mongoose

object modeling for node.js

Install

```
npm install --save mongoose
Var mongoose = require('mongoose');
mongoose.connect('mongodb://localhost:27017/{db_name}');
mongoose.connection.once('open',()=>{
 console.log('connected to mongoo')
})
mongoose.connection.on('error',(err)=>{
 console.log('connected to mongoo error ',err)
})
```

Models

```
var mongoose = require('mongoose');
var Schema = mongoose.Schema;
var userSchema = new Schema({
  name: {type: String, required: true},
  email: {type: String, required: true, unique: true,index: true},
  password: {type: String, required: true},
  age: {type: Number, required: true},
})
module.exports = User = mongoose.model('User', userSchema);
module.exports.getUserById = function (id, callback) {
  User.findById(id, callback)
}
//, { versionKey: false } // to remove __v
```

DataTypes

- String
- Number { type: Number, min: 18, max: 65 }
- Date { type: Date, default: Date.now }
- Boolean
- Array {type: [String] }

Insert Data

```
var User = require('../model/user');
var newUser = new User();
 newUser.name =’mina’;
 newUser.email = 'email5';
 newUser.password = 'password';
 newUser.age = 22;
 console.log(newUser)
 newUser.save(function (err, doc) {
   if (err) {
    throw error
   }
 });
```

Find Data

```
User.find(function(err,docs){
   if (err) {
     res.send(err);
   }else {
     console.log(docs);
   }
 })
User.find({age:{$gte:1}},function(err,docs){
   if (err) {
     res.send(err);
   }else {
     res.send(docs);
   }
})
//User.find().exec(function(err,docs)
```

Delete Data

```
User.findByIdAndRemove({'_id':'5ace6c8b9db9a6469e4d2bcd'},functio
n(err,doc){
   res.send(doc)
 })
doc.remove(function(err,doc){
})
```

Relations

```
var postSchema = Schema({
   user: { type: Schema.Types.ObjectId, ref: 'User' },
   title: String,
   content: String
});
module.exports = Post = mongoose.model('Post', postSchema);
var userSchema = new Schema({
   name: {type: String },
   posts: [{ type: Schema.Types.ObjectId, ref: 'Post' }],
})
module.exports = User = mongoose.model('User', userSchema);
```

Relations

For insert

user.posts.push(PostId)

Post.user = userId

Relations

For retrieve use **populate**

```
User.find().populate('posts').exec(function(err,docs){
        res.send(docs)
 })
Post.find().populate('author').exec(function(err,docs){
     res.send(docs)
 })
```