mrmongoo-cli

A command-line tool to quickly generate **Mongoose models** and their **Joi validations**, **controllers**, **routes** for Node.js projects.
This tool can create register, login and authentication middleware.
This tool can create picture logic and multer middleware

Installation

Global installation (recommended for CLI usage)

```
npm install -g mrmongoo-cli
```

Local installation (per project)

```
npm install mrmongoo-cli --save-dev
```

You can then run the CLI using `npx`:

```
npx mrmongoo-create Test name:string:true:John
```

Usage

3 command lines to create common models, users administration or pictures handling

To create a common **model** :

```
npx mrmongoo-create <ModelName> [field:type:required:default ...]
```

To create the **users administration** with jwtoken authentication (authentication middleware):
email and *password* fields will be automatically created in **User** model. You can add optionnal fields.

```
npx mrmongoo-security <ModelName> [optionnalField:type:required:default ...]
```

To create **pictures handling** with **multer** middleware
name and *alt* fields will be automatically created in **Picture** model. You can add optionnal fields.

```
npx mrmongoo-multer <ModelName> [optionnalField:type:required:default ...]
```

Arguments:

- `ModelName`: Name of the model to create.
- `field:type:required:default` (optional, repeatable for multiple fields):
  - `field` — attribute name
  - `type` — field type (`string`, `number`, `boolean`, `date`), default: `string`
  - `required` — `true` or `false` (default: `false`)
  - `default` — default value (optional)

Example 1:

```
npx mrmongoo-create article title:string:true content:string
isPublished:boolean:true:false
```

This generates:

1. `models/article.model.js` — Mongoose model with timestamps.
2. `validation/article.validation.js` — Joi validation function for the model.
3. `controllers/article.controller.js` — controller with CRUD functions for the model.
4. `routes/article.route.js` — routes for each function of the controller.

Example 2:

```
npx mrmongoo-security user name:string:true isAdmin:boolean:true
```

This generates:

1. `models/user.model.js` — Mongoose model with timestamps (with *email* and *password* fields. including hashing password).
2. `validation/user.validation.js` — Joi validation function for the model.
3. `controllers/user.controller.js` — controller with CRUD functions for the model (including register and login).
4. `routes/user.route.js` — routes for each function of the controller.
5. `middleware/auth.js` — authentication middleware (needs jsonwebtoken module)

Example 3:

```
npx mrmongoo-multer picture
```

This generates:

1. `models/picture.model.js` — Mongoose model with timestamps (with *name* and *alt* fields).
2. `validation/picture.validation.js` — Joi validation function for the model.
3. `controllers/picture.controller.js` — controller with CRUD functions for the model.
4. `routes/picture.route.js` — routes for each function of the controller (with *upload.single()* on *create route*).
5. `middleware/multer.js` — multer middleware (needs multer module)

Features

- Automatically creates `models`, `controllers`, `routes` and `validation` folders if they don't exist.
- Prevents overwriting existing files.
- Capitalizes the model name automatically.

Generated file structure

```
project/
├─ controllers/
│  └─ User.controller.js
├─ middlewares/
│  └─ auth.js
│  └─ multer.js
├─ models/
│  └─ User.model.js
├─ routes/
│  └─ User.route.js
└─ validations/
   └─ User.validation.js
```

Dependencies

Dependencies

- bcrypt
- cors
- dotenv
- express
- joi

- jsonwebtoken
- mongoose
- multer
- nodemon

Tips

- Use `npx mrmongoo-cli` for one-off usage without installing globally.
- Always increment your package version when publishing fixes or new features.
- You can add more fields by repeating `field:type:required:default` in the CLI command.

License

MIT

```
---
```