



Helix

de Asandei Ștefan-Alexandru

2023



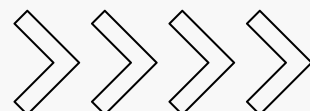
Clasa a IX-a

Colegiul Național Iași

Profesor: Pădurariu Emanuela-Tatiana

Cuprins

1. Introducere	
1.1. Scopul platformei.....	2
1.2. Publicul țintă.....	3
2. Prezentare generală	
2.1 Pagina principală.....	4
2.2 Forum.....	4
2.3 Online Judge.....	5
2.4 Code Runner.....	5
2.5 AI Coach.....	6
3. Prezentare tehnică	
3.1 Frontend	
3.1.1 Tehnologii.....	7
3.1.2 Accesibilitate.....	8
3.1.3 Performanță.....	8
3.1.4 Serializare.....	9
3.2 Backend	
3.2.1 Structură.....	10
3.2.2 Securitate.....	11
3.2.3 Rularea codului.....	12-13
3.2.4 Inteligența artificială.....	14
3.2.5 Baza de date.....	14
3.3 Infrastructură.....	15
4. Bibliografie.....	16



1. Introducere

1.1 Scopul platformei

Helix este o platformă adresată pasionaților de informatică de toate vârstele. Aceasta îmbină un catalog de probleme cu un mediu de dezvoltare integrat și o comunitate pentru a crea un loc cât mai prielnic pentru rezolvarea și discutarea problemelor de natură algoritmică.

Se pot identifica patru elemente, fiecare fiind axat pe una din paginile principale:

- exersare: Online Judge
- comunitate: Forum
- ajutor: AI Coach / Forum
- experimentare: Code Runner

Helix a fost gândit ca o platformă unde găsești tot ce ai nevoie pentru a-ți dezvolta abilitățile de gândire algoritmică, cât și a învăța lucruri noi din domeniul informaticii. Ce este diferit față de website-urile consacrate, de exemplu CodeForces sau PblInfo? Helix este construit cu tehnologii actuale de care beneficiază utilizatorul cel mai mult, în primul rând prin conveniența sa. Codul se poate scrie și rula în același loc, dacă ai nevoie de ajutor poți întreba pe forum sau pe asistentul AI.



1.2 Publicul țintă

Cui i se adresează Helix? Oricărui pasionat de informatică doritor să descopere mai mult! De la elevi de gimnaziu și liceu până la studenți și, de ce nu, chiar adulți ce vor să se pregătească pentru un interviu tehnic. Conținutul este generat și recomandat de către utilizatori, așa că, aceștia au puterea de decizie asupra conținutului prezentat.

Există două secțiuni principale ce prezintă conținut realizat și moderat de alți utilizatori:

- forumul: împărțit în comunități unde se postează
- online judge: probleme (fie originale, propuse de utilizatori, fie preluate din concursuri)

The screenshot displays the Helix website interface. On the left, a sidebar contains navigation icons: a home icon (highlighted with a red box), a chat icon, a code icon, a link icon, a document icon, and a settings icon. The main content area is divided into several sections:

- Upcoming contests:** A table with columns for TITLE, AUTHORS, START, and DURATION. It lists 'Helix Round #1' by StefanAsandei, starting on 01.11.2023 at 12:00, with a duration of 02:00.
- New courses:** A section stating 'Coming soon!'.
- Latest changes:** A list of updates: '27.06.2023: You can now participate in coding contests on Helix!' and '23.06.2023: Save & share files from the new CodeRunner'.
- Latest posts:** A list of recent forum posts. The first post is 'An introduction to C++ templates' by StefanAsandei, posted 7 days ago, with 0 likes. The second post is 'Hello' by AKA 449, posted 13 days ago, with 0 likes. The third post is 'Hello' by StefanAsandei, posted 17 days ago, with 1 like.

At the bottom, a footer indicates the user is logged in as 'StefanAsandei' and provides a random tip: 'You can learn the basics of programming right here, on Helix'.

2. Prezentare

2.1 Pagina principală

Pagina principală are un rol de a ghida utilizatorii către celelalte pagini. Aici, noii utilizatori pot descoperi funcționalități ale platformei.

2.2 Forum

Forumul este organizat în comunități. Acestea, de obicei, se pot baza pe interese comune asupra unui subiect, împărtășirea cunoștințelor, cât și oferirea ajutorului pe diferite teme. Oricine poate crea o comunitate, creatorul având și rol de moderator. Acesta decide subiectul și audiența. Orice utilizator poate accesa și participa în toate comunitățile.

O comunitate este organizată în postări. Acestea au un conținut text ce poate fi formatat folosind Markdown. Fiecare postare are și un scor. Acesta este decis de utilizatori, fiecare având opțiunea de a mări scorul cu un punct, sau de a-l scădea. În acest mod, postările cele mai relevante și bine realizate vor fi văzute de mai mulți utilizatori.



2.3 Online Judge

Aceasta este secțiunea website-ului dedicată problemelor de natură algoritmică. Problemele sunt prezentate sub forma unei liste cu informații importante, precum numele ei, sursa și autorul. Problemele pot fi adăugate de administratorul instanței (platformei), care își asumă rolul de moderator. Fiecare problemă are un enunț, exemple input/output și teste. Astfel, utilizatorul poate scrie codul și citi enunțul de pe aceeași pagină. Soluția poate fi testată apăsând butonul "Submit", ce va rula testele și va calcula scorul. Nu există o limita de soluții trimise.

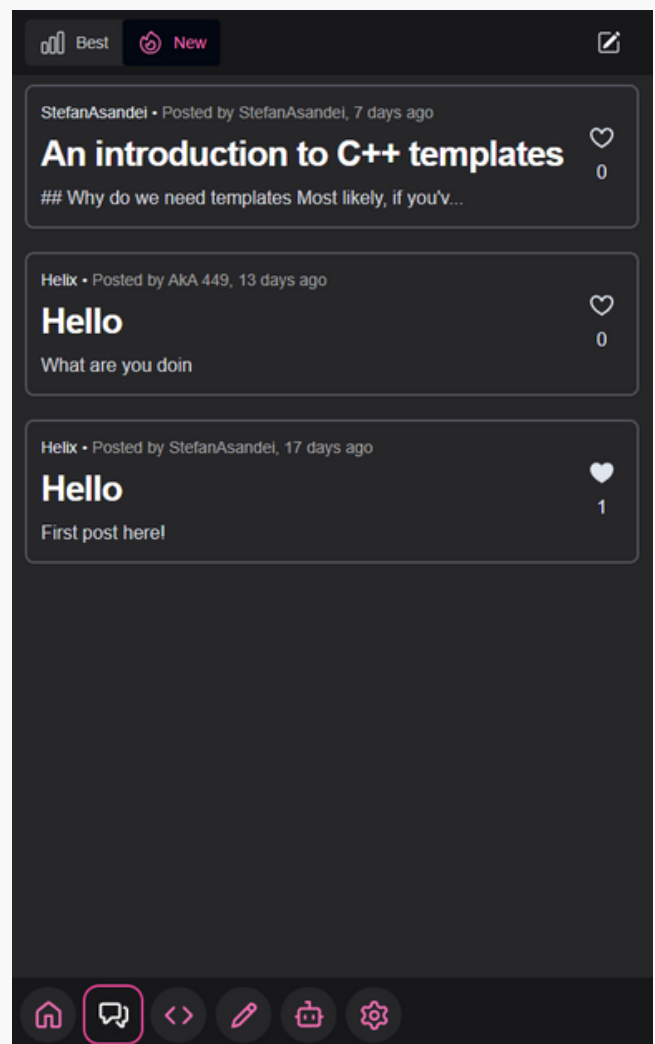
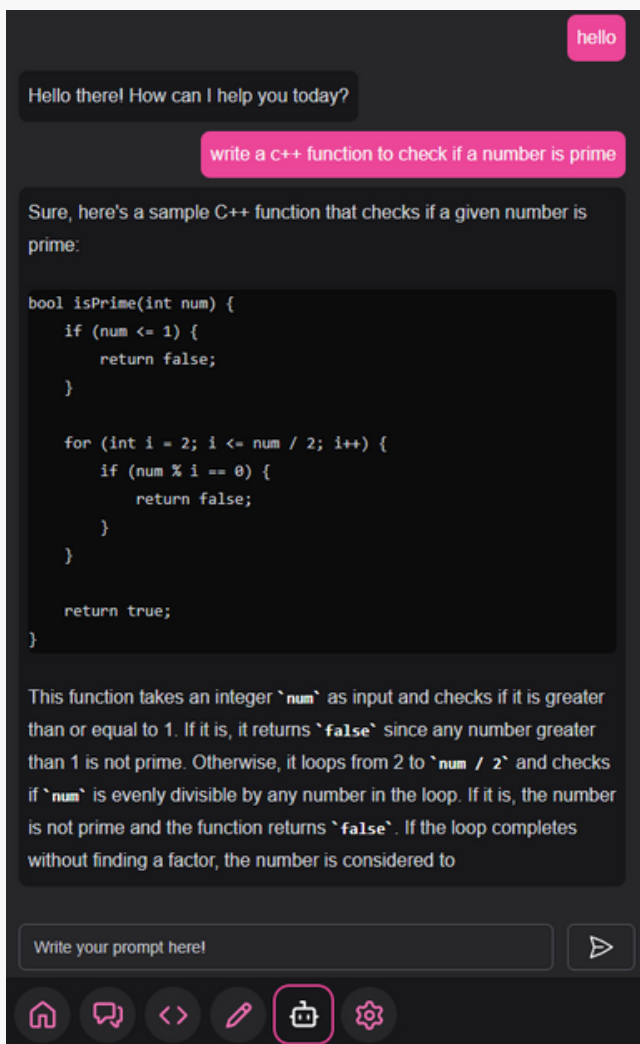
2.4 Code Runner

Aici, utilizatorii pot experimenta și testa, scriind codul într-un editor modern și rapid. Aceștia pot preciza datele de intrare, ce vor fi introduse automat. Odată ce codul este rulat, output-ul va fi afișat în partea dreaptă a ecranului. Peste 10 limbaje de programare sunt disponibile.



2.5 AI Coach

"Mentorul AI" este asistentul inteligent al platformei. Acesta poate comunica prin limbaj natural cu utilizatorul și poate rezolva și dezbate diverse subiecte. Scopul principal al acestuia este de a reprezenta un ajutor prietenos al celor ce au întâmpinat dificultăți în rezolvarea problemelor. Acesta poate scrie cod, cât și explica și răspunde la întrebări. Totuși, dacă utilizatorul dorește să discute subiecte off topic, acesta are cunoștințe din foarte multe domenii și are un orizont mult mai vast.



3.1.1 Tehnologii

Frontend-ul folosește următoarele tehnologii:



- TypeScript: limbajul de programare folosit, oferă un sistem complet de typing pentru structurile de date ce asigură structura corectă a informației din comunicarea dintre componente și client - server
- ReactJS: librăria pentru UI, asigură utilități pentru a face interfața cât mai interactivă
- Tailwind: o serie de clase CSS utilitare pentru a impune un style system consistent și a unifica codul pentru UI în componente TSX fără fișiere externe CSS
- Jotai: librărie pentru state management la nivel global între componente
- Storybook: librărie folosită pentru testarea componentelor și pentru a genera documentația acestora
- Cypress: librărie pentru testarea funcționalității paginilor și a interacțiunii utilizatorilor



3.1.2 Accesibilitate

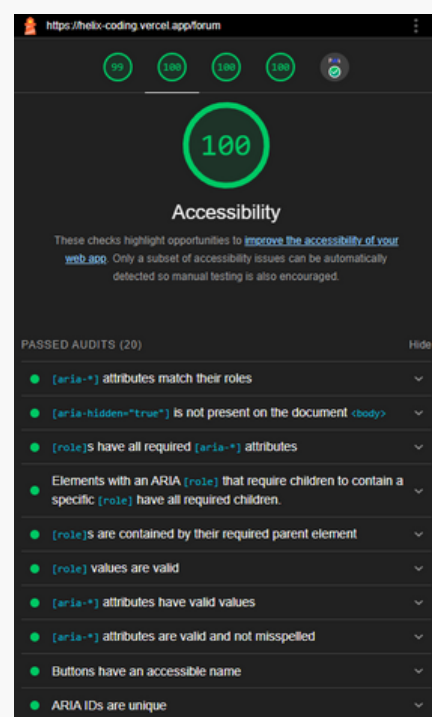
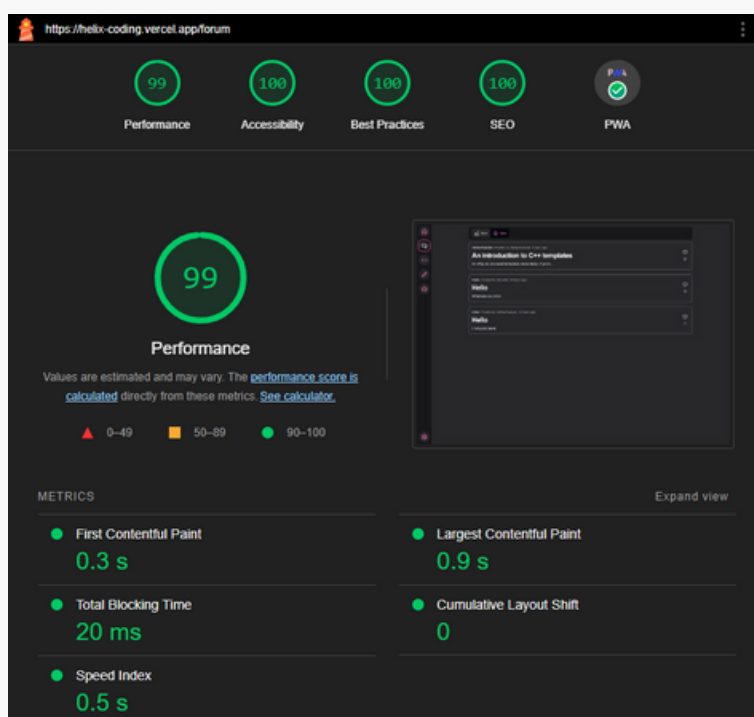
Au fost luate în considerare următoarele aspecte, privind accesibilitatea:

- contrastul dintre culori este optim
- toate paginile sunt optimizate pentru diferite mărimi ale ecranului (computer, tabletă și telefon)
- elementele au nume descriptive, imaginile au proprietatea "alt" și standardul ARIA este utilizat.

3.1.3 Performanța

Deși aspectele cele mai importante legate de performanța sunt pe partea de backend, au fost realizate următoarele optimizări pe frontend:

- imaginile sunt compresate
- utilizatorul poate interacționa instantaneu fără să aștepte încărcarea întregii pagini (non-blocking I/O)



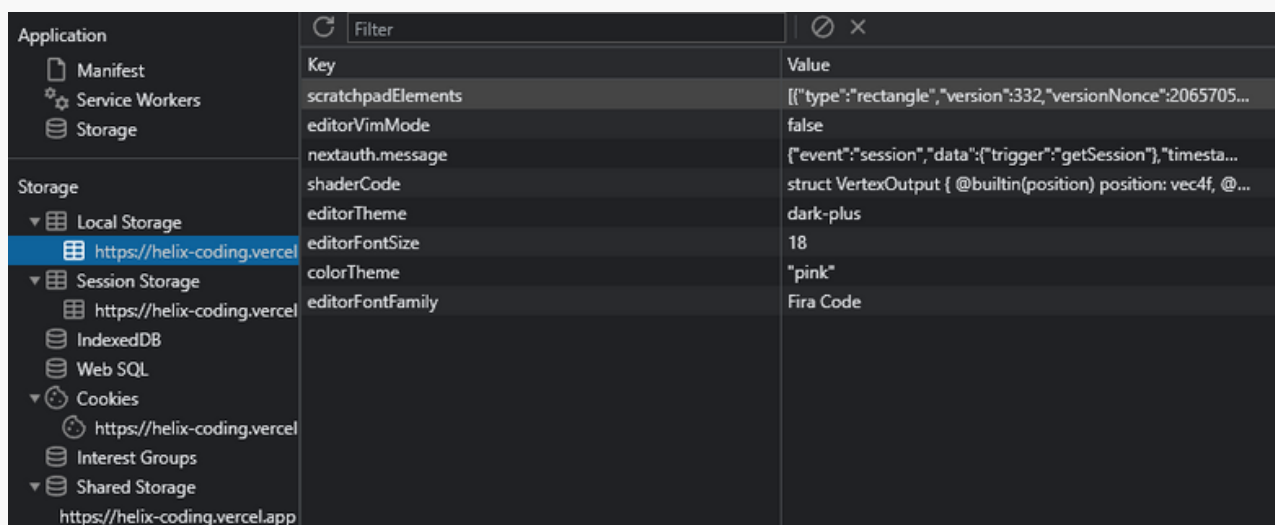
3.1.3 Serializare

Salvarea și persistarea datelor este gândită pentru două situații:

- accesul imediat (navigare de pe o pagină pe alta, revenirea pe website la scurt timp pe același device)
- accesul ulterior (accesarea website-ului oricând de pe orice device)

Pentru accesul direct este folosit local storage (format JSON) și cookie-urile (pentru sesiunea curentă, utilizatorul autentificat). Datele din sesiune sunt encriptate cu un secret unic.

Pentru accesul ulterior, cât și pentru stocarea datelor precum postările de pe forum sau problemele, este folosită o bază de date PostgreSQL, fiind accesibilă prin comunicarea cu backend-ul.



Application	Filter	Key	Value
Manifest		scratchpadElements	[{"type":"rectangle","version":332,"versionNonce":2065705...}
Service Workers		editorVimMode	false
Storage		nextauth.message	{"event":"session","data":{"trigger":"getSession"},"timesta...
		shaderCode	struct VertexOutput { @builtin(position) position: vec4f, @...
		editorTheme	dark-plus
		editorFontSize	18
		colorTheme	"pink"
		editorFontFamily	Fira Code
Local Storage			
https://helix-coding.vercel			
Session Storage			
https://helix-coding.vercel			
IndexedDB			
Web SQL			
Cookies			
https://helix-coding.vercel			
Interest Groups			
Shared Storage			
https://helix-coding.vercel.app			

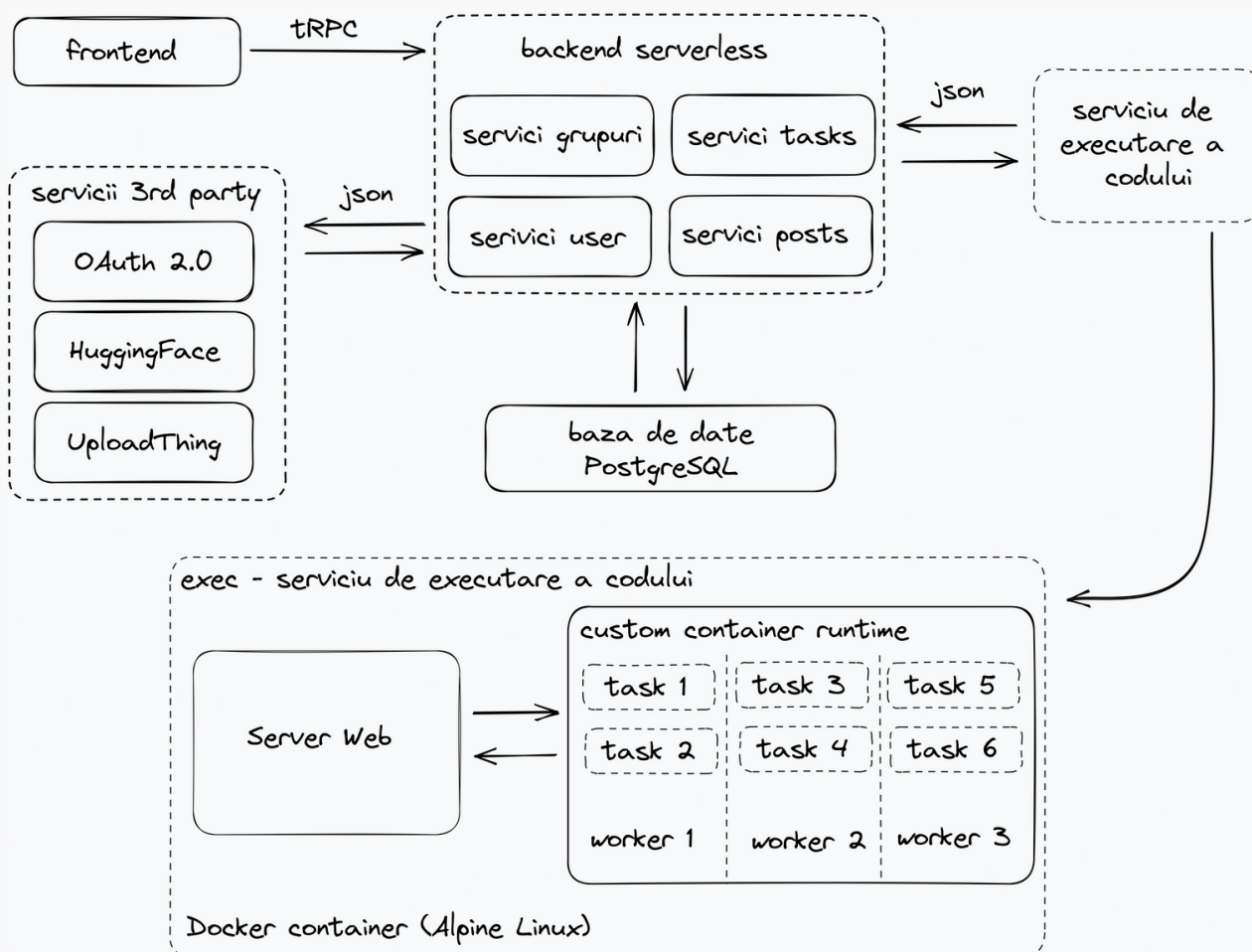


3.2.1 Structură

Backend-ul este structurat pe mai multe servicii separate. Acestea folosesc următoarele tehnologii:

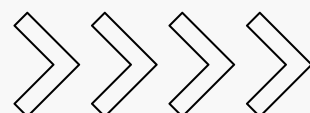
- NextJS
- tRPC
- Prisma
- Rust
- HuggingFace
- Docker
- Alpine Linux
- PostgreSQL

NEXT.js



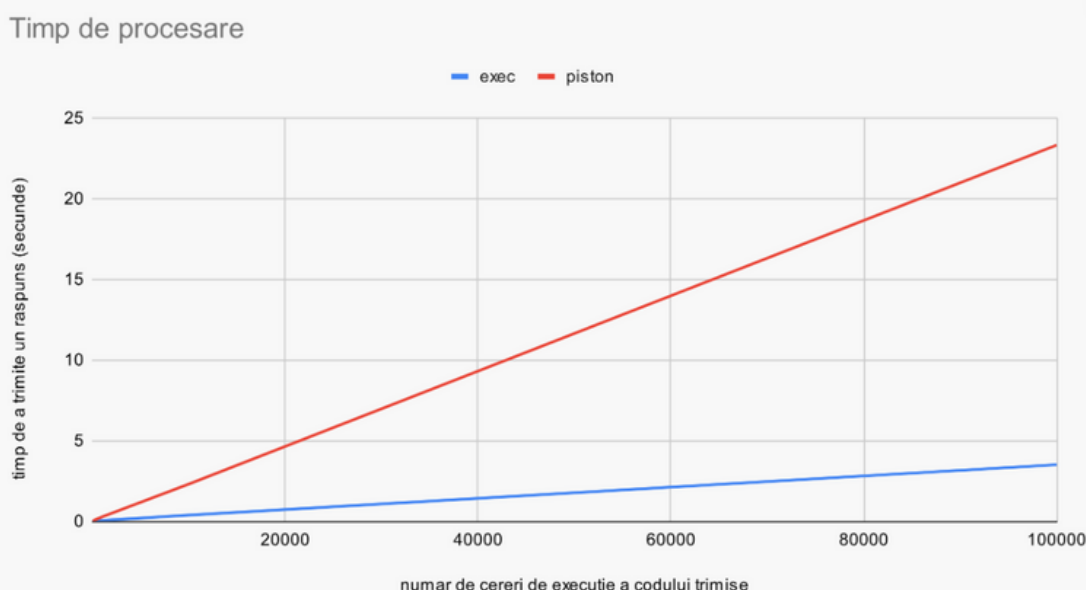
3.2.2 Securitate

- orice conținut introdus de utilizator este validat (protecție atacuri XSS, SQL injection), codul Markdown este sanitizat înainte să fie introdus în DOM
- utilizatorul nu poate face cereri directe către backend, acesta fiind protejat de CORS
- codul trimis de utilizatori este rulat într-un mediu izolat, resursele proceselor fiind strict limitate (mai multe detalii în descrierea motorului de execuție a codului)
- sunt verificări pentru permisiunile de scriere/citire ale utilizatorilor
- autentificarea este realizată doar prin OAuth 2.0 (Google, Twitter, GitHub și Discord)
- unele rute sunt protejate pe baza statutului utilizatorului (exemplu: probleme de la un concurs care încă nu a început)



3.2.3 Rularea codului

Una din funcționalitățile de bază ale platformei constă în execuția codului trimis de client și testarea acestuia. Pentru acest lucru este folosit un motor de execuție a codului scris în Rust, numit "Exec".



Piston este considerat cel mai rapid motor open source de execuție a codului. Dintr-un benchmark realizat pe un PC (AMD Ryzen 5 3600, 32 GB RAM), ambele servicii rulând de pe containere Docker, reiese că Exec este de aproximativ 5 ori mai rapid și se scalează mai eficient.

Acest lucru este posibil datorită unei tehnologii de izolare bazată pe un runtime de container propriu. Exec folosește o utilitate a kernelului Linux, namespaces, pentru a asigura izolarea procesului.



În ce consta izolarea procesului:

- procesul are rol de init(1), aparand ca fiind cel ce a inițializat sistemul și are PID = 1
- device-urile de rețea sunt alterate, astfel fiind imposibilă comunicarea cu internetul
- este creat un utilizator nou, fără privilegii ridicate
- este limitat timpul de execuție, număr de fișiere deschise și mărimea acestora, numărul de fire de execuție create

Server-ul Exec este scris în Rust și este compilat, ulterior rulat ca executabil. Operațiile I/O nu blochează, fiind folosite operații asincrone. De asemenea, cererile primite sunt distribuite pe mai multe fire de execuție pentru a asigura o performanță cât mai bună. Custom container runtime-ul este mult mai minimal și rapid de pornit decât unul tradițional ca Docker sau LXC. Acesta este scris în C, un exemplu pentru utilizarea namespace-urilor:

```
int main(int argc, char *argv[]) {
    if (argc < 2) {
        printf("Usage: %s [program] [args...]\n", argv[0]);
        exit(EXIT_FAILURE);
    }

    const long flags = CLONE_NEWIPC | CLONE_NEWNET |
                      CLONE_NEWNS | CLONE_NEWPID |
                      CLONE_NEWUTS;

    pid_t pid = clone(child_process,
                     child_stack + STACK_SIZE, flags | SIGCHLD, argv + 1);

    waitpid(pid, NULL, 0);

    return 0;
}
```

```
int child_process(void *arg) {
    setup_container();

    char **cmd = (char **)arg;

    if (execvp(cmd[0], cmd) == -1) {
        LOGE("execvp");
        exit(EXIT_FAILURE);
    }

    return 0;
}
```

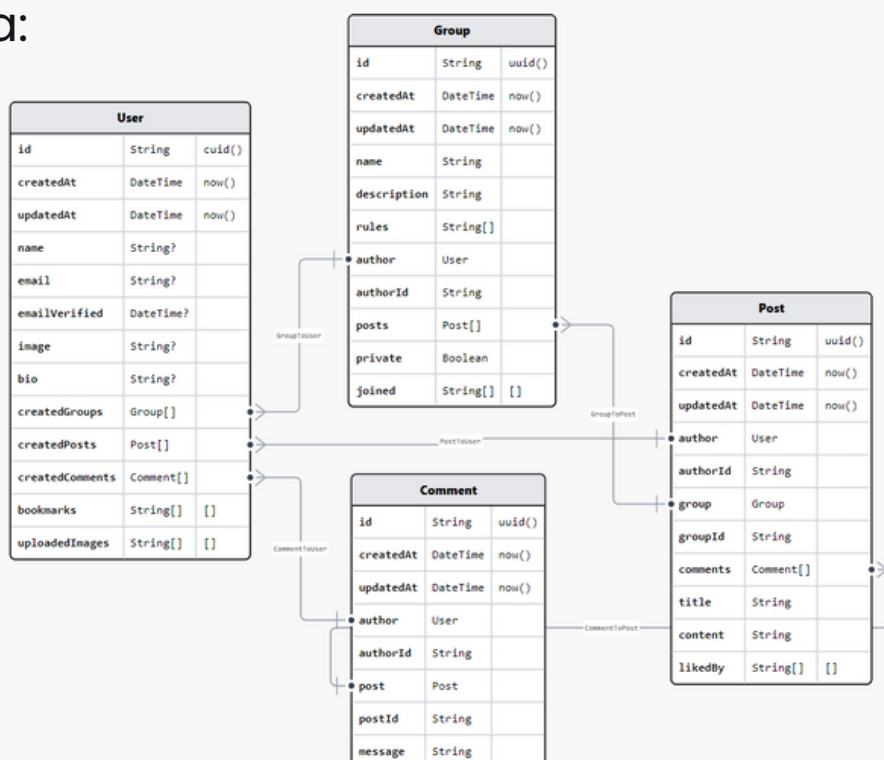


3.2.4 Inteligența Artificială

Helix oferă o pagină experimentală "AI Coach", aceasta oferă posibilitatea de a interacționa cu un model conversațional AI. Din cauza limitărilor hardware impuse de server, backend-ul apelează la rândul lui API-ul HuggingFace pentru răspunsul modelului. Pentru a maximiza raportul viteză / calitate, au fost analizate mai multe modele, printre care cele mai bune au reieșit OpenAssistant/oasst, bigscience/bloom și google/flan-t5.

3.2.5 Baza de date

Baza de date este una PostgreSQL și este hostată gratuit pe Supabase, cu o limită de spațiu de stocare de 500mb. Pentru scrieri și citiri este folosită biblioteca Prisma (ca ORM). Exemplu restrâns cu schema:

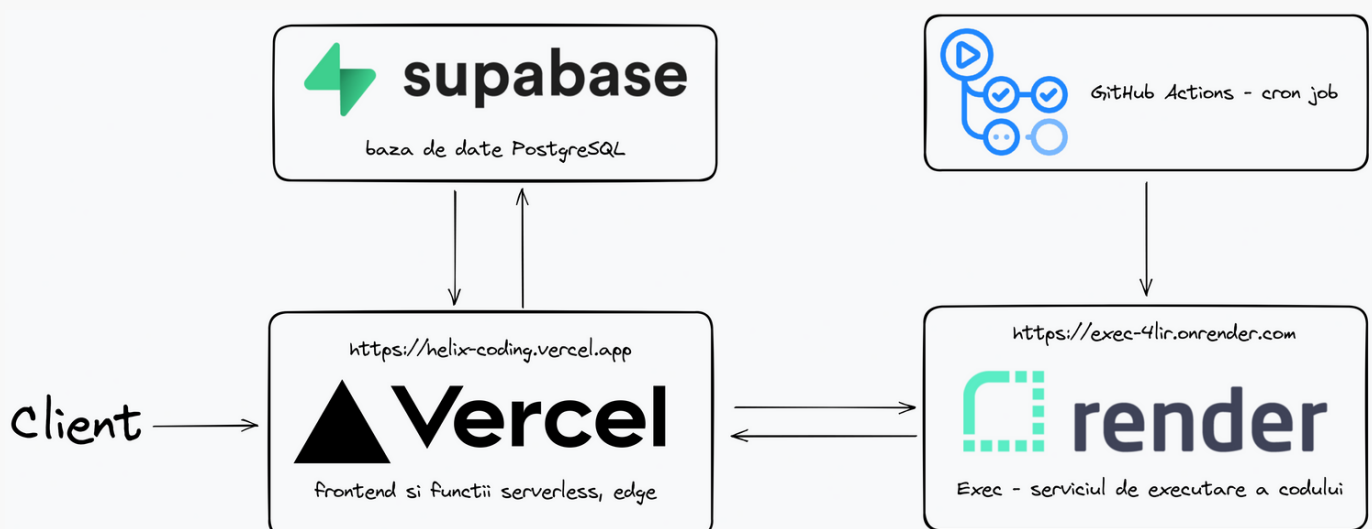


3.3 Infrastructură

Serviciul NextJS, ce servește frontend-ul și funcțiile serverless, este găzduit gratuit pe Vercel. Majoritatea rutelor din frontend sunt servite static (SSG), în timp ce altele folosesc ISR pentru a face streaming cu resursele de pe backend.

Motorul de execuție a codului ("Exec") este găzduit pe platforma Render.com, acesta rulând într-un container Docker. Deoarece instanța de compute este gratuită, resursele hardware sunt limitate la 512mb RAM și 0.25% dintr-un CPU core (model standard Render.com). Aceste limitări (și cold start-ul) constituie singurele probleme de performanță ale acestui serviciu.

Sunt folosite GitHub Actions pentru a seta un cron job de a apela Exec, o dată la 10 minute, pentru a minimiza cold start-ul.



4. Bibliografie

1. Documentațiile pentru tehnologiile folosite

- <https://create.t3.gg/>
- <https://vercel.com/docs>
- <https://www.prisma.io/docs>
- <https://tailwindcss.com/docs>
- <https://huggingface.co/docs>
- <https://docs.cypress.io/>

2. Proiecte open source

- <https://github.com/benawad/dogehouse>
- <https://github.com/t3dotgg/chirp>

Codul sursă este open-source și folosește licența GNU General Public License v3. Acesta este disponibil pe platforma GitHub:

- <https://github.com/nikolatesla13/helix>
- <https://github.com/nikolatesla13/exec>

