FER - GO
Zadatak 2023.

Generalna uputa: kolege prisutni 50+% predavanja do 11. 05. 2023. mogu pristupiti izradi zadatka. Zadatak možete raditi samostalno ili u grupi do 3 (tri) studenta. Obavezno javite do 18. 05. 2023. kako ćete se ekipirati. Zadatak mora biti predan do zadnjih predavanja ove akademske godine.

Napraviti HTTP baziranu aplikaciju koja kod pokretanja čita config.yaml u kojem se nalaze konfiguracijska postavke.

config.yaml:

```yaml
jmbag: 00363912345
http:
  address: 127.0.0.1
  port: 80
users:
  - name: imeprezime (dodati svoje)
    jmbag: 0036391058
    password: lozinka # extra - use bcrypt or something similar
  - name: admin
      password: adminpwd # extra - isto kao i za prethodnog usera
```

Aplikacija mora sadrzavati sljedeće HTTP endpointove:

- /jmbag
  - Vraća JMBAG specificiran u config.yaml file-u (u bilo kojem formatu, txt, json …)
  - Ovaj endpoint ne treba biti zasticen sa user/password kombinacijom, dok ostali **moraju** biti odbijeni ako pristupni podaci nisu korektni

- /sum?a=2&b=1
  - vraca json u obliku (pripazite da dodate i odgovarajuci header)

```
{
  "a": 2,
  "b": 1,
```

```
  "result": 3
}
```

- ako se ne dobiju dva broja ili a ili b nisu brojevi, vratiti gresku (status bad request)

- /multiply?a=2&b=1
  - Za grupe s vise od jednog studenta
  - vraca json u obliku (pripazite da dodate i odgovarajuci header)

```
{
  "a": 2,
  "b": 1,
  "result": 2
}
```

- /fetch
  - POST zahtjev koji prima url u tijelu zahtijeva obliku json zahtjeva

```
{"url":"https://example.com"}
```

  - Odraditi GET na primljeni url i vratiti sve header-e u json formatu
    Primjer rezultata:

```
{
  "Accept-Ranges": [
       "bytes"
  ],
  "Content-Length": [
       "442"
  ],
  ...
  "Vary": [
       "Accept-Encoding"
  ]
}
```

- /0036391234 (vas jmbag)
  - Svaki student definira svoj tip gdje
    - POST prima podatke i sprema ih u student1.txt (proizvoljno ime) dokument na disk
    - GET vraca podatke spremljene u student1.txt

Struktura projekta je proizvoljna, no **mora** sadrzavati go.mod i barem jedan extra direktorij unutar projekta

Implementirajte HTTP endpoint koji će moći izvršavati paralelizirati SAXPY operaciju. Zahtjevi su:

- Omogućiti slanje *n* dijelova podataka na server
- Omogućiti klasifikaciju poslanog podatka. To znači da korisnik može uzastopno poslati *n-ti* dio nekog skupa različitog skupa podataka. Morate omogućiti ispravan način kodiranja podatka kako bi sustav znao prepoznati sve dijelove skupa i kada ima cijeli skup, otpočeti paraleliziranu SAXPY operaciju
- Korisnik može poslati podataka, promijeniti već poslani podatak ili pobrisati *n-ti* dio podatka. Implementacija toga je slobodna, no možete se voditi i jednostavnim REST principom
- Iako nećemo provjeravati optimiziranost algoritma kojeg ste kreirali, on mora biti paraleliziran na način da je moguće simultano poznati *n* puta HTTP endpoint
- Rezultat morate vratiti prilikom kompletiranja svih potrebnih dijelova skupa za SAXPY operaciju
- Formati ulaza i izlaza podataka su proizvoljni ali moraju biti jasno dokumentirani kako bi ih mogli reproducirati