cd ~/Desktop/BroadStreet_InFlux && python3 - <<'PY' import os, csv, json, time, urllib.parse, urllib.request, subprocess, textwrap from PIL import Image from reportlab.lib.pagesizes import landscape, letter from reportlab.pdfgen import canvas

PROJECT = os.getcwd() OUTPUT = os.path.join(PROJECT, "output") os.makedirs(OUTPUT, exist_ok=True)

photographers = { "dante": "Dante Sisofo", "dylan": "Dylan Stone" }

all_rows = [] photos_for_zine = []

def reverse_geocode(lat, lon): url = "https://nominatim.openstreetmap.org/reverse?" + urllib.parse.urlencode({ "format": "json", "lat": lat, "lon": lon, "zoom": 18, "addressdetails": 1 }) req = urllib.request.Request(url, headers={"User-Agent": "BroadStreetInFlux/1.0"}) try: with urllib.request.urlopen(req, timeout=20) as r: data = json.loads(r.read().decode("utf-8")) return data.get("display_name", "") except Exception as e: return f"ADDRESS ERROR: {e}"

for folder, photographer in photographers.items(): photo_dir = os.path.join(PROJECT, folder, "photos") if not os.path.isdir(photo_dir): continue

```
files = sorted([
    os.path.join(photo_dir, f)
    for f in os.listdir(photo_dir)
    if f.lower().endswith((".jpg", ".jpeg"))
])

if not files:
    continue

cmd = [
    "exiftool", "-json", "-n",
    "-FileName", "-DateTimeOriginal", "-CreateDate",
    "-GPSLatitude", "-GPSLongitude"
] + files
```

```
metadata = json.loads(subprocess.check_output(cmd).decode("utf-
8"))

for item in metadata:
    filename = item.get("FileName", "")
    path = os.path.join(photo_dir, filename)
    date_time = item.get("DateTimeOriginal") or
item.get("CreateDate") or ""
    lat = item.get("GPSLatitude", "")
    lon = item.get("GPSLongitude", "")

    address = ""
    if lat and lon:
        address = reverse_geocode(lat, lon)
        time.sleep(1)

    caption = f"{date_time} — {address} — {photographer}"

    row = {
        "Title": filename,
        "Photographer": photographer,
        "DateTime": date_time,
        "Address": address,
        "Caption": caption,
        "Latitude": lat,
        "Longitude": lon
    }

    all_rows.append(row)
    photos_for_zine.append((path, caption))

    print(filename, "->", address)
```

csv_path = os.path.join(OUTPUT, "broad-street-in-flux-google-my-maps.csv") with open(csv_path, "w", newline="", encoding="utf-8") as f: fieldnames = ["Title", "Photographer", "DateTime", "Address", "Caption", "Latitude", "Longitude"] writer = csv.DictWriter(f, fieldnames=fieldnames) writer.writeheader() writer.writerows(all_rows)

pdf_path = os.path.join(OUTPUT, "broad-street-in-flux-captioned-zine.pdf")

page_w, page_h = landscape(letter) margin = 36 caption_area = 95 image_box_w = page_w - margin * 2 image_box_h = page_h - margin * 2 - caption_area

c = canvas.Canvas(pdf_path, pagesize=landscape(letter))

for path, caption in photos_for_zine: img = Image.open(path) w, h = img.size

```
scale = min(image_box_w / w, image_box_h / h)
draw_w = w * scale
draw_h = h * scale

img_x = (page_w - draw_w) / 2
img_y = margin + caption_area + (image_box_h - draw_h) / 2

c.drawImage(path, img_x, img_y, width=draw_w, height=draw_h)

c.setFont("Courier", 9)
wrapped = textwrap.wrap(caption, width=95)
y = margin + 55

for line in wrapped[:5]:
    c.drawCentredString(page_w / 2, y, line)
    y -= 12

c.showPage()
```

c.save()

print("\nDONE") print("CSV:", csv_path) print("ZINE:", pdf_path) PY