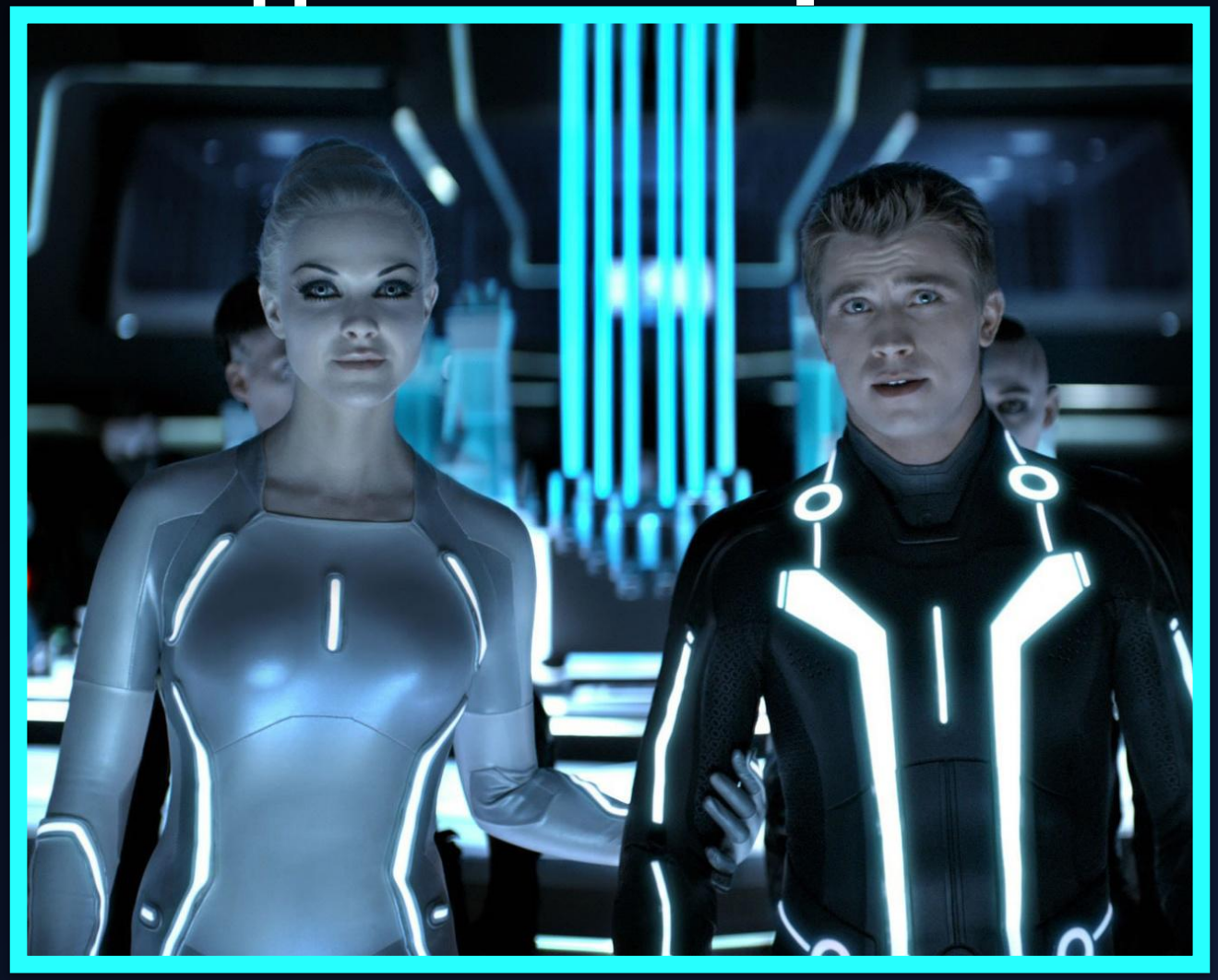
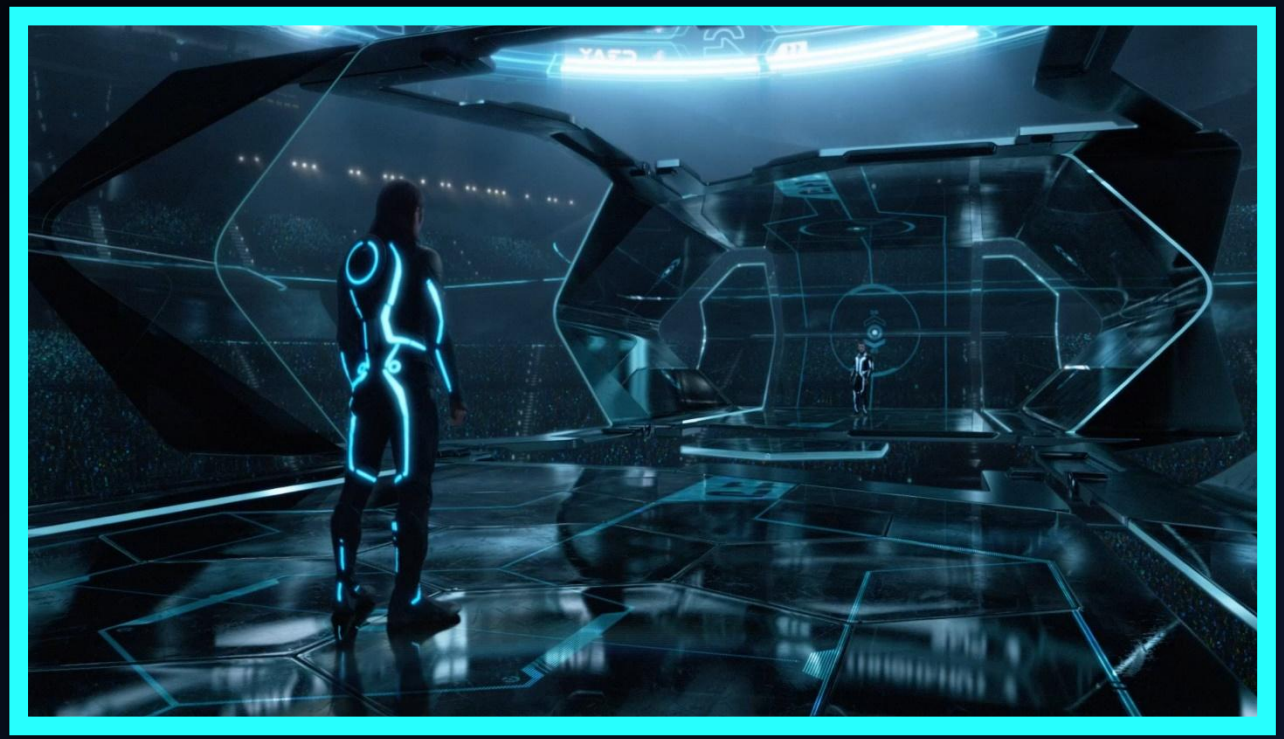
Короче говоря

МЫ СДЕЛАЛИ ТРОН

Трон — многопользовательская аркадная игра, основанная на одноименном фильме 1982 года.



Новое видение

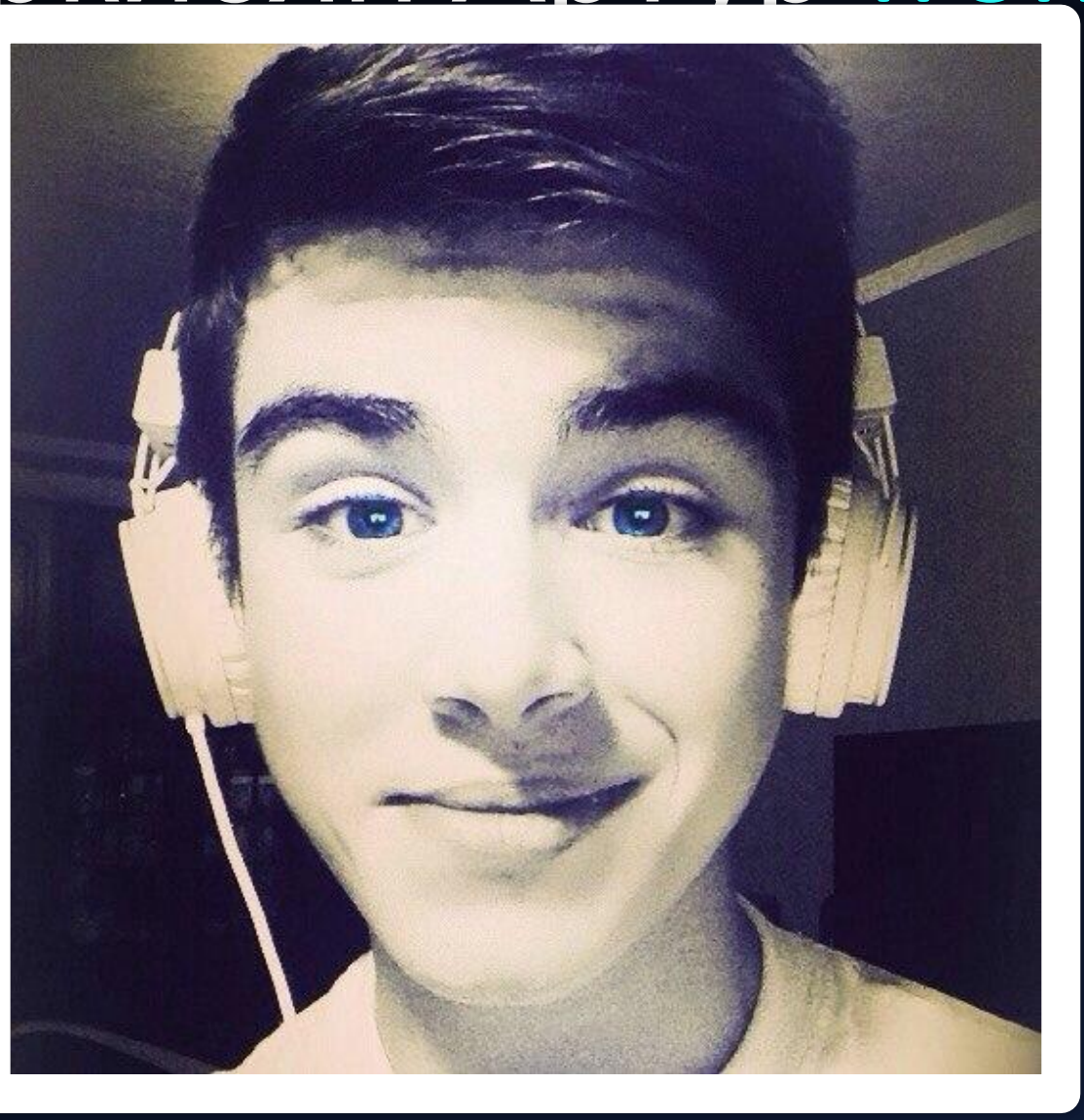


Codeloft

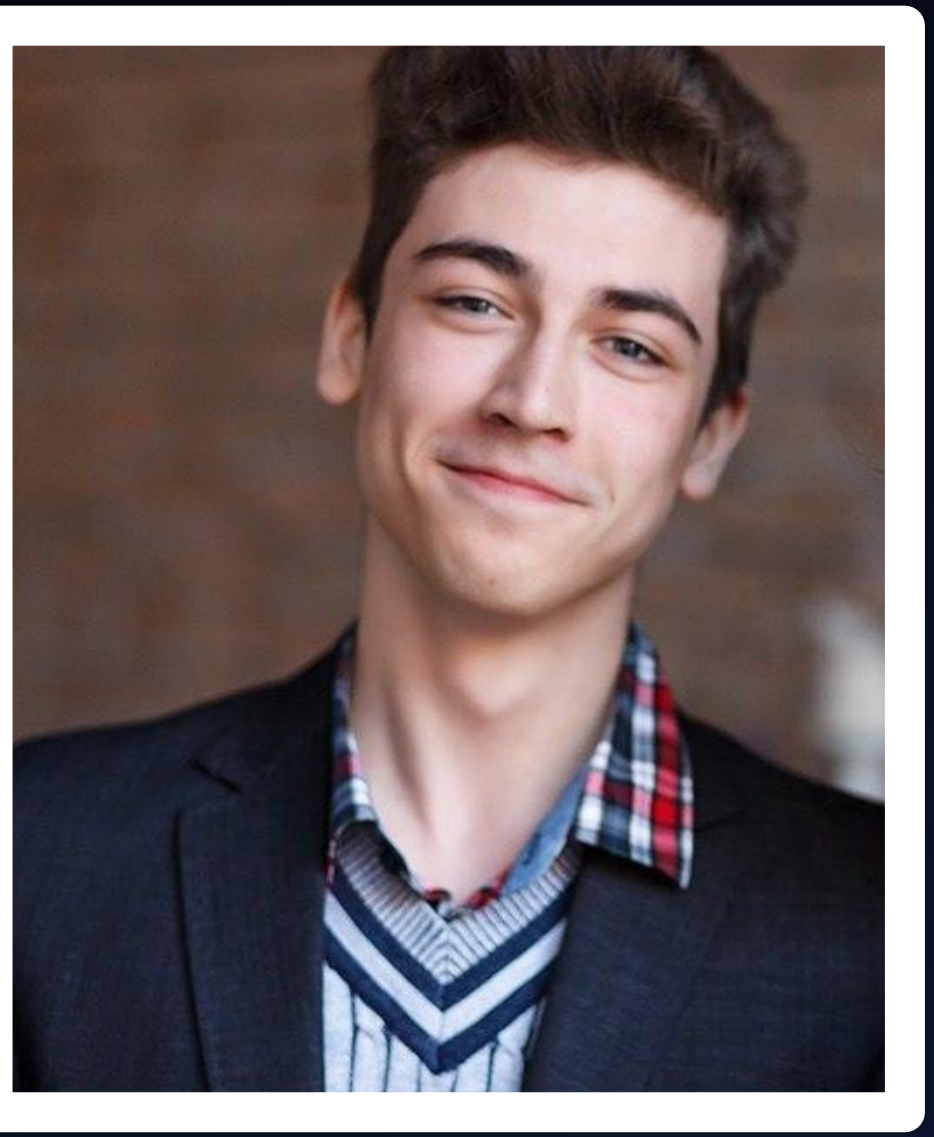
®

Игорь Дыров frontend

Саркисян Артур frontend



Анохин Данил backend

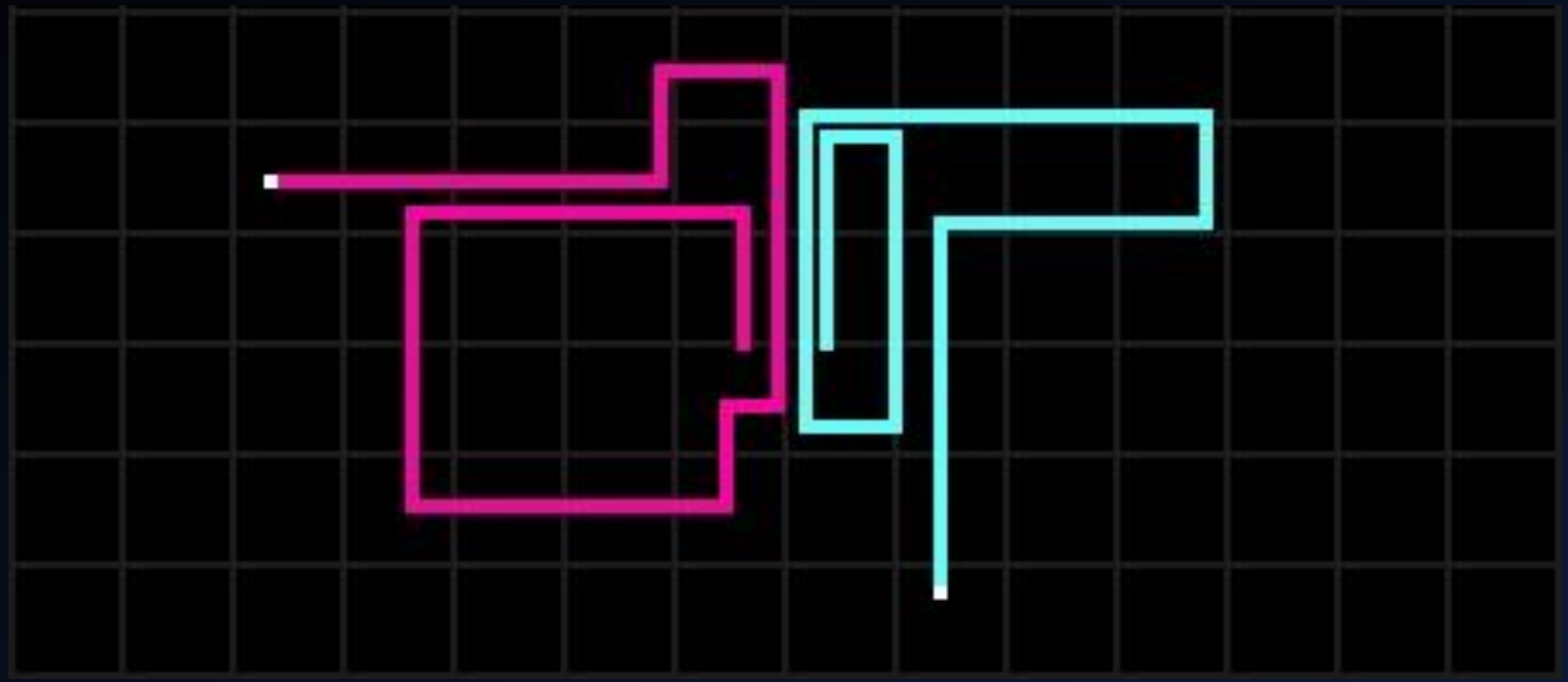


Рязанов Максим backend

Перескоков Владислав

ментор

Старый Трон

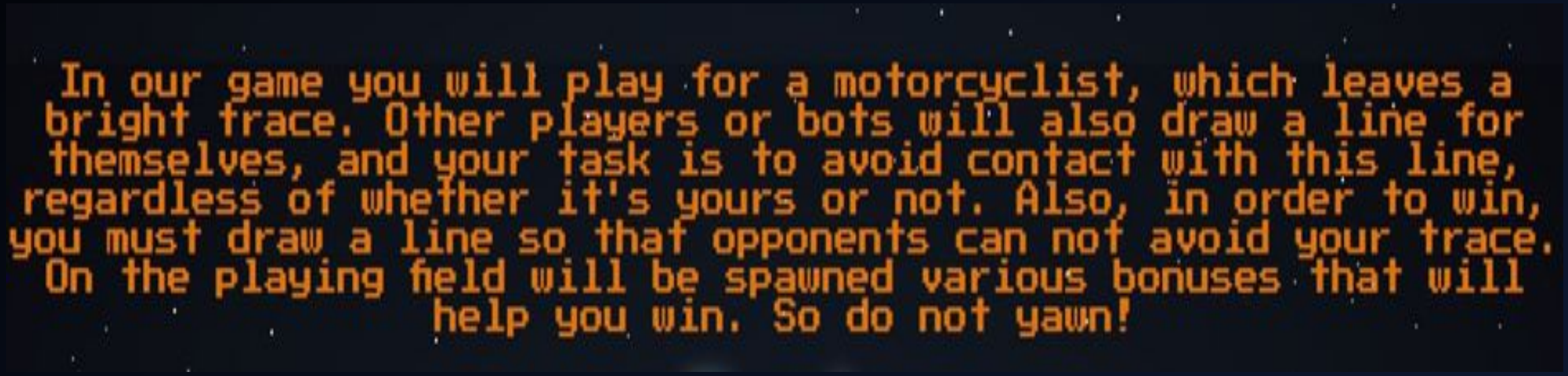


Когда-то было и так…

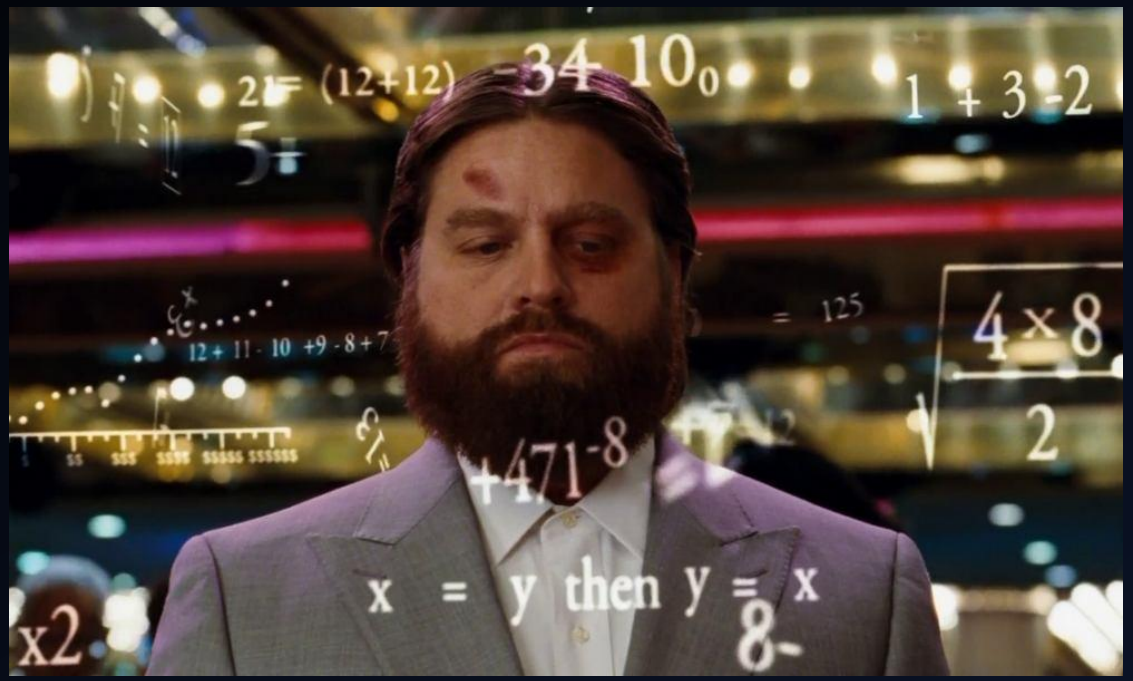
Развитие


Tron 2D
Sign In
Sign Up
Single Player
Rules
High score
CodeLoft®
en

Но были и не самые удачные варианты…



Матчасть



Технические особенности

```
preRender() {
        return Transport.Get('/user').then((responseJSON) => responseJSON.json())
                .then((response) => {
                        this.pageAmount = response.pagesCount;
                });
}
 afterRender() {
        return new Promise((resolve) => {
                this.mainEvent = this.submit;
                this.inputs = {
                        login: this.elementsArray[1],
                        password: this.elementsArray[3],
                };
                this.errorLabels = {
                        login: this.elementsArray[0],
                        password: this.elementsArray[2],
                };
                Object.keys(this.errorLabels).forEach((field) => {
                        this.errorLabels[field].hide();
                });
                this.validator = new Validator(this.inputs, this.errorLabels);
                Object.keys(this.inputs).forEach((input) => {
                        this.inputs[input].render().addEventListener('blur', () => {
                                this.validator.checkInput(input);
                        });
                });
                resolve();
        });
}
```

```
mux := http.NewServeMux()

mux.HandleFunc("/", handlers.MainPage)
mux.Handle("/user", &handlers.UserHandler{db.DataBase, sessManager})
mux.Handle("/session", &handlers.SessionHandler{db.DataBase, sessManager})
mux.Handle("/user/", &handlers.UserById{db.DataBase, sessManager})
```

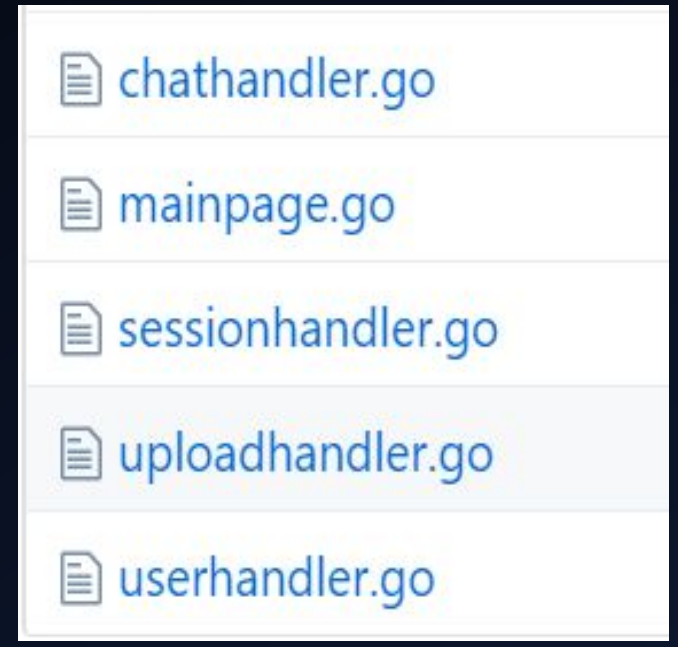
```
  logHandler := logMiddleware(mux)
  corsMW := c.Handler(logHandler)
  panicMW := panicMiddleware(corsMW)
  port := os.Getenv("PORT") // for heroku
  if port != "" {
          zap.S().Infow("get port from env: ", port)
  } else {
          port = "8080"
  }

  if len(os.Args) > 3 {
          port = os.Args[3]
          fmt.Println(port)
  }
  addr := fmt.Sprintf(":%s", port)

  fmt.Println("starting server on http://127.0.0.1:8080")
  http.ListenAndServe(addr, panicMW)
```

Handlers

- User handler
- Session handler
- Chat handler
- Upload handler

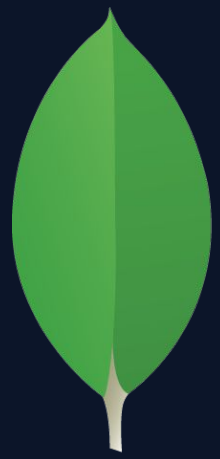


PostgreSQL

- database/sql
- github.com/lib/pq

```
create table if not exists users (
  id bigserial not null primary key,
  login citext unique ,
  password varchar(30),
  email varchar(30),
  score bigint default 0,
  lang varchar(2) default 'en',
  avatar varchar(30) default 'defaultimg.jpeg'
);

create table if not exists sessions (
  value text unique,
  id int references users(id) on delete cascade
);
```



mongoDB®

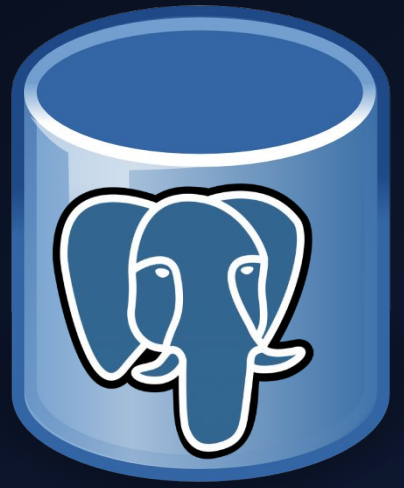

amazon
web services™


mLab

API

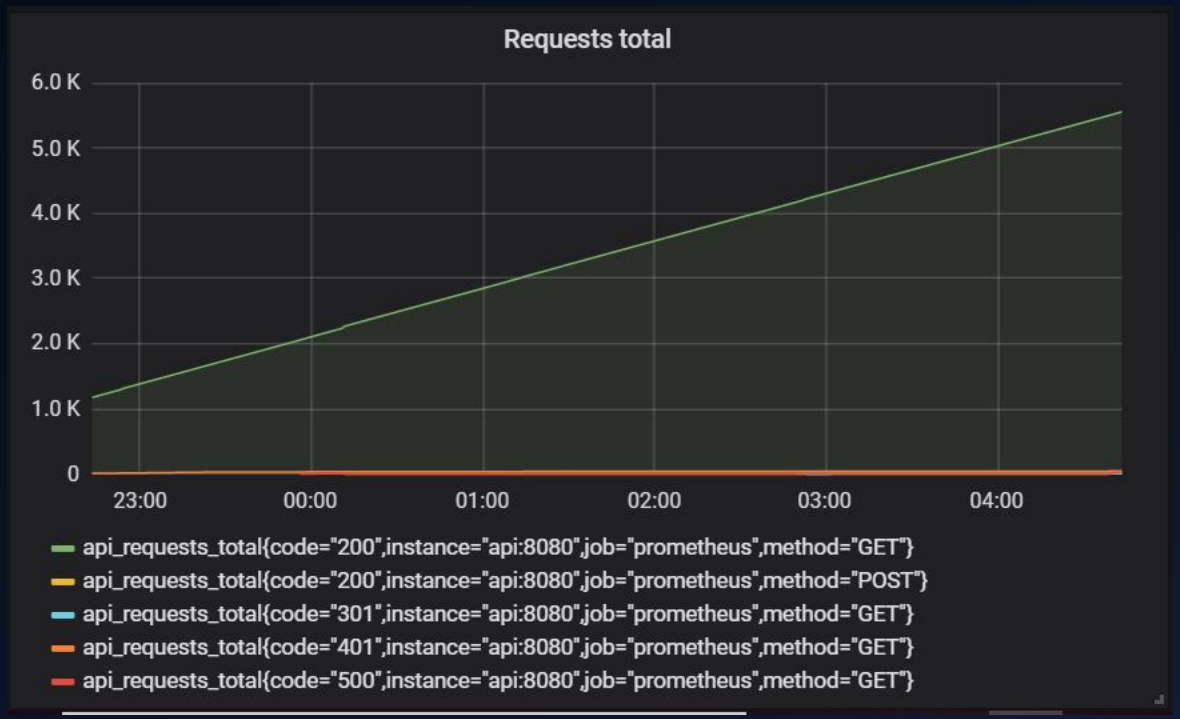
Auth

Game

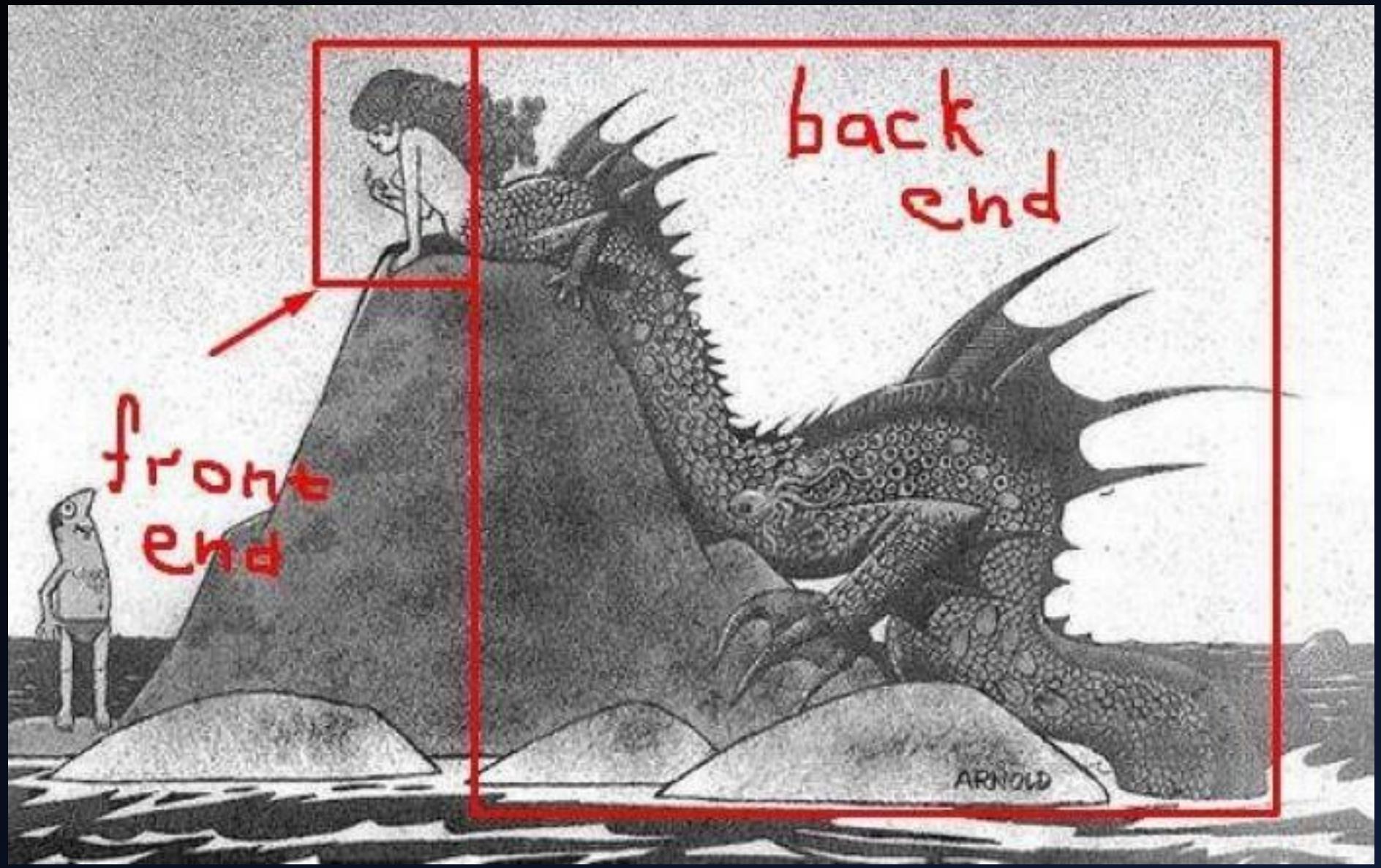



PROMETHEUS
GRAFANA


Rooms count
Last 10 minutes
2.5
2.0
1.5
1.0
0.5
0
04:34
04:36
04:38
04:40
04:42
game_rooms_count{instance="game:8082",job="prometheus"}

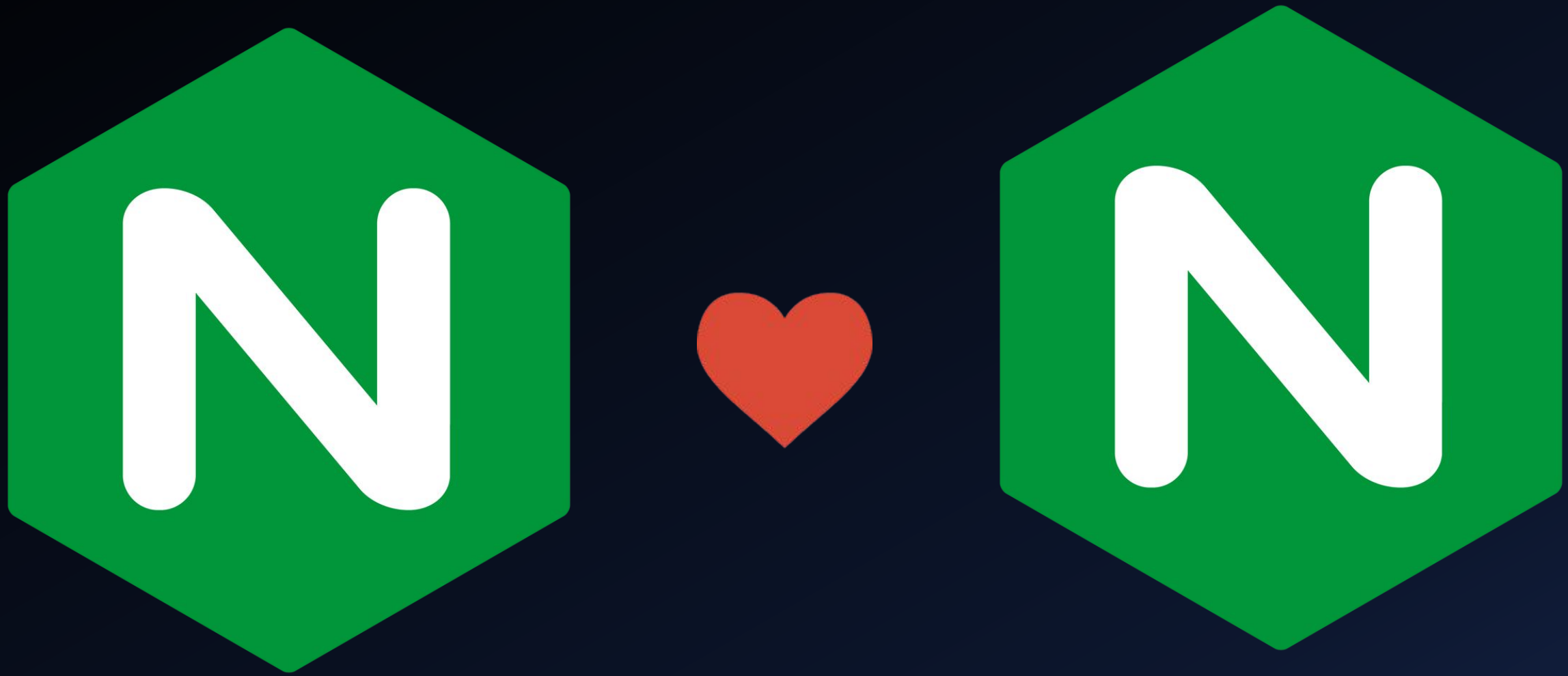

Requests total
6.0 K
5.0 K
4.0 K
3.0 K
2.0 K
1.0 K
0
23:00
00:00
01:00
02:00
03:00
04:00
api_requests_total{code="200",instance="api:8080",job="prometheus",method="GET"}
api_requests_total{code="200",instance="api:8080",job="prometheus",method="POST"}
api_requests_total{code="301",instance="api:8080",job="prometheus",method="GET"}
api_requests_total{code="401",instance="api:8080",job="prometheus",method="GET"}
api_requests_total{code="500",instance="api:8080",job="prometheus",method="GET"}

Интеграция Frontend и Backend

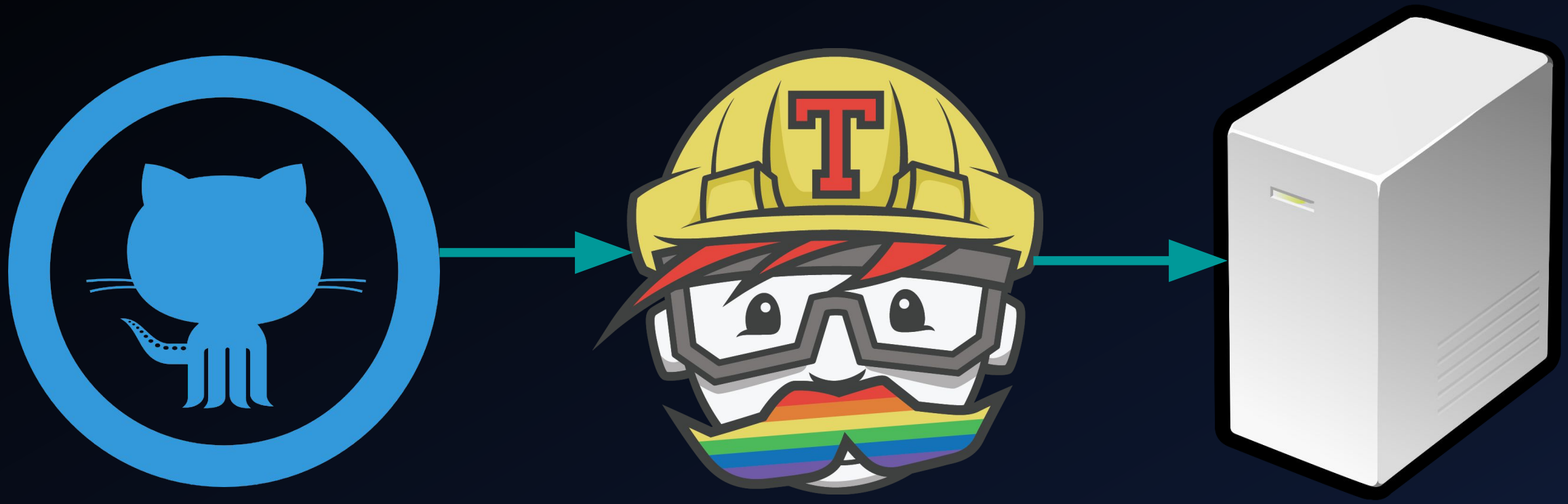


VPS

Один Nginx – хорошо, а два - лучше

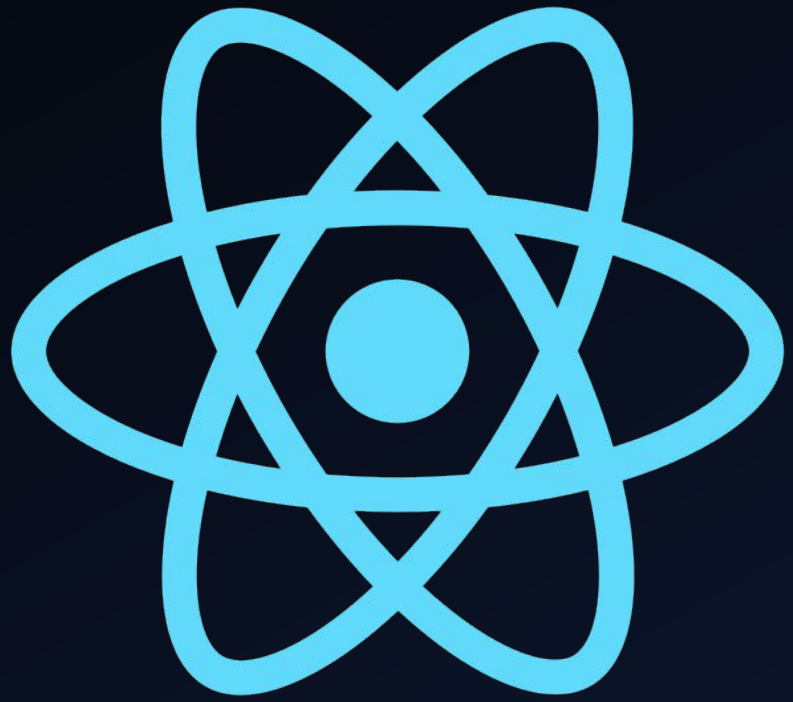


Деплой Frontend



Деплой Backend

React JS

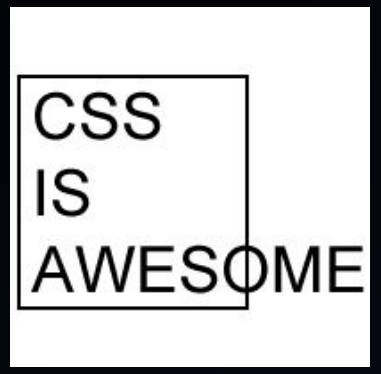


Redux

TypeScript

Давайте поиграем!

https://codeloft.ru


CSS
IS
AWESOME

JS

node
JS

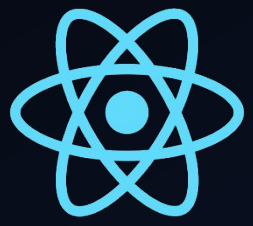
Sass

http2

handlebars

TS


N




Let's Encrypt

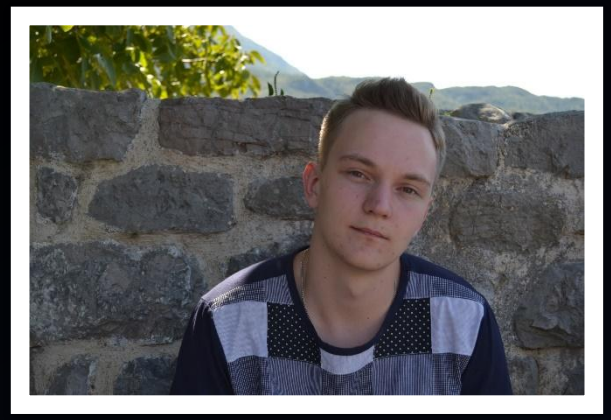
Prometheus + Grafana

GRPC

mongoDB®

protobuf
Protocol Buffers

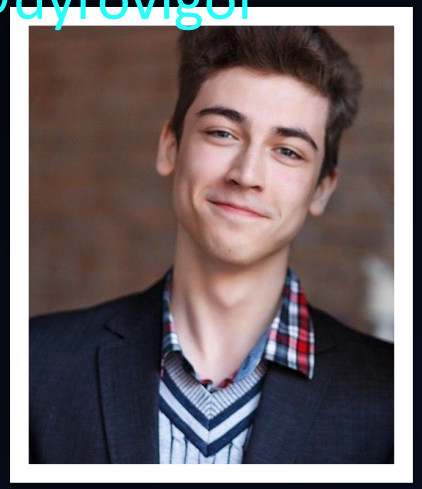
docker



Дыров Игорь
frontend
@dyrovigor

Саркисян Артур
frontend
@arthursa

Codeloft®



Анохин Данил
backend
@malefaro

Рязанов Максим
backend
@RyazMax