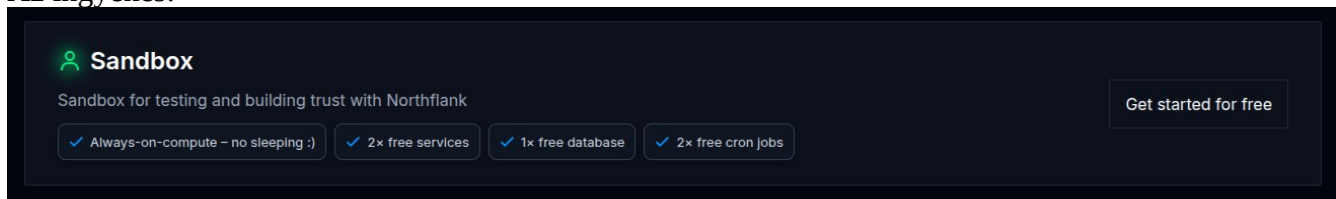


A projektet saját Githubon hoztam létre publikus repoban, külön brancheken. Funkcionalitásban azonosak. A Northflank és Cloudflare fiókok szintén saját adatokkal jöttek létre.

Northflank

Link: <https://p01--frontend--pxyfv22r4grv.code.run/>

Az ingyenes:



Regisztráció

Egyszerű folyamat. Bankkártya megadása szükséges, de ez csak akkor derült ki mikor már bekonfiguráltam az adatbázist és csak rá kellett nyomni a “Létrehozás”ra

Projekt létrehozása & Első deploy

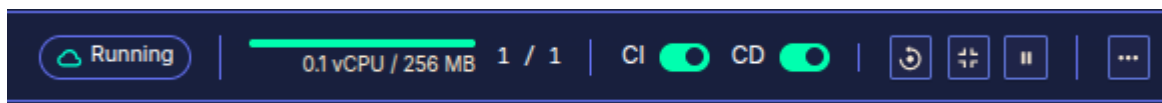
Nem volt nehéz:

a backend és frontend 1-1 külön service, az adatbázis egy 'addon', ezzel ki is van maxolva az ingyenes rész a 2 cron job-on kívül. Tartós fájlrendszer nincs ingyen, ezért a “kiskutya” a backend konténer része, backend/data/message.txt-ből jön. Github repo és branch megadás, mindenhez saját beállítás, pl network, build/runtime variables stb. A legtöbb helyre jó volt az alapértelmezett.

További deploy

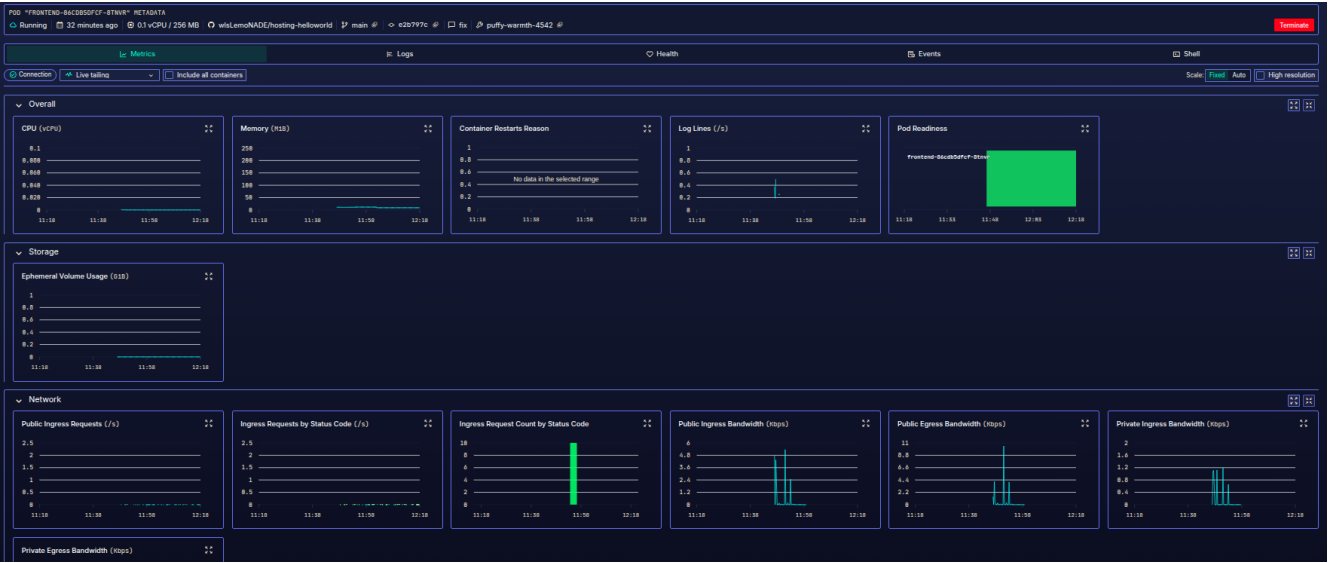
Auto CI/CD-hez össze kell kötni a Northflanket és a git szolgáltatót, Gitlab is támogatva van, de azzal nem próbáltam. A Githubos megoldás simán ment.

Az auto CI/CD könnyen kikapcsolható service-nként



Nyomon követés

Egyszerű felületen viszonylag sok metrika látható működés közben



A logokhoz egyszerűen, akár 2 kattintásból el lehet jutni, lehet bennük keresni, legutóbbi x időre szűrni

```
POD "BACKEND-767DC40B76-SH58V" METADATA
Running 44 minutes ago 0.1 vCPU / 256 MB wslLemoNADE/hosting-helloworld main e2b797c fix Kingly-nature-2089

Metrics Logs Health Events Shell
Connection Live tailing Search logs [ABC] Returned 28 lines

20 JUL 11:38:48.824 2026-07-20T11:38:47.827582531Z stdout F Filesystem message: /app/data/message.txt
20 JUL 11:38:48.824 2026-07-20T11:38:47.827575376Z stdout F Visit counter: primary.main-db--pxyfv22r4grv.addon.code.run:5432/helloworld_db
20 JUL 11:38:48.824 2026-07-20T11:38:47.826867219Z stdout F Backend listening on port 3808
20 JUL 11:38:48.824 2026-07-20T11:38:47.822246884Z stdout F Postgres schema ready
20 JUL 11:38:47.821 2026-07-20T11:38:46.902704212Z stderr F (Use 'node --trace-warnings ...' to show where the warning was created)
20 JUL 11:38:47.821 2026-07-20T11:38:46.902701982Z stderr F See https://www.postgresql.org/docs/current/libpq-ssl.html for libpq SSL mode definitions.
20 JUL 11:38:47.821 2026-07-20T11:38:46.902695163Z stderr F
20 JUL 11:38:47.821 2026-07-20T11:38:46.90269317Z stderr F - If you want libpq compatibility now, use 'useLibpqCompat=true&sslmode=require'
20 JUL 11:38:47.821 2026-07-20T11:38:46.902698497Z stderr F - If you want the current behavior, explicitly use 'sslmode=verify-full'
20 JUL 11:38:47.821 2026-07-20T11:38:46.902686745Z stderr F To prepare for this change:
20 JUL 11:38:47.821 2026-07-20T11:38:46.902683663Z stderr F
20 JUL 11:38:47.821 2026-07-20T11:38:46.902679626Z stderr F In the next major version (pg-connection-string v3.0.0 and pg v9.8.0), these modes will adopt standard libpq semantics, which have weaker security guarantees.
20 JUL 11:38:47.821 2026-07-20T11:38:46.902638869Z stderr F (node:15) Warning: SECURITY WARNING: The SSL modes 'prefer', 'require', and 'verify-ca' are treated as aliases for 'verify-full'.
20 JUL 11:38:42.811 2026-07-20T11:38:41.814839842Z stdout F
20 JUL 11:38:42.811 2026-07-20T11:38:41.814836489Z stdout F > node src/index.js
20 JUL 11:38:42.811 2026-07-20T11:38:41.814832482Z stdout F > hosting-helloworld-backend@1.0.0 start
20 JUL 11:38:42.811 2026-07-20T11:38:41.813994828Z stdout F
20 JUL 11:38:41.258 2026-07-20T11:38:36.221857531Z stdout F [2026-07-20T11:38:36Z INFO ] Starting container entrypoint...
20 JUL 11:38:41.258 2026-07-20T11:38:36.218769682Z stdout F [2026-07-20T11:38:36Z INFO ] Successfully fetched environment variables
20 JUL 11:38:41.258 2026-07-20T11:38:36.085948415Z stdout F [2026-07-20T11:38:36Z INFO ] Securely fetching environment variables
```

Van shell hozzáférés a fájlokhoz, kis negatívum hogyha átkattintunk pl a Logs-ra majd vissza akkor az ssh ablak reset-el



```
POD "BACKEND-767DC4DB76-SM58V" METADATA
Running an hour ago 0.1 vCPU / 256 MB wisLemoNADE/hosting-helloworld main e2b797c fix kingly-nature-2089 [Terminate]

Metrics Logs Health Events Shell
Connecting... Connected Help
Authenticated successfully!
Connected to instance "backend-767dc4db76-sm58v" (35e4661e-18d4-457a-9648-115c8a7ee5cb).
Use Ctrl+Shift+C to copy, Ctrl+V to paste.
ls
data node_modules package-lock.json package.json src
/app # cd data
/app/data # ls
message.txt
/app/data # cat message.txt
kiskutya
/app/data #
```

Vélemény

A felület egyszerűnek bizonyult, mindent megtaláltam benne amire (szerintem) szükség lehet. Nem okozott gondot a navigáció annak ellenére, hogy minden oldalon sok információ és lehetőség van egyszerre. Az alap domain nem túl bizalomgerjesztő.

Cloudflare

Link: <https://hosting-helloworld.pages.dev/>

Regisztráció

Egyszerű email+jelszó pár, bankkártya nem kellett.

Projekt létrehozása & Első deploy

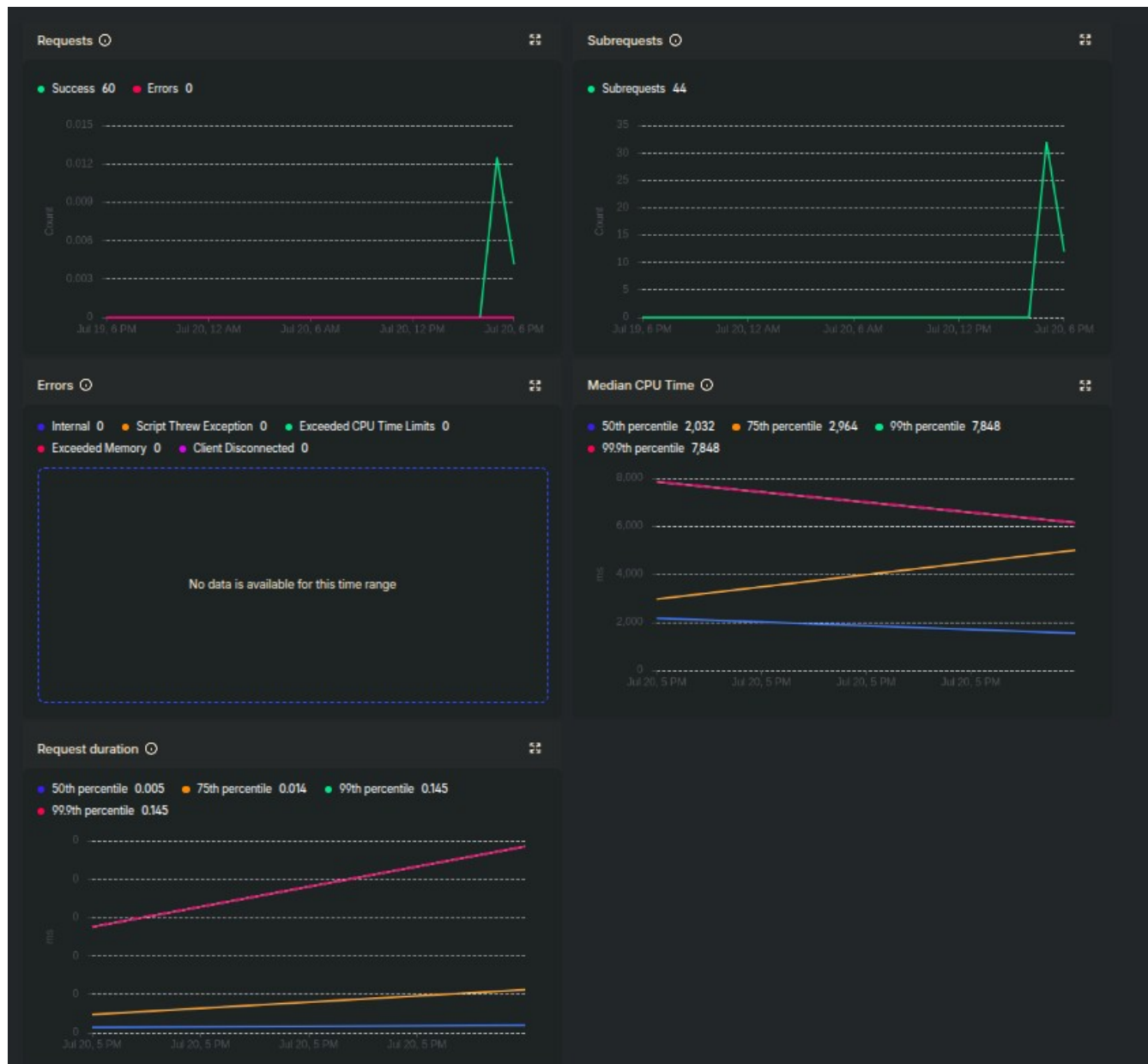
Kicsit nehezebb volt eligazodni azon, hogy mit is keresek, de összességében jóval gyorsabb volt mint a Northflank, bár ez inkább annak köszönhető, hogy a 80%-át megtudta csinálni a cursor. A projekt azonos repoban külön branchen módosításra került, mivel bizonyos részei máshogy működnek cloudflaren, vagy nem lett volna ingyenes. Nincs külön frontend/backend hanem egy közös Pages project van A kiskutya itt is projekten belüli fájlból jön, de ha minden igaz R2 Object Storage vagy worker Key-Value val is meg lehetne oldani ingyen.

További deploy

Auto CI/CD-hez össze kell kötni a Cloudflaret és a git szolgáltatót, Gitlab is támogatva van, de azzal nem próbáltam. A Githubos megoldás simán ment. Szintén kikapcsolható az auto CI/CD, de sokkal több kattintásba kerül.

Nyomon követés

Itt is vannak metrikák, bár sokkal kevesebb, projekt és account szinten is.



A logokhoz elég nehéz eljutni hacsak nincs bent a legutóbbi elemek között, illetve nem találtam hozzá funkciókat mint pl keresés/szűrés. Vannak extra log manager részec, de azok különálló modulok és pénzért.

Build logAssets uploadedFunctionsRedirectsHeaders

Build log

✓ initialize4s

✓ clone repo2s

✓ build4s

✓ deploy10s

Copy logDownload log

7-----
2026-07-20T16:11:06.83799ZUsing v2 root directory strategy
82026-07-20T16:11:06.858486ZSuccess: Finished cloning repository files
92026-07-20T16:11:08.929677ZChecking for configuration in a Wrangler configuration file (BETA)
102026-07-20T16:11:08.930253Z
112026-07-20T16:11:08.930953ZFound wrangler.toml file. Reading build configuration...
122026-07-20T16:11:08.939009Zpages_build_output_dir: public
132026-07-20T16:11:08.9392ZBuild environment variables: (none found)
142026-07-20T16:11:09.112564ZSuccessfully read the Wrangler configuration file.
152026-07-20T16:11:09.113327ZNo build command specified. Skipping build step.
162026-07-20T16:11:09.113933ZFound Functions directory at /functions. Uploading.
172026-07-20T16:11:09.121341Z🔥 wrangler 3.114.17
182026-07-20T16:11:09.121517Z-----
192026-07-20T16:11:10.059036Z🌟 Compiled Worker successfully
202026-07-20T16:11:10.179468ZFound _routes.json in output directory. Uploading.
212026-07-20T16:11:10.190294ZValidating asset output directory
222026-07-20T16:11:11.680688ZDeploying your site to Cloudflare's global network...
232026-07-20T16:11:16.863655ZUploading... (1/2)
242026-07-20T16:11:17.37712ZUploading... (2/2)
252026-07-20T16:11:17.378607Z🌟 Success! Uploaded 1 files (1 already uploaded) (0.82 sec)
262026-07-20T16:11:17.783321Z🌟 Upload complete!
272026-07-20T16:11:19.571331ZSuccess: Assets published!
282026-07-20T16:11:21.254305ZSuccess: Your site was deployed!
29
30

Nincs ssh access, nincs meg minden fájl egy helyen áttekintés szinten sem.

Vélemény

Bár jóval hamarabb “beindult” sokkal kevésbé értettem meg, hogy hogyan is tette és hogyan működik, nagyon szokatlan és eltérő az eddigiekhez képest, főleg a docker hiánya és a helyette lévő más felépítés miatt (Functions, public/, etc.) A felület nekem nehezebben kezelhető, kevésbé intuitív és jobban keresgélős, amin beépített ai chatte próbálnak segíteni. Széleskörű szolgáltatások, de csak külön modulokként, amiknek vagy van ingyen része vagy nincs.

Működtetésbeli különbségek

	Northflank	Cloudflare Pages + Functions
Process model	Always-on containers	On-demand isolates (wake per request, de <1s ahogy néztem)
How many “services”	Separate frontend + backend (+ DB)	One project; routes decide static vs Function
Networking	Private DNS (backend:3000), proxy in nginx	Same origin; /api/* is just Functions
Runtime	Full Node + Linux filesystem	Workers runtime (limited Node APIs)
Scaling	Scale containers	Cloudflare scales Functions automatically
Cost shape	Pay for running services/DB even idle	Free tier: pay mainly for requests/DB ops

Teljesítménybeli különbségeket nem tudtam tesztelni.

Előnyök, hátrányok

Northflank	
Pros	Cons
Megszokott model/működtetés	A több különálló rész baj is lehet
Egyértelmű elválasztás komponensek között	Ha fizetésre kerül a sor az idle időért is fizetni kell (Nem tudom, hogy többbe kerülne-e, mint a cf)
A backend nem kell hogy publikusan elérhető legyen	Kötelező bankkártya
Hamar megszokható kezelhető felület	Az alap domain úgy néz ki mintha a májamat szeretné
Valós Always-On	

Cloudflare	
Pros	Cons
Egyben deployolható minden része	Eltérő felépítés
Idle idő nem költség	Nehezebben kezelhető felület
Sokkal szélesebb userbase / „Nagyobb nevű”	Bizonyos funkciók külön modulokban vannak (bár semmi ellehetetlenítő)
Nem kell semmilyen fizetési mód	Nagymértékű újratervezést igényel a projekt
Szebb alap domain	Lehet mégis van minimális cold start, már nem tudom követni, de nem tűnt fel