RESTful Web API Design with Node.js

Saleh Hamadeh

Video 2.2

Sample Application Code Architecture

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

- Data sources
- Source code directory structure
- Server code architecture
- Client code architecture

Data Sources

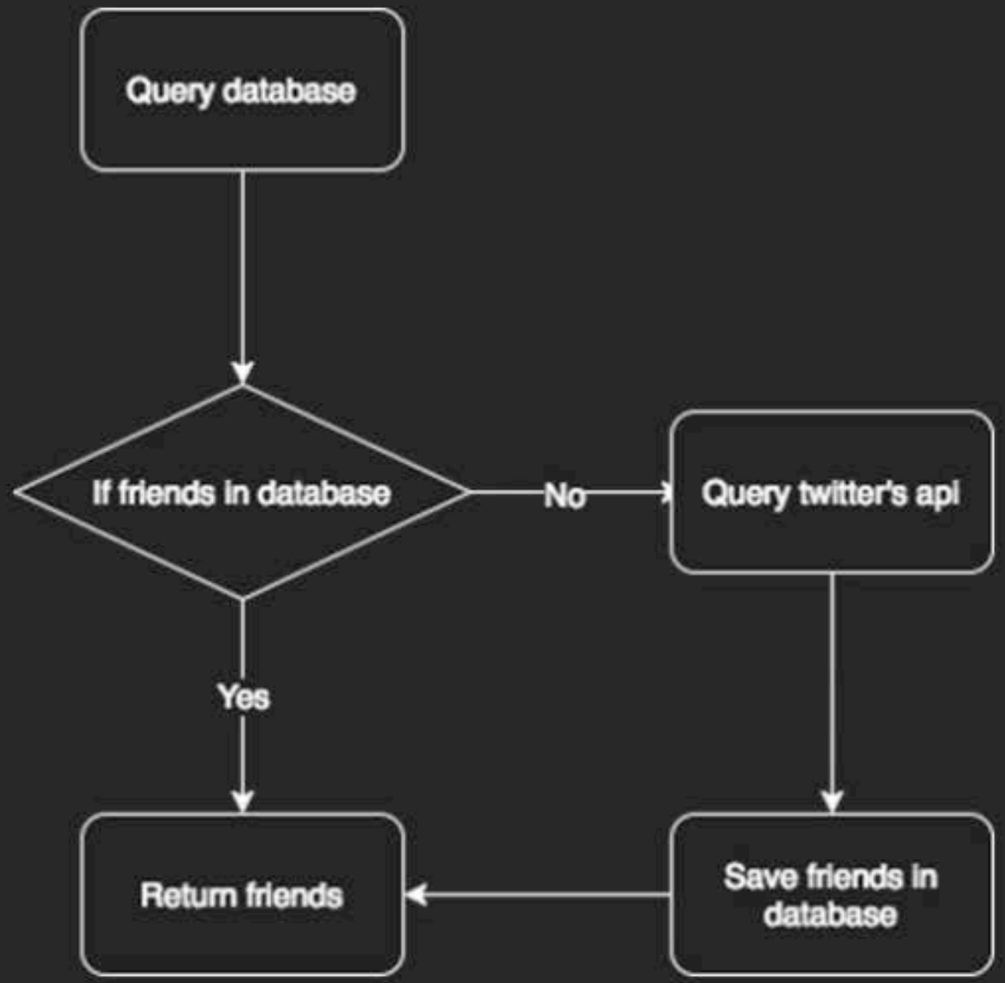
Twitter API

authenticator.js

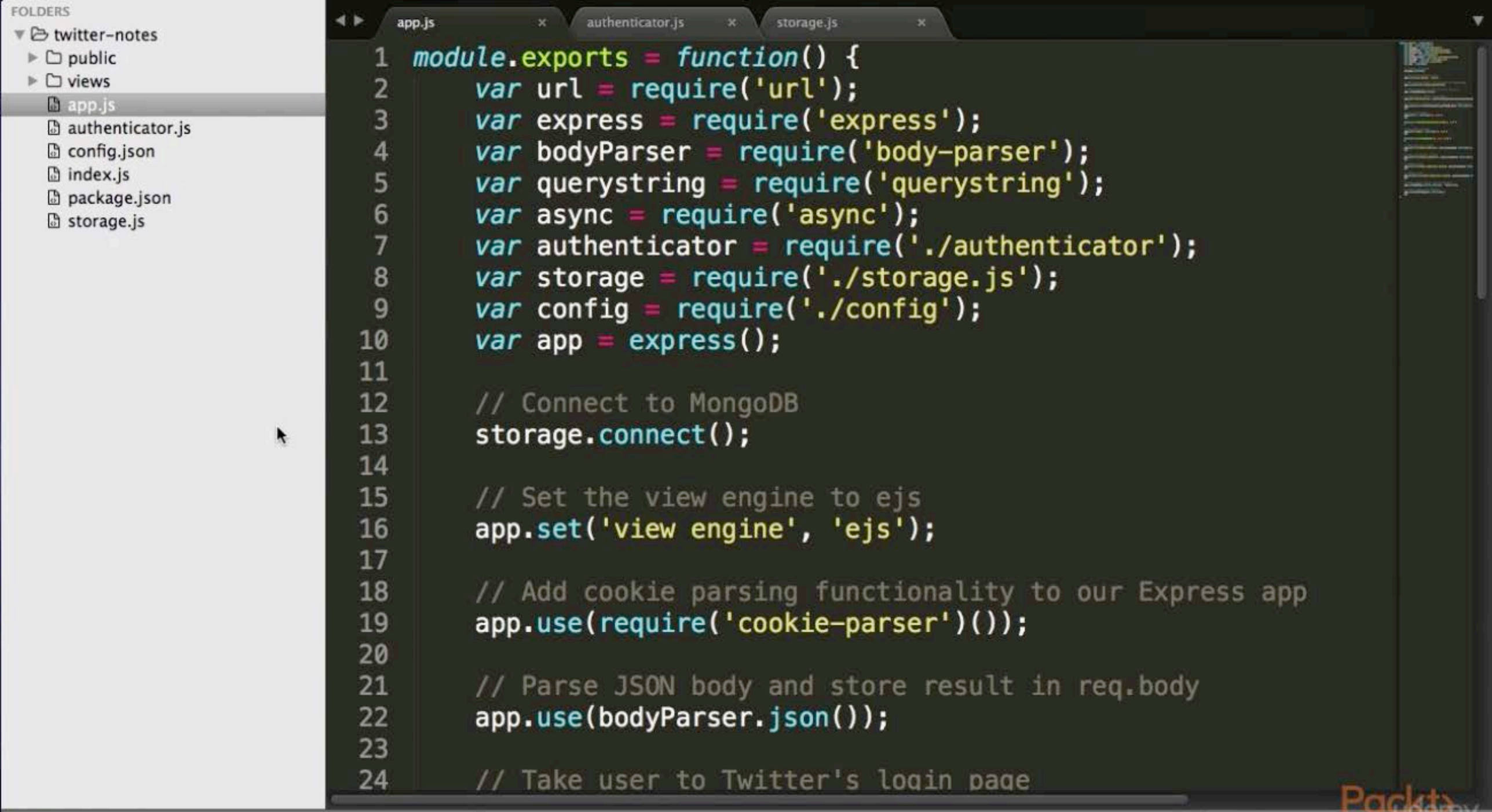
Application Database

storage.js

Getting Friends



Getting Notes

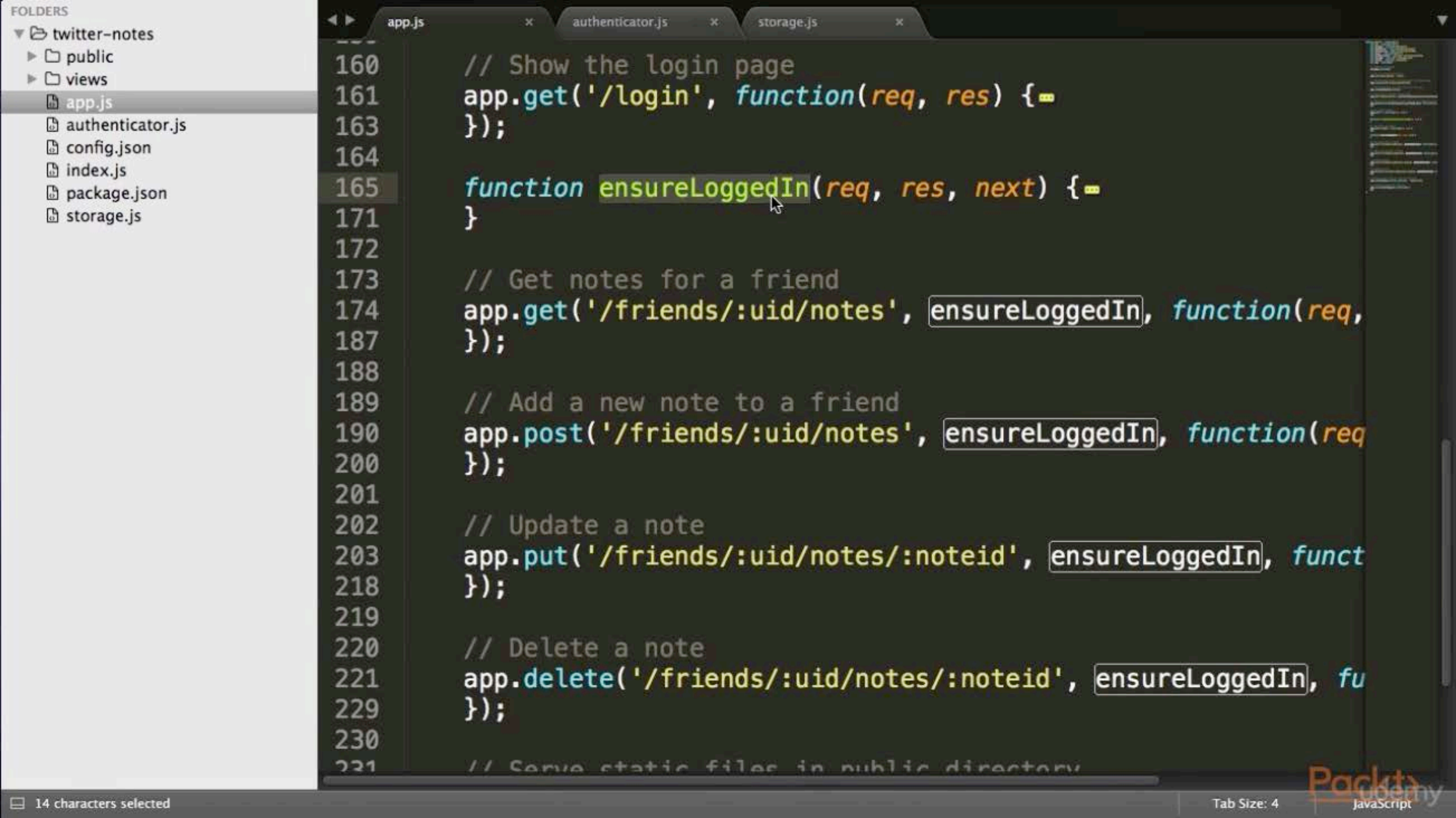

FOLDERS
twitter-notes
public
views
app.js
authenticator.js
config.json
index.js
package.json
storage.js
app.js
authenticator.js
storage.js
module.exports = function() {
var url = require('url');
var express = require('express');
var bodyParser = require('body-parser');
var querystring = require('querystring');
var async = require('async');
var authenticator = require('./authenticator');
var storage = require('./storage.js');
var config = require('./config');
var app = express();
// Connect to MongoDB
storage.connect();
// Set the view engine to ejs
app.set('view engine', 'ejs');
// Add cookie parsing functionality to our Express app
app.use(require('cookie-parser')());
// Parse JSON body and store result in req.body
app.use(bodyParser.json());
// Take user to Twitter's login page
Line 36, Column 8
Tab Size: 4
JavaScript

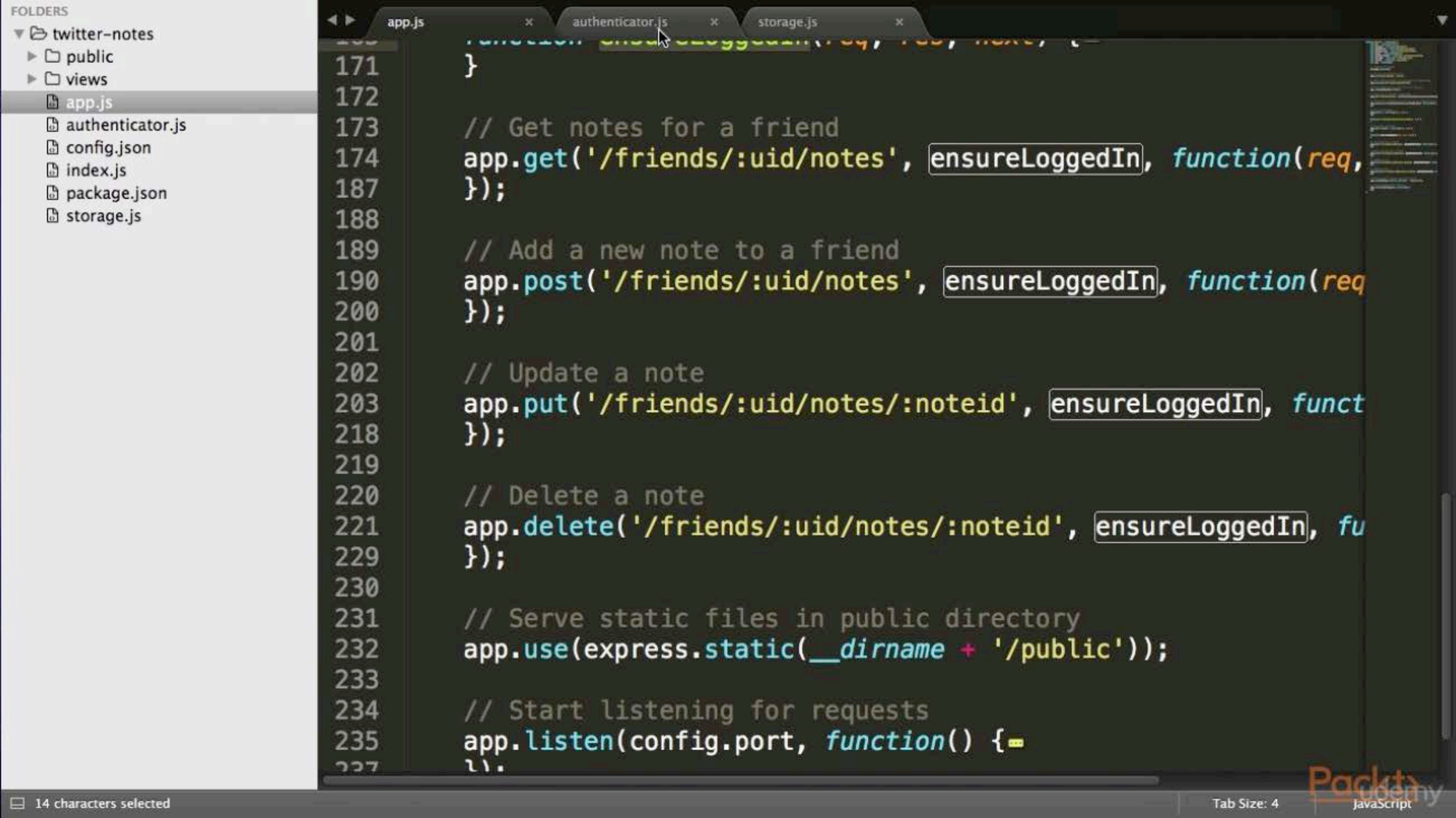
FOLDERS

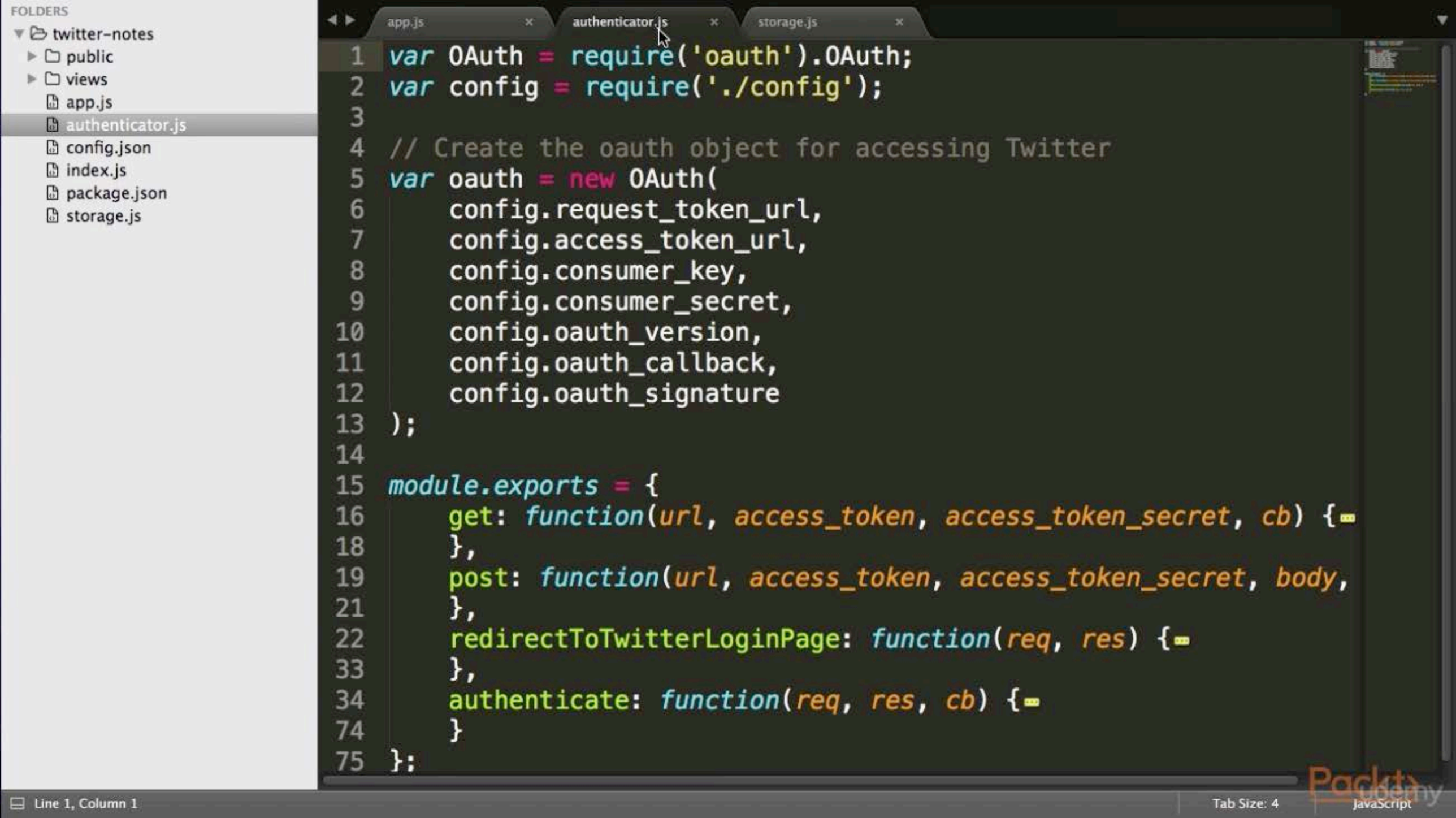
- twitter-notes
  - public
    - login.css
    - main.js
    - sign-in-with-twitter-gray.png
    - style.css
  - views
  - app.js
  - authenticator.js
  - config.json
  - index.js
  - package.json
  - storage.js

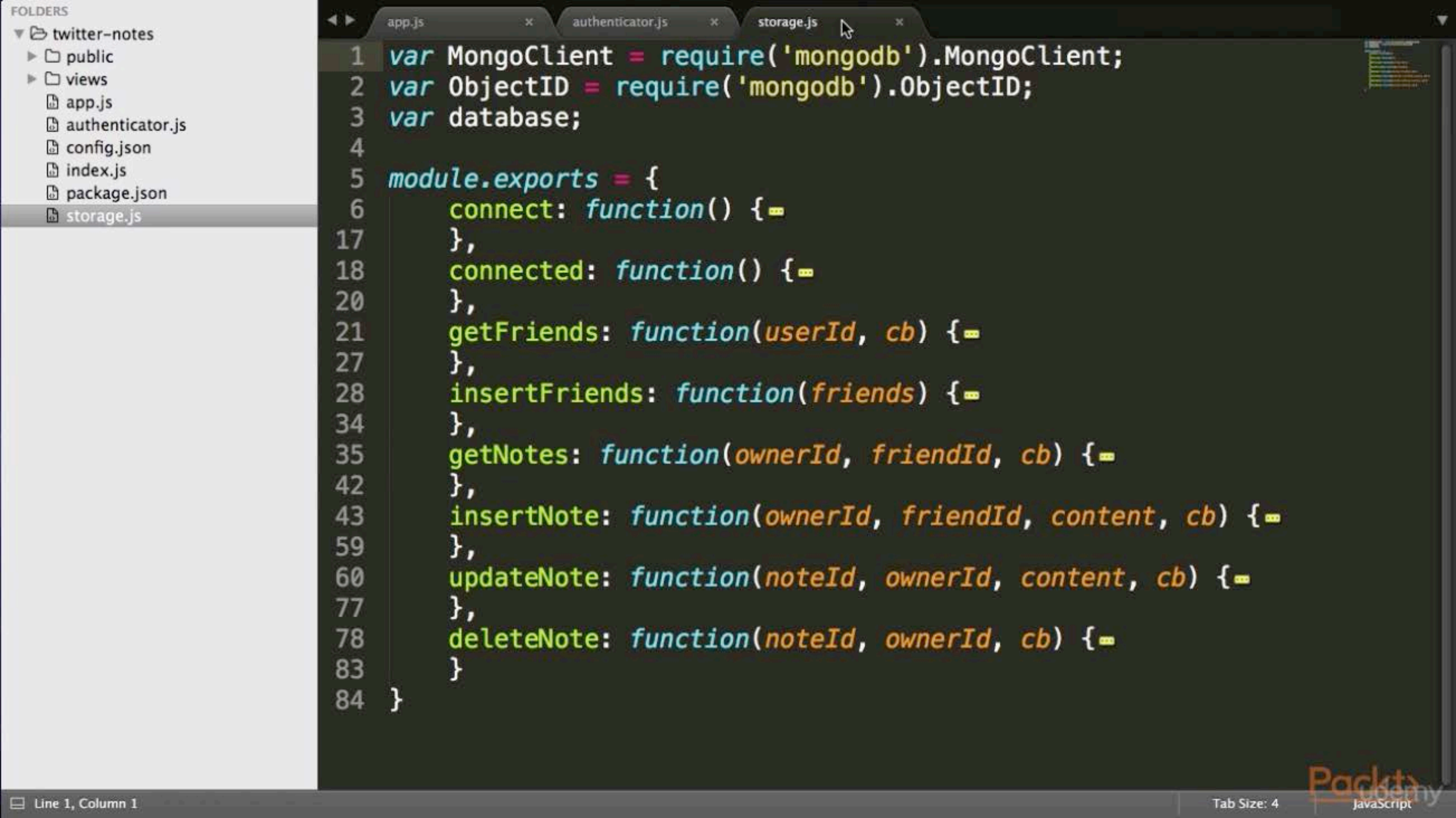
app.js | authenticator.js | storage.js

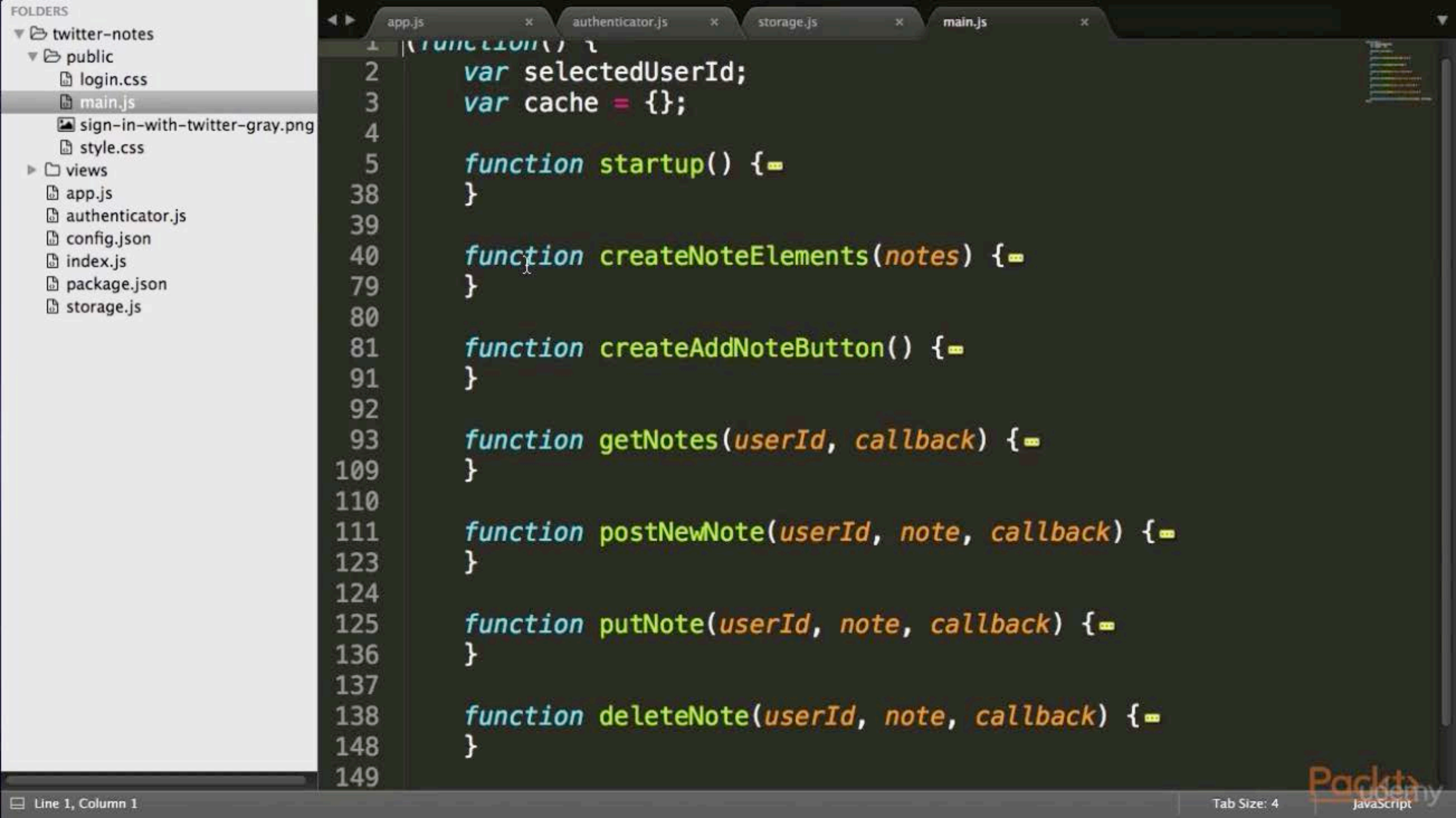
```
1  module.exports = function() {
2      var url = require('url');
3      var express = require('express');
4      var bodyParser = require('body-parser');
5      var querystring = require('querystring');
6      var async = require('async');
7      var authenticator = require('./authenticator');
8      var storage = require('./storage.js');
9      var config = require('./config');
10     var app = express();
11
12     // Connect to MongoDB
13     storage.connect();
14
15     // Set the view engine to ejs
16     app.set('view engine', 'ejs');
17
18     // Add cookie parsing functionality to our Express app
19     app.use(require('cookie-parser')());
20
21     // Parse JSON body and store result in req.body
22     app.use(bodyParser.json());
23
24     // Take user to Twitter's login page
```


FOLDERS
twitter-notes
public
views
app.js
authenticator.js
config.json
index.js
package.json
storage.js
app.js
authenticator.js
storage.js
// Show the login page
app.get('/login', function(req, res) {
});
function ensureLoggedIn(req, res, next) {
}
// Get notes for a friend
app.get('/friends/:uid/notes', ensureLoggedIn, function(req,
});
// Add a new note to a friend
app.post('/friends/:uid/notes', ensureLoggedIn, function(req
});
// Update a note
app.put('/friends/:uid/notes/:noteid', ensureLoggedIn, funct
});
// Delete a note
app.delete('/friends/:uid/notes/:noteid', ensureLoggedIn, fu
});
14 characters selected
Tab Size: 4
JavaScript


FOLDERS
twitter-notes
public
views
app.js
authenticator.js
config.json
index.js
package.json
storage.js
app.js
authenticator.js
storage.js
171 }
172
173 // Get notes for a friend
174 app.get('/friends/:uid/notes', ensureLoggedIn, function(req,
187 });
188
189 // Add a new note to a friend
190 app.post('/friends/:uid/notes', ensureLoggedIn, function(req
200 });
201
202 // Update a note
203 app.put('/friends/:uid/notes/:noteid', ensureLoggedIn, funct
218 });
219
220 // Delete a note
221 app.delete('/friends/:uid/notes/:noteid', ensureLoggedIn, fu
229 });
230
231 // Serve static files in public directory
232 app.use(express.static(__dirname + '/public'));
233
234 // Start listening for requests
235 app.listen(config.port, function() {
14 characters selected
Tab Size: 4
JavaScript


FOLDERS
twitter-notes
public
views
app.js
authenticator.js
config.json
index.js
package.json
storage.js
app.js
authenticator.js
storage.js
var OAuth = require('oauth').OAuth;
var config = require('./config');
// Create the oauth object for accessing Twitter
var oauth = new OAuth(
config.request_token_url,
config.access_token_url,
config.consumer_key,
config.consumer_secret,
config.oauth_version,
config.oauth_callback,
config.oauth_signature
);
module.exports = {
get: function(url, access_token, access_token_secret, cb) {
post: function(url, access_token, access_token_secret, body,
redirectToTwitterLoginPage: function(req, res) {
authenticate: function(req, res, cb) {
};
Line 1, Column 1
Tab Size: 4
JavaScript


FOLDERS
twitter-notes
public
views
app.js
authenticator.js
config.json
index.js
package.json
storage.js
app.js
authenticator.js
storage.js
var MongoClient = require('mongodb').MongoClient;
var ObjectID = require('mongodb').ObjectID;
var database;
module.exports = {
connect: function() {
},
connected: function() {
},
getFriends: function(userId, cb) {
},
insertFriends: function(friends) {
},
getNotes: function(ownerId, friendId, cb) {
},
insertNote: function(ownerId, friendId, content, cb) {
},
updateNote: function(noteId, ownerId, content, cb) {
},
deleteNote: function(noteId, ownerId, cb) {
}
}
Line 1, Column 1
Tab Size: 4
JavaScript


FOLDERS
twitter-notes
public
login.css
main.js
sign-in-with-twitter-gray.png
style.css
views
app.js
authenticator.js
config.json
index.js
package.json
storage.js
app.js
authenticator.js
storage.js
main.js
(function() {
var selectedUserId;
var cache = {};
function startup() {
}
function createNoteElements(notes) {
}
function createAddNoteButton() {
}
function getNotes(userId, callback) {
}
function postNewNote(userId, note, callback) {
}
function putNote(userId, note, callback) {
}
function deleteNote(userId, note, callback) {
}
Line 1, Column 1
Tab Size: 4
JavaScript

Next Video

Creating an OAuth Login Request