Corso di Progettazione di Applicazioni Web e Mobile

Mirko Calvaresi

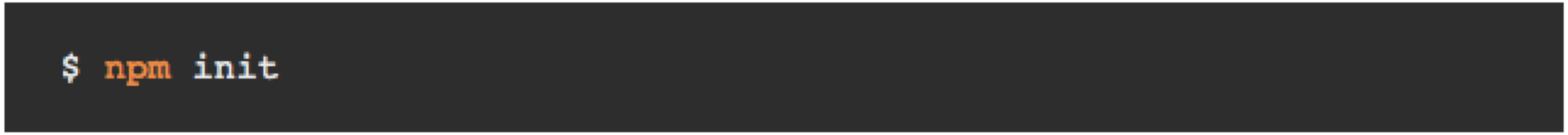
BUILDING A SIMPLE WEB APPLICATION WITH A BACKEND

NODE JS EXPRESS AND SQLITE

THE WEB APPLICATION

In order to build the application we are going to use the NODE JS stack

A good introduction can be found here
https://zellwk.com/blog/crud-express-mongodb/



```
$ npm init
```

THE WEB APPLICATION

We are going to use

Node JS

For the server

Express

To handle the routing

Sqlite

To manage the data

EJS

as template engine

THE WEB APPLICATION

We are going to use

- Node JS
- Express
- Sqlite

https://expressjs.com/en/starter/installing.html

```
npm init
npm install express --save
npm install —save express-session
npm install --save body-parser
npm install --save express-partials
npm install ejs --save
npm install --save sqlite3
npm install cookie-session --save
npm install cookie-parser --save
```

THE WEB APPLICATION

Starting the server with the first template
https://www.codementor.io/naeemshaikh27/node-with-express-and-ejs-du107lnk6

```
const express = require('express');
const app = express();
app.set('view engine', 'ejs');

app.get('/', function(req, res){
res.render('index',{user: "Great User",title:"homepage"});
});

app.listen(3000,function(){
console.log("Live at Port 3000");
});
```

NODE JS BASE ROUTING

The idea in NodeJS routing is to associate a specific handler for the single path and method.

The HTTP request and response object are passed to the handler.

```
const express = require('express');
const app = express();
app.set('view engine', 'ejs');

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

app.post('/login', function(req, res){
…
});
```

NODE JS SQL CLIENT

In order to integrate a SQL client we need to create a specific module to expose the DB calls

```
const sqlite3 = require('sqlite3').verbose();
const database = './student.db';

module.exports = {
getStudents: function (callback) {
let db = new sqlite3.Database(database);

var students = []


let sql = `SELECT * FROM STUDENT ORDER BY NAME DESC`;
```

INTEGRATE DB CALLS IN THE VIEWS

In order to integrate a SQL client we need to create a specific module to expose the DB calls

```
const sqlite3 = require('sqlite3').verbose();
const database = './student.db';

module.exports = {
getStudents: function (callback) {
let db = new sqlite3.Database(database);

var students = []


let sql = `SELECT * FROM STUDENT ORDER BY NAME DESC`;
```

NODE JS SQL CLIENT

In order to integrate a SQL in the view we can pass a specific call to the sql module handler

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

        sqllite.getStudents( function (students) {
                res.render('students', {
                "students": students
                });

        })

});
```

THE WEB APPLICATION

To watch and reload the application

```
npm install nodemom
nodemon app.js
```

GIT HUB Repository

All the code is present on github
https://github.com/mccalv/unicam

```
git clone git@github.com:mccalv/unicam.git
cd unicam
npm install
nodemon app-ej.js
```