Yes — same exact setup for Crawl4AI on Render free tier.
Unlimited private web crawling (markdown, JSON, screenshots, structured data, deep crawls, etc.) for your AI agents, no API costs, full privacy.

Crawl4AI complements your SearXNG perfectly — use SearXNG for search, Crawl4AI for deep page scraping.

1. Create One GitHub Repo (`crawl4ai-render`)

Go to GitHub → New repo → Name: `crawl4ai-render`

Add these **3 files**:

`Dockerfile`

```
dockerfile
FROM unclecode/crawl4ai:latest

# Render free tier fixes (dynamic $PORT + no rate limits)
COPY config.yml /app/config.yml
COPY entrypoint.sh /entrypoint.sh

RUN chmod +x /entrypoint.sh && \
    apt-get update && apt-get install -y sed && rm -rf
/var/lib/apt/lists/*

ENTRYPOINT ["/entrypoint.sh"]
```

`entrypoint.sh`

```bash
#!/bin/sh
set -e

# Make it listen on Render's dynamic $PORT
sed -i "s/port: 11235/port: ${PORT}/g" /app/config.yml

echo "🚀 Crawl4AI starting on port $PORT (Render free tier)"

# Run the official server
exec "$@"
```

`config.yml` (maxed out for unlimited + free tier)

```yaml
app:
  host: "0.0.0.0"
  port: 11235   # ← Will be auto-changed to $PORT by entrypoint

llm:
  provider: "openai/gpt-4o-mini"   # Change to your preferred (or leave
blank)

security:
  enabled: false                    # No auth for personal use
  jwt_enabled: false

rate_limiting:
  enabled: false                    # ← UNLIMITED CALLS

crawler:
  memory_threshold_percent: 85.0    # Gentle on free tier RAM
  timeouts:
    stream_init: 60.0
    batch_process: 600.0

# Optional: Enable everything
browser:
  pool_size: 2                      # Keep light for free tier
  headless: true
```

Commit & push.

2. Deploy on Render (Free)

1. Go to dashboard.render.com → **New** → **Web Service**
2. Connect your `crawl4ai-render` repo
3. **Name**: `crawl4ai` (or anything)
4. **Branch**: main
5. **Runtime**: **Docker**
6. **Instance Type**: **Free**
7. **Environment Variables** (add these):
    - `OPENAI_API_KEY` → your key (for LLM extraction, optional)
    - `ANTHROPIC_API_KEY` → optional
    - Any other LLM keys you use
8. Click **Create Web Service**

Wait 3–5 minutes for build (it pulls the official image + applies our tweaks).

3. First-Time Fix (Important)

After first deploy:

- Go to your service → **Environment** tab
- Update any env vars if needed
- Click **Redeploy**

4. Test It

- **Web UI (Playground)**: `https://crawl4ai-xxx.onrender.com/playground`
- **Health**: `https://crawl4ai-xxx.onrender.com/health`
- **Simple crawl** (via curl):

```
Bash

curl -X POST https://crawl4ai-xxx.onrender.com/crawl \
  -H "Content-Type: application/json" \
  -d '{"urls": ["https://example.com"], "extract": {"type":
"markdown"}}'
```

- **MCP ready** (for later): `/mcp/sse`

Step 11: Keep It Awake (Free UptimeRobot)

Same as SearXNG:

- Add monitor for `https://your-crawl4ai.onrender.com/health` → every 5 min

Free Tier Notes (Same as SearXNG)

- 750 hours/month → fine for one service
- Sleeps after 15 min → UptimeRobot prevents it
- 512MB RAM → good for personal crawls (deep crawls may be slower)
- No persistent storage → configs baked in (edit `config.yml` → push → redeploy)

Done in 10 minutes.

Your stack is now god-tier:

- SearXNG → unlimited search
- Crawl4AI → unlimited deep crawling

When you want to add MCP integration later, just say the word — 2-minute setup.

Ping me with the Render URL when live, I'll test a heavy crawl for you.