**Main.js**

(function() {

var selectedUserId;

var cache = {};

function startup() {

var friends = document.getElementsByClassName('friend');

for (var i = 0; i < friends.length; i++) {

friends[i].addEventListener('click', function() {

// Deselect last selected option

for (var j = 0; j < friends.length; j++) {

friends[j].className = 'friend';

}

// Select friend

this.className += ' active';

// Get notes for selected person

selectedUserId = this.getAttribute('uid');

var notes = getNotes(selectedUserId,   
 function(notes) {

var docFragment =   
 document.createDocumentFragment();

// Add notes

var notesElements =   
 createNoteElements(notes);

notesElements.forEach(function(element) {

docFragment.appendChild(element);

});

// Add the new note button

var newNoteButton = createAddNoteButton();

docFragment.appendChild(newNoteButton);

// Render the downloaded notes

document.getElementById('notes').innerHTML =   
 "";

document.getElementById('notes').  
 appendChild(docFragment);

});

});

}

}

function createNoteElements(notes) {

return notes.map(function(note) {

var element = document.createElement('li');

element.className = "note";

element.setAttribute('contenteditable', true);

element.textContent = note.content;

element.addEventListener('blur', function() {

note.content = this.textContent;

if (note.content == "") {

if (note.\_id) {

deleteNote(selectedUserId, note,   
 function() {

document.getElementById('notes').  
 removeChild(element);

});

} else {

document.getElementById('notes').  
 removeChild(element);

}

} else if (!note.\_id) {

postNewNote(selectedUserId,   
 {content: this.textContent},   
 function(newNote) {

note.\_id = newNote.\_id;

});

} else {

putNote(selectedUserId, note,   
 function() {});

}

});

element.addEventListener('keydown', function(e) {

// If Enter is pressed

if (e.keyCode == 13) {

e.preventDefault();

if (element.nextSibling.className == "add-  
 note") {

element.nextSibling.click();

} else {

element.nextSibling.focus();

}

}

});

return element;

});

}

function createAddNoteButton() {

var element = document.createElement('li');

element.className = "add-note";

element.textContent = "Add a new note...";

element.addEventListener('click', function() {

var noteElement = createNoteElements([{}])[0];

document.getElementById('notes').  
 insertBefore(noteElement, this);

noteElement.focus();

});

return element;

}

function getNotes(userId, callback) {

if (cache[userId]) {

return callback(cache[userId]);

}

var xhttp = new XMLHttpRequest();

xhttp.onreadystatechange=function() {

if (xhttp.readyState == 4 && xhttp.status == 200) {

var notes =   
 JSON.parse(xhttp.responseText) || [];

cache[userId] = notes;

callback(notes);

}

};

xhttp.open("GET", "/friends/" +   
 encodeURIComponent(userId) + "/notes", true);

xhttp.send();

}

function postNewNote(userId, note, callback) {

var xhttp = new XMLHttpRequest();

xhttp.onreadystatechange=function() {

if (xhttp.readyState == 4 && xhttp.status == 200) {

var serverNote = JSON.parse(xhttp.responseText)   
 || {};

// cache[userId] exists because it was created   
 during the GET request.

cache[userId].push(serverNote);

callback(serverNote);

}

};

xhttp.open("POST", "/friends/" +   
 encodeURIComponent(userId) + "/notes", true);

xhttp.setRequestHeader("Content-Type",   
 "application/json;charset=UTF-8");

xhttp.send(JSON.stringify(note));

}

function putNote(userId, note, callback) {

var xhttp = new XMLHttpRequest();

xhttp.onreadystatechange=function() {

if (xhttp.readyState == 4 && xhttp.status == 200) {

var serverNote =   
 JSON.parse(xhttp.responseText).  
 content || {};

callback(serverNote);

}

};

xhttp.open("PUT", "/friends/" +   
 encodeURIComponent(userId) + "/notes/" +   
 encodeURIComponent(note.\_id), true);

xhttp.setRequestHeader("Content-Type",   
 "application/json;charset=UTF-8");

xhttp.send(JSON.stringify(note));

}

function deleteNote(userId, note, callback) {

var xhttp = new XMLHttpRequest();

xhttp.onreadystatechange=function() {

if (xhttp.readyState == 4 && xhttp.status == 200) {

cache[userId] =   
 cache[userId].  
 filter(function(localNote) {   
 return localNote.\_id != note.\_id; });

callback();

}

};

xhttp.open("DELETE", "/friends/" +   
 encodeURIComponent(userId) + "/notes/" +   
 encodeURIComponent(note.\_id), true);

xhttp.send();

}

document.addEventListener('DOMContentLoaded', startup);

})();

**Exercise02\_04\_01 – Step 2**

storage.js

var MongoClient = require('mongodb').MongoClient;

var url = 'mongodb://localhost:27017';

var dbName = 'twitter\_notes';

var database;

------------------------------------

storage.js

module.exports = {

connect: function() {

MongoClient.connect(url, function(err, client) {

if (err) {

return console.log('Error: ' + err);

}

database = client.db(dbName);

console.log('Connected to database: ' + dbName);

});

}

}

------------------------------------

index.js

var app = express();

storage.connect();

------------------------------------

storage.js

},

connected: function () {

return typeof database != 'undefined';

}

------------------------------------

index.js

if (!storage.connected()) {

console.log('Loading friends from Twitter');

renderMainPageFromTwitter(req, res);

}

**Exercise02\_04\_01 – Step 3**

index.js

},

insertFriends: function(friends) {

database.collection('friends').insert(friends,   
 function(err) {

if (err) {

console.log('Cannot insert friends into   
 database.');

}

});

}

------------------------------------

index.js

app.get('/', function(req, res) {

var credentials = authenticator.getCredentials();

if (!credentials.access\_token ||   
 !credentials.access\_token\_secret) {

return res.redirect('/login');

}

if (!storage.connected()) {

console.log('Loading friends from Twitter');

renderMainPageFromTwitter(req, res);

}

});

------------------------------------

index.js

res.render('index', { friends: friends });

if (storage.connected) {

storage.insertFriends(friends);

}

**Exercise02\_04\_01 – Step 4**

storage.js

},

getFriends: function(userId, callback) {

var cursor = database.collection('friends').  
 find({ for\_user: userId});

cursor.toArray(callback);

}

------------------------------------

index.js

console.log('Loading friends from MongoDB');

storage.getFriends(credentials.twitter\_id, function(err,   
 friends) {

if (err) {

return res.status(500).send(error);

}

if (friends.length > 0) {

console.log('Friends successfully loaded from   
 MongoDB');

friends.sort(function(a, b) {

return a.name.toLowerCase().  
 localeCompare(b.name.toLowerCase());

});

res.render('index', { friends: friends});

}

else {

console.log('Loading friends from Twitter');

renderMainPageFromTwitter(req, res);

}

});

**Exercise02\_04\_01 – Step 5**

index.js

------------------------------------

index.js

------------------------------------

index.js

------------------------------------

index.js

------------------------------------

index.js

------------------------------------

index.js

------------------------------------

index.js

------------------------------------

index.js

------------------------------------

index.js

------------------------------------

index.js