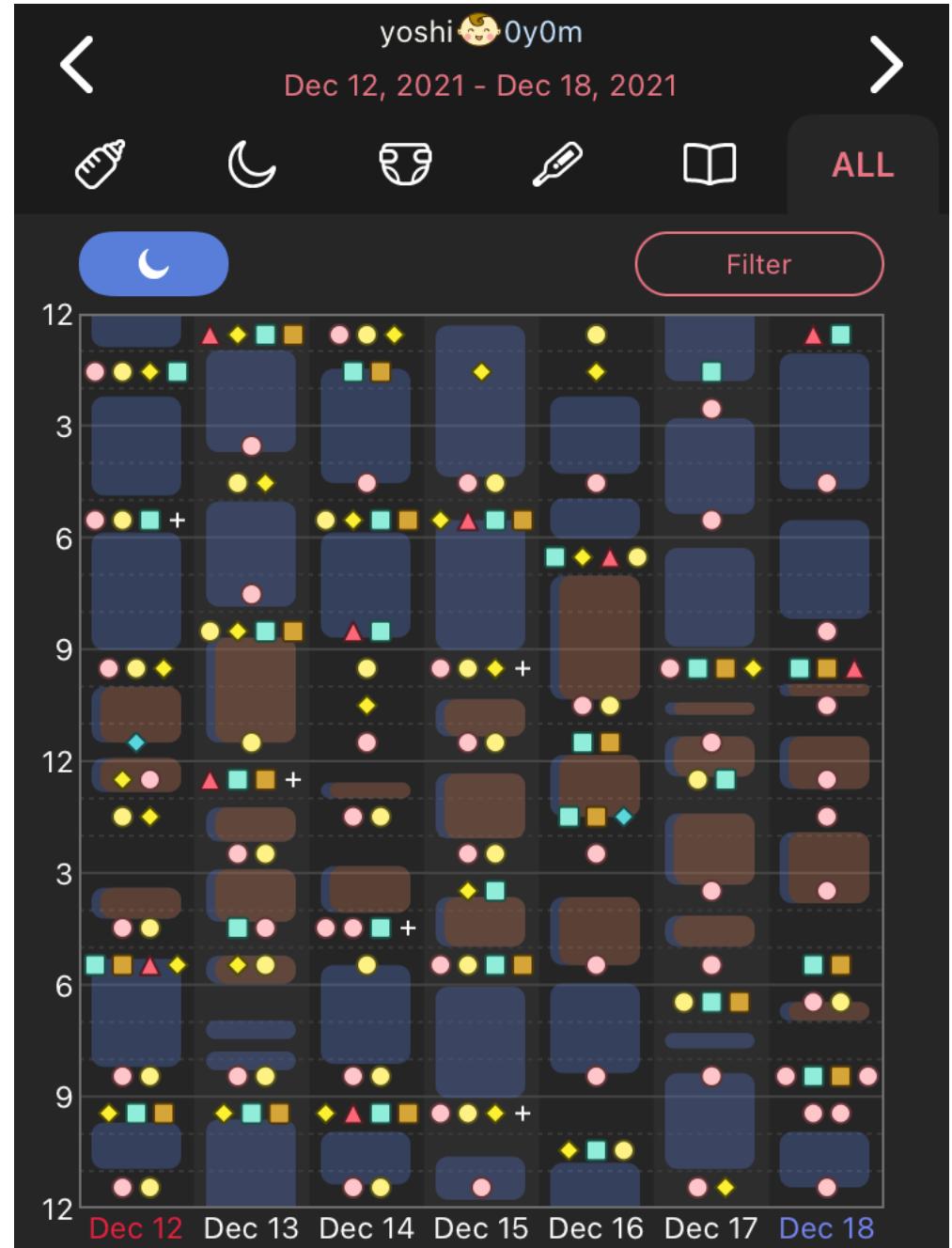
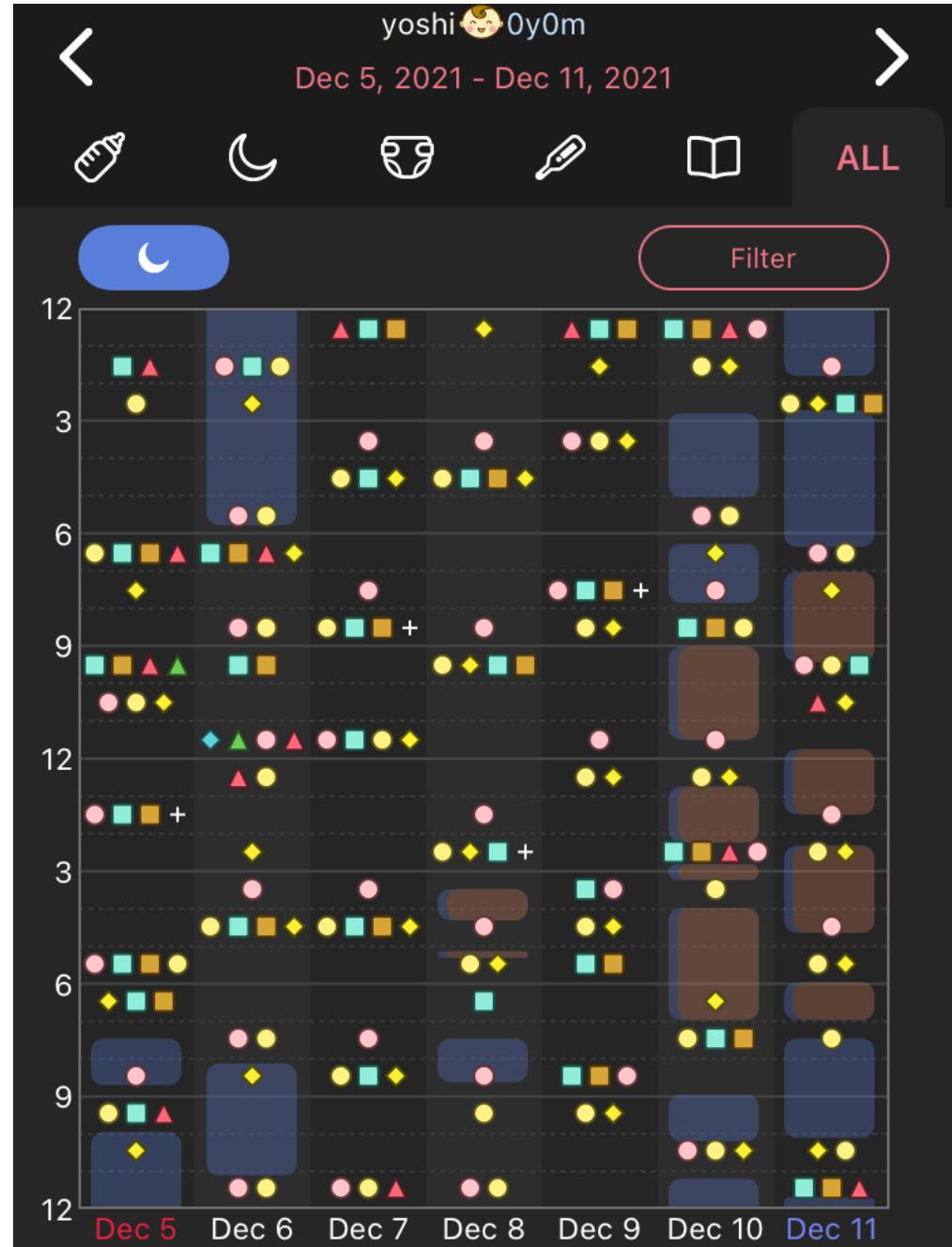
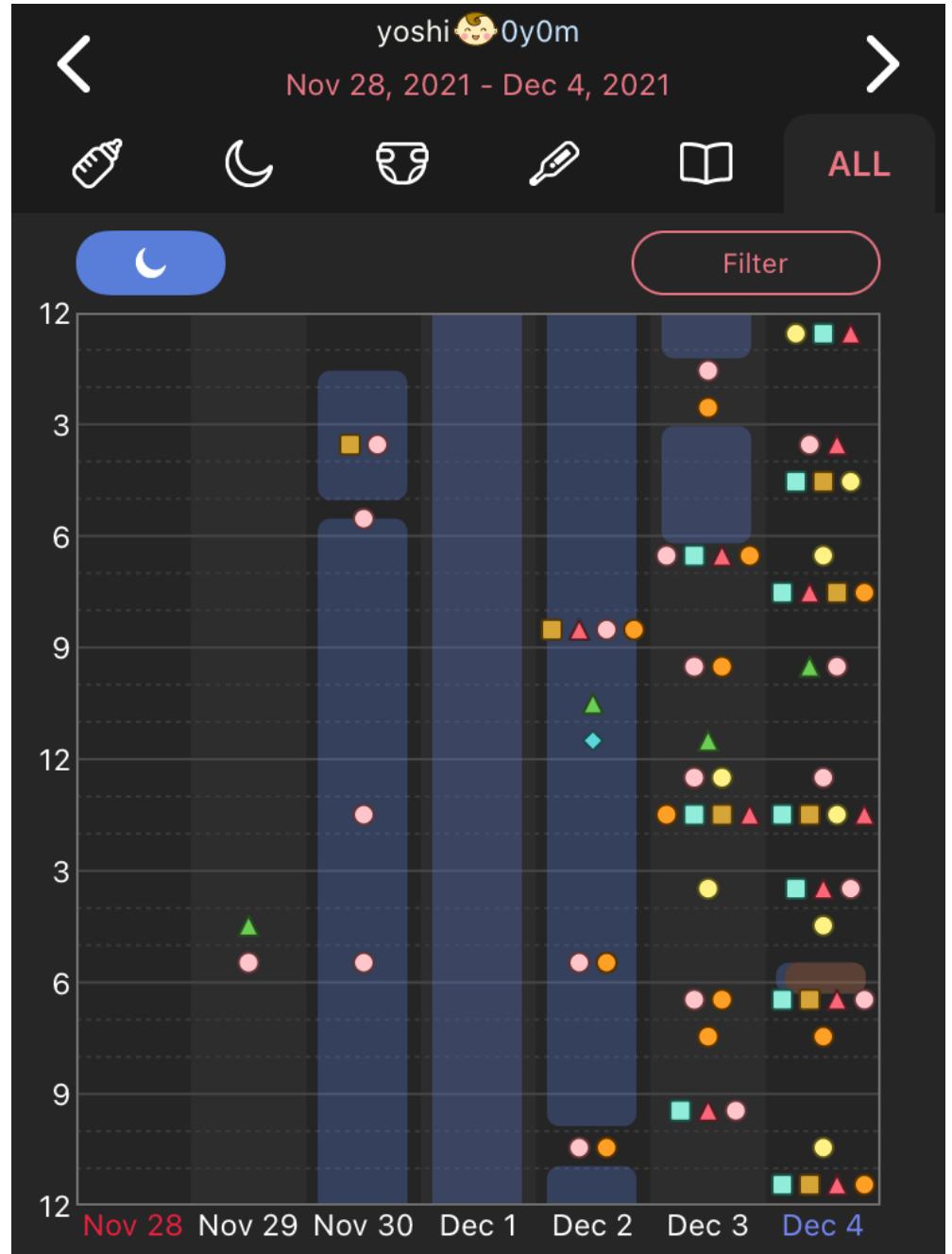
<https://data.rivm.nl/covid-19/>



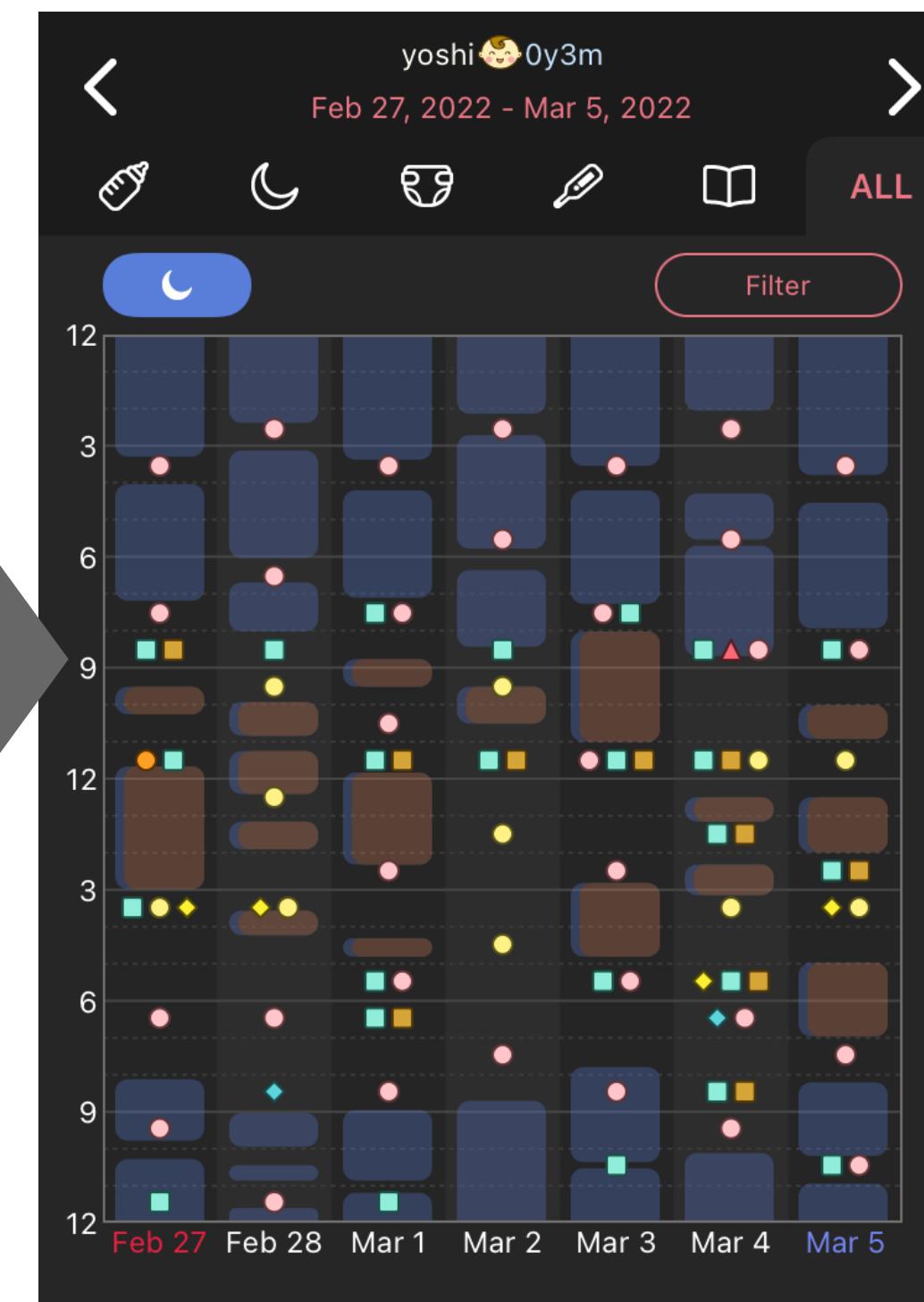
Data Digitalization → + Math, Data, Computing →



dear data



Data Digitalization



Data collection

Tue, Mar 1, 2022
yoshi (0y3m1d)

03:20 AM Wake-up (3h 40m)
03:25 AM Nursing R 28m
04:10 AM Sleep
07:05 AM Wake-up (2h 55m)
07:10 AM Pee
07:20 AM Nursing L 28m
08:45 AM Sleep
09:30 AM Wake-up (0h 45m)
10:30 AM Nursing L 6m ← R 25m
11:15 AM Pee
11:15 AM Poop
11:50 AM Sleep
02:20 PM Wake-up (2h 30m)
02:25 PM Nursing L 30m
04:20 PM Sleep
04:50 PM Wake-up (0h 30m)
05:30 PM Pee
05:35 PM Nursing R 30m
06:30 PM Poop
06:30 PM Pee
08:35 PM Nursing L 30m |
09:00 PM Sleep
10:55 PM Wake-up (1h 55m)
11:00 PM Pee
11:15 PM Sleep

Total nursing time L 94m / R 83m
Formula Total: 0 times 0ml
Total sleep 12h40m
Peed 5times
Pooped 2times

Intermediate data

A screenshot of a Microsoft Excel spreadsheet titled "yoshi-data - Saved to my Mac". The spreadsheet has columns labeled A through K. Column A contains the row numbers. Columns B through K contain the data for each event. The data includes the date, time, activity type, and duration. The last few rows show data for Wednesday, March 2, 2022.

Machine-readable Data

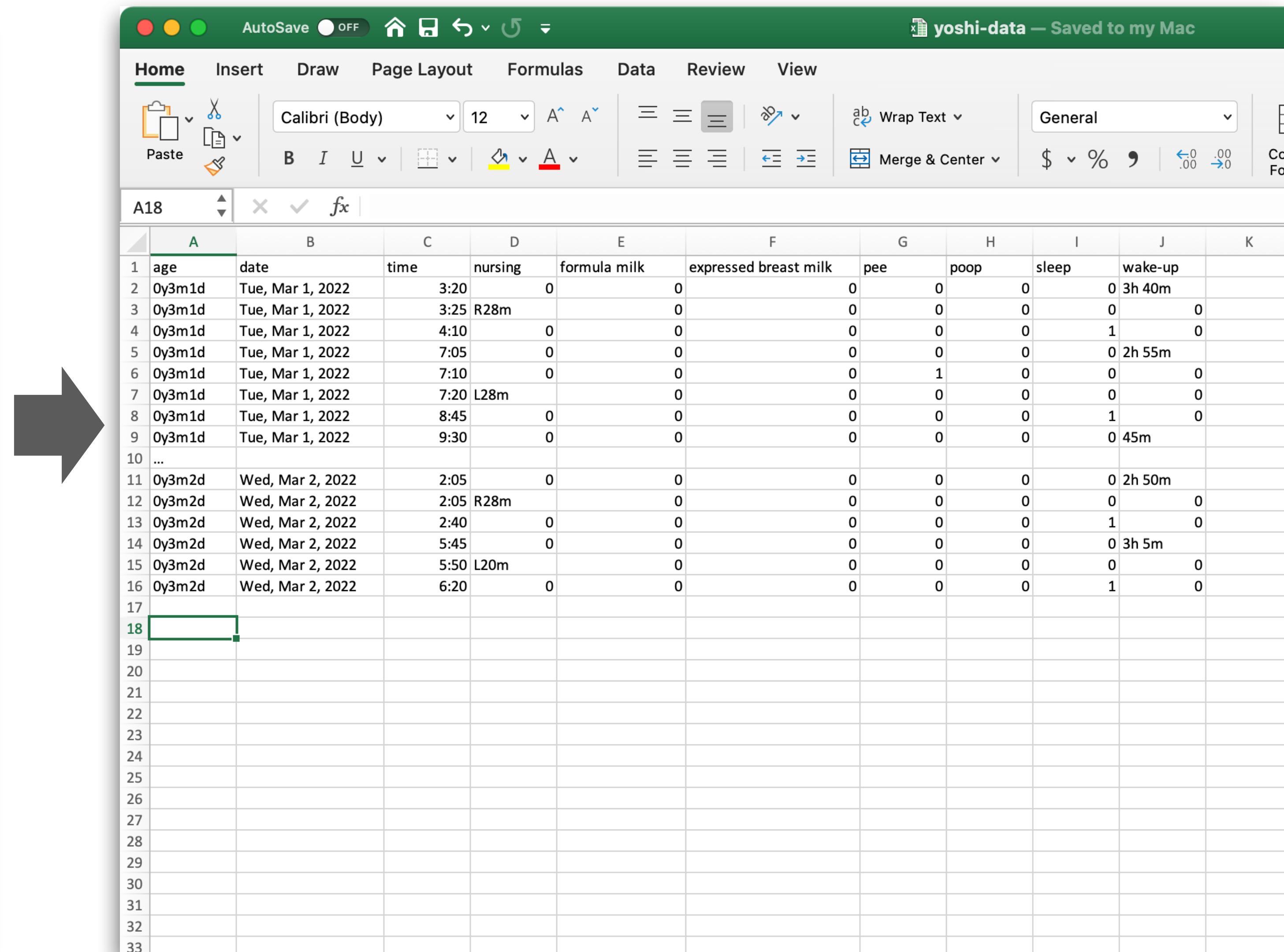
Real-world experience

Machine-Readable Data

Tue, Mar 1, 2022
yoshi (0y3m1d)

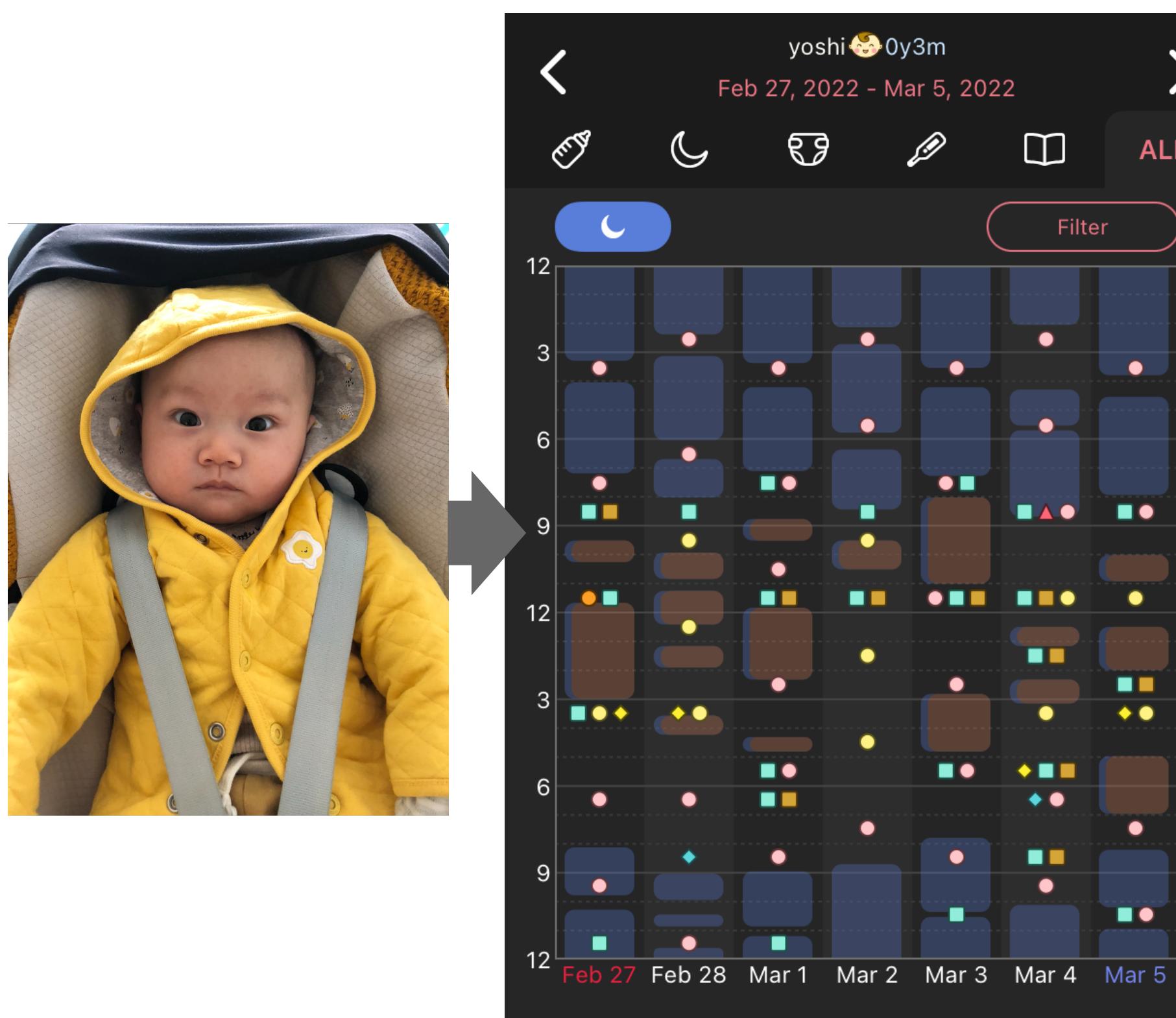
03:20 AM	Wake-up (3h 40m)
03:25 AM	Nursing R 28m
04:10 AM	Sleep
07:05 AM	Wake-up (2h 55m)
07:10 AM	Pee
07:20 AM	Nursing L 28m
08:45 AM	Sleep
09:30 AM	Wake-up (0h 45m)
10:30 AM	Nursing L 6m ← R 25m
11:15 AM	Pee
11:15 AM	Poop
11:50 AM	Sleep
02:20 PM	Wake-up (2h 30m)
02:25 PM	Nursing L 30m
04:20 PM	Sleep
04:50 PM	Wake-up (0h 30m)
05:30 PM	Pee
05:35 PM	Nursing R 30m
06:30 PM	Poop
06:30 PM	Pee
08:35 PM	Nursing L 30m
09:00 PM	Sleep
10:55 PM	Wake-up (1h 55m)
11:00 PM	Pee
11:15 PM	Sleep

Total nursing time L 94m / R 83m
Formula Total: 0 times 0ml
Total sleep 12h40m
Peed 5times
Pooped 2times



“Data Digitalization” Exercise (15 mins)

- Step1: Create a spreadsheet and define your columns
 - Step2: “Digitalize” your Dear Data



Tue, Mar 1, 2022
yoshi (0y3m1d)

Break

Working with AI, Data and Pattern

prompt-to-code, data pattern exploration



The shareable in-browser notebook

EXAMPLE NOTEBOOKS

Introduction to Starboard

Data Visualization in JS

Python: Pandas and Matplotlib

Procedural Art

Starboard is an [Open Startup](#).

Starboard is in beta, you can read more about Starboard [here](#).

Support Markdown, Latex, HTML, CSS, Javascript, and Python



[View source](#)

MARKDOWN

Introducing Starboard Notebook

Starboard brings cell-by-cell notebooks to the browser, no code is running on the backend here!

It's probably the quickest way to visualize some data with interactivity, do some prototyping, or build a rudimentary dashboard.

H4 Some features

- Mix Markdown, L^AT_EX, HTML, CSS, Javascript, and Python.
- The file format is a plaintext file, which plays nice with version control systems like git.
- Runs entirely in your browser, everything is static: no server, no setup and no build step.
- You can embed a fully functional notebook on your website.

Let's see it in action!

Tip: Press the ► Play button on the left to run a cell's code.

</>

JAVASCRIPT

```
1 // You write vanilla Javascript
2 const greeting = "Hello world!";
3
4 // The last statement in a cell will be displayed if it is not undefined.
5 greeting
```



**DATA
FOUNDRY**

<https://data.id.tue.nl>

What is Data Foundry?

The screenshot shows the Data Foundry website interface. On the left is a vertical navigation sidebar with links: Login, Register, Community (which is highlighted in green), Explore, Guides, and Support. The main content area has a header with the Data Foundry logo, a search icon, and 'Register' and 'Login' buttons. The main text on the page reads: "Welcome! This is Data Foundry @ ID Eindhoven! With this service you can collect data from a variety of sources and store data in a unified format. You can then export data for analysis or stream back in real-time into prototypes, products and systems." Below this, another text block says: "We have a documentation site where you can find more. [This way](#). Find out more about the project progress in the [development blog](#). If you need support, you have questions or feedback, head over to [support](#)." At the bottom, it says: "Find a few open projects below. Sign-in or register to get your own thing going." Below this, there are three cards for open projects: 1. TEMP_HUMIDITY_HO... by Eva van der Born (Study on how air quality affects your sleep pattern). 2. SMARTPHONE USAGE I... by Marco Peters (Project focuses on smartphone usage in the...). 3. TESTING EXAMPLE CO... by Pablo Traversat (Project consists of getting acquainted with ...).

**DATA
FOUNDRY**

Login Register Login

Community

Explore

Guides

Support

Welcome! This is Data Foundry @ ID Eindhoven!

With this service you can collect data from a variety of sources and store data in a unified format. You can then export data for analysis or stream back in real-time into prototypes, products and systems.

We have a documentation site where you can find more. [This way](#). Find out more about the project progress in the [development blog](#). If you need support, you have questions or feedback, head over to [support](#).

Find a few open projects below. Sign-in or register to get your own thing going.

TEMP_HUMIDITY_HO...

Study on how air quality affects your sleep pattern...

Eva van der Born

1/1 0/1 2 ENTITY EXISTING

FORM FITBIT ANNOTATION DIARY IOT

MEDIA

SMARTPHONE USAGE I...

This project focuses on smartphone usage in the...

Marco Peters

2/6 4 IOT

ENTITY ANNOTATION DIARY

IOT MEDIA

TESTING EXAMPLE CO...

This project consists of getting acquainted with ...

Pablo Traversat

3/4 10 ENTITY ANNOTATION DIARY

IOT MEDIA

What is Data Foundry?

The screenshot shows the Data Foundry website homepage. The header features a green logo with a stylized 'D' icon and the text 'DATA FOUNDRY'. To the right are links for 'Login', 'Register', and 'Login'. Below the header, a dark banner displays the text 'Welcome! This is Data Foundry® ID Eindhoven!' and 'Data foundry (<https://data.id.tue.nl>)'. It describes the service as allowing users to collect data from various sources and store it in a unified format, export it for analysis, or stream it back in real-time. The main content area lists two key features:

- a data platform that support researchers to collect data from a variety of sources and store data in a unified format (and GDPR-safe).
- Researchers can : (1) manage their design research projects with participants and devices, (2) collaborate and share data and (3) mash-up and export data

Below these features, a call-to-action encourages users to "Find a few open projects below. Sign-in or register to get your own thing going." At the bottom, three project cards are shown:

- TEMP_HUMIDITY_HO...** by **Eva van der Born**. Study on how air quality affects you sleep patte... Status: 1/1 FORM, 0/1 FITBIT, 2 ENTITY, EXISTING, MEDIA, IOT, ANNOTATION, DIARY, IOT.
- SMARTPHONE USAGE I...** by **Marco Peters**. This project focuses on smartphone usage in th... Status: 2/6 FORM, 4 ENTITY, IOT.
- TESTING EXAMPLE CO...** by **Pablo Traversat**. This project consists of getting acquainted with ... Status: 3/4 FORM, 10 ENTITY, ANNOTATION, DIARY, MEDIA, IOT.

Use Starboard for interactive prototyping

The screenshot shows the Data Foundry interface. On the left, a sidebar menu includes options like Portfolio, My projects, Archive, Community, Collaborations, Subscriptions, Explore, Data tools, Guides, and Support. The main area displays a project titled "OBJECT DETECTOR USING ML5.JS". It features sections for "Diary Dataset" (Data by participants as diary entries), "Media Dataset" (Media files (images)), "Movement Dataset" (Import one or more files of a movement dataset (GPX)), and "Experience Sampling Dataset". A red box highlights the "Existing Dataset" section, which describes it as "One or more files of an existing dataset (data, text, images, audio, html)".

Step 1: Create an existing dataset

The screenshot shows the Data Foundry interface. On the left, a sidebar menu includes options like Portfolio, My projects, Archive, Community, Collaborations, Subscriptions, Explore, Data tools, Guides, and Support. The main area displays a project titled "Home > Notebooks". It features a "NOTEBOOKS" section with a description: "What are notebooks? Data Foundry notebooks is a web-based interactive computational environment for creating notebook documents. A notebook document is a browser-based REPL containing an ordered list of input/output cells which can contain code, text (using Markdown), mathematics, plots and rich media. Underneath the interface, a notebook is a simple text document, usually ending with the ".gg" extension." Below this, there is a list of notebooks with columns for "Notebook", "Description", and "Uploaded". A red box highlights the "ADD A NEW NOTEBOOK" button at the bottom right.

Step 3: Add a new notebook

The screenshot shows the Data Foundry interface. On the left, a sidebar menu includes options like Portfolio, My projects, Archive, Community, Collaborations, Subscriptions, Explore, Data tools, Guides, and Support. The main area displays a project titled "DCB150 Digital Craftsmanship". It features a "DATASETS (3)" section with three items: "CHATBOT2" (id: 7967, SCRIPT), "CHATBOT INTERFACE" (id: 8163, EXISTING), and "NOTEBOOK FOR CODING PRACTICE" (id: 8164, EXISTING). A red box highlights the "DCB150 Digital Craftsmanship --> Notebook for coding practice" entry. The right side of the screen shows a "NOTEBOOKS" section with a list of notebooks, similar to the previous screenshot.

Step 4: Add a notebook into the newly created existing dataset

DATA FOUNDRY

- Portfolio
- My projects
- Archive
- Community**
- Collaborations
- Subscriptions
- Explore
- Data tools
- Guides
- Support

Home > DCB150 Digital Craftsmanship

DCB150 DIGITAL CRAFTSMANSHIP

AI tool for Digital Craftsmanship

project id: 3483 | PUBLIC | MIT

2023-03-24 | 2024-05-01

DATASETS (3)

- CHATBOT2 (id: 7967) SCRIPT
- CHATBOT INTERFACE (id: 8163) EXISTING
- NOTEBOOK FOR CODING PRACTICE (id: 8164) EXISTING**

ADD DATASET OR SCRIPT choose a dataset or script to add

Manage participants, wearables and devices in this project.

TIMELINE

	STUDY MANAGEMENT	RESEARCHERS AND PARTICIPANTS	CONNECTIONS	LAB NOTES
Project	Project phase	Post project phase (stopped data collection)	Archive project (remove participant info)	Project archived (create export bundle)
Datasets	ChatBot2	Chatbot Interface	Notebook for coding practice	
2023-04-01 2023-05-01 2023-06-01 2023-07-01 2023-08-01 2023-09-01 2023-10-01 2023-11-01 2023-12-01 2024-01-01 2024-02-01 2024-03-01 2024-04-01 2024-05-01 2024-06-01 2024-07-01 2024-08-01				

Home > DCB150 Digital Craftsmanship > Notebook for coding practice

NOTEBOOK FOR CODING PRACTICE

2023-05-01 | 2024-05-01

Starboard notebook for coding practice

INFO

License: MIT

MANAGE RESOURCES

EXPORT

REVIEW REQUEST

RESEARCHERS

Janet Huang

I-Tang Chiang

Add collaborator

VIEW DATA

DOWNLOAD

UPLOAD FILE(S)

DATASET FILES

File name	Description	Uploaded	Actions
Coding with ChatGPT.gg		May 08 at 11:55	download edit delete

CONFIGURATION

CSV/JSON TOKEN LINK

WEB-ACCESS

OOCST STREAM

p5.js

[Home](#)[Editor](#)[Download](#)[Donate](#)[Get Started](#)[Reference](#)[Libraries](#)[Learn](#)[Teach](#)[Examples](#)[Books](#)[Community](#)[Showcase](#)[Forum](#)

Hello!

Search p5js.org

p5.js is a JavaScript library for creative coding, with a focus on making coding accessible and inclusive for artists, designers, educators, beginners, and anyone else! p5.js is free and open-source because we believe software, and the tools to learn it, should be accessible to everyone.

Using the metaphor of a sketch, p5.js has a full set of drawing functionality. However, you're not limited to your drawing canvas. You can think of your whole browser page as your sketch, including HTML5 objects for text, input, video, webcam, and sound.

[Join the p5.js Discord!](#)

[Start creating with the p5 Editor!](#)

Community

We are a community of, and in solidarity with, people from every gender identity and expression, sexual orientation, race, ethnicity, language,

<https://p5js.org>

<https://editor.p5js.org>

Pattern exploration with AI

Notebook: Coding with ChatGPT.gg

Edit your notebook below, don't forget to save your changes. You can [access your datasets](#) with a bit of JavaScript or Python, depending on the notebook cell.

[Save notebook](#)

H2 Session 0: Setup for using ChatGPT

H3 Step 1: define request function for further POST call

In starboard, you can only use Pyodide to get data from the url. But, it is currently impossible to use the requests since sockets are not available in Pyodide. To make POST calls, you can use Web APIs in pyodide.

```
+ 1 # code is borrowed from https://docs.pyascript.net/latest/guides/http-requests.html
2 from pyodide.http import pyfetch, FetchResponse
3 from typing import Optional, Any
4
5 async def request(url: str, method: str = "GET", body: Optional[str] = None,
6                   headers: Optional[dict[str, str]] = None, **fetch_kwarg: Any) -> FetchResponse:
7
8     """"
9     Async request function. Pass in Method and make sure to await!
10    Parameters:
11        url: str = URL to make request to
12        method: str = {"GET", "POST", "PUT", "DELETE"} from `JavaScript` global fetch()
13        body: str = body as json string. Example, body=json.dumps(my_dict)
14        headers: dict[str, str] = header as dict, will be converted to string...
15            Example, headers=json.dumps({"Content-Type": "application/json"})
16        fetch_kwarg: Any = any other keyword arguments to pass to `pyfetch` (will be passed to `fetch`)
17
18    Return:
19        response: pyodide.http.FetchResponse = use with .status or await.json(), etc.
20
21    kwargs = {"method": method, "mode": "cors"} # CORS: https://en.wikipedia.org/wiki/Cross-origin_resource_shari
22    if body and method not in ["GET", "HEAD"]:
23        kwargs["body"] = body
24    if headers:
25        kwargs["headers"] = headers
26    kwargs.update(fetch_kwarg)
27
28    response = await pyfetch(url, **kwargs)
29    return response
```

Python initialization complete

H3 Step 2: Get your personal information from DF project for enabling OpenAI api

Copy the following information from your project and paste it to the following code.

(1) api_key
(2) project_id

```
+ 1 #api_key = '<api_key>' #use your api_key
2 #project_id = '<project_id>' #use your project_id
3 api_key = 'df-YitjZDNYa3ZiRjg2TkLHZTVkb2N5V0RJc3I0dEoyUk1Da1BTUlBzTFN1az0='
4 project_id = '3483'
5 headers = {'Content-Type': 'application/json'}
6
7 # system_role = f"""
8 # You are a new media artist who uses code and technology to create patterns. You are good at programming language
9 #
10 # setup a system role
```

Prompt-to-Code

Use starboard to explore pattern generation through ChatGPT

specify the role of ChatGPT and constraint the AI model's response

System prompt

You are a new media artist who uses code and technology to create patterns. You are good at using p5.js to generate patterns.

user's input/question/instruction for a specific task

User prompt

Write p5.js sketch that that draws the following:

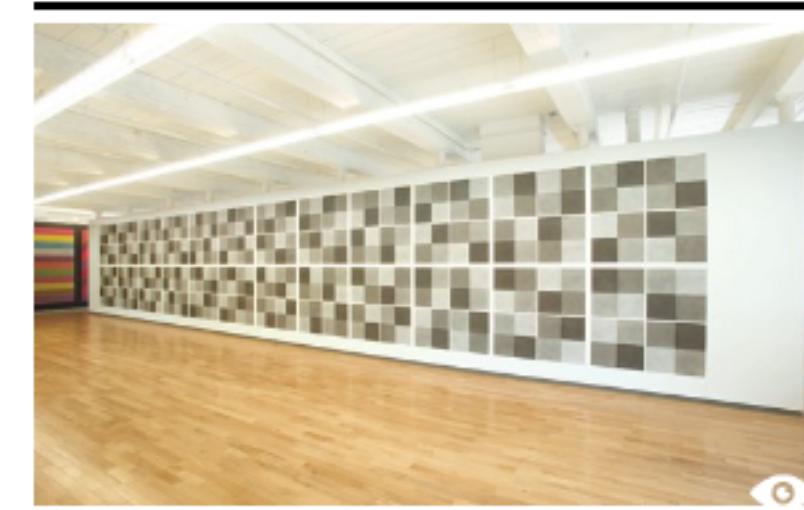
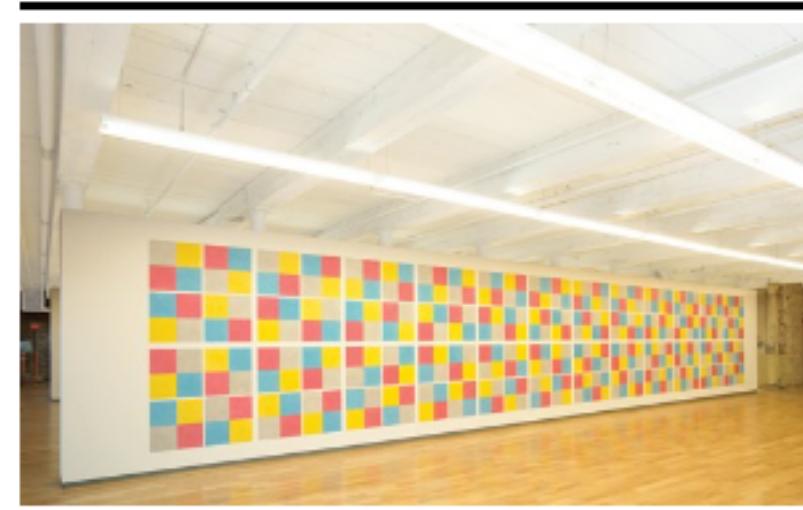
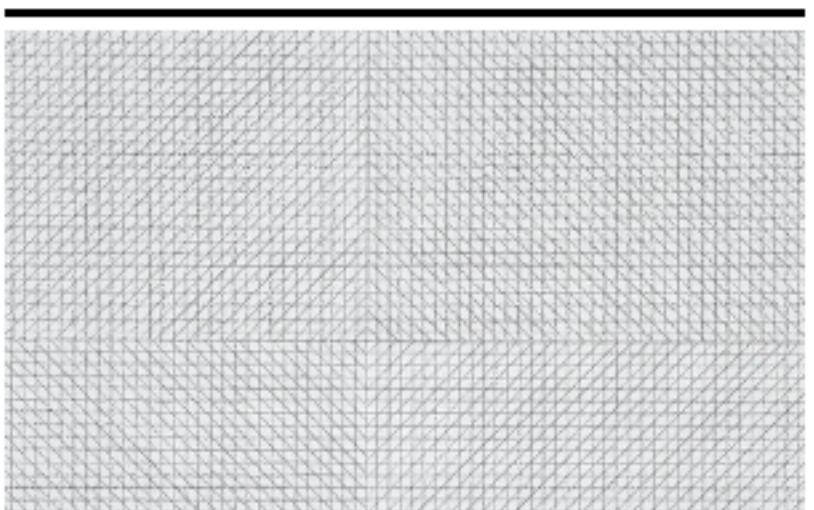
“Vertical lines, not straight, not touching, covering the wall evenly.”

pattern description

SOL LEWITT: A WALL DRAWING RETROSPECTIVE

possible by Agnes Gund.

<https://massmoca.org/sol-lewitt/>



Wall Drawing 11

A wall divided horizontally and vertically into four equal parts. Within each part, three of the four kinds of lines are... [Read More](#)

Wall Drawing 16

Bands of lines 12 inches (30 cm) wide, in three directions (vertical, horizontal, diagonal right) intersecting. Septembe... [Read More](#)

Wall Drawing 17

Four-part drawing with a different line direction in each part. September 1969 Black pencil Albright Knox Art Gallery, B... [Read More](#)

Wall Drawing 413

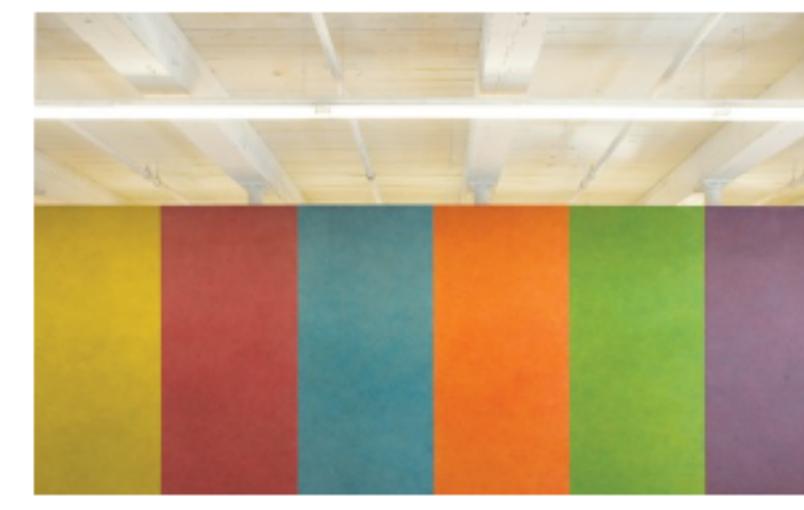
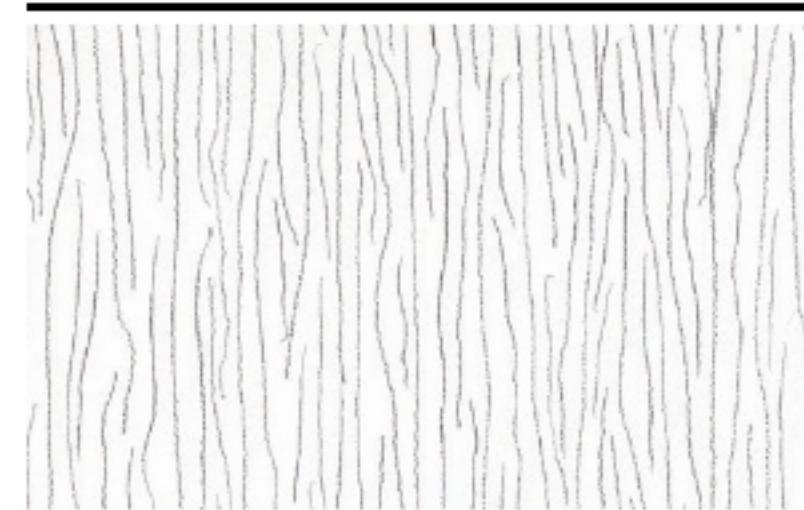
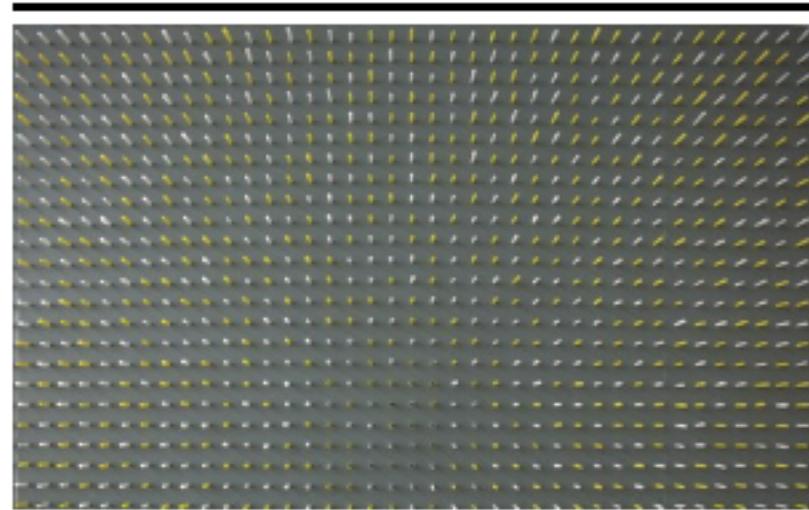
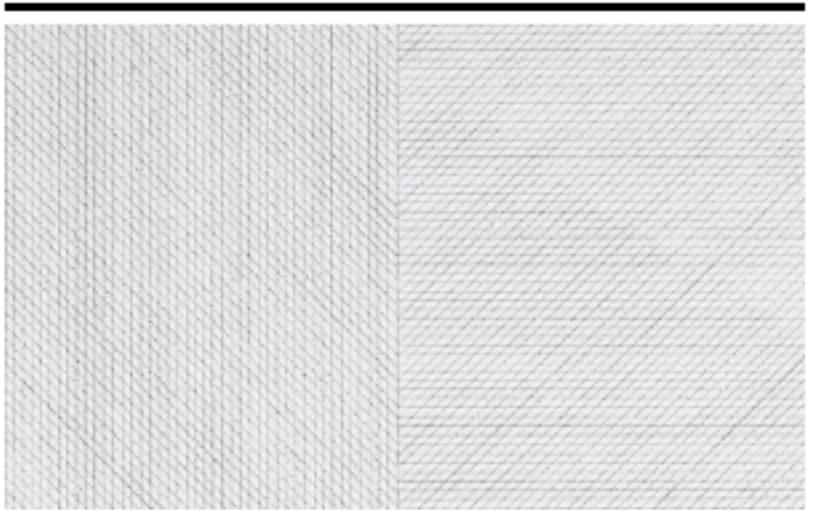
Drawing Series IV (A) with color ink washes. (24 drawings.) March 1984 Color ink wash LeWitt Collection, Chester, Connec... [Read More](#)

Wall Drawing 414

Drawing Series IV (A) with India ink washes. (24 Drawings.) March 1984 India ink wash LeWitt Collection, Chester, Connec... [Read More](#)

Wall Drawing 415D

Double Drawing. Right: Isometric Figure (Cube) with progressively darker gradations of gray on each of three planes; Le... [Read More](#)



Wall Drawing 19

A wall divided vertically into six equal parts, with two of the four kinds of line directions superimposed in each part.... [Read More](#)

Wall Drawing 38

Tissue paper cut into 1½-inch (4 cm) squares and inserted into holes in the gray pegboard walls. All holes in the walls... [Read More](#)

Wall Drawing 46

Vertical lines, not straight, not touching, covering the wall evenly. May 1970 Black pencil LeWitt Collection, Chester, ... [Read More](#)

Wall Drawing 419

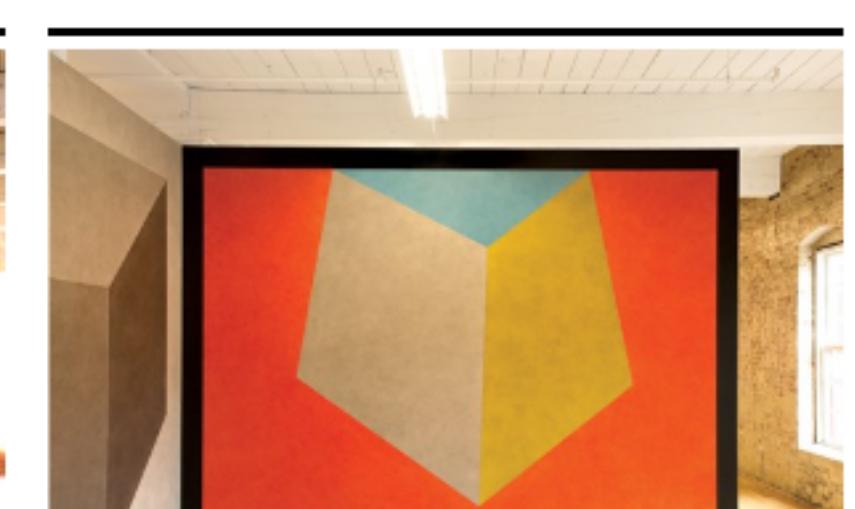
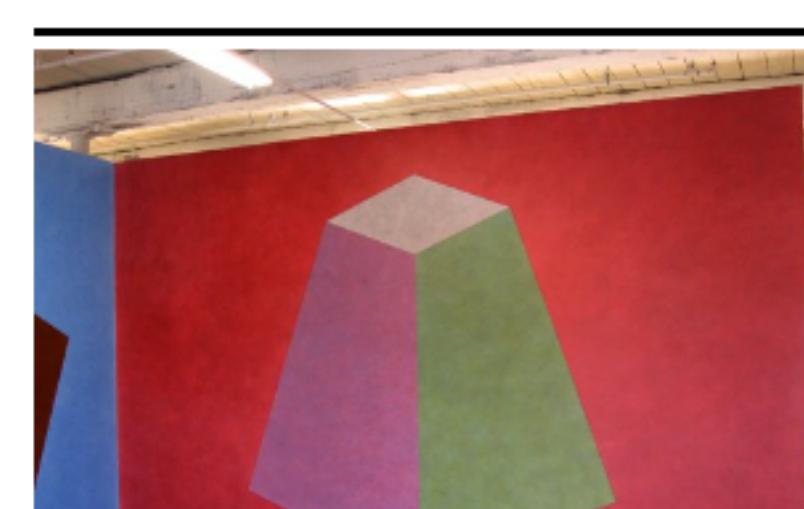
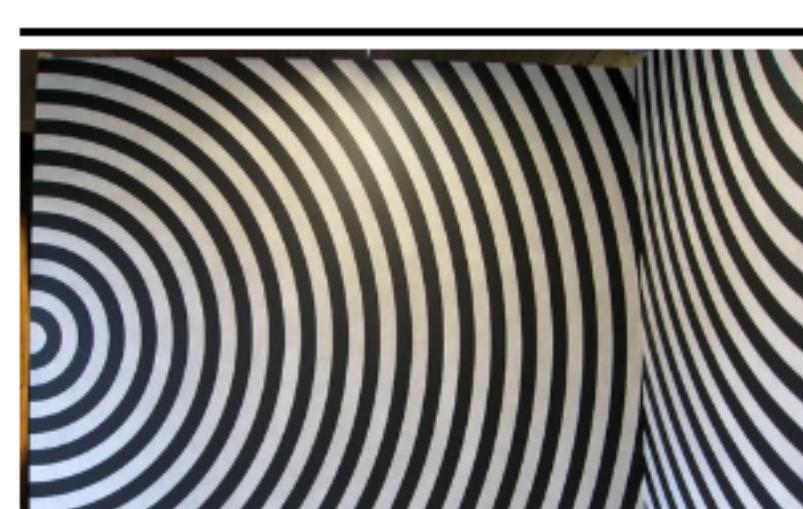
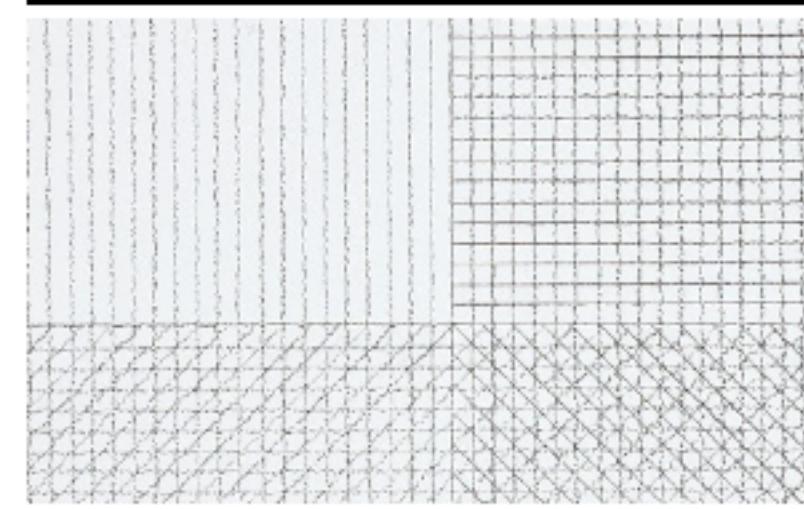
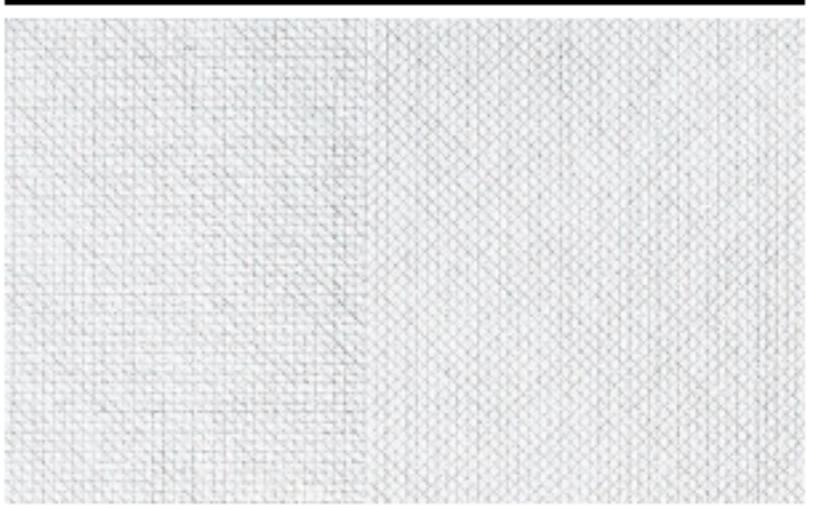
The wall is bordered and divided horizontally and vertically into four equal parts with a 6-inch (15 cm) black ink band.... [Read More](#)

Wall Drawing 422

The room (or wall) is divided vertically into fifteen parts. All one-, two-, three-, and four-part combinations of four ... [Read More](#)

Wall Drawing 439

Asymmetrical pyramid with color ink washes superimposed. May 1985 Color ink wash Cuomo Collection First installation Sal... [Read More](#)



Wall Drawing 47

A wall divided into fifteen equal parts, each with a different line direction, and all combinations. June 1970 Black pen... [Read More](#)

Wall Drawing 51

All architectural points connected by straight lines. June 1970 Blue snap lines LeWitt Collection, Chester, Connecticut ... [Read More](#)

Wall Drawing 56

A square is divided horizontally and vertically into four equal parts, each with lines in four directions superimposed p... [Read More](#)

Wall Drawing 462

On four walls, one room, arcs 4 inches (10 cm) wide, from the midpoints of four sides, drawn with alternating bands of g... [Read More](#)

Wall Drawing 527

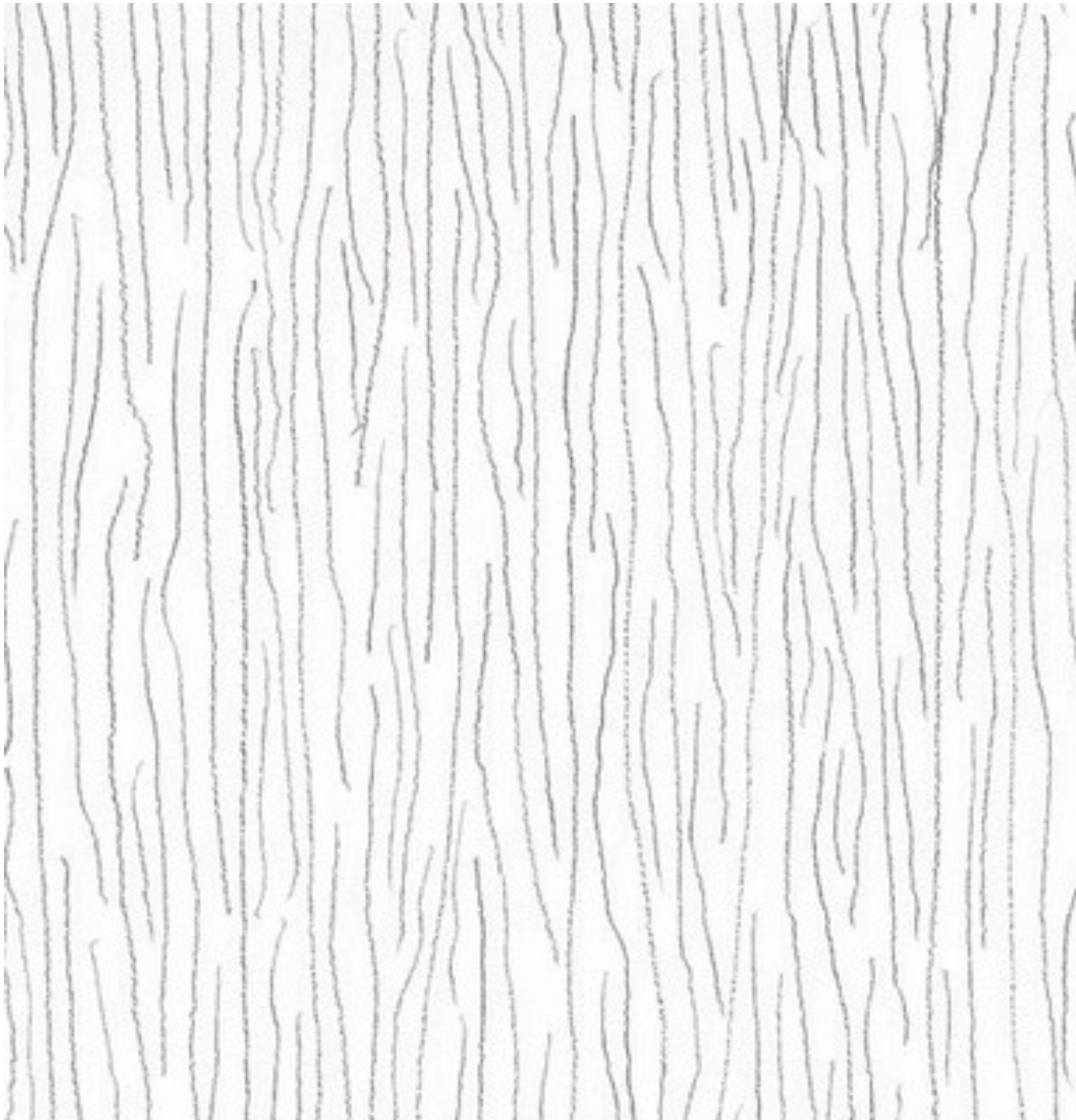
Two flat-topped pyramids with color ink washes superimposed. April 1987 Color ink wash Courtesy of the Estate of Sol LeW... [Read More](#)

Wall Drawing 552D

Tilted forms with color ink washes superimposed. December 1987 Color ink wash Courtesy of the Estate of Sol LeWitt First... [Read More](#)

Wall Drawing 46

Sol LeWitt



Sol LeWitt

"Wall Drawing 46," May 1970

Vertical lines, not straight, not touching,
covering the wall evenly.

Black pencil

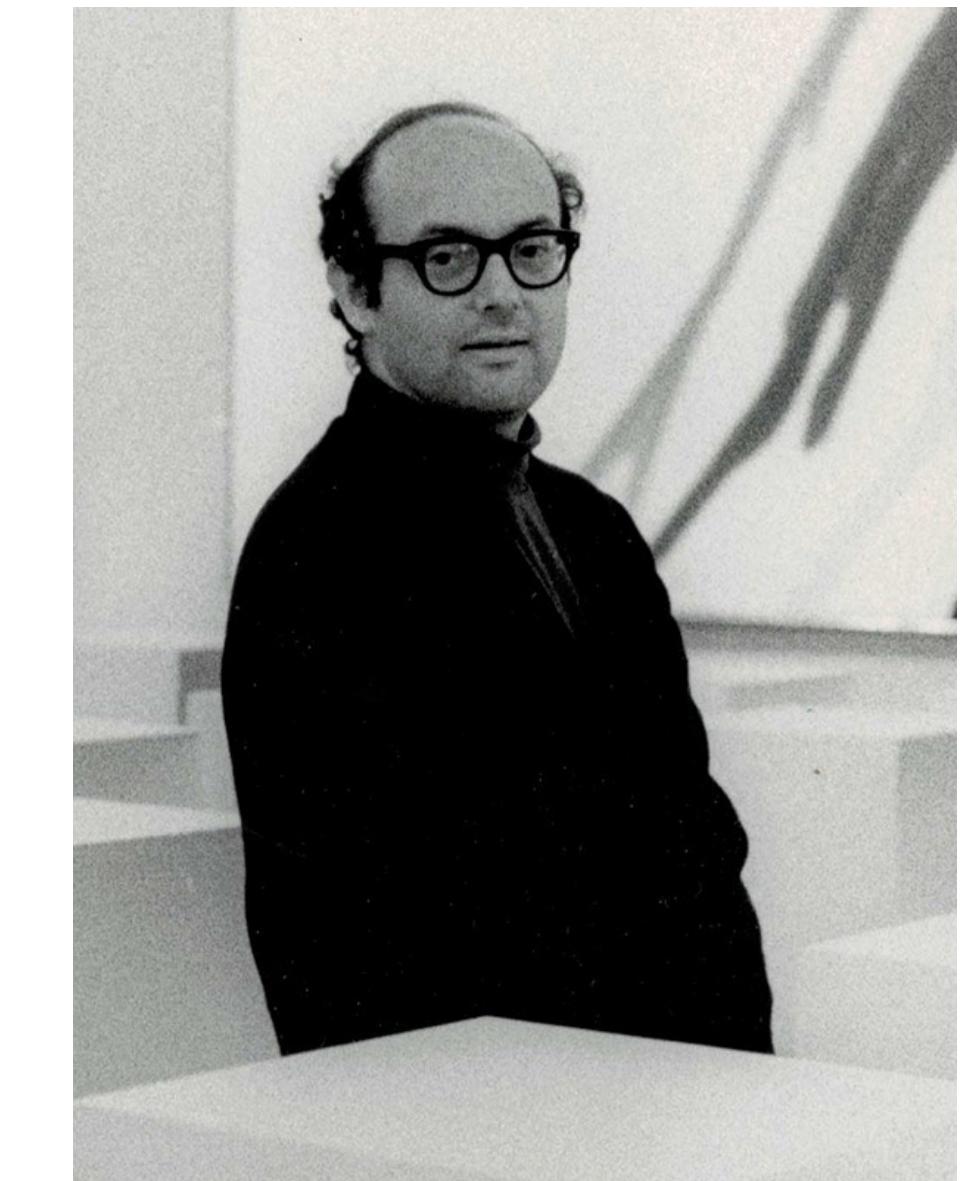


Image source 1: <https://massmoca.org/event/walldrawing46/>

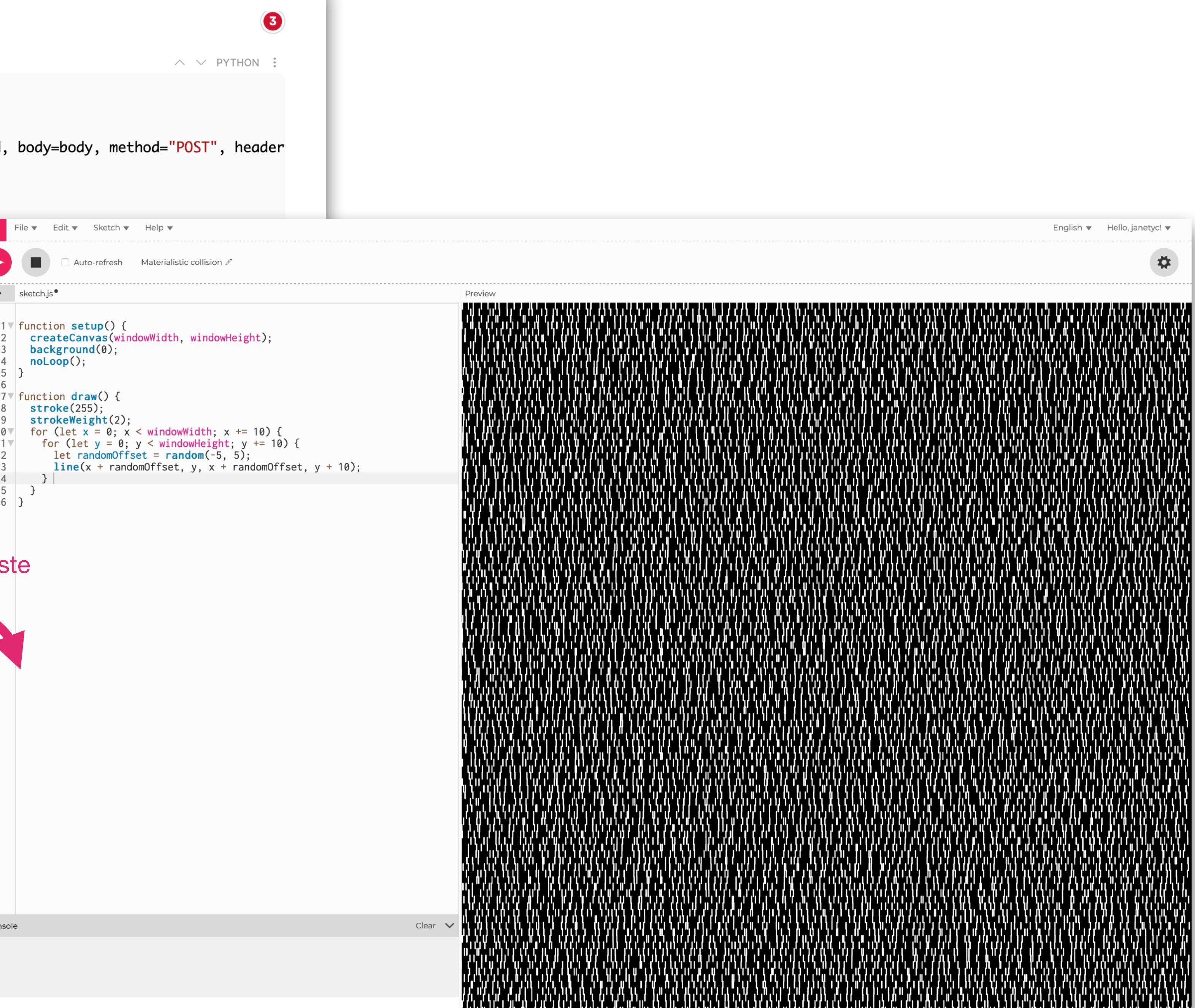
Image source 2: <https://www.ragoarts.com/auctions/2021/06/prints-multiples/544>

Step 2: Call DF's OpenAI api to generating patterns through p5.js code

```
1 import json
2
3 body = json.dumps(data)
4 new_post = await request(f"https://data.id.tue.nl/api/vendor/openai/" + project_id, body=body, method="POST", header
5 results = await new_post.json()
6 print(f"POST request=> status:{new_post.status}, json:{results}")
7 print("-----")
8
9 #print out the important information: cost and output
10 print("cost:%d" % results["cost"])
11 print(results["content"])
```

POST request=> status:200, json:{'role': 'assistant', 'content': "Here's a p5.js sketch that draws g, covering the wall evenly:\n```\nfunction setup() {\n createCanvas(windowWidth, windowHeight);\n stroke(255);\n strokeWeight(2);\n for (let x = 0; x < windowHeight; x += 10) {\n for (let y = 0; y < windowHeight; y += 10) {\n let randomOffset = random(-5, 5);\n line(x + randomOffset, y, x + randomOffset, y + 10);\n }\n }\n}\n\nfunction draw() {\n background(0);\n stroke(255);\n strokeWeight(2);\n for (let x = 0; x < windowHeight; x += 10) {\n for (let y = 0; y < windowHeight; y += 10) {\n let randomOffset = random(-5, 5);\n line(x + randomOffset, y, x + randomOffset, y + 10);\n }\n }\n}\n\n```\n\nThis code creates a canvas that fills the entire window, sets the background color to black, and th ps. Each line is drawn 2 pixels wide with a white stroke color. The x and y values are incremented are evenly spaced. The `randomOffset` variable adds a random value between -5 and 5 to the x coord right and not touching."}

copy and paste



 Auto-refresh

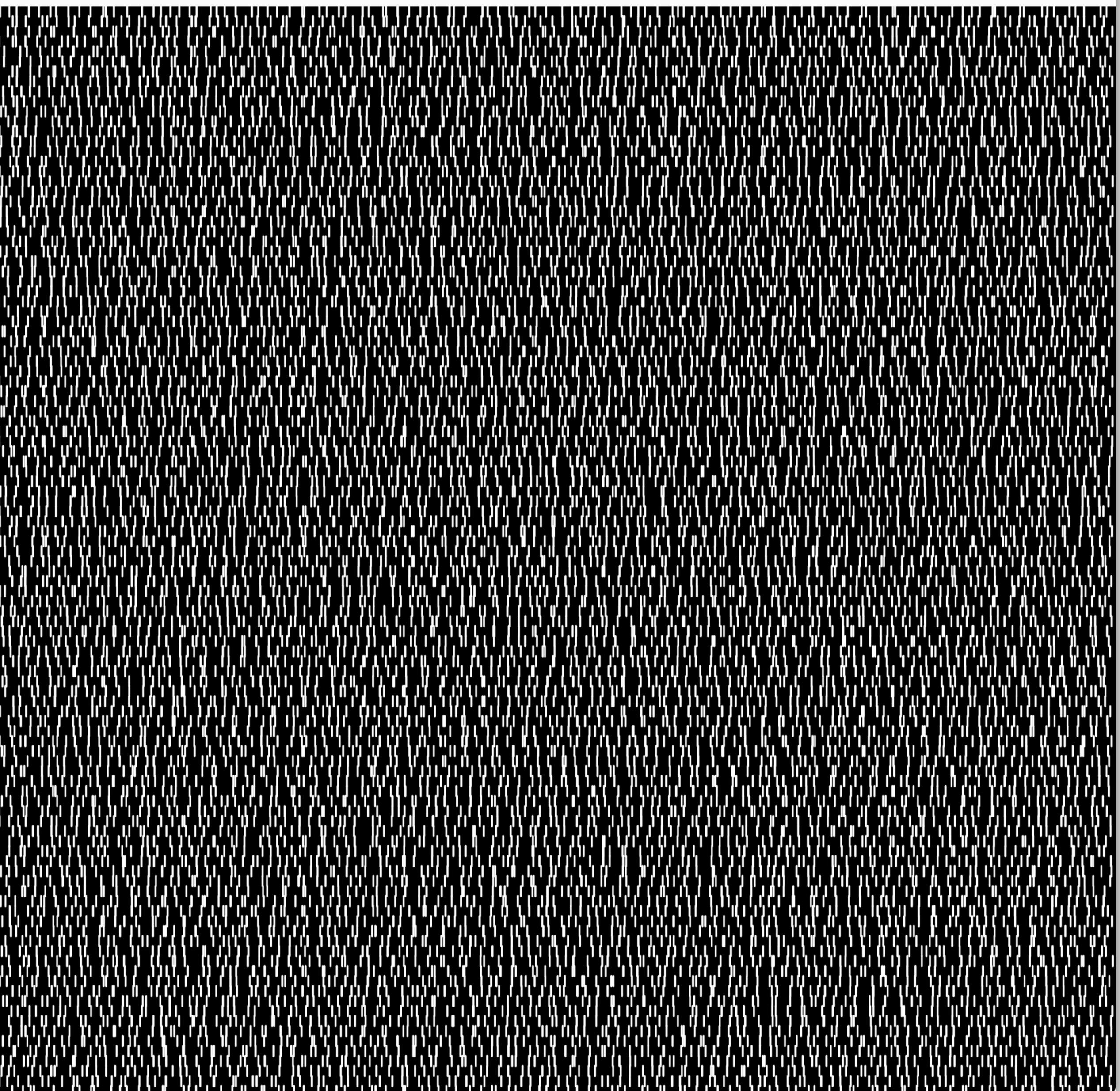
Materialistic collision



> sketch.js*

```
1▼ function setup() {
2  createCanvas(windowWidth, windowHeight);
3  background(0);
4  noLoop();
5 }
6
7▼ function draw() {
8  stroke(255);
9  strokeWeight(2);
10▼ for (let x = 0; x < windowHeight; x += 10) {
11▼   for (let y = 0; y < windowHeight; y += 10) {
12    let randomOffset = random(-5, 5);
13    line(x + randomOffset, y, x + randomOffset, y + 10);
14  }
15 }
16 }
```

Preview



Console

Clear

>

 Auto-refresh

Materialistic collision by janetyc

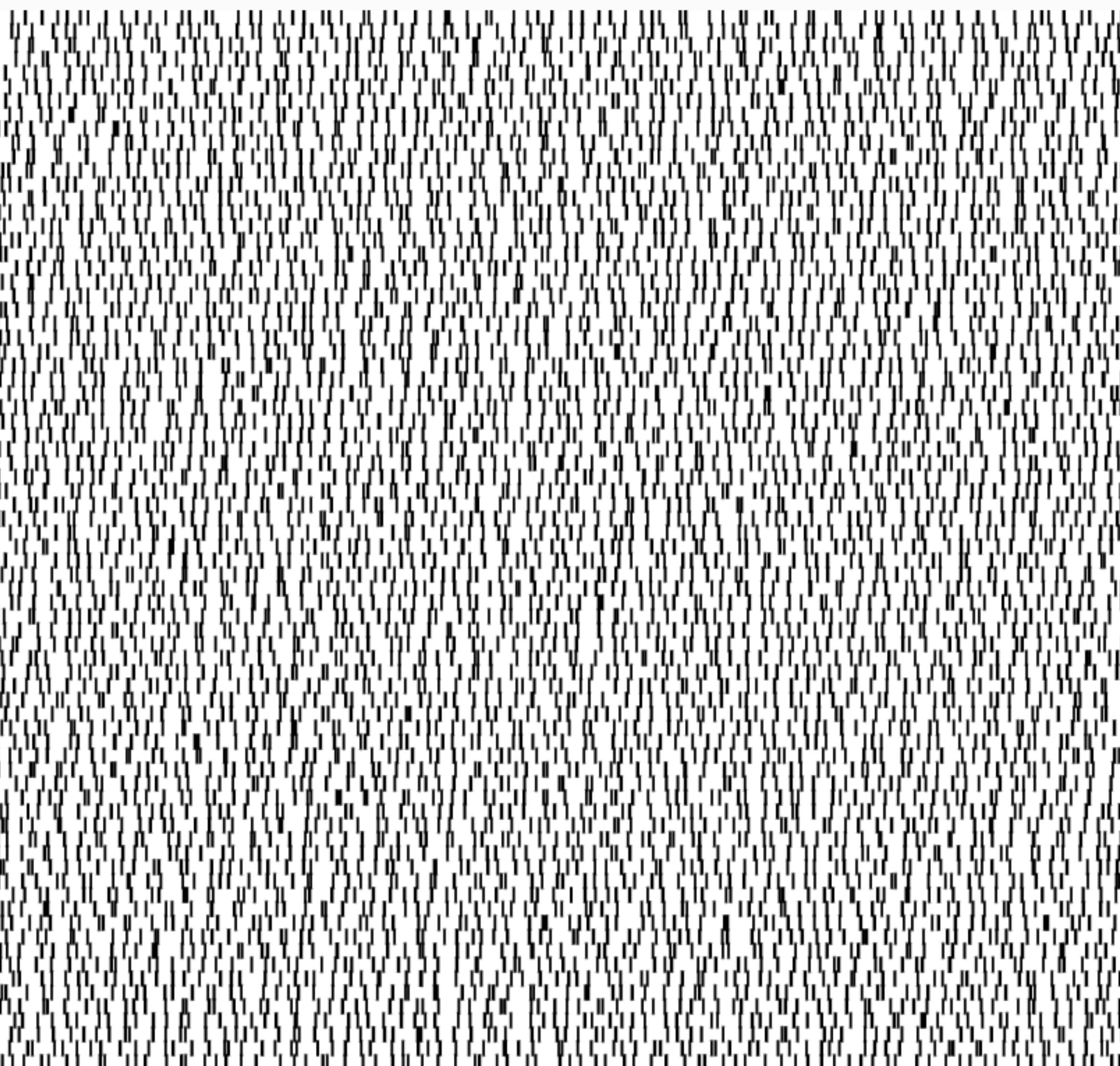


> sketch.js

Saved: 25 seconds ago

Preview

```
1
2 function setup() {
3   createCanvas(windowWidth, windowHeight);
4   background(255);
5   noLoop();
6 }
7
8 function draw() {
9   stroke(0);
10  strokeWeight(2);
11  for (let x = 0; x < windowHeight; x += 10) {
12    for (let y = 0; y < windowHeight; y += 10) {
13      let randomOffset = random(-5, 5);
14      line(x + randomOffset, y, x + randomOffset, y + 10);
15    }
16  }
17 }
```



Console

Clear ▾

 Auto-refresh

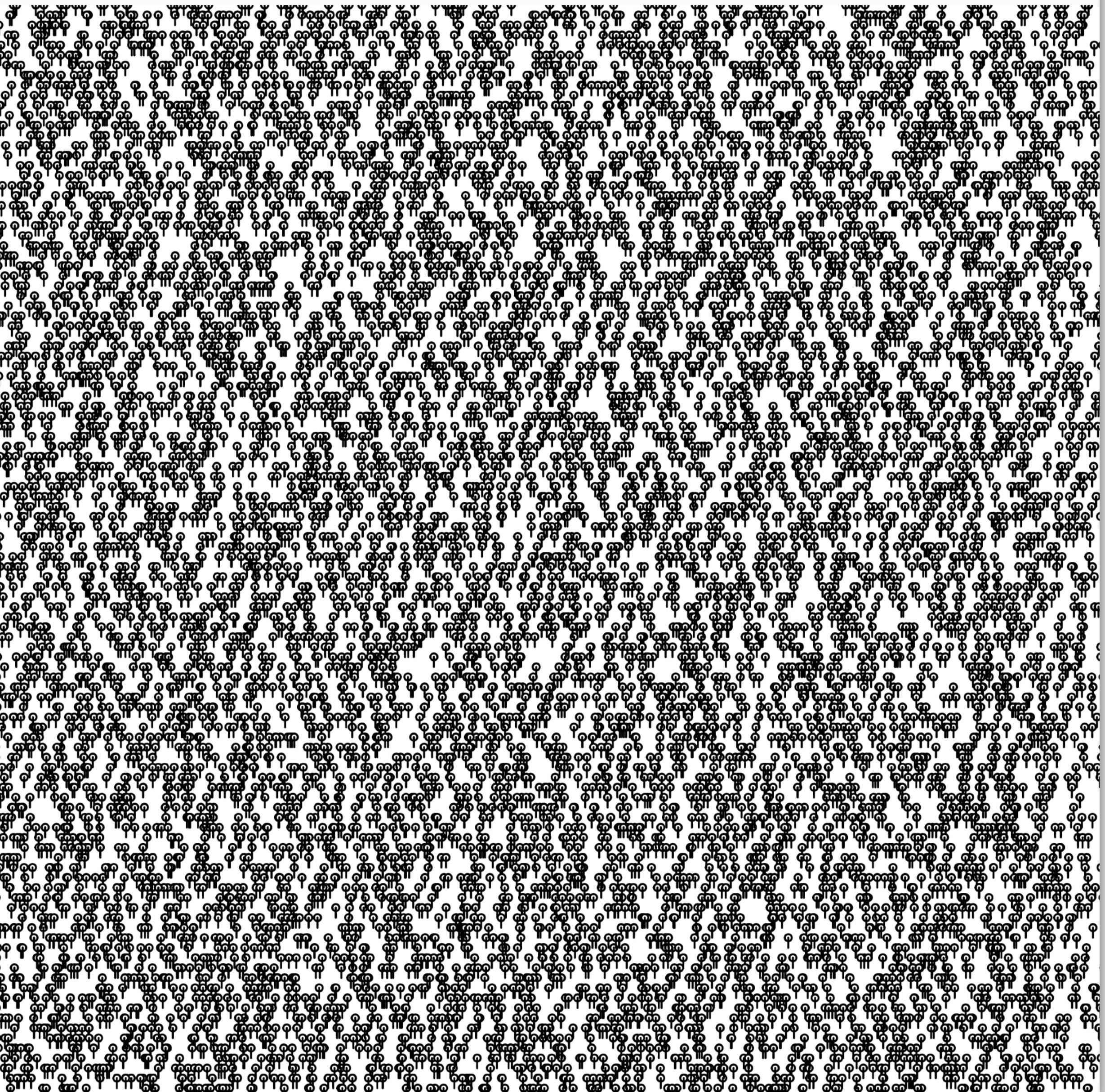
Materialistic collision



> sketch.js

```
1 function setup() {
2   createCanvas(windowWidth, windowHeight);
3   background(255);
4   noLoop();
5 }
6
7 function draw() {
8   stroke(0);
9   strokeWeight(2);
10  for (let x = 0; x < windowHeight; x += 10) {
11    for (let y = 0; y < windowHeight; y += 10) {
12      let randomOffset = random(-50, 50);
13
14      circle(x + randomOffset, y, 7);
15      line(x + randomOffset, y, x + randomOffset, y + 10);
16    }
17  }
18}
```

Preview



Console

Clear

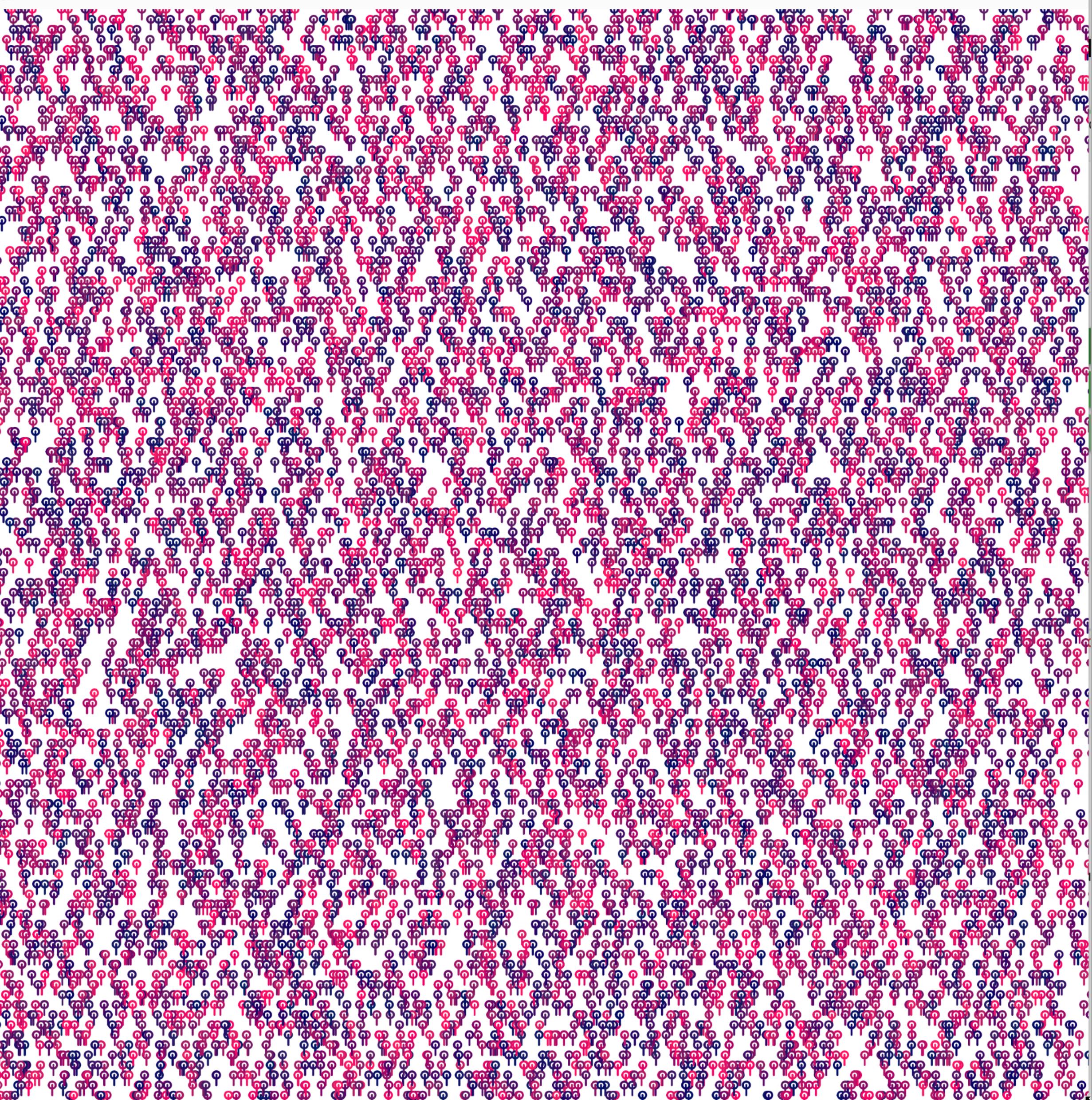
 Auto-refresh

Materialistic collision



```
> sketch.js•  
1 function setup() {  
2   createCanvas(windowWidth, windowHeight);  
3   background(255);  
4   noLoop();  
5 }  
6  
7 function draw() {  
8  
9   strokeWeight(2);  
10  for (let x = 0; x < windowHeight; x += 10) {  
11    for (let y = 0; y < windowHeight; y += 10) {  
12      let randomOffset = random(-50, 50);  
13      stroke(random(255), random(30, 100));  
14      circle(x + randomOffset, y, 7);  
15      line(x + randomOffset, y, x + randomOffset, y + 10);  
16    }  
17  }  
18}
```

Preview



Console

Clear

Add data into pattern

```
[base] ~ ▶ curl "https://archive-api.open-meteo.com/v1/era5?latitude=52.52&longitude=13.41&start_date=2021-01-01&end_date=2021-12-31&hourly=temperature_2m"
{"latitude":52.5,"longitude":13.400009,"generationtime_ms":0.39899349212646484,"utc_offset_seconds":0,"timezone":"GMT","timezone_abbr":eviation":"GMT","elevation":38.0,"hourly_units":{"time":"iso8601","temperature_2m":"°C"},"hourly":[{"time":["2021-01-01T00:00","2021-01-01T01:00","2021-01-01T02:00","2021-01-01T03:00","2021-01-01T04:00","2021-01-01T05:00","2021-01-01T06:00","2021-01-01T07:00","2021-01-01T08:00","2021-01-01T09:00","2021-01-01T10:00","2021-01-01T11:00","2021-01-01T12:00","2021-01-01T13:00","2021-01-01T14:00","2021-01-01T15:00","2021-01-01T16:00","2021-01-01T17:00","2021-01-01T18:00","2021-01-01T19:00","2021-01-01T20:00","2021-01-01T21:00","2021-01-01T22:00","2021-01-01T23:00","2021-01-02T00:00","2021-01-02T01:00","2021-01-02T02:00","2021-01-02T03:00","2021-01-02T04:00","2021-01-02T05:00","2021-01-02T06:00","2021-01-02T07:00","2021-01-02T08:00","2021-01-02T09:00","2021-01-02T10:00","2021-01-02T11:00","2021-01-02T12:00","2021-01-02T13:00","2021-01-02T14:00","2021-01-02T15:00","2021-01-02T16:00","2021-01-02T17:00","2021-01-02T18:00","2021-01-02T19:00","2021-01-02T20:00","2021-01-02T21:00","2021-01-02T22:00","2021-01-02T23:00","2021-01-03T00:00","2021-01-03T01:00","2021-01-03T02:00","2021-01-03T03:00","2021-01-03T04:00","2021-01-03T05:00","2021-01-03T06:00","2021-01-03T07:00","2021-01-03T08:00","2021-01-03T09:00","2021-01-03T10:00","2021-01-03T11:00","2021-01-03T12:00","2021-01-03T13:00","2021-01-03T14:00","2021-01-03T15:00","2021-01-03T16:00","2021-01-03T17:00","2021-01-03T18:00","2021-01-03T19:00","2021-01-03T20:00","2021-01-03T21:00","2021-01-03T22:00","2021-01-03T23:00","2021-01-04T00:00","2021-01-04T01:00","2021-01-04T02:00","2021-01-04T03:00","2021-01-04T04:00","2021-01-04T05:00","2021-01-04T06:00","2021-01-04T07:00","2021-01-04T08:00","2021-01-04T09:00","2021-01-04T10:00","2021-01-04T11:00","2021-01-04T12:00","2021-01-04T13:00","2021-01-04T14:00","2021-01-04T15:00","2021-01-04T16:00","2021-01-04T17:00","2021-01-04T18:00","2021-01-04T19:00","2021-01-04T20:00","2021-01-04T21:00","2021-01-04T22:00","2021-01-04T23:00","2021-01-05T00:00","2021-01-05T01:00","2021-01-05T02:00","2021-01-05T03:00","2021-01-05T04:00","2021-01-05T05:00","2021-01-05T06:00","2021-01-05T07:00","2021-01-05T08:00","2021-01-05T09:00","2021-01-05T10:00","2021-01-05T11:00","2021-01-05T12:00","2021-01-05T13:00","2021-01-05T14:00","2021-01-05T15:00","2021-01-05T16:00","2021-01-05T17:00","2021-01-05T18:00","2021-01-05T19:00","2021-01-05T20:00","2021-01-05T21:00","2021-01-05T22:00","2021-01-05T23:00","2021-01-06T00:00","2021-01-06T01:00","2021-01-06T02:00","2021-01-06T03:00","2021-01-06T04:00","2021-01-06T05:00","2021-01-06T06:00","2021-01-06T07:00","2021-01-06T08:00","2021-01-06T09:00","2021-01-06T10:00","2021-01-06T11:00","2021-01-06T12:00","2021-01-06T13:00","2021-01-06T14:00","2021-01-06T15:00","2021-01-06T16:00","2021-01-06T17:00","2021-01-06T18:00","2021-01-06T19:00","2021-01-06T20:00","2021-01-06T21:00","2021-01-06T22:00","2021-01-06T23:00","2021-01-07T00:00","2021-01-07T01:00","2021-01-07T02:00","2021-01-07T03:00","2021-01-07T04:00","2021-01-07T05:00","2021-01-07T06:00","2021-01-07T07:00","2021-01-07T08:00","2021-01-07T09:00","2021-01-07T10:00","2021-01-07T11:00","2021-01-07T12:00","2021-01-07T13:00","2021-01-07T14:00","2021-01-07T15:00","2021-01-07T16:00","2021-01-07T17:00","2021-01-07T18:00","2021-01-07T19:00","2021-01-07T20:00","2021-01-07T21:00","2021-01-07T22:00","2021-01-07T23:00","2021-01-08T00:00","2021-01-08T01:00","2021-01-08T02:00","2021-01-08T03:00","2021-01-08T04:00","2021-01-08T05:00","2021-01-08T06:00","2021-01-08T07:00","2021-01-08T08:00","2021-01-08T09:00","2021-01-08T10:00","2021-01-08T11:00","2021-01-08T12:00","2021-01-08T13:00","2021-01-08T14:00","2021-01-08T15:00","2021-01-08T16:00","2021-01-08T17:00","2021-01-08T18:00","2021-01-08T19:00","2021-01-08T20:00","2021-01-08T21:00","2021-01-08T22:00","2021-01-08T23:00","2021-01-09T00:00","2021-01-09T01:00","2021-01-09T02:00","2021-01-09T03:00","2021-01-09T04:00","2021-01-09T05:00","2021-01-09T06:00","2021-01-09T07:00","2021-01-09T08:00","2021-01-09T09:00","2021-01-09T10:00","2021-01-09T11:00","2021-01-09T12:00","2021-01-09T13:00","2021-01-09T14:00","2021-01-09T15:00","2021-01-09T16:00","2021-01-09T17:00","2021-01-09T18:00","2021-01-09T19:00","2021-01-09T20:00","2021-01-09T21:00","2021-01-09T22:00","2021-01-09T23:00","2021-01-10T00:00","2021-01-10T01:00","2021-01-10T02:00","2021-01-10T03:00","2021-01-10T04:00","2021-01-10T05:00","2021-01-10T06:00","2021-01-10T07:00","2021-01-10T08:00","2021-01-10T09:00","2021-01-10T10:00","2021-01-10T11:00","2021-01-10T12:00","2021-01-10T13:00","2021-01-10T14:00","2021-01-10T15:00","2021-01-10T16:00","2021-01-10T17:00","2021-01-10T18:00","2021-01-10T19:00","2021-01-10T20:00","2021-01-10T21:00","2021-01-10T22:00"}]
```

https://api.open-meteo.com/v1/forecast?latitude=52.52&longitude=13.41&past_days=10&hourly=temperature_2m,relativehumidity_2m,windspeed_10m

json data structure

```
▼ {latitude: 52.52, longitude: 13.419998, generationtime_ms: 0.7320642471313477, utc_offset_seconds: 0, time zone: "GMT"...}
  latitude: 52.52
  longitude: 13.419998
  generationtime_ms: 0.7320642471313477
  utc_offset_seconds: 0
  timezone: "GMT"
  timezone_abbreviation: "GMT"
  elevation: 38
  ▼ hourly_units: Object
    time: "iso8601"
    temperature_2m: "°C"
    relativehumidity_2m: "%"
    windspeed_10m: "km/h"
  ▼ hourly: Object
    ► time: Array(408)
      ► temperature_2m: Array(408) get temperature data
      ► relativehumidity_2m: Array(408)
      ► windspeed_10m: Array(408)
```

```
relativehumidity_2m: "%"
windspeed_10m: "km/h"
▼ hourly: Object
  ► time: Array(408)
    ▼ temperature_2m: Array(408)
      0: 8.2
      1: 7.2
      2: 6.2
      3: 5.1
      4: 4.5
      5: 4.4
      6: 5.5
      7: 6.9
      8: 8.2
      9: 9.7
      10: 10.9
      11: 11.9
      12: 12.8
      13: 13.6
      14: 14
      15: 14.4
      16: 14
      17: 13.4
      18: 13.2
      19: 11.5
      20: 10
      21: 8.7
      22: 8.1
```

 Auto-refresh

Materialistic collision by janetyc



Sketch Files ▾

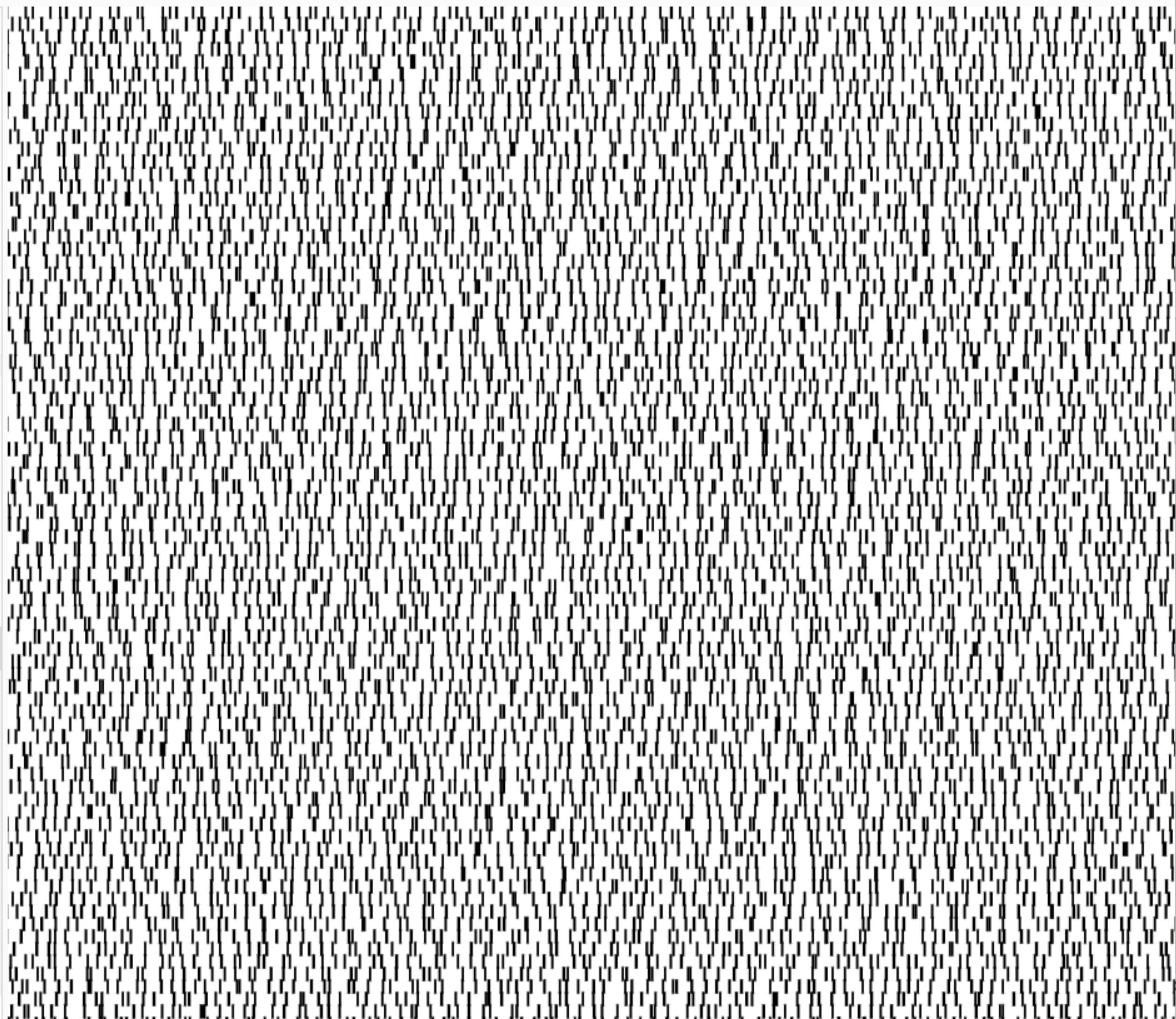
< sketch.js*

- era5.json
- forecast.json
- index.html
- sketch.js
- style.css

```
1 function setup() {
2   createCanvas(windowWidth, windowHeight);
3   background(255);
4   stroke(0);
5   strokeWeight(2)
6
7   for (let x = 0; x < windowHeight; x += 10) {
8     for (let y = 0; y < windowHeight; y += 10) {
9       let randomOffset = random(-5, 5);
10      line(x + randomOffset, y, x + randomOffset, y + 10);
11    }
12  }
13
14 }
15
16 function draw() {
17 }
```

Saved: 25 seconds ago

Preview



Console

Clear ▾

>

 Auto-refresh

Materialistic collision ↗ by janetyc



Sketch Files ▾

< sketch.js*

```
1 let temperature_data;
2 let i;
3 function setup() {
4   createCanvas(windowWidth, windowHeight);
5   background(255);
6   stroke(0);
7   strokeWeight(2);
8   i=0;
```

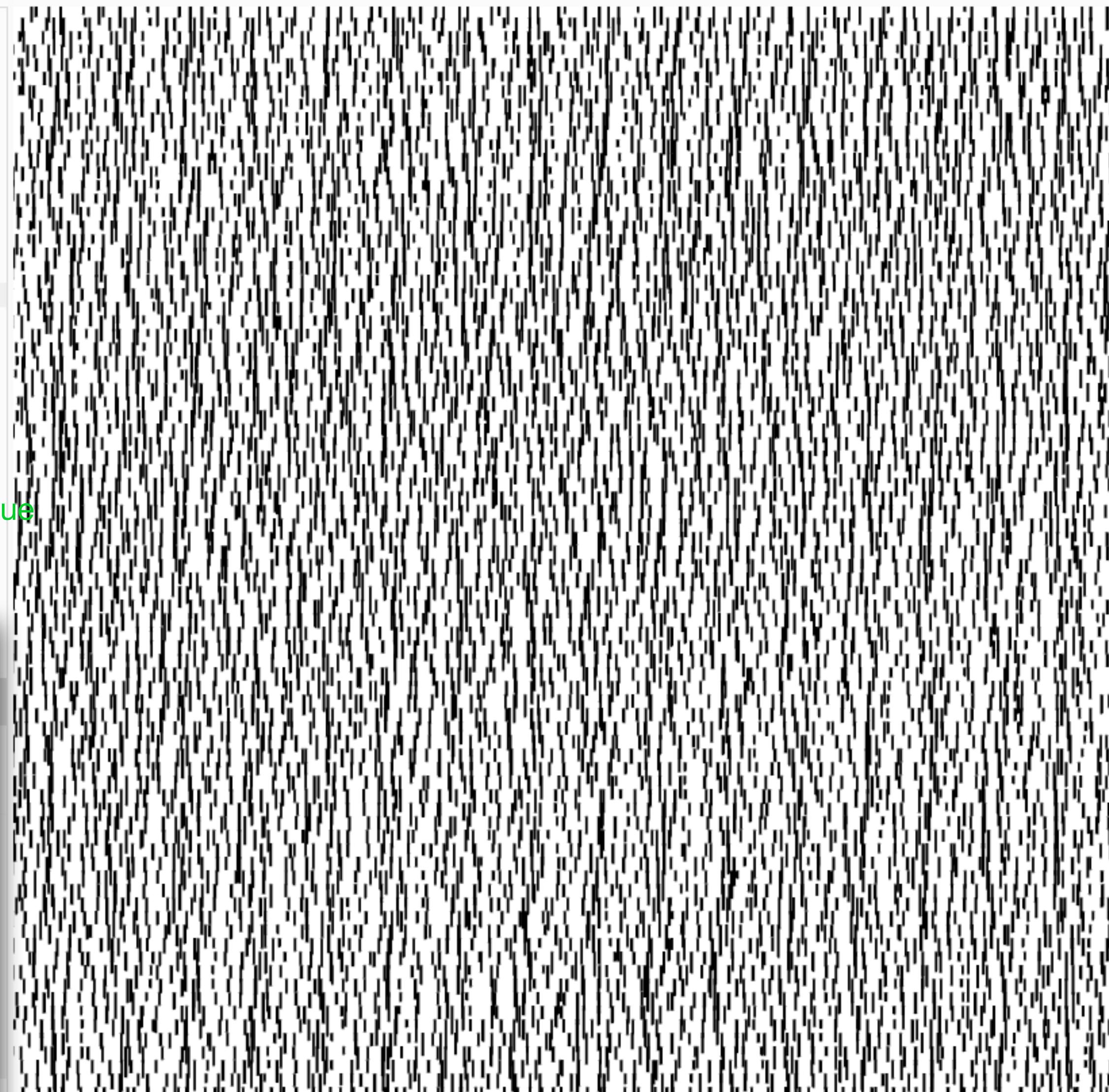
read temperature data from forecast.json

```
10 //read data from json
11 d3.json("forecast.json").then((data) => {
12   console.log(data);
13   temperature_data = data.hourly.temperature_2m;
14
15   for (let x = 0; x < windowHeight; x += 10) {
16     for (let y = 0; y < windowHeight; y += 10) {
17       let randomOffset = random(-5, 5);
18       line(x + randomOffset, y, x + randomOffset, y +
temperature_data[i%temperature_data.length]);
19       i++;
20     }
21   }
22 });
23
```

change length of vertical line based on temperature value

Saved: 2 minutes ago

Preview



Sketch Files ▾

< index.html

Saved: 1 minute ago

```
1 era5.json
2 forecast.json
3 index.html ▾
4 sketch.js
5 style.css
```

```
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/1.6.0/p5.js">
5   </script>
6     <script
src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/1.6.0/addons/p5.sound.min.j
s"></script>
7
8   <script src="https://cdn.jsdelivr.net/npm/d3-fetch@3"></script>
9   <link rel="stylesheet" type="text/css" href="style.css">
10  <meta charset="utf-8" />
```

add d3-fetch library in index.html

```
11
12
13
14
15  <script src="sketch.js"></script>
16
17
18
```

 Auto-refresh

Materialistic collision ↕ by janetyc



Sketch Files



sketch.js

Saved: just now

Preview

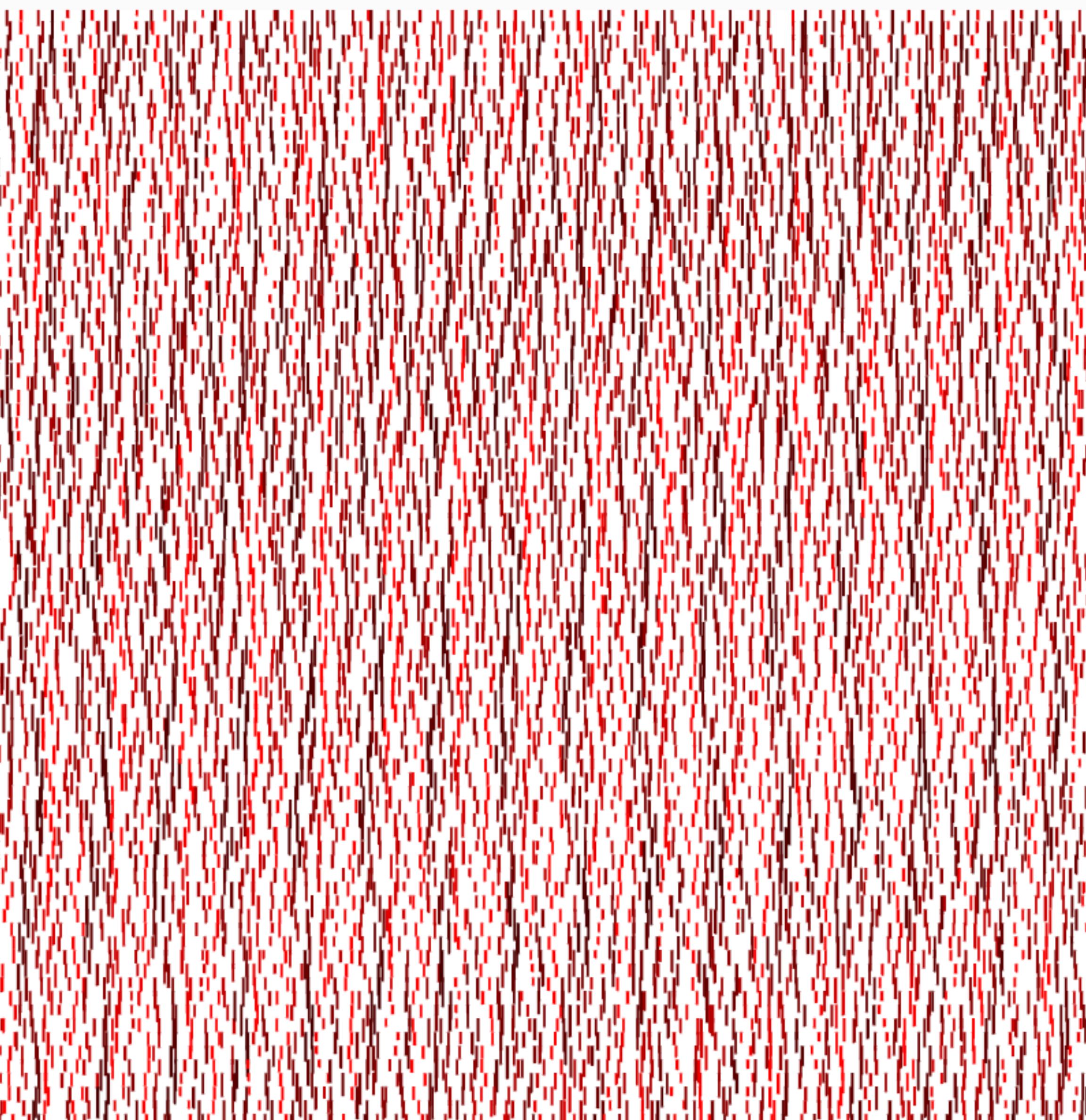
```
era5.json
forecast.json
index.html
sketch.js
style.css

1 let temperature_data;
2 let humidity_data;
3 function setup() {
4   createCanvas(windowWidth, windowHeight);
5   background(255);
6   strokeWeight(2)
7   i=0;
8
9   //read data from json
10  d3.json("forecast.json").then((data) => {
11    console.log(data);
12    temperature_data = data.hourly.temperature_2m;
13    humidity_data = data.hourly.relativehumidity_2m;
14
15   for (let x = 0; x < windowHeight; x += 10) {
16     for (let y = 0; y < windowHeight; y += 10) {
17       let randomOffset = random(-5, 5);
18       stroke(humidity_data[i%humidity_data.length]*3, 0, 0);
19       line(x + randomOffset, y, x + randomOffset, y +
temperature_data[i%temperature_data.length]);
20       i++;
21     }
22   }
23 });
24
25 }
```

Console

Clear ▾

```
▶ {latitude: 52.52, longitude: 13.419998, generationtime_ms: 0.73
20642471313477, utc_offset_seconds: 0, timezone: "GMT"...}
```



p5*

File ▾ Edit ▾ Sketch ▾ Help ▾

English ▾ Hello, janetyc! ▾



Auto-refresh

Materialistic collision by janetyc



Sketch Files



sketch.js*

Saved: 1 minute ago

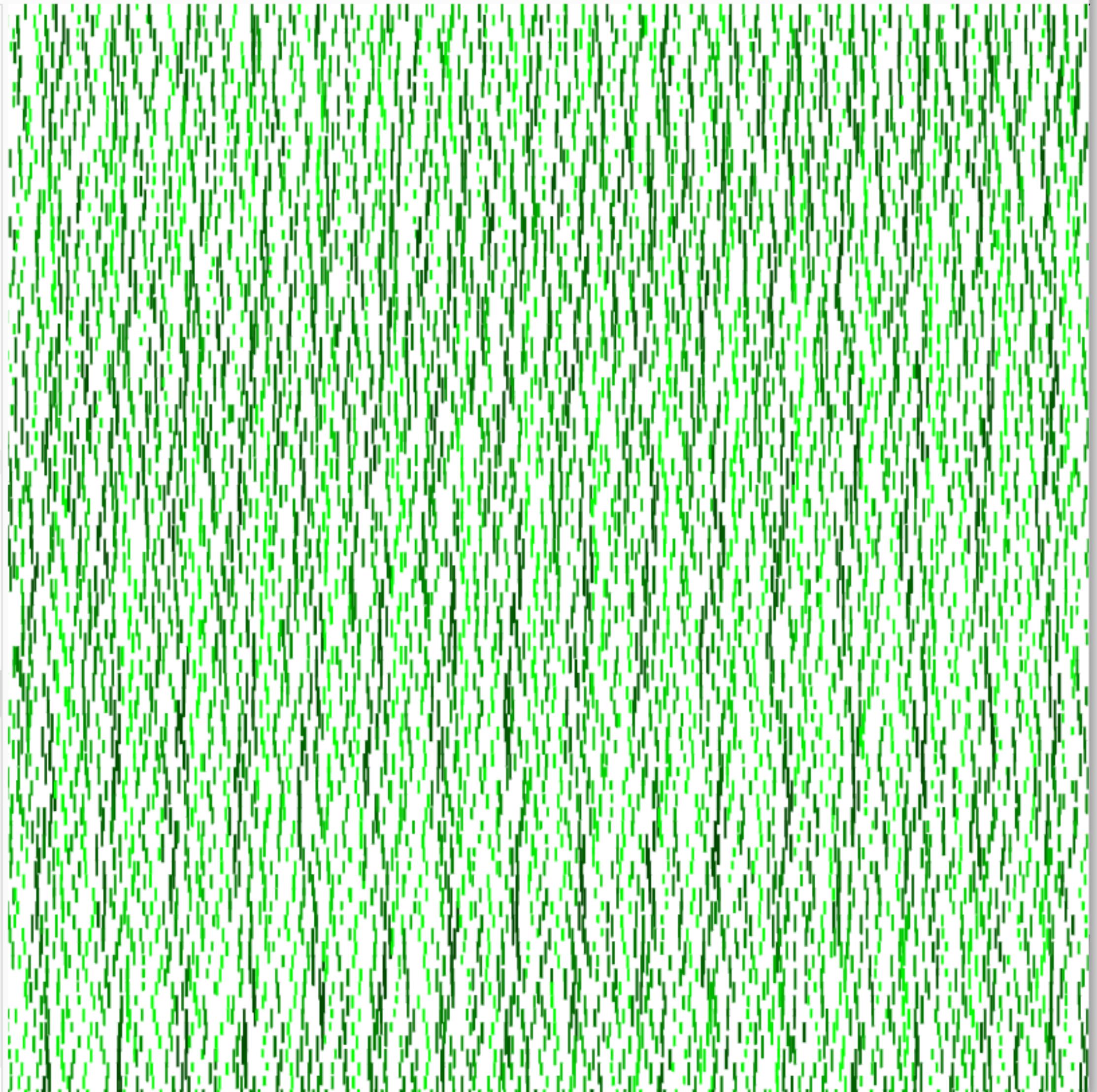
Preview

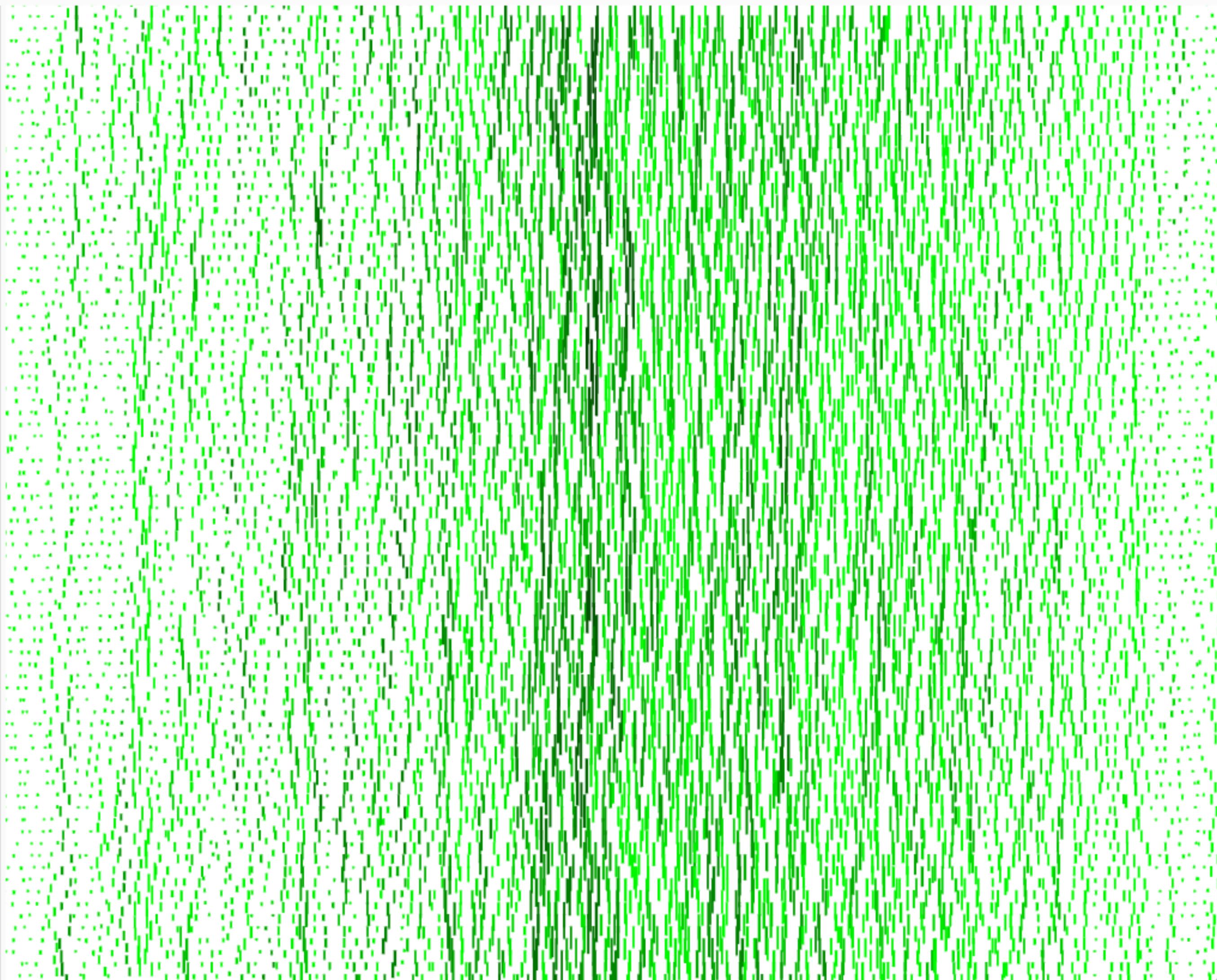
```
1 let temperature_data;
2 let humidity_data;
3 function setup() {
4   createCanvas(windowWidth, windowHeight);
5   background(255);
6   strokeWeight(2);
7   i=0;
8
9   //read data from json
10  d3.json("forecast.json").then((data) => {
11    console.log(data);
12    temperature_data = data.hourly.temperature_2m;
13    humidity_data = data.hourly.relativehumidity_2m;
14
15   for (let x = 0; x < windowHeight; x += 10) {
16     for (let y = 0; y < windowHeight; y += 10) {
17       let randomOffset = random(-5, 5);
18       stroke(0, humidity_data[i%humidity_data.length]*3, 0);
19       line(x + randomOffset, y, x + randomOffset, y +
temperature_data[i%temperature_data.length]);
20       i++;
21     }
22   }
23 });
24
25 }
```

Console

Clear ▾

```
▶ {latitude: 52.52, longitude: 13.419998, generationtime_ms: 0.73
20642471313477, utc_offset_seconds: 0, timezone: "GMT"...}
```





The image shows a p5.js sketch interface. On the left, the code editor displays `sketch.js` with the following content:

```
Sketch Files < sketch.js
Saved: 15 seconds ago Preview
era5.json
forecast.json
index.html
sketch.js
style.css

1 let temperature_data;
2 let humidity_data;
3 function setup() {
4   createCanvas(windowWidth, windowHeight);
5   background(255);
6   strokeWeight(2);
7   i=0;
8
9   //read data from json
10  d3.json("era5.json").then((data) => {
11    console.log(data);
12    temperature_data = data.hourly.temperature_2m;
13    humidity_data = data.hourly.relativehumidity_2m;
14
15    for (let x = 0; x < windowHeight; x += 10) {
16      for (let y = 0; y < windowHeight; y += 10) {
17        let randomOffset = random(-5, 5);
18        stroke(0, humidity_data[i%humidity_data.length]*3,
19          0);
20        line(x + randomOffset, y, x + randomOffset, y +
21          temperature_data[i%temperature_data.length]);
22        i++;
23      }
24    });
}

Console
▶ {latitude: 52.5, longitude: 13.400009, generationtime_ms:
0.800013542175293, utc_offset_seconds: 0, timezone: "GMT"...}
```

The code reads data from `era5.json` and uses it to draw a grid of green vertical lines on a black background. The lines are offset by a random value between -5 and 5 pixels. The color of each line is determined by the relative humidity at that specific location.

“Working with AI, data and patterns” Exercise (30 mins)

- Step1: Create DF project to upload the starboard notebook
 - Step2: Write a prompt to describe you “dear data” pattern and generate patterns through code
 - Step3: Read weather data and generate data pattern

Step 1: Setup

Notebook: Coding with ChatGPT.gg

Edit your notebook below, don't forget to save your changes. You can [access your datasets](#) with a bit of JavaScript or Python, depending on the notebook cell.

Step 2: Prompt-to-code

Session 0: Setup for using ChatGPT

Step 1: define request function for further POST call

In starboard, you can only use Pyodide to get data from the url. But, it is currently impossible to use `</>` not available in Pyodide. To make POST calls, you can use Web APIs in pyodide.

```
# code is borrowed from https://docs.pyascript.net/latest/guides/http-requests.html
from pyodide.http import pyfetch, FetchResponse
from typing import Optional, Any

async def request(url: str, method: str = "GET", body: Optional[str] = None,
                  headers: Optional[dict[str, str]] = None, **fetch_kwarg: Any) -> FetchResponse:
    """
    Async request function. Pass in Method and make sure to await!
    Parameters:
        url: str = URL to make request to
        method: str = {"GET", "POST", "PUT", "DELETE"} from `JavaScript` global fetch
        body: str = body as json string. Example, body=json.dumps(my_dict)
        headers: dict[str, str] = header as dict, will be converted to string...
            Example, headers=json.dumps({"Content-Type": "application/json"})
        fetch_kwarg: Any = any other keyword arguments to pass to `pyfetch` (will be passed to JavaScript's fetch)
    Return:
        response: pyodide.http.FetchResponse = use with .status or await.json(), etc.
    """
    kwargs = {"method": method, "mode": "cors"} # CORS: https://en.wikipedia.org/wiki/Cross-origin_resource_sharing
    if body and method not in ["GET", "HEAD"]:
        kwargs["body"] = body
    if headers:
        kwargs["headers"] = headers
    kwargs.update(fetch_kwarg)

    response = await pyfetch(url, **kwargs)
    return response
```

Python initialization complete

Step 2: Get your personal information from DF project for enabling OpenAI api

Copy the following information from your project and paste it to the following code.
(1).api.key

The screenshot shows the p5.js IDE interface. The left sidebar lists sketch files: era5.json, forecast.json, index.html, sketch.js (selected), and style.css. The main code editor contains the following sketch.js code:

```
function setup() {
  createCanvas(windowWidth, windowHeight);
  background(255);
  stroke(0);
  strokeWeight(2);

  for (let x = 0; x < windowWidth; x += 10) {
    for (let y = 0; y < windowHeight; y += 10) {
      let randomOffset = random(-5, 5);
      line(x + randomOffset, y, x + randomOffset, y + 10);
    }
  }
}

function draw() {
```

The right side of the interface shows a preview area displaying a black canvas with a grid of thin white lines, each shifted slightly in a random direction along its length.

Step 3: Read data to generate data pattern

(base) ~ ▶ curl "https://archive-api.open-meteo.com/v1/era5?latitude=52.52&longitude=13.41&start_date=2021-01-01&end_date=2021-12-31&hourly=temperature_2m"

```
[{"latitude":52.5,"longitude":13.400009,"generationtime_ms":0.39899349212646484,"utc_offset_seconds":0,"timezone":"GMT","timezone_abbr":"GMT","elevation":38.0,"hourly_units":{"time":"iso8601","temperature_2m":"°C"},"hourly":[{"time":["2021-01-01T00:00","2021-01-01T01:00","2021-01-01T02:00","2021-01-01T03:00","2021-01-01T04:00","2021-01-01T05:00","2021-01-01T06:00","2021-01-01T07:00","2021-01-01T08:00","2021-01-01T09:00","2021-01-01T10:00","2021-01-01T11:00","2021-01-01T12:00","2021-01-01T13:00","2021-01-01T14:00","2021-01-01T15:00","2021-01-01T16:00","2021-01-01T17:00","2021-01-01T18:00","2021-01-01T19:00","2021-01-01T20:00","2021-01-01T21:00","2021-01-01T22:00","2021-01-01T23:00","2021-01-02T00:00","2021-01-02T01:00","2021-01-02T02:00","2021-01-02T03:00","2021-01-02T04:00","2021-01-02T05:00","2021-01-02T06:00","2021-01-02T07:00","2021-01-02T08:00","2021-01-02T09:00","2021-01-02T10:00","2021-01-02T11:00","2021-01-02T12:00","2021-01-02T13:00","2021-01-02T14:00","2021-01-02T15:00","2021-01-02T16:00","2021-01-02T17:00","2021-01-02T18:00","2021-01-02T19:00","2021-01-02T20:00","2021-01-02T21:00","2021-01-02T22:00","2021-01-03T00:00","2021-01-03T01:00","2021-01-03T02:00","2021-01-03T03:00","2021-01-03T04:00","2021-01-03T05:00","2021-01-03T06:00","2021-01-03T07:00","2021-01-03T08:00","2021-01-03T09:00","2021-01-03T10:00","2021-01-03T11:00","2021-01-03T12:00","2021-01-03T13:00","2021-01-03T14:00","2021-01-03T15:00","2021-01-03T16:00","2021-01-03T17:00","2021-01-03T18:00","2021-01-03T19:00","2021-01-03T20:00","2021-01-03T21:00","2021-01-03T22:00","2021-01-04T00:00","2021-01-04T01:00","2021-01-04T02:00","2021-01-04T03:00","2021-01-04T04:00","2021-01-04T05:00","2021-01-04T06:00","2021-01-04T07:00","2021-01-04T08:00","2021-01-04T09:00","2021-01-04T10:00","2021-01-04T11:00","2021-01-04T12:00","2021-01-04T13:00","2021-01-04T14:00","2021-01-04T15:00","2021-01-04T16:00","2021-01-04T17:00","2021-01-04T18:00","2021-01-04T19:00","2021-01-04T20:00","2021-01-04T21:00","2021-01-04T22:00","2021-01-04T23:00","2021-01-05T00:00","2021-01-05T01:00","2021-01-05T02:00","2021-01-05T03:00","2021-01-05T04:00","2021-01-05T05:00","2021-01-05T06:00","2021-01-05T07:00","2021-01-05T08:00","2021-01-05T09:00","2021-01-05T10:00","2021-01-05T11:00","2021-01-05T12:00","2021-01-05T13:00","2021-01-05T14:00","2021-01-05T15:00","2021-01-05T16:00","2021-01-05T17:00","2021-01-05T18:00","2021-01-05T19:00","2021-01-05T20:00","2021-01-06T00:00","2021-01-06T01:00","2021-01-06T02:00","2021-01-06T03:00","2021-01-06T04:00","2021-01-06T05:00","2021-01-06T06:00","2021-01-06T07:00","2021-01-06T08:00","2021-01-06T09:00","2021-01-06T10:00","2021-01-06T11:00","2021-01-06T12:00","2021-01-06T13:00","2021-01-06T14:00","2021-01-06T15:00","2021-01-06T16:00","2021-01-06T17:00","2021-01-06T18:00","2021-01-06T19:00","2021-01-06T20:00","2021-01-06T21:00","2021-01-06T22:00","2021-01-06T23:00","2021-01-07T00:00","2021-01-07T01:00","2021-01-07T02:00","2021-01-07T03:00","2021-01-07T04:00","2021-01-07T05:00","2021-01-07T06:00","2021-01-07T07:00","2021-01-07T08:00","2021-01-07T09:00","2021-01-07T10:00","2021-01-07T11:00","2021-01-07T12:00","2021-01-07T13:00","2021-01-07T14:00","2021-01-07T15:00","2021-01-07T16:00","2021-01-07T17:00","2021-01-07T18:00","2021-01-07T19:00","2021-01-07T20:00","2021-01-07T21:00","2021-01-07T22:00","2021-01-07T23:00","2021-01-08T00:00","2021-01-08T01:00","2021-01-08T02:00","2021-01-08T03:00","2021-01-08T04:00","2021-01-08T05:00","2021-01-08T06:00","2021-01-08T07:00","2021-01-08T08:00","2021-01-08T09:00","2021-01-08T10:00","2021-01-08T11:00","2021-01-08T12:00","2021-01-08T13:00","2021-01-08T14:00","2021-01-08T15:00","2021-01-08T16:00","2021-01-08T17:00","2021-01-08T18:00","2021-01-08T19:00","2021-01-08T20:00","2021-01-08T21:00","2021-01-08T22:00","2021-01-09T00:00","2021-01-09T01:00","2021-01-09T02:00","2021-01-09T03:00","2021-01-09T04:00","2021-01-09T05:00","2021-01-09T06:00","2021-01-09T07:00","2021-01-09T08:00","2021-01-09T09:00","2021-01-09T10:00","2021-01-09T11:00","2021-01-09T12:00","2021-01-09T13:00","2021-01-09T14:00","2021-01-09T15:00","2021-01-09T16:00","2021-01-09T17:00","2021-01-09T18:00","2021-01-09T19:00","2021-01-09T20:00","2021-01-09T21:00","2021-01-09T22:00","2021-01-10T00:00","2021-01-10T01:00","2021-01-10T02:00","2021-01-10T03:00","2021-01-10T04:00","2021-01-10T05:00","2021-01-10T06:00","2021-01-10T07:00","2021-01-10T08:00","2021-01-10T09:00","2021-01-10T10:00","2021-01-10T11:00","2021-01-10T12:00","2021-01-10T13:00","2021-01-10T14:00","2021-01-10T15:00","2021-01-10T16:00","2021-01-10T17:00","2021-01-10T18:00","2021-01-10T19:00","2021-01-10T20:00","2021-01-10T21:00","2021-01-10T22:00","2021-01-11T00:00","2021-01-11T01:00","2021-01-11T02:00"}]
```

English ▾ Hello, janetycl !

p5' File ▾ Edit ▾ Sketch ▾ Help ▾

Auto-refresh Materialistic collision by janetycl

Sketch Files < sketch.js Saved: 15 seconds ago Preview

era5.json forecast.json index.html sketch.js style.css

```
let temperature_data;
let humidity_data;
function setup() {
  createCanvas(windowWidth, windowHeight);
  background(255);
  strokeWeight(2);
  i=0;
  //read data from json
  d3.json("era5.json").then((data) => {
    console.log(data);
    temperature_data = data.hourly.temperature_2m;
    humidity_data = data.hourly.relativehumidity_2m;
    for (let x = 0; x < windowWidth; x += 10) {
      for (let y = 0; y < windowHeight; y += 10) {
        let randomOffset = random(-5, 5);
        stroke(0, humidity_data[i%humidity_data.length]*3,
        0);
        line(x + randomOffset, y, x + randomOffset, y +
        temperature_data[i%temperature_data.length]);
        i++;
      }
    }
  });
}
```

Console Clear

latitude: 52.5, longitude: 13.400009, generationtime_ms: 0.800013542175293, utc_offset_seconds: 0, timezone: "GMT"...

Data Foundry x OpenAI API



Hey, you are interested in using GPT and soon chatGPT models in your design prototyping? Right on!

Thanks already for filling this form. We will credit your API key with extra credits as a small 🎉.

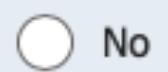
Hi, Janet. When you submit this form, the owner will see your name and email address.

* Required

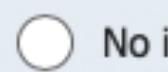
1. Have you used GPT models or chatGPT before? *



Yes

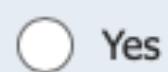


No

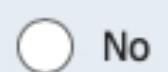


No idea

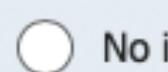
2. Have you used Data Foundry before in your design (research) projects? *



Yes



No



No idea

Submit

This content is created by the owner of the form. The data you submit will be sent to the form owner. Microsoft is not responsible for the privacy or security practices of its customers, including those of this form owner. Never give out your password.

Powered by Microsoft Forms | [Privacy and cookies](#) | [Terms of use](#)

Need more extra API credits?

<https://forms.office.com/e/entYh9Pjs6>

Technical support from TU/e ID

TU/e ID Tech Support [Link]

The screenshot shows a Microsoft Teams interface. On the left, there's a sidebar with icons for Activity, Chat, Teams (selected), Assignments, Calendar, Files, Calls, and more. The main area is a channel named "Data Foundry help". The channel header includes the TU/e ID logo, a search bar, and a profile picture for TU Eindhoven.

Data Foundry help Posts Files +

2 replies from Peter and Mathias

Reply

April 9, 2021

Jansen, Anniek 4/8 12:30 PM
Hello, I am trying to upload pdf files in a dataset (type existing) on Data Foundry and while this worked before, I now receive the error: Only text, image or audio files allowed. Word files also don't work but images do work. The dataset is still active. Does someone know what might go wrong?

2 replies from Mathias

Reply

April 12, 2021

Funk, Mathias 4/9 9:14 PM
Data Foundry feature updates: auto-transcription + RAWgraphs integration thumb up 2
hey all, happy to announce two really nice new features:

(1) with **auto-transcription**, you can upload audio and video files to Data Foundry and the speech in English and Dutch will be extracted automatically. This is great for user interviews and can save you some hours of typing. Don't forget that there is machine learning behind this so the results need to be checked. The whole thing is built to be GDPR safe. [See more](#)

Heuvel, Roy van den 4/12 9:32 AM heart 1
Very nice! I love the RAWgraph feature! Very handy for quick dataviz! I'll give the transcription function a go soon as well, that might be a huge timesaver for students who do interviews etc. The OCR feature might also be great for people using note-taking or diary-style user-input. Great work! 😊

Reply

Want to know more?

- You can use the DF to collect your movement data via mobile phones, collect sensor data via smart things, track your web activities, and so on.
- Please check the use cases in the Data foundry website ([https://data.id.tue.nl/
documentation/use-cases](https://data.id.tue.nl/documentation/use-cases))

TU/e ID Tech Support

The screenshot shows the homepage of the Data Foundry website. On the left is a sidebar with links: Login, Register, Community, Explore, Guides, and Support. The main content area has a header with a search icon, 'Register', and 'Login'. Below the header, a welcome message says: 'Welcome! This is Data Foundry @ ID Eindhoven! With this service you can collect data from a variety of sources and store data in a unified format. You can then export data for analysis or stream back in real-time into prototypes, products and systems.' It also mentions a documentation site, a development blog, and support. A section for open projects shows three cards: 'TEMP_HUMIDITY_HO...', 'SMARTPHONE USAGE I...', and 'TESTING EXAMPLE CO...'. Each card includes a lock icon, the project name, a brief description, the creator's name, and a grid of data source icons.

Welcome! This is Data Foundry @ ID Eindhoven!

With this service you can collect data from a variety of sources and store data in a unified format. You can then export data for analysis or stream back in real-time into prototypes, products and systems.

We have a documentation site where you can find more. [This way](#). Find out more about the project progress in the [development blog](#). If you need support, you have questions or feedback, head over to [support](#).

Find a few open projects below. Sign-in or register to get your own thing going.

TEMP_HUMIDITY_HO...

Study on how air quality affects you sleep pat...

Eva van der Born

1/1 0/1 2 ENTITY EXISTING
FORM FITBIT ANNOTATION DIARY IOT
MEDIA

SMARTPHONE USAGE I...

This project focuses on smarphone usage in th...

Marco Peters

2/6 4 IOT

TESTING EXAMPLE CO...

This project consists of getting acquainted with ...

Pablo Traversat

3/4 10 ENTITY ANNOTATION DIARY
IOT MEDIA

Any problems related to Data Foundry or other technical issues, please contact [Eden Chiang \(@d.search lab\)](#) i.chiang@tue.nl

Recap

- What is data?
- Data as a creative material
- “Dear Data” exercise
- “Data Digitalization” exercise
- “Working with AI” exercise
- Your task: working with AI to explore data pattern
 - Step 1: collect your data (small data or big data)
 - Step 2: sketching/designing your data
 - Step 3: data digitalization
 - Step 4: working with AI, data and pattern

Q&A

Thank you!