Sequelize

https://cybersoft.edu.vn/

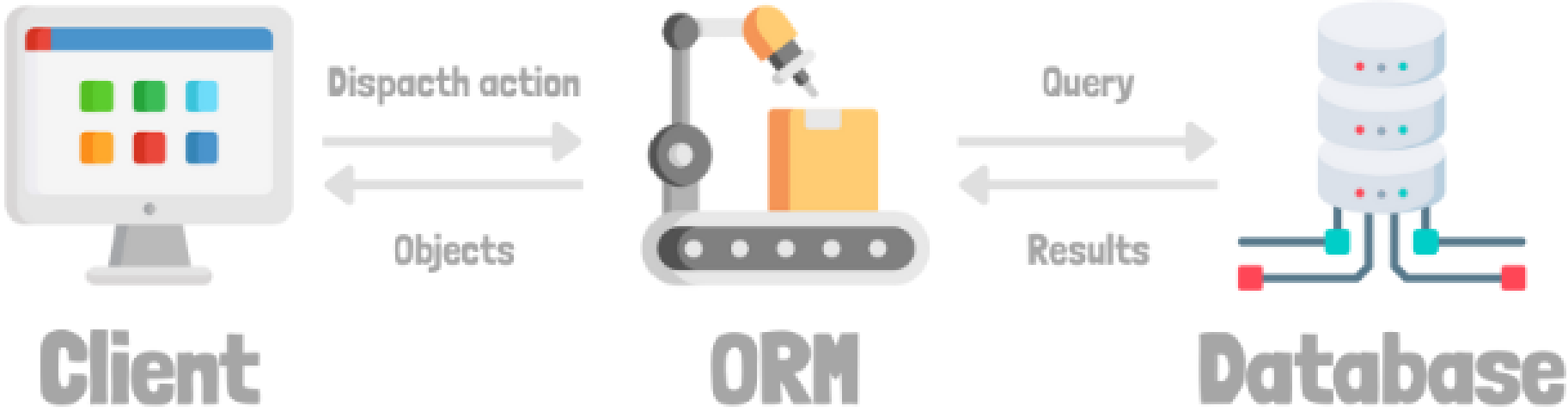
Mục Lục

Why Sequelize ?

- Sử dụng cơ chế ORM (Object Relational Mapping).
- Tránh thao tác trực tiếp với tầng database => không cần viết lệnh SQL dài dòng
- Cú pháp đơn giản dễ sử dụng
- Hỗ trợ migrates Code first

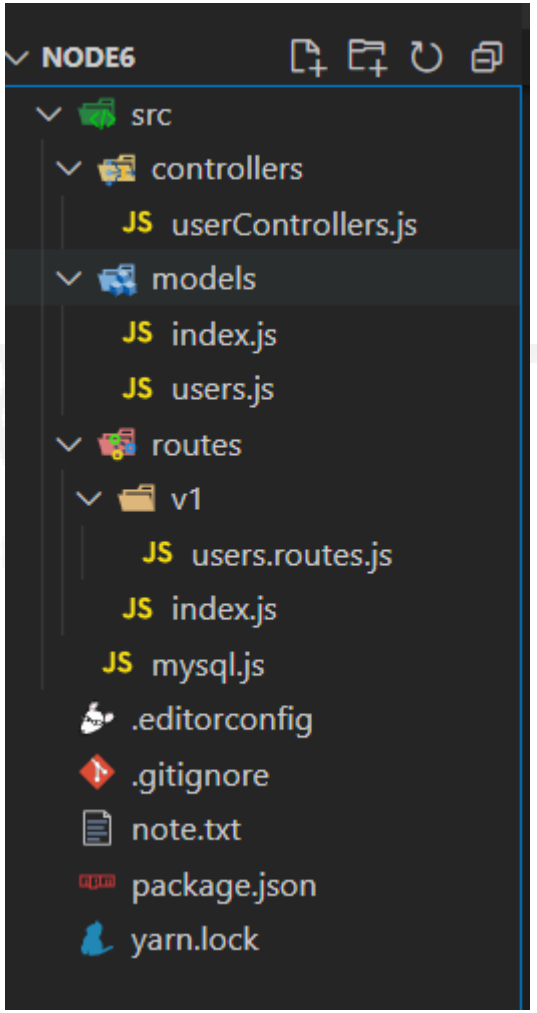
Kết quả đạt được

Thao tác với database lấy dữ liệu nhanh, đơn giản hơn



Cài đặt

- Trang chủ: https://sequelize.org
- Thêm thư viện: **yarn add sequelize**
- Structure

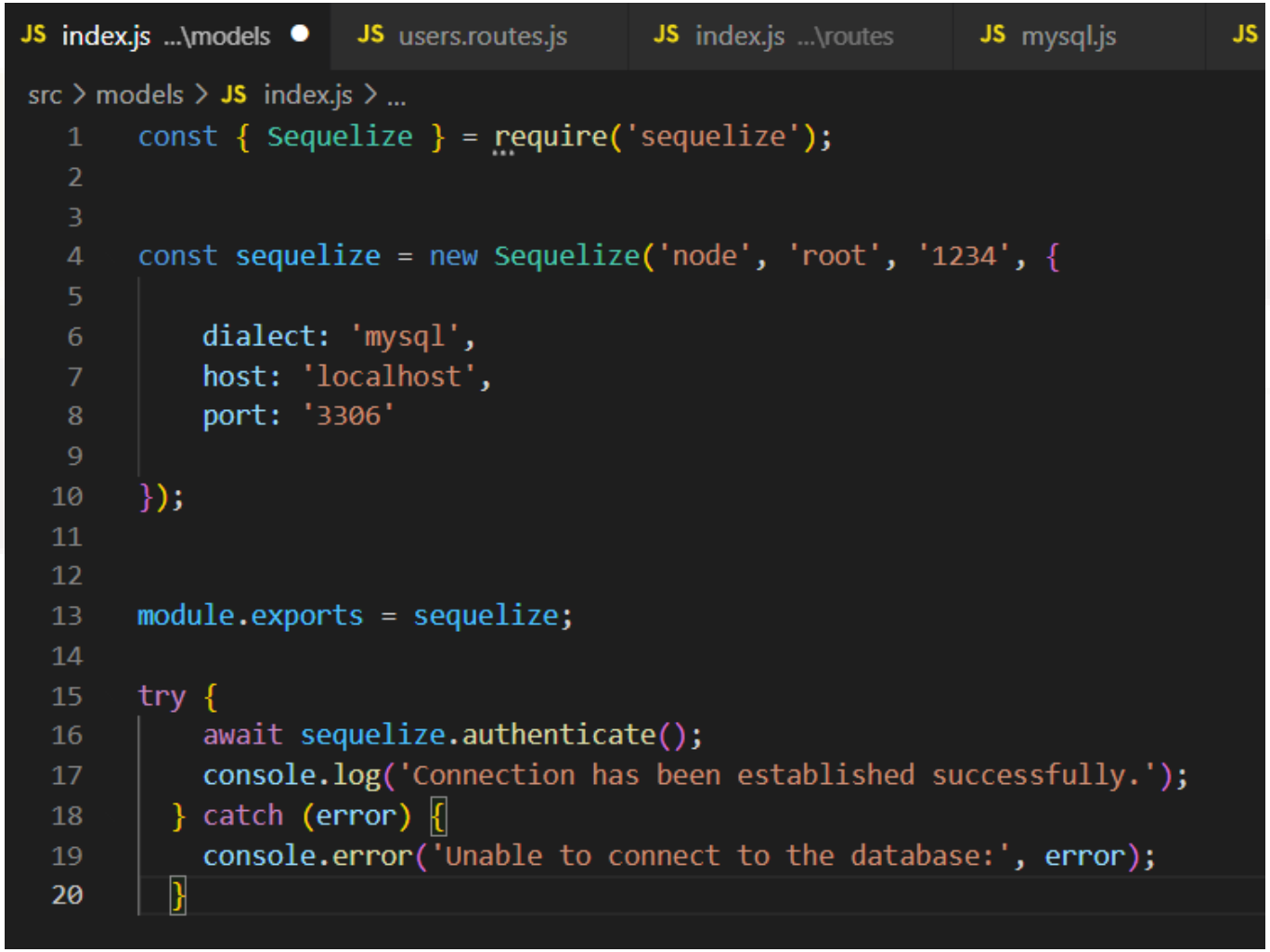


Cài đặt

```
const { Sequelize } = require('sequelize');

// Option 1: Passing a connection URI
const sequelize = new Sequelize('sqlite::memory:') // Example for sqlite
const sequelize = new Sequelize('postgres://user:pass@example.com:5432/dbname') // Example for postgres

// Option 2: Passing parameters separately (sqlite)
const sequelize = new Sequelize({
  dialect: 'sqlite',
  storage: 'path/to/database.sqlite'
});

// Option 3: Passing parameters separately (other dialects)
const sequelize = new Sequelize('database', 'username', 'password', {
  host: 'localhost',
  dialect: /* one of 'mysql' | 'mariadb' | 'postgres' | 'mssql' */
});
```

Cài đặt: Kết nối database (model/index.js)

- Tham khảo: https://sequelize.org/docs/v6/getting-started/



Cài đặt

Using `sequelize.define`:

```
const { Sequelize, DataTypes } = require('sequelize');
const sequelize = new Sequelize('sqlite::memory:');

const User = sequelize.define('User', {
  // Model attributes are defined here
  firstName: {
    type: DataTypes.STRING,
    allowNull: false
  },
  lastName: {
    type: DataTypes.STRING
    // allowNull defaults to true
  }
}, {
  // Other model options go here
});

// `sequelize.define` also returns the model
console.log(User === sequelize.models.User); // true
```

Extending Model

```
const { Sequelize, DataTypes, Model } = require('sequelize');
const sequelize = new Sequelize('sqlite::memory:');

class User extends Model {}

User.init({
  // Model attributes are defined here
  firstName: {
    type: DataTypes.STRING,
    allowNull: false
  },
  lastName: {
    type: DataTypes.STRING
    // allowNull defaults to true
  }
}, {
  // Other model options go here
  sequelize, // We need to pass the connection instance
  modelName: 'User' // We need to choose the model name
});

// the defined model is the class itself
console.log(User === sequelize.models.User); // true
```

Cài đặt:

Tạo model (user.js)

- Tham khảo: https://sequelize.org/docs/v6/core-concepts/model-basics/
- Properties:

  primaryKey

  allowNull

  autoIncrement

  unique

  validate

  defaultValue

```
JS users.js   JS index.js ...\models   JS users.routes.js   JS ind
src > models > JS users.js > ...
1  const { DataTypes, Model } = require('sequelize');
2  const sequelize = require('./index');
3
4
5  class User extends Model { }
6
7  User.init({
8      id: {
9          type: DataTypes.INTEGER,
10         primaryKey: true,
11         allowNull: false, // khong duoc rong
12         autoIncrement:true,
13         unique: true, // khong duoc trung
14         validate: {
15             isNumeric: {
16                 msg: "id is number",
17             }
18         }
19     },
20     hoTen: {
21         type: DataTypes.STRING
22     },
23     email: {
24         type: DataTypes.STRING,
25         unique: true
26
27     }
28 }, {
29
30     sequelize,
31     modelName: 'User',
32     tableName: 'user',
33     timestamps: false
34 });
35
36 module.exports = User;
```

Cài đặt: Datatypes

Numbers

```
DataTypes.INTEGER              // INTEGER
DataTypes.BIGINT               // BIGINT
DataTypes.BIGINT(11)           // BIGINT(11)

DataTypes.FLOAT                // FLOAT
DataTypes.FLOAT(11)            // FLOAT(11)
DataTypes.FLOAT(11, 10)        // FLOAT(11,10)

DataTypes.REAL                 // REAL           PostgreSQL only.
DataTypes.REAL(11)             // REAL(11)       PostgreSQL only.
DataTypes.REAL(11, 12)         // REAL(11,12)    PostgreSQL only.

DataTypes.DOUBLE               // DOUBLE
DataTypes.DOUBLE(11)           // DOUBLE(11)
DataTypes.DOUBLE(11, 10)       // DOUBLE(11,10)

DataTypes.DECIMAL              // DECIMAL
DataTypes.DECIMAL(10, 2)       // DECIMAL(10,2)
```

Strings

```
DataTypes.STRING               // VARCHAR(255)
DataTypes.STRING(1234)         // VARCHAR(1234)
DataTypes.STRING.BINARY        // VARCHAR BINARY
DataTypes.TEXT                 // TEXT
DataTypes.TEXT('tiny')         // TINYTEXT
DataTypes.CITEXT               // CITEXT          PostgreSQL and SQLite only.
DataTypes.TSVECTOR             // TSVECTOR        PostgreSQL only.
```

Boolean

```
DataTypes.BOOLEAN              // TINYINT(1)
```

Dates

```
DataTypes.DATE          // DATETIME for mysql /
DataTypes.DATE(6)       // DATETIME(6) for mysql
DataTypes.DATEONLY      // DATE without time
```

Querying database

- C: **User.create()**
- R: **User.findAll()**

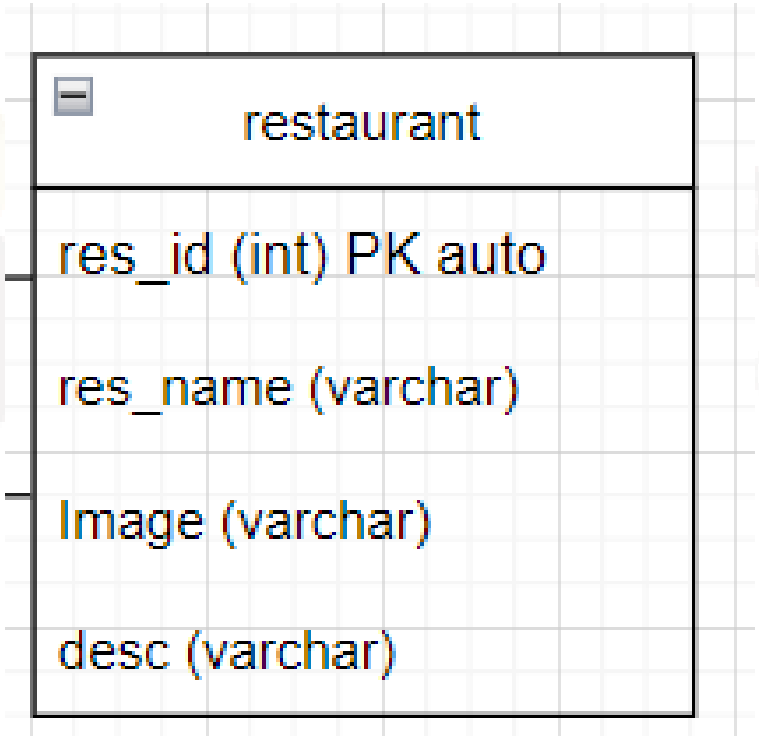
  User.findByPk()
- U: **User.update()**
- D: **User.destroy()**
- Tham khảo:

https://sequelize.org/docs/v6/core-concepts/model-querying-basics/

```
const User = require('../models/users');
const { Sequelize, where } = require('sequelize');
const Product = require('../models/products');
const Product_Type = require('../models/product_type');
const User_Product = require('../models/user_product');

const Op = Sequelize.Op;

//sequelize.query()

const getUsers = async (req, res) => {
    try {
        //get one
        const listUser = await User.findAll({
            where: {
                hoTen: {
                    [Op.like]: '%B%'
                }
            }
        });
        res.send(listUser);

    } catch (err) {
        res.send(err);
    }
}
```

Thực hành

Tạo API cho table restaurant



Relationships

```
Product.belongsTo(Product_Type, { foreignKey: 'typeId' });
Product_Type.hasMany(Product, { foreignKey: 'typeId' });
```

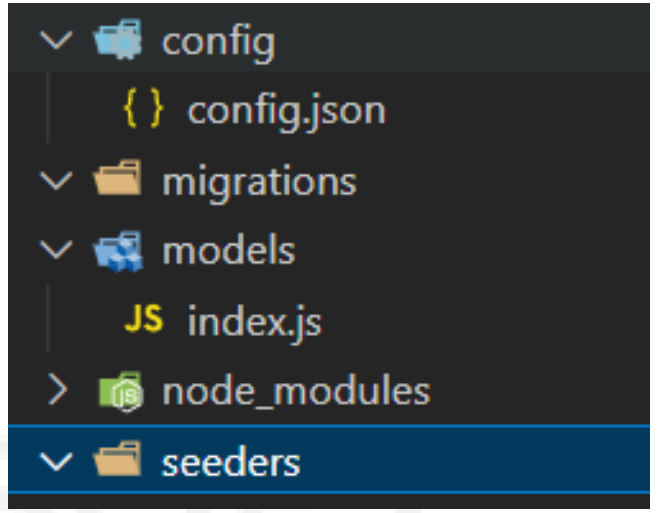
```
const getUser = await model.products.findAll({ include: Model });
```

- 1 – 1:
- 1 – n:
- n – n:

- The `HasOne` association
- The `BelongsTo` association
- The `HasMany` association
- The `BelongsToMany` association

Migrates

- Thư viện: **yarn add sequelize-cli**
- Run: **sequelize-cli init**
- Structure
- Lưu ý tên database phải tồn tại.
- Tạo một model User:



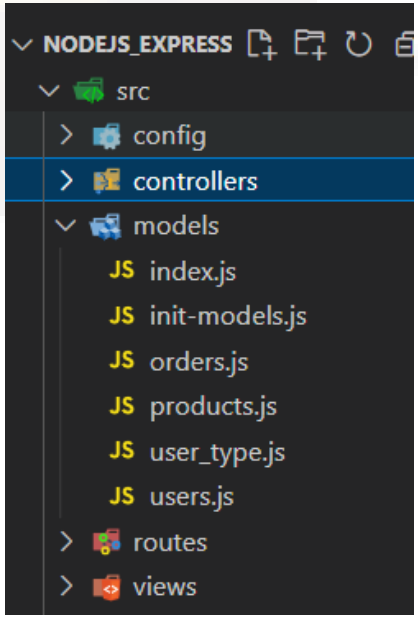
yarn sequelize-cli model:generate --name User --attributes firstName:string,lastName:string,email:string

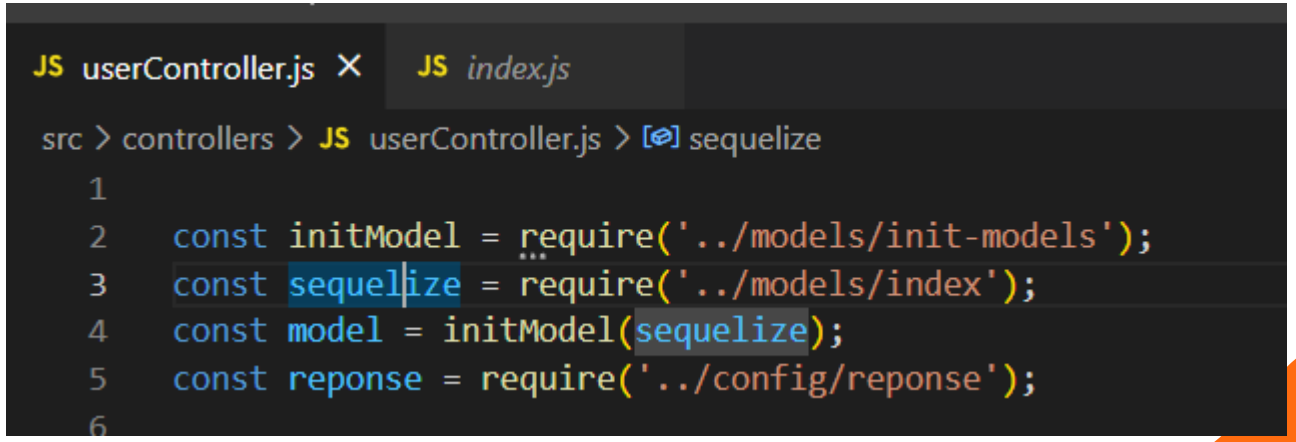
- Tạo table trong database:

sequelize-cli db:migrate

Migrates Database First

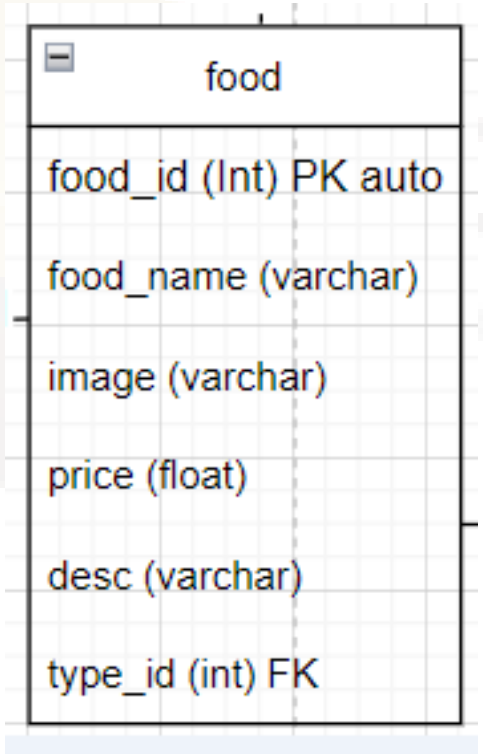
- Thư viện: **yarn add sequelize-auto**
- Run: **yarn sequelize-auto -h <host> -d <database> -u <user> -x [password] -p [port] --dialect [dialect] -o [/path/to/models] -l es6**
- Structure

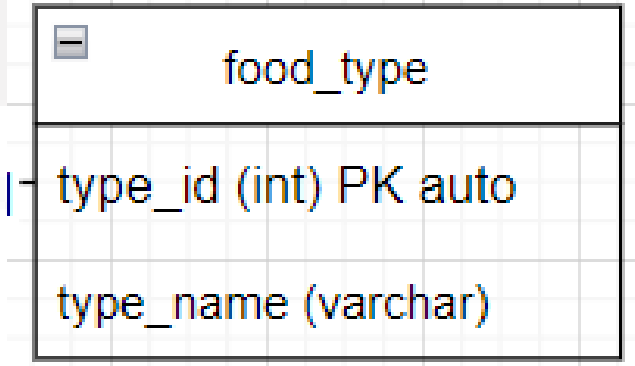




Thực hành

Tạo API cho table food và food_type





Upload

Cài đặt: **yarn add multer**

```
const multer = require('multer');

const upload = multer({ dest: './public/img' })
```

```
app.use(express.static("."))
const multer = require('multer');


const storage = multer.diskStorage({
    destination: (req, file, cb) => {
        cb(null, './public/img_compress');
    },
    filename: (req, file, cb) => {
        const uniqueSuffix = Date.now() + '_' + file.originalname;
        cb(null, uniqueSuffix);
    }
});
const upload = multer({ storage })

userRouter.post("/upload", upload.single('image'), async (req, res) => {
    // const result = fs.renameSync(`${process.cwd()}/public/${req.file.fi
```

Optimal

Cài đặt: **yarn add compress-images**

```
userRouter.post("/upload", upload.single('image'), async (req, res) => {

  const result = await compress_images(`${process.cwd()}/public/img_compress/${req.file.filename}`, `./public/img/`,
        { compress_force: false, statistic: true, autoupdate: true },
        false,
        { jpg: { engine: "mozjpeg", command: ["-quality", "25"] } },
        { png: { engine: "pngquant", command: ["--quality=20-50", "-o"] } },
        { svg: { engine: "svgo", command: "--multipass" } },
        { gif: { engine: "gifsicle", command: ["--colors", "64", "--use-col=web"] } },
        function (error, completed, statistic) {
            if (completed) {
                fs.unlinkSync(statistic.input);
                res.send(statistic.path_out_new)
            }

        })
        console.log(result)

})
```