**Node js documentation**

**Local Server**

Install node on local server

**Live Server**

Terminal access require

Putty.exe

Open cmd or putty shell and go to ur directory in my case folder is in c:/scoket

cd\

c:\> cd socket

c:\socket> node inde.js

**npm packages**

|  |  |  |
| --- | --- | --- |
| name | description | uses |
| [forever](https://www.npmjs.com/package/forever) | **Use for node server run continuously** | Forever start index.js |
| **Cheerio** | Use for jquery selector in nodejs | var request = require('request')  , cheerio = require('cheerio')  , async = require('async')  , format = require('util').format;  var reddits = [ 'programming', 'javascript', 'node' ]  , concurrency = 2;  async.eachLimit(reddits, concurrency, function (reddit, next) {    var url = format('http://reddit.com/r/%s', reddit);  request(url, function (err, response, body) {  if (err) throw err;  var $ = cheerio.load(body);  $('a.title').each(function () {    console.log('%s (%s)', $(this).text(), $(this).attr('href'));  });  next();  });  }); |
| fs-extra | File management or file system extra function |  |
| Crypto | Encryption | crypto.createHash('md5').update(s).digest('hex') |
| request | Use for send http request like curl in php | var request = require("request");  var url = 'http://www.mp3afghan.com/wdprs/index.php';  request.post({url:url, form: {'login[username]':data.username, 'login[password]':data.password}}, function(err, headers, body){  if(fn.isJsonString(body)){  body = fn.parseJson(body);  response = body;  if(fn.isset(response.data.user\_url)){response.data.user\_url = undefined;}  if(fn.isset(response.data.user\_pass)){response.data.user\_pass = undefined;}  if(fn.isset(response.data.user\_status)){response.data.user\_status = undefined;}  if(fn.isset(response.data.user\_activation\_key)){response.data.user\_activation\_key = undefined;}  if(fn.isset(response.data.ID)){response.data.user\_url = parseInt(response.data.ID);}  socket.userProfile = response.data;  socket.emit('userLogin', response);  response=undefined; return;  }  else{  response.message="No Response Received";  socket.emit('userLogin', response);  response=undefined; return;  }  }); |
|  |  |  |
|  |  |  |

**FRAMEWORK**

1. **Loopback** is a nodejs framework like yii auto generate crud system <http://loopback.io/>
2. **Socketstream** [**http://www.socketstream.org/**](http://www.socketstream.org/)
3. **meteor** [**https://www.meteor.com/**](https://www.meteor.com/)
4. **geddy**  auto generate curd system [**http://geddyjs.org/**](http://geddyjs.org/)
5. **sleekjs** <http://sleekjs.com/>

**Autogenerate curd framework**

* Meteor - [http://www.meteor.com](http://www.meteor.com/). See [Why Meteor](http://www.meteorpedia.com/read/Why_Meteor).
* Derby - <http://derbyjs.com/>
* Sails - <http://sailsjs.org/>
* SANE - <http://sanestack.com/>
* MEAN.io - [http://mean.io](http://mean.io/)
* MEAN.js - [http://meanjs.org](http://meanjs.org/)
* Tower.js - [http://towerjs.org](http://towerjs.org/)
* Nombo - <http://nombo.io/>
* Geddy - [http://geddyjs.org](http://geddyjs.org/)
* Compound - <http://compoundjs.com/> (formerly known as Railway)
* Yahoo! Mojito - [http://developer.yahoo.com/cockt...](http://developer.yahoo.com/cocktails/mojito/)
* Wakanda - [http://wakanda.org](http://wakanda.org/)
* Opa - [http://opalang.org](http://opalang.org/)
* Deployd - [Deployd](http://www.deployd.com/" \t "_blank)

**/////////////////**

**Loopback relationship query**

**Go to models common\models\desire-model.json**

In relationship add this

"sender": {

"type": "belongsTo",

"model": "users",

"foreignKey": "senderID"

}

Here

Sender : relationship name

Type: belongsTo

Model : enter model name here

foreignKey: foreignKey name

**Fullcode here**

"relations": {

"sender": {

"type": "belongsTo",

"model": "users",

"foreignKey": "senderID"

}

}

In your desire controller

Enter this code in your query

{include: [{relation:'sender', scope: {fields: ['name']}}

Full code

var fieldnotshow = {addedby: false, addedtype: false, updatedat: false, updatedby: false, updatedtype: false, approved: false, status: false};

UsersRating.find**({include: [{relation:'sender', scope: {fields: ['name']}}** , {relation:'receiver', scope: {fields: ['name']} } ] ,where: {receiverID: data.user\_id}, fields: fieldnotshow}, function (err, data) {

if (err)

socket.emit('user:getUserRatingsByID', {success: false, message: 'Error getting data from database'});

else {

socket.emit('user:getUserRatingsByID', {success: true, message: "get user review list list", data: data});

}