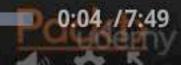
### RESTful Web API Design with Node.js

Saleh Hamadeh

Video 2.8

Saving Data in MongoDB













# In this Video, we are going to take a look at...

- Turning our program into a web application:
  - o Routes
  - o Views
  - Static files

- Accessing MongoDB from Node.js:
  - o Saving data
  - o Loading data



₩ twitter-notes

▶ public

▶ views

authenticator.js config.json index.js package.json storage.js

B

```
var MongoClient = require('mongodb').MongoClient;
             var database;
authenticator.js
             module.exports = {
                  connect: function() {
package.json
                      MongoClient.connect('mongodb://localhost:27017/twitter_notes', function(err
           6
                          if (err) {
                               return console.log("Error: " + err);
           8
          9
         10
                          db.open(function(err, db) {
         11
         12
                              database = db;
                              console.log("Connected to database.");
         13
                          });
         14
                      });
         15
         16
                  },
         17
                  connected: function() {
         18
                      return typeof database != 'undefined';
         19
                  },
         20
                 getFriends: function(userId, cb) {
         21
                      var cursor = database.collection('friends').find({
         22
                          for_user: userId
         23
                      });
         24
                      cursor.toArray(cb);
         25
         26
                  },
         27
                  insertFriends: function(friends) {
                      database.collection('friends').insert(friends. function(err) {
         28
```

▼ twitter-notes

config.json

index.js

storage.js

public

views

```
var MongoClient = require('mongodb').MongoClient;
              var database;
authenticator.js
config.json
             module.exports = {
                  connect: function() {
package.json
                      MongoClient.connect('mongodb://localhost:27017/twitter_notes', function(err)
           6
                          if (err) {
                               return console.log("Error: " + err);
           8
           9
         10
                          db.open(function(err, db) {
         11
         12
                               database = db;
                               console.log("Connected to database.");
         13
                          });
         14
                      });
         15
         16
                  },
         17
                  connected: function() {
         18
                      return typeof database != 'undefined';
         19
                  },
         20
                  getFriends: function(userId, cb) {
         21
                      var cursor = database.collection('friends').find({
         22
                          for_user: userId
         23
                      });
         24
                      cursor.toArray(cb);
         25
         26
                  },
         27
                  insertFriends: function(friends) {
                      database.collection('friends').insert(friends. function(err) {
         28
```

▼ twitter-notes

index.js

storage.js

public

views

```
FOLDERS

▼ twitter-notes

                   <!DOCTYPE html>
 ₩ public
                   <html>
    login.css
    sign-in-with-tv
    style.css
                     <head>
 <link rel="stylesheet" href="login.css">
    index.ejs
                6
                     </head>
    lo n.ejs
   authenticator.js
  config.json
                     <body>
   index.js
                        <div>
  package.json
              10
                          <h1>Twitter Notes</h1>
  storage.js
                          <h2>Your personal friends diary</h2>
              11
              12
                          <a href="/auth/twitter"><img src="sign-in-with-twitter-gray.png"></a>
              13
                        </div>
              14
                     </body>
              15
              16
                   </html>
```

```
FOLDERS

▼ twitter-notes

               <!DOCTYPE html>

▼ public

               <html>
   login.css
   sign-in-with-tv
   style.css
                 <head>
 <link rel="stylesheet" href="style.css">
                 </head>
   login.ejs
  authenticator.js
  config.json
                 <body>
             8
  index.js
                   <div id="wrapper">
  package.json
            10
                      <div id="friends-column" class="column">
  storage.js
                        <h1>Your Friends</h1>
            11
            12
                        d="friends">
                          <% friends.forEach(function(friend) { %>
            13
            14
                            uid="<%= friend.twitter_id %>"class="friend">
            15
                              <img class="profile-image" src="<%= friend.profile_image_url %>">
            16
                              >
            17
                                <span class="name"><%= friend.name %></span>
            18
                                <span class="screen-name">@<%= friend.screen_name %></span>
            19
                              20
                              <%= friend.location %>
                            21
                          <% }); %>
            22
            23
                        </div><!--
            24
            25
                      --><div id="notes-column" class="column">
            26
                        <h1>Notes</h1>
            27
                        </hi>
            28
```

Spaces: 2

```
FOLDERS
              index.js

▼ twitter-notes

                  var url = require('url');

▼ public

                  var express = require('express');
   login.css
   sign-in-with-tv
                  var querystring = require('querystring');
   style.css
                  var async = require('async');
 ▼ views
                  var authenticator = require('./authenticator');
   index.ejs
                  var storage = require('./storage.js');
   login.ejs
                  var config = require('./config');
  authenticator.js
  config.json
                  var app = express();
               9
  package.json
                  // Connect to MongoDB
  storage.js
                  storage.connect();
              12
                 // Set the view engine to ejs
                  app.set('view engine', 'ejs');
              15
                  // Add cookie parsing functionality to our Express app
                  app.use(require('cookie-parser')());
              18
                  // Take user to Twitter's login page
                  app.get('/auth/twitter', authenticator.redirectToTwitterLoginPage);
              21
                  // This is the callback url that the user is redirected to after signing in
                  app.get(url.parse(config.oauth_callback).path, function(req, res) {
              24
                      authenticator.authenticate(req, res, function(err) {
                           if (err) {
              26
                               res.redirect('/login');
              27
                           } else {
              28
                               res_redirect('/'):
```

Tab Size: 4

```
FOLDERS
              index.js

▼ twitter-notes

                  // And congre harstill intertainertely to our expices abb
 ₩ public
                  app.use(require('cookie-parser')());
   login.css
              18
   sign-in-with-tv
                  // Take user to Twitter's login page
   style.css
                  app.get('/auth/twitter', authenticator.redirectToTwitterLoginPage);
 ▼ views
   index.ejs
              21
   login.ejs
                  // This is the callback url that the user is redirected to after signing in
  authenticator.js
                  app.get(url.parse(config.oauth_callback).path, function(req, res) {
  config.json
                      authenticator.authenticate(reg, res, function(err) {
              24
  index.js
                           if (err) {
  package.json
  storage.js
              26
                                res.redirect('/login');
              27
                           } else {
                                res.redirect('/');
              28
              29
                       });
              30
              31
                  });
              32
                  // Main page handler
                  app.get('/,', function(req, res) {
              35
                       if (!req.cookies.access_token | !req.cookies.access_token_secret | !req.cook:
              36
                           return res.redirect('/login');
              37
              38
              39
                      // If the app couldn't connect to the database, get data from Twitter's API
                      if (!storage.connected()) {
              40
                           return renderMainPageFromTwitter(req, res);
              41
              42
              43
```

```
FOLDERS
              index.js

▼ twitter-notes

               ...
                                         ₩ public
              47
   login.css
              48
   sign-in-with-tv
                               (friends.length > 0) {
              49
   style.css
                                 console.log("Data loaded from MongoDB");
              50
 ▼ views
              51
   index.ejs
   login.ejs
              52
                                 // Sort the friends alphabetically by name
  authenticator.js
              53
                                 friends.sort(function(a, b) {
  config.json
                                     return a.name.toLowerCase().localeCompare(b.name.toLowerCase());
              54
  index.js
                                });
              55
  package.json
              56
  storage.js
              57
                                 // Render the main application
              58
                                 res.render('index', {
                                     friends: friends
              59
                                });
              60
              61
                            } else {
                                 renderMainPageFromTwitter(req, res);
              62
              63
              64
                       });
              65
                  });
              66
                   function renderMainPageFromTwitter(reg, res) {
              68
                       async.waterfall([
              69
                            // Get friends' IDs
                            function(cb) {
              70
              71
                                 var cursor = -1;
              72
                                 var ids = [];
              73
```

25 characters selected

index.js ▼ twitter-notes location: friend.location, 138 ▼ public login.css profile\_image\_url: friend.profile\_image\_url 139 sign-in-with-tv 140 **}**; style.css }); 141 **▼** views 142 index.ejs // Render the main application 143 login.ejs authenticator.js res.render('index', { 144 config.json friends: friends 145 index.js }); 146 package.json 147 storage.js 148 In the background, save the friends to MongoDB (storage.connected()) { 149 150 storage.insertFriends(friends); 151 **});** 152 153 1); 154 155 156 // Show the login page 157 app.get('/login', function(req, res) { 158 res.render('login'); 159 }); 160 161 // Serve static files in public directory 162 app.use(express.static(\_\_dirname + '/public')); 163 164 // Ctart lictoring for requests

**FOLDERS** 

5 characters selected Tab Size: 4 Java

```
FOLDERS
              index.js

▼ twitter-notes

                      app -
                             CAPICSS(/,
 ▼ public
               9
   login.css
                  // Connect to MongoDB
   sign-in-with-tv
                  storage.connect();
                                                                                                                 style.css
 ▼ views
              12
   index.ejs
                  // Set the view engine to ejs
   login.ejs
                  app.set('view engine', 'ejs');
  authenticator.js
              15
  config.json
              16
                  // Add cookie parsing functionality to our Express app
  index.is
                  app.use(require('cookie-parser')());
  package.json
  storage.js
              18
                  // Take user to Twitter's login page
              19
                  app.get('/auth/twitter', authenticator.redirectToTwitterLoginPage);
              20
              21
                  // This is the callback url that the user is redirected to after signing in
                  app.gct(url.parse(config.oauth_callback).path, function(req, res) {
              24
                       authenticator.authenticate(reg, res, function(err) {
                           if (err) {
              26
                                res.redirect('/login');
              27
                           } else {
                                res.redirect('/');
              28
              29
              30
                       });
              31
                  });
              32
                  // Main page handler
                  app.get('/', function(req, res) {
              35
                          (!req.cookies.access_token | !req.cookies.access_token_secret | |
```

3 characters selected Tab Size: 4 JavaScript



```
index.js
              var url = require('url');
              var express = require('express');
              var querystring = require('querystring');
sign-in-with-tv
              var async = require('async');
              var authenticator = require('./authenticator');
              var storage = require('./storage.js');
              var config = require('./config');
authenticator.js
              var app = express();
           9
              // Connect to MongoDB
              storage.connect();
          12
              // Set the view engine to ejs
              app.set('view engine', 'ejs');
          15
              // Add cookie parsing functionality to our Express app
              app.use(require('cookie-parser')());
          18
              // Take user to Twitter's login page
              app.get('/auth/twitter', authenticator.redirectToTwitterLoginPage);
          21
              // This is the callback url that the user is redirected to after signing in
              app.get(url.parse(config.oauth_callback).path, function(req, res) {
          24
                  authenticator.authenticate(req, res, function(err) {
                      if (err) {
          26
                           res.redirect('/login');
          27
                      } else {
                           res_redirect('/'):
          78
```

▼ twitter-notes

login.css

style.css

index.ejs

login.ejs

config.json

package.json

index.js

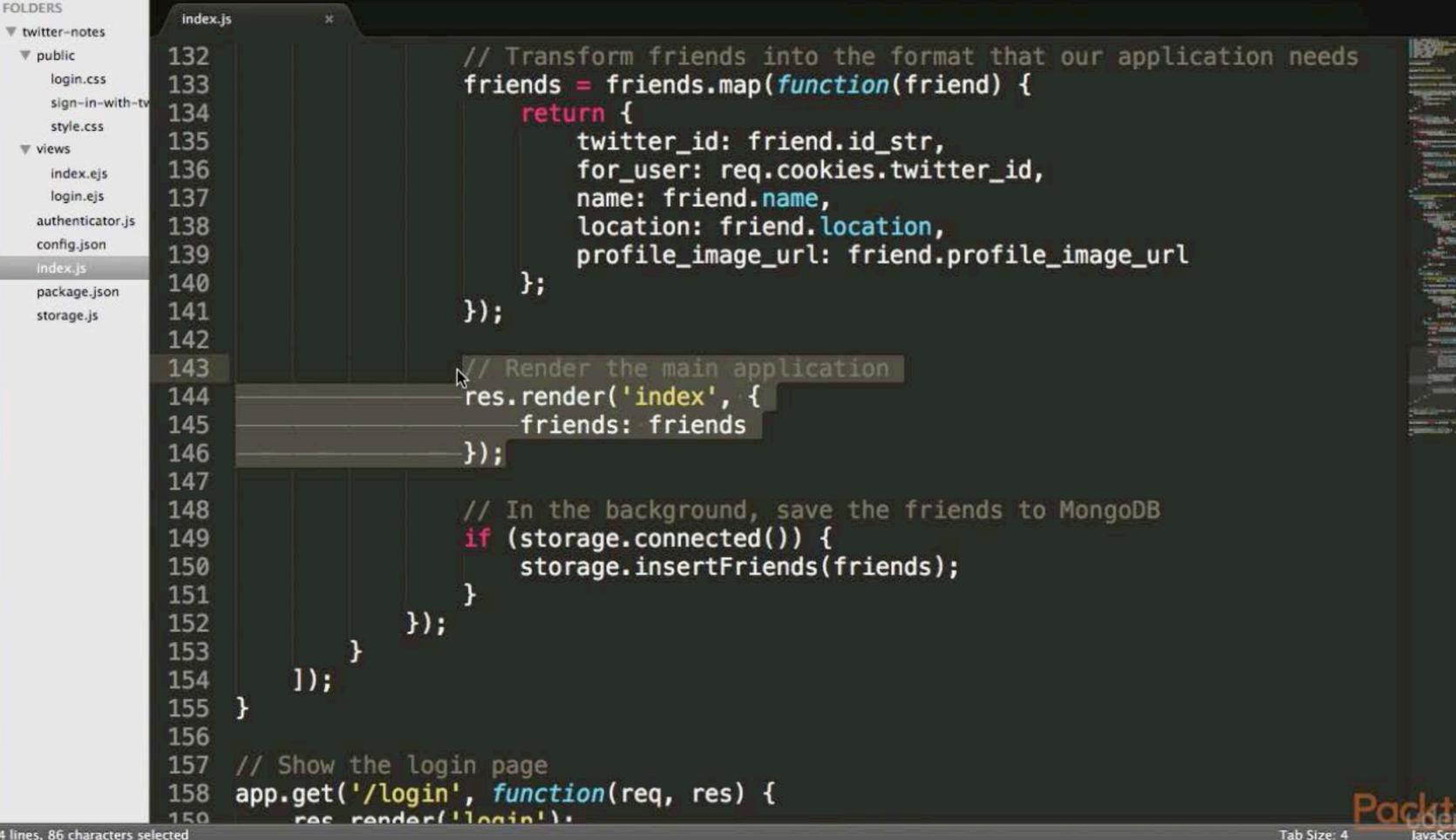
storage.js

▼ public

2 lines, 31 characters selected Tab Size: 4

```
FOLDERS
              index.js
 ▼ public
                            return renderMainPageFromTwitter(req, res);
              41
   login.css
              42
   sign-in-with-tv
              43
   style.css
                       storage.getFriends(req.cookies.twitter_id, function(err, friends) {
              44
 45
                               (err) {
   index.ejs
   login.ejs
                                return res.status(500).send(err);
              46
  authenticator.js
              47
  config.json
              48
  index.js
                            if (friends.length > 0) {
              49
  package.json
                                console.log("Data loaded from MongoDB");
              50
  storage.js
              51
              52
                                // Sort the friends alphabetically by name
              53
                                friends.sort(function(a, b) {
                                     return a.name.toLowerCase().localeCompare(b.name.toLowerCase());
              54
                                });
              55
              56
              57
                                 // Render the main application
                                res.render('index', {
              58
              59
                                     friends: friends
              60
                                }); &
              61
                            } else {
              62
                                renderMainPageFromTwitter(req, res);
              63
                       });
              64
              65
                  });
              66
                  function renderMainPageFromTwitter(reg, res) {
```

3 lines, 49 characters selected JavaScript



4 lines, 86 characters selected

**FOLDERS** index.js ▼ twitter-notes 141 511 ₩ public 142 login.css 143 // Render the main application sign-in-with-tv res.render('index', { 144 style.css friends: friends 145 index.ejs }); 146 login.ejs 147 authenticator.js 148 // In the background, save the friends to MongoDB config.json 149 (storage.connected()) { index.js package.json storage.insertFriends(friends); 150 storage.js 151 152 }); 153 1); 154 155 156 157 // Show the login page app.get('/login', function(req, res) { 158 159 red render('login'); 160 }); 161 162 // Serve static files in public directory app.use(express.static(\_\_dirname + '/public')); 163 164 165 // Start listening for requests 166 app.listen(config.port, function() { console.log("Listening on port " + config.port); 167 168 });

2 lines, 22 characters selected Tab Size: 4 JavaScript

index.js ▼ twitter-notes 141 SII ₩ public 142 login.css 143 // Render the main application sign-in-with-tv res.render('index', { 144 style.css **▼** views friends: friends 145 index.ejs }); 146 login.ejs 147 authenticator.js 148 // In the background, save the friends to MongoDB config.json 149 (storage.connected()) { index.js package.json storage.insertFriends(friends); 150 storage.js 151 }); 152 153 1); 154 155 156 157 // Show the login page app.get('/login', function(req, res) { 158 159 res.render('login'); 160 }); 161 162 // Serve static files in public directory app.use(express.static(\_\_dirname '/public')); 163 164 165 // Start listening for requests 166 app.listen(config.port, function() { console.log("Listening on port " + config.port); 167 168 });

**FOLDERS** 

2 lines, 48 characters selected Tab Size: 4 JavaSo

lawn-143-215-115-165:Desktop saleh\$ mongod --dbpath database/db/

```
lawn-143-215-115-165:Desktop saleh$ mongod --dbpath database/db/
                                        [initandlisten] journal dir=database/db/journal
2016-04-13T11:13:50.374-0400 I JOURNAL
2016-04-13T11:13:50.375-0400 I JOURNAL
                                        [initandlisten] recover : no journal files present, no recovery n
eeded
2016-04-13T11:13:50.455-0400 I JOURNAL
                                        [durability] Durability thread started
                                        [journal writer] Journal writer thread started
2016-04-13T11:13:50.455-0400 I JOURNAL
2016-04-13T11:13:50.456-0400 I CONTROL
                                        [initandlisten] MongoDB starting : pid=785 port=27017 dbpath=data
base/db/ 64-bit host=lawn-143-215-115-165.lawn.gatech.edu
                                        [initandlisten]
2016-04-13T11:13:50.456-0400 I CONTROL
2016-04-13T11:13:50.456-0400 I CONTROL
                                        [initandlisten] ** WARNING: soft rlimits too low. Number of files
is 256, should be at least 1000
2016-04-13T11:13:50.456-0400 I CONTROL
                                        [initandlisten] db version v3.0.7
                                        [initandlisten] git version: 6ce7cbe8c6b899552dadd907604559806aa2
2016-04-13T11:13:50.456-0400 I CONTROL
e9bd
2016-04-13T11:13:50.456-0400 I CONTROL
                                        [initandlisten] build info: Darwin mci-osx108-13.build.10gen.cc 1
2.3.0 Darwin Kernel Version 12.3.0: Sun Jan 6 22:37:10 PST 2013; root:xnu-2050.22.13~1/RELEASE_X86_64 x8
6_64 B00ST_LIB_VERSION=1_49
2016-04-13T11:13:50.456-0400 I CONTROL
                                        [initandlisten] allocator: system
2016-04-13T11:13:50.456-0400 I CONTROL
                                        [initandlisten] options: { storage: { dbPath: "database/db/" } }
2016-04-13T11:13:50.533-0400 I NETWORK
                                        [initandlisten] waiting for connections on port 27017
```



```
index.js
              var MongoClient = require('mongodb').MongoClient;
              var database;
 sign-in-with-tv
              module.exports = {
                  connect: function() {
                      MongoClient.connect('mongodb://localhost:27017/twitter_notes', function(err)
                           if (err) {
authenticator.js
                               return console.log("Error: " + err);
package.json
          10
                           db.open(function(err, db) {
          11
          12
                               database = db;
          13
                               console.log("Connected to database.");
                           });
          14
                       });
          15
          16
                  },
          17
                  connected: function() {
          18
                       return typeof database != 'undefined';
          19
                  },
          20
                  getFriends: function(userId, cb) {
          21
                       var cursor = database.collection('friends').find({
          22
                           for_user: userId
          23
                       });
          24
          25
                       cursor.toArray(cb);
          26
                  },
          27
                  insertFriends: function(friends) {
                       database.collection('friends').insert(friends. function(err) {
          28
                                                                                                Tab Size: 4
```

▼ twitter-notes

login.css

style.css

index.ejs

login.ejs

config.json

storage.js,

index.js

▼ public

**▼ views** 

```
FOLDERS
              index.js
                             storage.is

▼ twitter-notes

                  var url = require('url');

▼ public

                  var express = require('express');
   login.css
   sign-in-with-tv
                  var querystring = require('querystring');
   style.css
                  var async = require('async');
 ▼ views
                  var authenticator = require('./authenticator');
   index.ejs
                  var storage = require('./storage.js');
   login.ejs
                  var config = require('./config');
  authenticator.js
  config.json
                  var app = express();
  index.js
               9
  package.json
                  // Connect to MongoDB
  storage.js
                  storage.connect();
              12
                  // Set the view engine to ejs
                  app.set('view engine', 'ejs');
              15
                  // Add cookie parsing functionality to our Express app
                  app.use(require('cookie-parser')());
              18
                  // Take user to Twitter's login page
                  app.get('/auth/twitter', authenticator.redirectToTwitterLoginPage);
              21
                  // This is the callback url that the user is redirected to after signing in
                  app.get(url.parse(config.oauth_callback).path, function(req, res) {
              24
                      authenticator.authenticate(req, res, function(err) {
                           if (err) {
              26
                                res.redirect('/login');
              27
                           } else {
              28
                                res. redirect('/'):
```

2 lines, 19 characters selected Tab Size: 4

```
index.js
              var MongoClient = require('mongodb').MongoClient;
              var database;
 sign-in-with-tv
             module.exports = {
                  connect: function() {
                      MongoClient.cognect('mongodb://localhost:27017/twitter_notes', function(err)
           6
                           if (err) {
authenticator.js
                               return console.log("Error: " + err);
         10
                           db.open(function(err, db) {
         11
         12
                               database = db;
         13
                               console.log("Connected to database.");
                           });
         14
                      });
         15
         16
                  },
         17
                  connected: function() {
         18
                      return typeof database != 'undefined';
         19
                  },
         20
                  getFriends: function(userId, cb) {
         21
                      var cursor = database.collection('friends').find({
         22
                           for_user: userId
         23
                      });
         24
         25
                      cursor.toArray(cb);
         26
                  },
         27
                  insertFriends: function(friends) {
                      database.collection('friends').insert(friends. function(err) {
         28
                                                                                               Tab Size: 4
```

▼ twitter-notes

login.css

style.css

index.ejs

login.ejs

config.json

package.json

index.js

storage.js

▼ public

**▼ views** 

```
FOLDERS
               index.js

▼ twitter-notes

                  var MongoClient = require('mongodb').MongoClient;
  ▼ public
                  var database;
    login.css
    sign-in-with-tv
    style.css
                  module.exports = {
  connect: function() {
    index.ejs
                           MongoClient.connect('mongodb://localhost:27017/twitter_notes', function(err)
    login.ejs
                                if (err) {
   authenticator.js
   config.json
                                     return console.log("Error: " + err);
   index.js
   package.json
              10
   storage.js
              11
                                db.open(function(err, db) {
                                     database = db;
              12
                                     console.log("Connected to database.");
              13
                                });
              14
                            });
              15
              16
                       },
              17
                       connected: function() {
              18
                            return typeof database != 'undefined';
              19
                       },
              20
                       getFriends: function(userId, cb) {
              21
                            var cursor = database.collection('friends').find({
              22
                                for_user: userId
              23
                            });
              24
                            cursor.toArray(cb);
              25
              26
                       },
              27
                       insertFriends: function(friends) {
                            database.collection('friends').insert(friends. function(err) {
2 lines, 31 characters selected
```

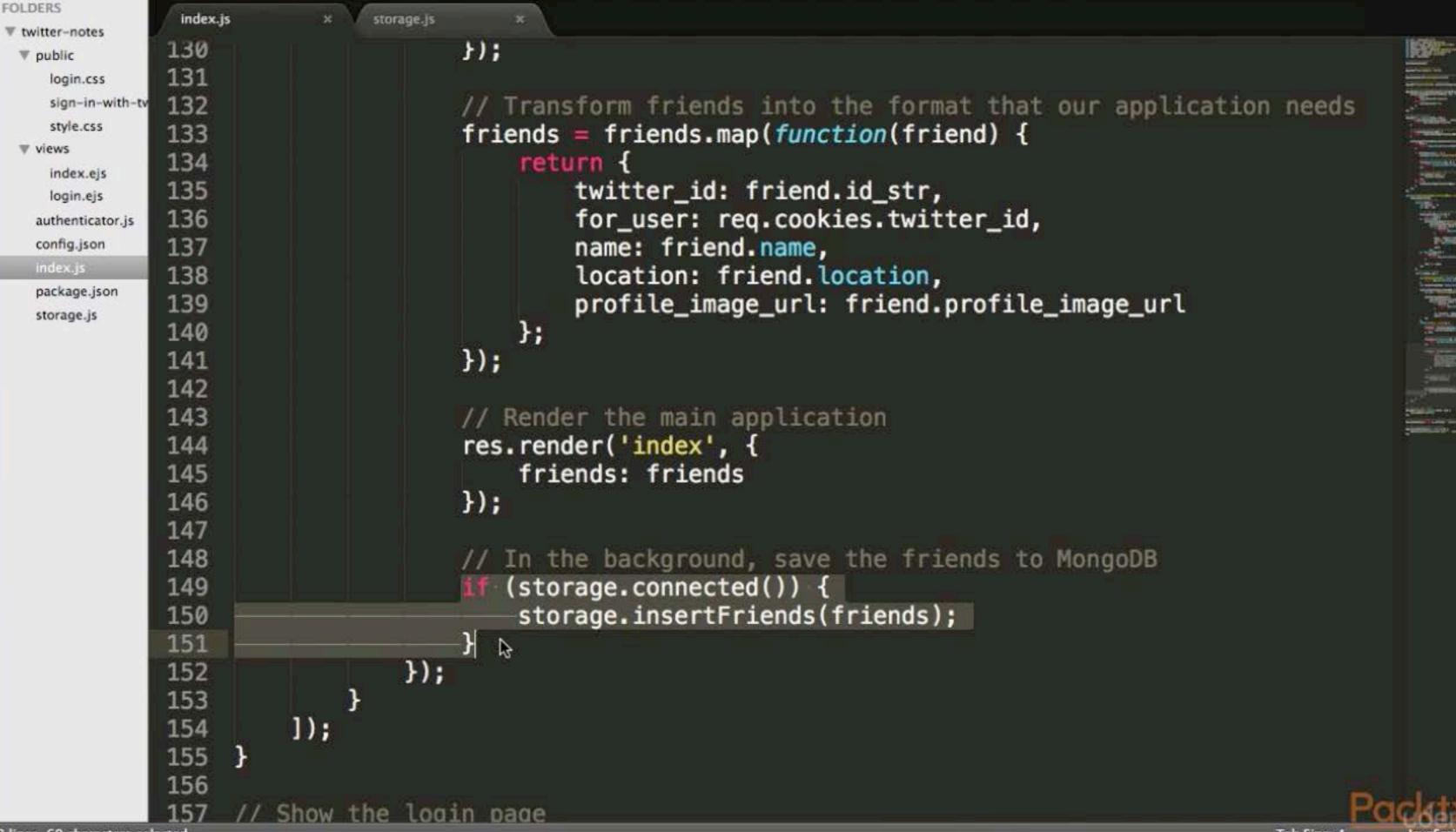
```
FOLDERS
               index.js

▼ twitter-notes

▼ public

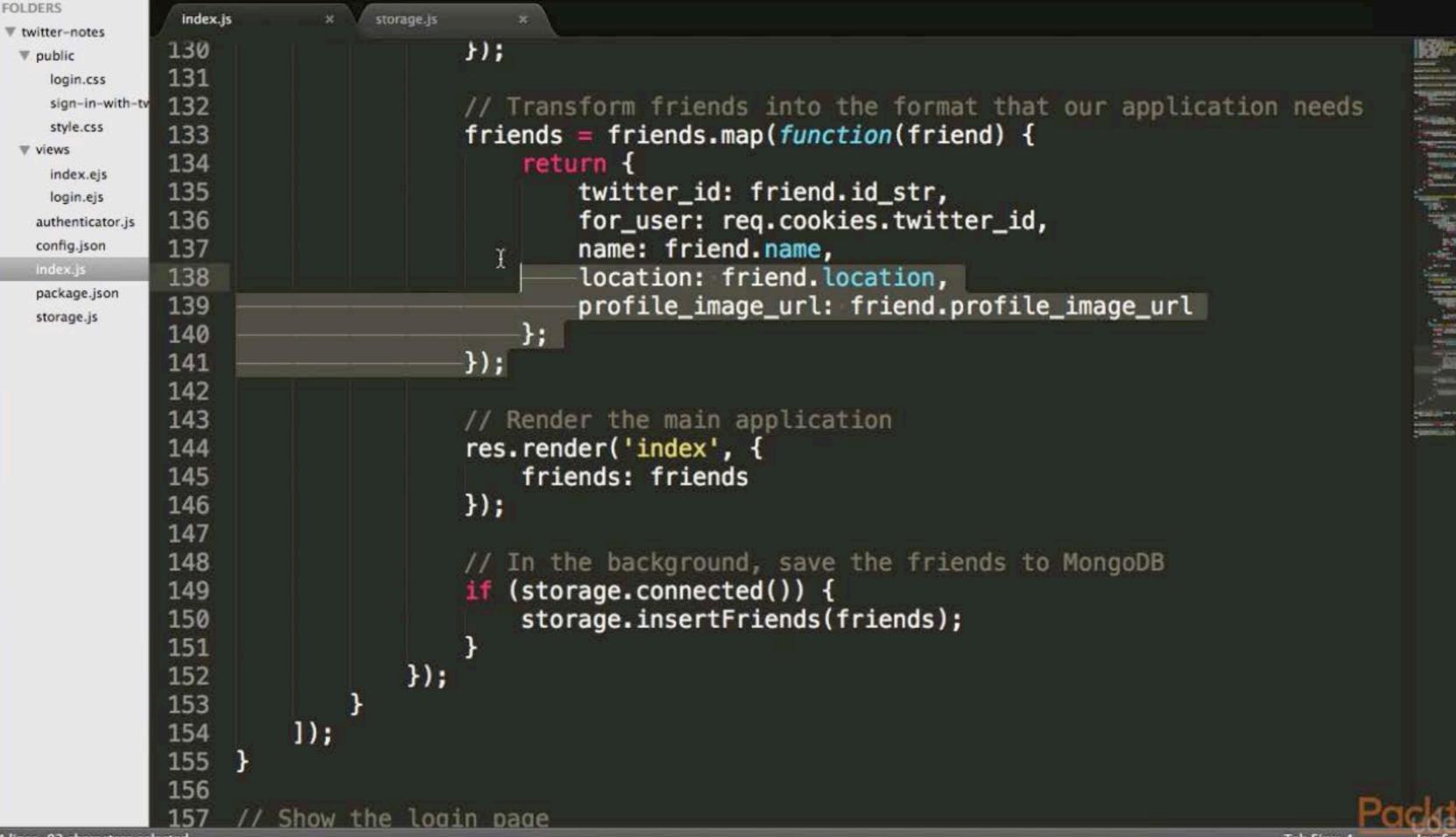
               51
    login.css
                                  // Sort the friends alphabetically by name
    sign-in-with-tv
               53
                                  friends.sort(function(a, b) {
    style.css
                                       return a.name.toLowerCase().localeCompare(b.name.toLowerCase());
               54
 ▼ views
               55
                                  });
    index.ejs
    login.ejs
               56
  authenticator.js
               57
                                  // Render the main application
  config.json
                                  res.render('index', {
               58
  index.js
               59
                                       friends: friends
  package.json
               60
                                  });
  storage.js
               61
                             } else {
               62
                                  renderMainPageFromTwitter(req, res);
               63
                        });
               64
               65
                   });
               66
               67
                   function renderMainPageFromTwitter(reg, res) {
               68
                        async.waterfall([
               69
                             // Get friends' IDs
                             function(cb) {
               70
               71
                                  var cursor = -1;
               72
                                  var ids = [];
               73
               74
                                  // Get IDs by traversing the cursored collection
               75
                                  async.whilst(function() {
               76
                                       return cursor != 0;
                                  }, function(cb) {
```

2 lines, 47 characters selected Tab Size: 4



3 lines, 69 characters selected JavaScr

**FOLDERS** index.js storage.js ▼ twitter-notes LI (CII) 1 **▼** public return console.log("Error: " + err); login.css sign-in-with-tv 10 style.css **▼** views db.open(function(err, db) { 11 index.ejs database = db; login.ejs console.log("Connected to database."); 13 authenticator.js }); 14 config.json }); 15 index.js package.json 16 }, storage.js 17 connected: function() { 18 return typeof database != 'undefined'; 19 }, getFriends: function(userId, cb) { 20 21 var cursor = database.collection('friends').find({ 22 for\_user: userId 23 }); 24 25 cursor.toArray(cb); 26 }, 27 insertFriends: function(friends) { database.collection('friends').insert(friends, function(err) { 28 29 (err) { 30 console.log("Cannot insert friends to database."); 31 }); 32 33 34



4 lines, 93 characters selected Tab Size: 4 JavaScr

```
FOLDERS
              index.js
                             storage.|s
                       3);
              30

▼ public

              31
                  });
   login.css
              32
   sign-in-with-tv
   style.css
                  // Main page handler
 app.get('/', function(req, res) {
   index.ejs
              35
                       if (!req.cookies.access_token | !req.cookies.access_token_secret | !req.cooki
   login.ejs
              36
                            return res.redirect('/login');
  authenticator.js
              37
  config.json
  index.is
              38
  package.json
              39
                       // If the app couldn't connect to the database, get data from Twitter's API
  storage.js
              40
                          (!storage.connected()) {
                            return renderMainPageFromTwitter(req, res);
              41
              42
              43
              44
                       storage.getFriends(req.cookies.twitter_id, function(err, friends) {
              45
                               (err) {
              46
                                return res.status(500).send(err);
              47
              48
              49
                           if (friends.length > 0) {
              50
                                console.log("Data loaded from MongoDB");
              51
              52
                                // Sort the friends alphabetically by name
                                friends.sort(function(a, b) {
              53
              54
                                     return a.name.toLowerCase().localeCompare(b.name.toLowerCase());
                                });
              55
              56
              57
                                // Render the main application
```

10 characters selected

**FOLDERS** index.js storage.js ▼ twitter-notes Li (err) t **▼** public return console.log("Error: " + err); login.css sign-in-with-tv 10 style.css **▼** views db.open(function(err, db) { 11 index.ejs database = db; login.ejs console.log("Connected to database."); 13 authenticator.js }); 14 config.json }); 15 index.js package.json 16 }, storage.js 17 connected: function() { 18 return typeof database != 'undefined'; }, 19 getFriends: function(userId, cb) { 20 var cursor = database.collection('friends').find({ 21 22 for\_user: userId 23 }); 24 25 cursor.toArray(cb); 26 }, 27 insertFriends: function(friends) { database.collection('friends').insert(friends, function(err) { 28 if (err) { 29 console.log("Cannot insert friends to database."); 30 31 **});** 32 33 34

**FOLDERS** index.js storage.js ▼ twitter-notes Li (ell) i **▼** public return console.log("Error: " + err); login.css sign-in-with-tv 10 style.css **▼** views db.open(function(err, db) { 11 index.ejs database = db; login.ejs console.log("Connected to database."); 13 authenticator.js }); 14 config.json }); 15 index.js package.json 16 }, storage.js 17 connected: function() { 18 return typeof database != 'undefined'; 19 }, getFriends: function(userId, cb) { 20 var cursor = database.collection('friends').find({ 21 22 for\_user: userId **});**| ₽ 23 24 25 cursor.toArray(cb); 26 }, 27 insertFriends: function(friends) { database.collection('friends').insert(friends, function(err) { 28 if (err) { 29 console.log("Cannot insert friends to database."); 30 31 **});** 32 33 34

**FOLDERS** index.js storage.js ▼ twitter-notes LI (CII) 1 **▼** public return console.log("Error: " + err); login.css sign-in-with-tv 10 style.css db.open(function(err, db) { 11 index.ejs database = db; login.ejs console.log("Connected to database."); 13 authenticator.js }); 14 config.json }); 15 index.js package.json 16 }, storage.js 17 connected: function() { 18 return typeof database != 'undefined'; 19 }, getFriends: function(userId, cb) { 20 var cursor = database.collection('friends').find({ 21 22 for\_user: userId 23 }); 24 25 curspr.toArray(cb); 26 }, 27 insertFriends: function(friends) { database.collection('friends').insert(friends, function(err) { 28 if (err) { 29 console.log("Cannot insert friends to database."); 30 31 **});** 32 33 34

2 lines, 22 characters selected Tab Size: 4 JavaScript

```
index.js
                            storage.|s
             38

▼ public

             39
                      // If the app couldn't connect to the database, get data from Twitter's API
  login.css
                     if (!storage.connected()) {
             40
  sign-in-with-tv
  style.css
                          return renderMainPageFromTwitter(req, res);
             41
42
  index.ejs
             43
  login.ejs
                      storage.getFriends(req.cookies.twitter_id, function(err, friends) {
             44
 authenticator.js
             45
                          if (err) {
 config.json
 index.is
                               return res.status(500).send(err);
             46
 package.json
             47
 storage.js
             48
             49
                             (friends.length > 0) {
             50
                               console.log("Data loaded from MongoDB");
             51
             52
                               // Sort the friends alphabetically by name
                               friends.sort(function(a, b) {
             53
                                    return a.name.toLowerCase().localeCompare(b.name.toLowerCase());
             54
             55
                               });
             56
             57
                               // Render the main application
             58
                               res.render('index', {
                                    friends: friends
             59
                               });
             60
                          } else {
             61
             62
                               renderMainPageFromTwitter(req, res);
             63
             64
                      });
             65
```

3 lines, 105 characters selected

**FOLDERS** 

```
index.js
                            storage.|s
             38
▼ public
                     // If the app couldn't connect to the database, get data from Twitter's API
             39
  login.css
                     if (!storage.connected()) {
             40
  sign-in-with-tv
                          return renderMainPageFromTwitter(req, res);
  style.css
             41
42
  index.ejs
             43
  login.ejs
                      storage.getFriends(req.cookies.twitter_id, function(err, friends) {
             44
 authenticator.js
             45
                          if (err) {
 config.json
 index.is
                               return res.status(500).send(err);
             46
 package.json
             47
 storage.js
             48
                          if (friends.length > 0) {
             49
                               console.log("Data loaded from MongoDB");
             50
             51
             52
                               // Sort the friends alphabetically by name
             53
                               friends.sort(function(a, b) {
                                    return a.name.toLowerCase().localeCompare(b.name.toLowerCase());
             54
                               });
             55
             56
            57
                                  Render the main application
                               res.render('index', {
             58
             59
                                    friends: friends
             60
                               });
                          } else {
            61
             62
                               renderMainPageFromTwitter(req, res);
             63
                     });
             64
             65
```

4 lines, 83 characters selected Tab Size: 4 Ja

index.js storage.|s 38 ▼ public // If the app couldn't connect to the database, get data from Twitter's API 39 login.css if (!storage.connected()) { 40 sign-in-with-tv return renderMainPageFromTwitter(req, res); style.css 41 42 index.ejs 43 login.ejs storage.getFriends(req.cookies.twitter\_id, function(err, friends) { 44 authenticator.js 45 if (err) { config.json index.js return res.status(500).send(err); 46 package.json 47 storage.js 48 49 if (friends.length > 0) { 50 console.log("Data loaded from MongoDB"); 51 52 // Sort the friends alphabetically by name 53 friends.sort(function(a, b) { return a.name.toLowerCase().localeCompare(b.name.toLowerCase()); 54 }); 55 56 57 // Render the main application 58 res.render('index', { friends: friends 59 60 }); 61 } else { 62 renderMainPageFromTwitter(req, res); 63 }); 64 65

**FOLDERS** 

2 lines, 40 characters selected JavaS

## Summary

- OAuth 1.0a
  - o Parties
  - o Process
- Twitter's REST API
  - o Tweeting
  - o Searching for tweets
  - o Finding friends
- Async.js
- MongoDB



### Next Section

Building a RESTful API

