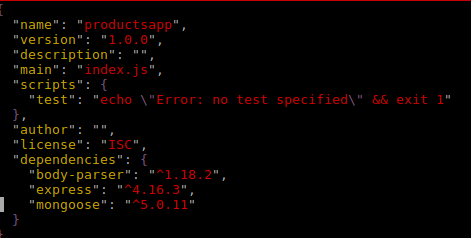
**Name – Saburi Sadashiv Giri**

* Build a RESTful API that can /create/read/update/delete

**npm init**



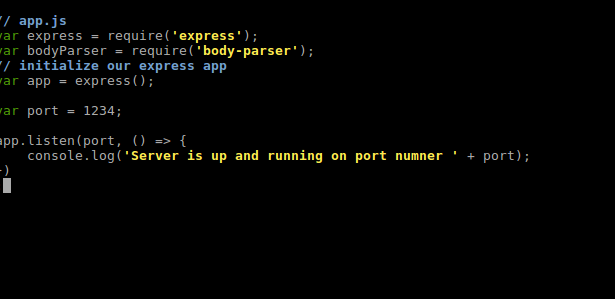


Initializing the Server:-

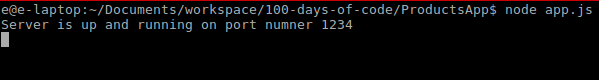
touch app.js

// app.js**const** express = require('express');  
**const** bodyParser = require('body-parser');**//** initialize our express app **const** app = express();

**let** port = 1234;  
  
app.listen(port, () => {  
 console.log('Server is up and running on port numner ' + port);  
});



node app.js



**const** mongoose = require('mongoose');  
**const** Schema = mongoose.Schema;  
  
**let** ProductSchema = **new** Schema({  
 name: {type: String, required: **true**, max: 100},  
 price: {type: Number, required: **true**},  
});  
  
  
// Export the model  
module.exports = mongoose.model('Product', ProductSchema);

***Routes***:

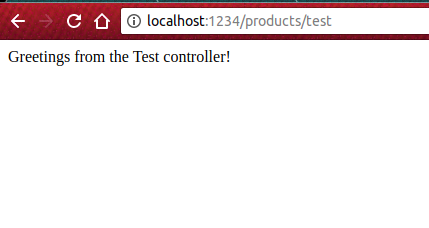
**const** express = require('express');  
**const** router = express.Router();  
  
// Require the controllers WHICH WE DID NOT CREATE YET!!  
**const** product\_controller = require('../controllers/product.controller');  
  
  
// a simple test url to check that all of our files are communicating correctly.  
router.get('/test', product\_controller.test);module.exports = router;

**Controllers:**

**const** Product = require('../models/product.model');  
  
//Simple version, without validation or sanitation  
exports.test = **function** (req, res) {  
 res.send('Greetings from the Test controller!');  
};

//app.js**const** express = require('express');  
**const** bodyParser = require('body-parser');**const** product = require('./routes/product.route'); // Imports routes for the products  
**const** app = express();app.use('/products', product);**let** port = 1234;app.listen(port, () => {  
 console.log('Server is up and running on port numner ' + port);  
});

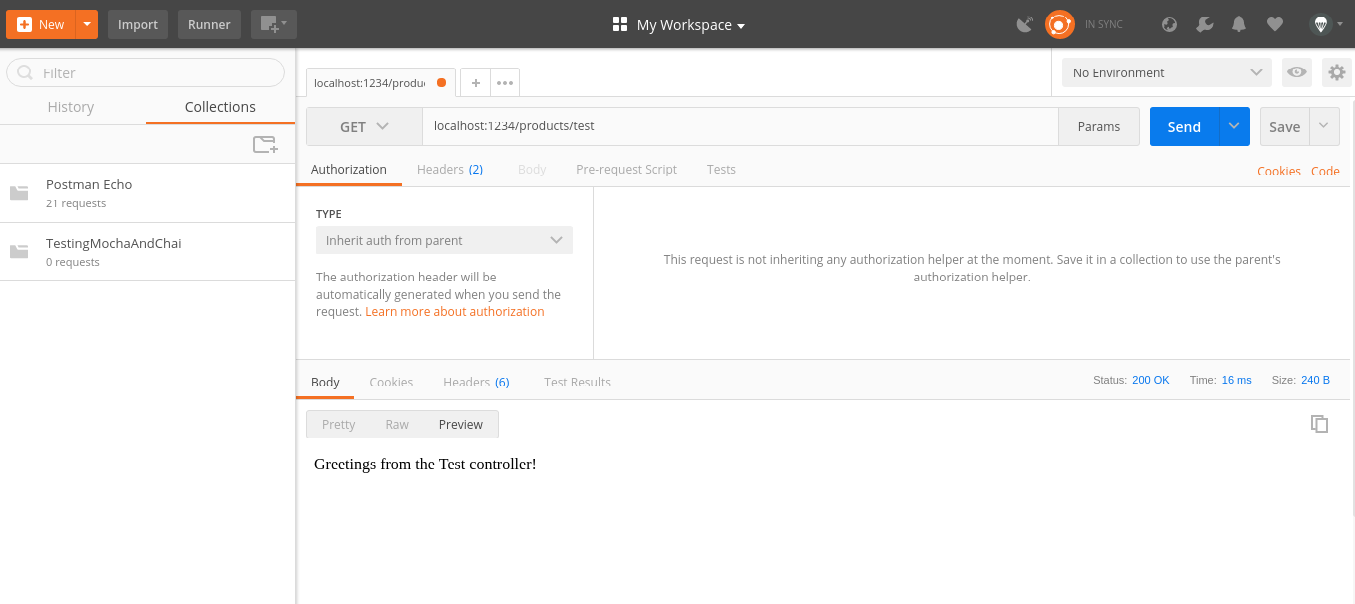
Now head to your browser and try the following link: <http://localhost:1234/products/test>



Validating that our test route is working…

# Postman:

1. Install Postman
2. Open the app, make sure it’s a GET request and type the following url ‘localhost:1234/products/test’.



Connecting to Database

// Set up mongoose connection  
**const** mongoose = require('mongoose');  
**let** dev\_db\_url = 'mongodb://someuser:abcd1234@ds123619.mlab.com:23619/productstutorial';  
**let** mongoDB = process.env.MONGODB\_URI || dev\_db\_url;  
mongoose.connect(mongoDB);  
mongoose.Promise = global.Promise;  
**let** db = mongoose.connection;  
db.on('error', console.error.bind(console, 'MongoDB connection error:'));

# Body Parser

In you ***app.js*** file, add the following couple of lines.

app.use(bodyParser.json());  
app.use(bodyParser.urlencoded({extended: **false**}));

Here is how our full ***app.js*** file looks like

// app.js  
  
**const** express = require('express');  
**const** bodyParser = require('body-parser');  
  
**const** product = require('./routes/product.route'); // Imports routes for the products  
**const** app = express();  
  
// Set up mongoose connection  
**const** mongoose = require('mongoose');  
**let** dev\_db\_url = 'mongodb://someuser:abcd1234@ds123619.mlab.com:23619/productstutorial';  
**const** mongoDB = process.env.MONGODB\_URI || dev\_db\_url;  
mongoose.connect(mongoDB);  
mongoose.Promise = global.Promise;  
**const** db = mongoose.connection;  
db.on('error', console.error.bind(console, 'MongoDB connection error:'));  
  
app.use(bodyParser.json());  
app.use(bodyParser.urlencoded({extended: **false**}));  
app.use('/products', product);  
  
**let** port = 1234;  
  
app.listen(port, () => {  
 console.log('Server is up and running on port numner ' + port);  
});

## **CREATE**

// routes/products.route.js...  
router.post('/create', product\_controller.product\_create);

// controllers/products.jsexports.product\_create = **function** (req, res) {  
 **let** product = **new** Product(  
 {

name: req.body.name,

price: req.body.price,

id: req.body.id,

qty: req.body.qty,

instock: req.body.instock,

discontinued: req.body.discontinued,

Cid: req.body.Cid,

Cname: req.body.Cname  
 }  
 );  
  
 product.save(**function** (err) {  
 **if** (err) {  
 **return** next(err);  
 }  
 res.send('Product Created successfully')  
 })  
};

Last step would be validating that we can easily create a new product. Let’s open Postman. Let’s send a POST request to the following url ‘localhost:1234/products/create’ and specify the POST data as name: apple and price: 15 as a test example.

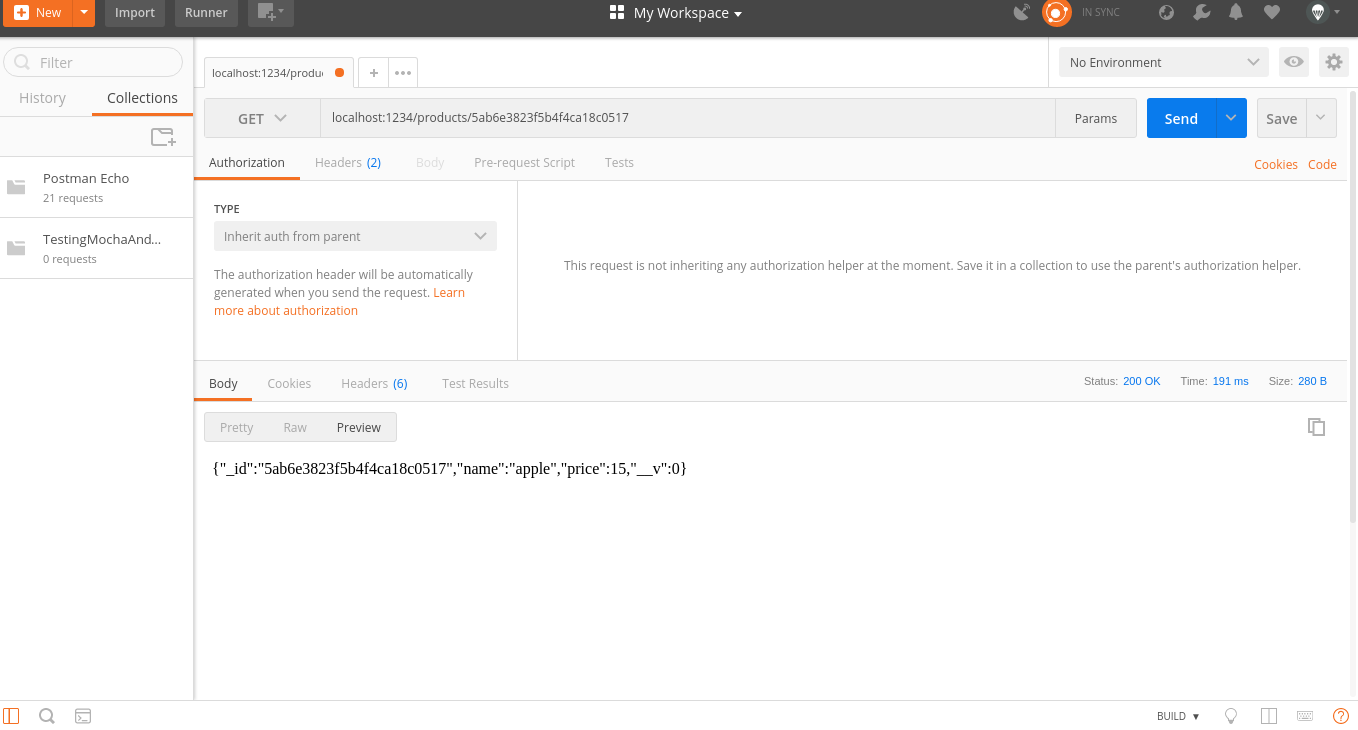
We can see that the response is ‘Product Created successfully. This means that the router and the controller are working correctly. To double check that an ‘Apple’ product was created.

## **Read**

// routes/products.route.js...  
router.get('/:id', product\_controller.product\_details);

// controllers/products.controller.jsexports.product\_details = **function** (req, res) {  
 Product.findById(req.params.id, **function** (err, product) {  
 **if** (err) **return** next(err);  
 res.send(product);  
 })  
};

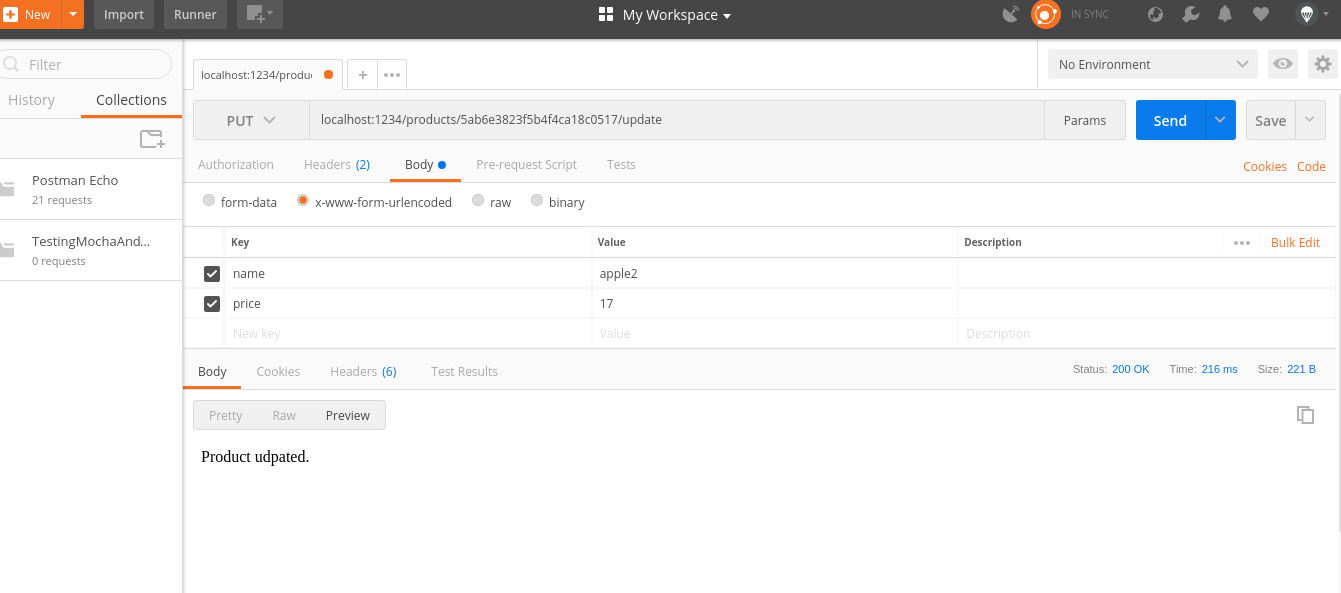
Now let’s head to Postman and try-out our new endpoint. Call the following url ‘localhost:1234/products/PRODUCT\_ID’



## **Update**

// routes/products.route.js...  
router.put('/:id/update', product\_controller.product\_update);

// controllers/products.controller.js...  
exports.product\_update = **function** (req, res) {  
 Product.findByIdAndUpdate(req.params.id, {$set: req.body}, **function** (err, product) {  
 **if** (err) **return** next(err);  
 res.send('Product udpated.');  
 });  
};



We have updated the product name to ‘apple2’ and we can see a response saying ‘Product updated.’

## **Delete**

// routes/products.route.js...  
router.delete('/:id/delete', product\_controller.product\_delete);

// controllers/products.controller.jsexports.product\_delete = **function** (req, res) {  
 Product.findByIdAndRemove(req.params.id, **function** (err) {  
 **if** (err) **return** next(err);  
 res.send('Deleted successfully!');  
 })  
};

