[**RESTful API Using Node and Express 4**](https://codeforgeek.com/2015/03/restful-api-node-and-express-4/)

* [SHARE ON FACEBOOK](https://www.facebook.com/sharer/sharer.php?u=https://codeforgeek.com/2015/03/restful-api-node-and-express-4/)
* [TWEET THIS](https://twitter.com/intent/tweet?source=https://codeforgeek.com/2015/03/restful-api-node-and-express-4/&text=RESTful%20API%20Using%20Node%20and%20Express%204)
* [SHARE ON GOOGLE+](https://plus.google.com/share?url=https://codeforgeek.com/2015/03/restful-api-node-and-express-4/)
* [PIN THIS](http://pinterest.com/pin/create/button/?url=https://codeforgeek.com/2015/03/restful-api-node-and-express-4/&description=RESTful%20API%20Using%20Node%20and%20Express%204)
* AUTHOR

[Shahid (UnixRoot) Shaikh](https://codeforgeek.com/author/shahid/)

* LAST EDITED

[Mar 16, 2015](https://codeforgeek.com/2015/03/restful-api-node-and-express-4/)

* COMMENTS

124

In this tutorial we’ll learn how to create basic REST API’s using Node.js and Express which retrieve data from MySQL database and give response in JSON format.

This tutorial is for the beginners and intermediate users of Node.js.

**Last update :**

This post is updated with the request of users at 3/21/2015. Following are the change log.

* Make SQL queries prepared statement.
* Make passwords hash, do not use it as plain text.
* Updating REST api’s.

Code is updated on Github with above changes so you can download latest one from there.

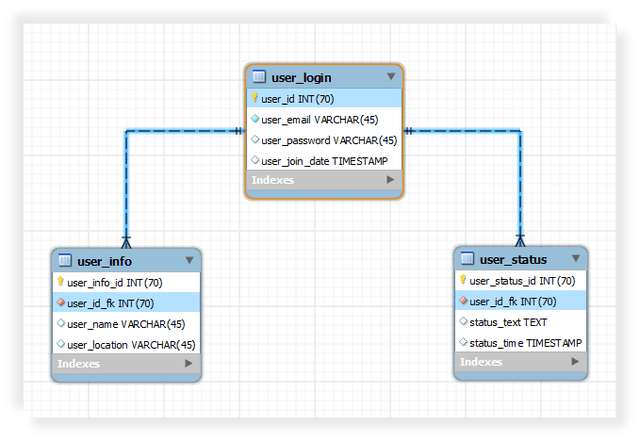
**Introduction :**

What is REST ? As per WikiPedia it is

Representational State Transfer (REST) is a software architecture style consisting of guidelines and best practices for creating scalable web services.REST is a coordinated set of constraints applied to the design of components in a distributed hypermedia system that can lead to a more performant and maintainable architecture.

Practically speaking if web services supports Uniform behavior, are stateless, allows caching, and provides high level of abstraction of data then it is more likely called as REST api’s.

**Database design :**

  
**user\_login** will store login information for particular user and each user gets an unique ID. **user\_info** will store more information about user and will have foreign key constraint to **user\_login** table and similarly for **user\_status** table.

[DOWNLOAD CODE](https://github.com/codeforgeek/RESTful-api-using-nodejs/archive/master.zip)

Create database name as **“restful\_api\_demo”** in MySQL via phpmyadmin and go to SQL tab and copy / paste following code to create above tables.

SQL code

SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0;  
SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0;  
SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='TRADITIONAL,ALLOW\_INVALID\_DATES';  
  
CREATE SCHEMA IF NOT EXISTS `restful\_api\_demo` DEFAULT CHARACTER SET latin1 ;  
USE `restful\_api\_demo` ;  
  
*-- -----------------------------------------------------*  
*-- Table `restful\_api\_demo`.`user\_login`*  
*-- -----------------------------------------------------*  
DROP TABLE IF EXISTS `restful\_api\_demo`.`user\_login` ;  
  
CREATE TABLE IF NOT EXISTS `restful\_api\_demo`.`user\_login` (  
  `user\_id` INT(70) NOT NULL AUTO\_INCREMENT,  
  `user\_email` VARCHAR(45) NOT NULL,  
  `user\_password` VARCHAR(45) NULL,  
  `user\_join\_date` TIMESTAMP NULL DEFAULT CURRENT\_TIMESTAMP,  
  PRIMARY KEY (`user\_id`),  
  UNIQUE INDEX `user\_email\_UNIQUE` (`user\_email` ASC))  
ENGINE = InnoDB;  
  
  
*-- -----------------------------------------------------*  
*-- Table `restful\_api\_demo`.`user\_info`*  
*-- -----------------------------------------------------*  
DROP TABLE IF EXISTS `restful\_api\_demo`.`user\_info` ;  
  
CREATE TABLE IF NOT EXISTS `restful\_api\_demo`.`user\_info` (  
  `user\_info\_id` INT(70) NOT NULL AUTO\_INCREMENT,  
  `user\_id\_fk` INT(70) NOT NULL,  
  `user\_name` VARCHAR(45) NULL,  
  `user\_location` VARCHAR(45) NULL,  
  PRIMARY KEY (`user\_info\_id`),  
  UNIQUE INDEX `user\_id\_fk\_UNIQUE` (`user\_id\_fk` ASC),  
  CONSTRAINT `user\_info\_foreign\_key`  
    FOREIGN KEY (`user\_id\_fk`)  
    REFERENCES `restful\_api\_demo`.`user\_login` (`user\_id`)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION)  
ENGINE = InnoDB;  
  
  
*-- -----------------------------------------------------*  
*-- Table `restful\_api\_demo`.`user\_status`*  
*-- -----------------------------------------------------*  
DROP TABLE IF EXISTS `restful\_api\_demo`.`user\_status` ;  
  
CREATE TABLE IF NOT EXISTS `restful\_api\_demo`.`user\_status` (  
  `user\_status\_id` INT(70) NOT NULL AUTO\_INCREMENT,  
  `user\_id\_fk` INT(70) NOT NULL,  
  `status\_text` TEXT NULL DEFAULT NULL,  
  `status\_time` TIMESTAMP NULL DEFAULT CURRENT\_TIMESTAMP,  
  PRIMARY KEY (`user\_status\_id`),  
  UNIQUE INDEX `user\_id\_fk\_UNIQUE` (`user\_id\_fk` ASC),  
  CONSTRAINT `user\_status\_foreign\_key`  
    FOREIGN KEY (`user\_id\_fk`)  
    REFERENCES `restful\_api\_demo`.`user\_login` (`user\_id`)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION)  
ENGINE = InnoDB;  
  
SET SQL\_MODE=@OLD\_SQL\_MODE;  
SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS;  
SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS;

You can use this code and run it in PHPMyadmin to create databases and tables.

**Our project :**

Before moving to code , let me explain you directory structure.

-----+ node\_modules ( folder )  
--------+ package.json //contains package information.  
--------+ Server.js //contains Express code  
--------+ REST.js // contains api code

First of all let’s install dependencies we need. Create project directory and create package.json file and paste following code.

Package.json

{  
  "name": "RESTful-API",  
  "version": "0.0.1",  
  "scripts": {  
    "start": "node Server.js"  
  },  
  "dependencies": {  
    "express": "~4.12.2",  
    "mysql": "~2.5.5",  
    "body-parser": "~1.12.0",  
    "MD5": "~1.2.1"  
  }  
}

Install dependencies by typing

npm install

Once you have done installing it move to next step !

**Let’s say “Hello World !”**

Let’s build one simple api which will return “Hello World” every time we do hit it. Here is Server.js file. I am using prototype programming style to modular the code.

Server.js

var express = require("express");  
var mysql   = require("mysql");  
var bodyParser  = require("body-parser");  
var md5 = require('MD5');  
var rest = require("./REST.js");  
var app  = express();  
  
function REST(){  
    var self = this;  
    self.connectMysql();  
};  
  
REST.prototype.connectMysql = function() {  
    var self = this;  
    var pool      =    mysql.createPool({  
        connectionLimit : 100,  
        host     : 'localhost',  
        user     : 'root',  
        password : '',  
        database : 'restful\_api\_demo',  
        debug    :  false  
    });  
    pool.getConnection(function(err,connection){  
        if(err) {  
          self.stop(err);  
        } else {  
          self.configureExpress(connection);  
        }  
    });  
}  
  
REST.prototype.configureExpress = function(connection) {  
      var self = this;  
      app.use(bodyParser.urlencoded({ extended: true }));  
      app.use(bodyParser.json());  
      var router = express.Router();  
      app.use('/api', router);  
      var rest\_router = new rest(router,connection,md5);  
      self.startServer();  
}  
  
REST.prototype.startServer = function() {  
      app.listen(3000,function(){  
          console.log("All right ! I am alive at Port 3000.");  
      });  
}  
  
REST.prototype.stop = function(err) {  
    console.log("ISSUE WITH MYSQL n" + err);  
    process.exit(1);  
}  
  
new REST();

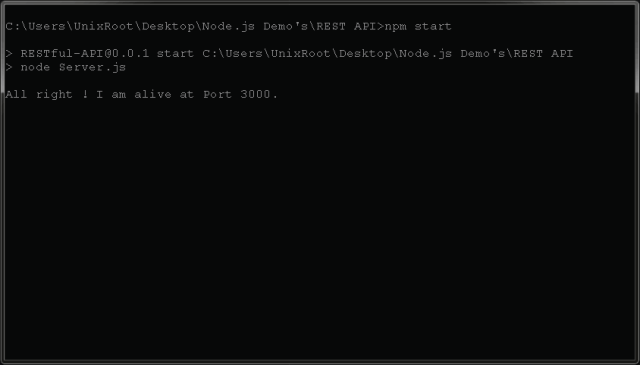
Here is REST.js.

REST.js

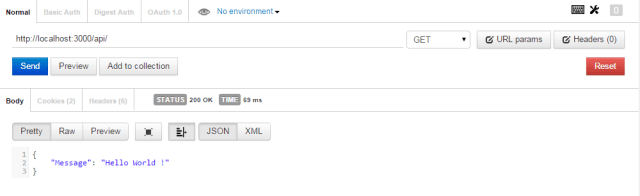
function REST\_ROUTER(router,connection,md5) {  
    var self = this;  
    self.handleRoutes(router,connection,md5);  
}  
  
REST\_ROUTER.prototype.handleRoutes= function(router,connection,md5) {  
    router.get("/",function(req,res){  
        res.json({"Message" : "Hello World !"});  
    })  
}  
  
module.exports = REST\_ROUTER;

Save both of the file. Make sure [MySQL](http://codeforgeek.com/2014/12/cluster-node-js-performance/) is running and your system have npm and node installed. Go to project directory and type

npm start

to execute the project. You should see something like this on console.  
  
Now as we have our server running on port 3000, we can test our api. You can use any REST simulator but recommended for you is POSTMAN. Download it for Chrome from [here](https://chrome.google.com/webstore/detail/postman-rest-client/fdmmgilgnpjigdojojpjoooidkmcomcm?hl=en).

Here is how to hit the api using Postman.



Now we have our “Hello World” api ready, let’s develop some real stuff !

**Building API’s :**

Here is list of api’s we are going to build. Changes will be done in **REST.js** file only.

|  |  |  |
| --- | --- | --- |
| **Api** | **Type** | **Description** |
| /users | POST | Takes email and password as input data and add new user. |
| /users | GET | Returns every users from database |
| /users/:userId | GET | Returns users from database with match of userId. |
| /users/:email/ | PUT | Update password of user by passing email ID. |
| /users/:email | DELETE | Delete user from database. |

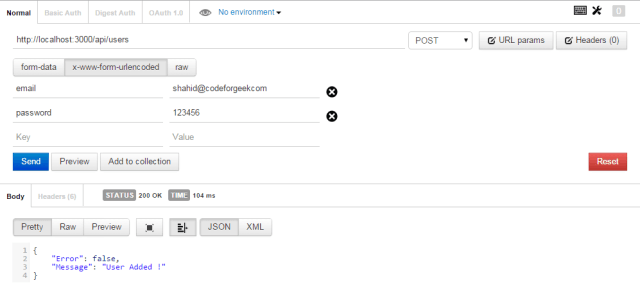
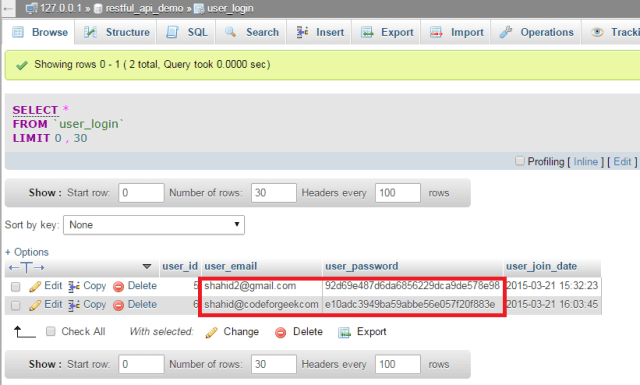
**#1: POST /users**

This api will take email and password and inserts in the database. Here is the code

In practical scenarios, never store password in plain text. Always encrypt and store in DB  
We are using MD5 module to hash the password and insert them in database.

REST.js

var mysql = require("mysql");  
function REST\_ROUTER(router,connection,md5) {  
    var self = this;  
    self.handleRoutes(router,connection,md5);  
}  
  
REST\_ROUTER.prototype.handleRoutes= function(router,connection,md5) {  
    router.get("/",function(req,res){  
           ...................  
    });  
    router.post("/users",function(req,res){  
        var query = "INSERT INTO ??(??,??) VALUES (?,?)";  
        var table =["user\_login","user\_email","user\_password",req.body.email,md5(req.body.password)];  
        query = mysql.format(query,table);  
        connection.query(query,function(err,rows){  
            if(err) {  
                res.json({"Error" : true, "Message" : "Error executing MySQL query"});  
            } else {  
                res.json({"Error" : false, "Message" : "User Added !"});  
            }  
        });  
    });  
}  
  
module.exports = REST\_ROUTER;

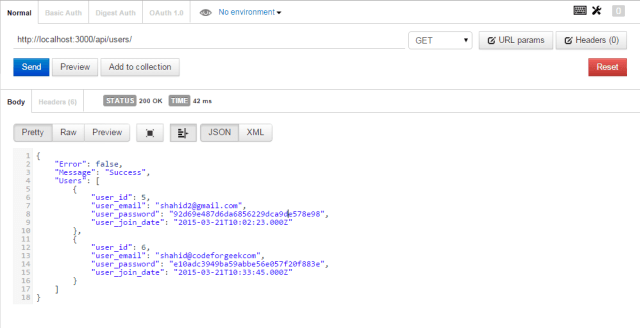
Here is the output of above code.  
  
Have a look at database where new user is added.  


**#2 : GET /users**

This api returns all users from database. Here is the code.

REST.js

var mysql = require("mysql");  
function REST\_ROUTER(router,connection,md5) {  
    var self = this;  
    self.handleRoutes(router,connection,md5);  
}  
  
REST\_ROUTER.prototype.handleRoutes= function(router,connection,md5) {  
    router.get("/",function(req,res){  
              ...........  
    });  
  
    router.post("/users",function(req,res){  
              ...........  
    });  
   router.get("/users",function(req,res){  
        var query = "SELECT \* FROM ??";  
        var table = ["user\_login"];  
        query = mysql.format(query,table);  
        connection.query(query,function(err,rows){  
            if(err) {  
                res.json({"Error" : true, "Message" : "Error executing MySQL query"});  
            } else {  
                res.json({"Error" : false, "Message" : "Success", "Users" :rows});  
            }  
        });  
    });  
  
    router.get("/users/:user\_id",function(req,res){  
        var query = "SELECT \* FROM ?? WHERE ??=?";  
        var table = ["user\_login","user\_id",req.params.user\_id];  
        query = mysql.format(query,table);  
        connection.query(query,function(err,rows){  
            if(err) {  
                res.json({"Error" : true, "Message" : "Error executing MySQL query"});  
            } else {  
                res.json({"Error" : false, "Message" : "Success", "Users" :rows});  
            }  
        });  
    });  
  
}  
  
module.exports = REST\_ROUTER;

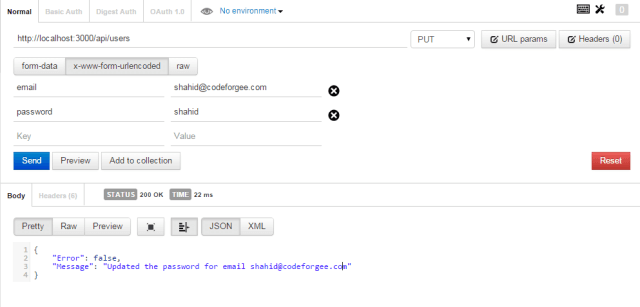
Here is the output of above api’s.  
**Case 1: Get all users.**  
  
**Case 2: Get user by ID.**  


**#3 : PUT /users**

This API uses PUT HTTP verb and it will update the password of user by passing its email ID. Here is the code.

REST.js

var mysql = require("mysql");  
function REST\_ROUTER(router,connection,md5) {  
    var self = this;  
    self.handleRoutes(router,connection,md5);  
}  
  
REST\_ROUTER.prototype.handleRoutes= function(router,connection,md5) {  
    router.get("/",function(req,res){  
            .................  
    });  
  
    router.post("/users",function(req,res){  
            .................  
    });  
  
    router.get("/users",function(req,res){  
            .................  
    });  
  
    router.get("/users/:userId",function(req,res){  
            .................  
    });  
  
    router.put("/users",function(req,res){  
        var query = "UPDATE ?? SET ?? = ? WHERE ?? = ?";  
        var table =["user\_login","user\_password",md5(req.body.password),"user\_email",req.body.email];  
        query = mysql.format(query,table);  
        connection.query(query,function(err,rows){  
            if(err) {  
                res.json({"Error" : true, "Message" : "Error executing MySQL query"});  
            } else {  
                res.json({"Error" : false, "Message" : "Updated the password for email "+req.body.email});  
            }  
        });  
    });  
}  
  
module.exports = REST\_ROUTER;

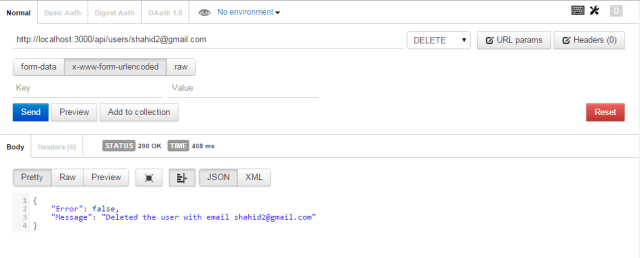
Here is the output of above API.  


**#4 : DELETE /users/:email**

This API will delete user from database by taking email ID as input. Here is the code.

REST.js

var mysql = require("mysql");  
function REST\_ROUTER(router,connection,md5) {  
    var self = this;  
    self.handleRoutes(router,connection,md5);  
}  
  
REST\_ROUTER.prototype.handleRoutes= function(router,connection,md5) {  
    router.get("/",function(req,res){  
                 ....................  
    });  
  
    router.post("/users",function(req,res){  
                 ....................  
    });  
  
    router.get("/users/:user\_id",function(req,res){  
                 ....................  
    });  
  
    router.get("/users",function(req,res){  
                 ....................  
    });  
  
    router.put("/users",function(req,res){  
                 ....................  
    });  
  
    router.delete("/users/:email",function(req,res){  
        var query = "DELETE from ?? WHERE ??=?";  
        var table = ["user\_login","user\_email",req.params.email];  
        query = mysql.format(query,table);  
        connection.query(query,function(err,rows){  
            if(err) {  
                res.json({"Error" : true, "Message" : "Error executing MySQL query"});  
            } else {  
                res.json({"Error" : false, "Message" : "Deleted the user with email "+req.params.email});  
            }  
        });  
    });  
}  
  
module.exports = REST\_ROUTER;

Here is the output of above code.  
  
You can develop more api’s depending upon your requirement and data model.

**Conclusion:**

REST api’s are very useful for any web app, mobile app, system software etc. You can develop REST api and use it anywhere because of its resource sharing feature.

With the above explanation i hope you understand the basics of REST api’s and also how to develop one for your system.

**Home work:**

Try to develop API to add new status in **user\_status** table and fetch that using user id or fetch all status.

You can also create another tables and explore it more ! Let me know if you stuck some where !