**Créer RESTAPI avec NodeJS / Express, MongoDB utilisant Mongoose**

Dans le tutoriel, nous montrons comment construire des RestAPI CRUD avec NodeJS / Express et MongoDB en utilisant Mongoose.

**Objectif**

Concevoir, créer des modèles et interagir avec MongoDB en utilisant Mongoose.

**Etapes**

Créer un projet NodeJs / Express : nodejs-restapi-mongodb

Créer l’arborescence suivante :

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | /nodejs-restapi-mongodb  /app  /config  mongodb.config.js  /controllers  customer.controller.js  /models  customer.model.js  /routes  customer.routes.js  /node\_modules  package.json  server.js |

Installer mongoose :

|  |  |
| --- | --- |
| 1 | npm install mongoose |
|  |  |

Installez Express / Body-parse / mongoose en ligne de commande:

|  |  |
| --- | --- |
| 1 | npm install express body-parser mongoose --save |

***Schéma de Mongoose***

Dans le dossier racine, créez un fichier 'app/models/customer.model.js' comme ci-dessous:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | const mongoose = require('mongoose');    const CustomerSchema = mongoose.Schema({      firstname: String,      lastname: String,  age: { type: Number, min: 18, max: 65, required: true }  });    module.exports = mongoose.model('Customer', CustomerSchema); |

***Configurer l'URL de MongoDB***

Dans le dossier racine, créez un fichier 'app/config/mongodb.config.js' comme ci-dessous:

|  |  |
| --- | --- |
| 1  2  3 | module.exports = {      url: 'mongodb://localhost:27017/nodejs-demo'  } |

***Se connecter à MongoDB***

Dans le fichier 'server.js', configurez la connexion mongodb:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31 | var express = require('express');  var app = express();  var bodyParser = require('body-parser');  app.use(bodyParser.json())    // Configuring the database  const dbConfig = require('./app/config/mongodb.config.js');  const mongoose = require('mongoose');    mongoose.Promise = global.Promise;    // Connecting to the database  mongoose.connect(dbConfig.url)  .then(() => {      console.log("Successfully connected to MongoDB.");  }).catch(err => {      console.log('Could not connect to MongoDB.');      process.exit();  });    require('./app/routes/customer.routes.js')(app);    // Create a Server  var server = app.listen(8081, function () {      var host = server.address().address    var port = server.address().port      console.log("App listening at http://%s:%s", host, port)    }) |

**Implémenter des API CRUD**

***Routes***

Dans le dossier racine, créez un fichier 'app/routes/customer.routes.js' comme ci-dessous le contenu:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | module.exports = function(app) {        var customers = require('../controllers/customer.controller.js');        // Create a new Customer      app.post('/api/customers', customers.create);        // Retrieve all Customer      app.get('/api/customers', customers.findAll);        // Retrieve a single Customer by Id      app.get('/api/customers/:customerId', customers.findOne);        // Update a Customer with Id      app.put('/api/customers/:customerId', customers.update);        // Delete a Customer with Id      app.delete('/api/customers/:customerId', customers.delete);  } |

***Contrôleurs***

Créez un fichier '/app/controllers/customer.controller.js' contenant les méthodes d'exécution des demandes d'URL ci-dessus:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107 | const Customer = require('../models/customer.model.js');      // POST a Customer  exports.create = (req, res) => {      // Create a Customer      const customer = new Customer({          firstname: req.body.firstname,  lastname: req.body.lastname,          age: req.body.age      });        // Save a Customer in the MongoDB      customer.save()      .then(data => {          res.send(data);      }).catch(err => {          res.status(500).send({              message: err.message          });      });  };      // FETCH all Customers  exports.findAll = (req, res) => {      Customer.find()      .then(customers => {          res.send(customers);      }).catch(err => {          res.status(500).send({              message: err.message          });      });  };      // FIND a Customer  exports.findOne = (req, res) => {      Customer.findById(req.params.customerId)      .then(customer => {          if(!customer) {              return res.status(404).send({                  message: "Customer not found with id " + req.params.customerId              });          }          res.send(customer);      }).catch(err => {          if(err.kind === 'ObjectId') {              return res.status(404).send({                  message: "Customer not found with id " + req.params.customerId              });          }          return res.status(500).send({              message: "Error retrieving Customer with id " + req.params.customerId          });      });  };    // UPDATE a Customer  exports.update = (req, res) => {      // Find customer and update it      Customer.findByIdAndUpdate(req.params.customerId, {          firstname: req.body.firstname,  lastname: req.body.lastname,  age: req.body.age      }, {new: true})      .then(customer => {          if(!customer) {              return res.status(404).send({                  message: "Customer not found with id " + req.params.customerId              });          }          res.send(customer);      }).catch(err => {          if(err.kind === 'ObjectId') {              return res.status(404).send({                  message: "Customer not found with id " + req.params.customerId              });          }          return res.status(500).send({              message: "Error updating customer with id " + req.params.customerId          });      });  };    // DELETE a Customer  exports.delete = (req, res) => {      Customer.findByIdAndRemove(req.params.customerId)      .then(customer => {          if(!customer) {              return res.status(404).send({                  message: "Customer not found with id " + req.params.customerId              });          }          res.send({message: "Customer deleted successfully!"});      }).catch(err => {          if(err.kind === 'ObjectId' || err.name === 'NotFound') {              return res.status(404).send({                  message: "Customer not found with id " + req.params.customerId              });          }          return res.status(500).send({              message: "Could not delete customer with id " + req.params.customerId          });      });  }; |

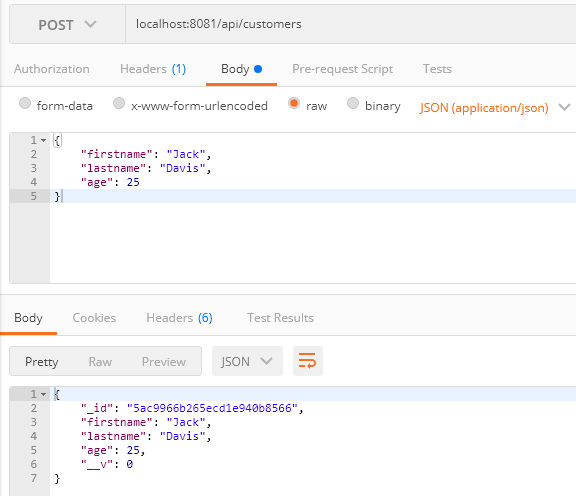
**Résultats Run & Check**

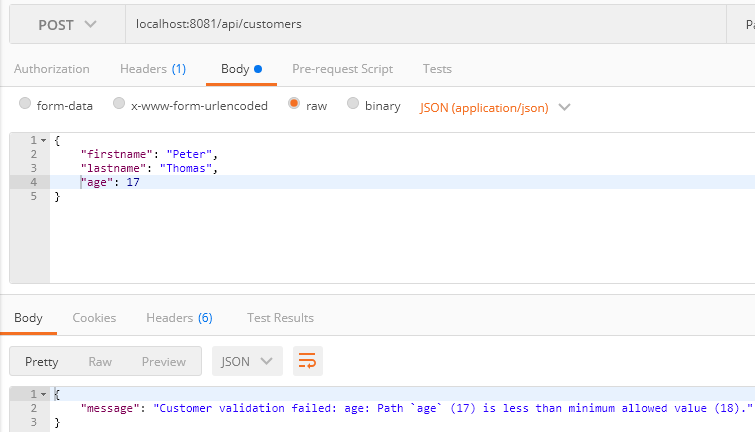
Exécutez le projet NodeJs / Express:

|  |  |
| --- | --- |
| 1  2  3 | nodejs-restapi-mongodb>node server.js  App listening at http://:::8081  Successfully connected to MongoDB. |

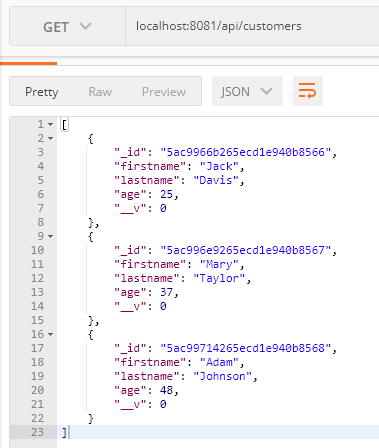
Utilisez **Postman** pour vérifier le résultat ->

- POST un client:

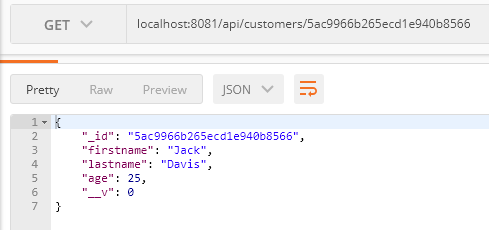




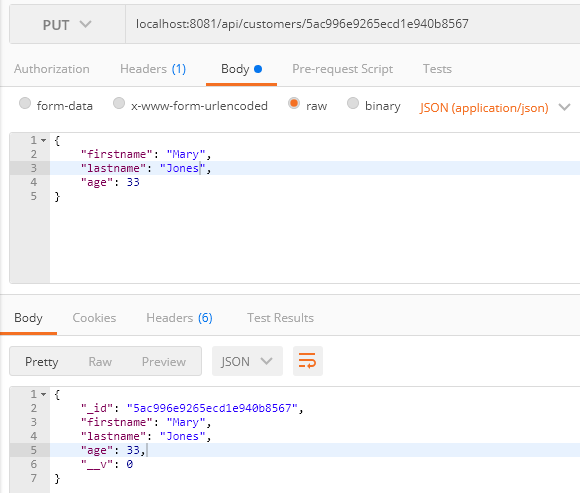
- GET tous les clients:



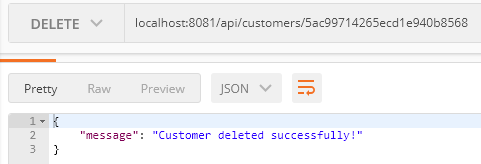
- GET UN CLIENT PAR ID:

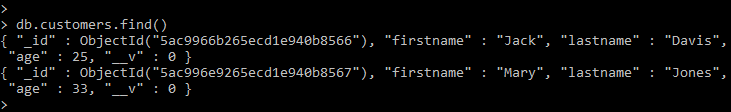


- PUT un client:



- SUPPRIMER un client:





Source : <https://grokonez.com/node-js/crud-restapis-with-nodejs-express-mongodb>