بخش اول: راه‌اندازی و نصب MongoDB، Node.Js و Express
چگونه در 60 دقیقه یک صفحه وب بسازیم؟

امروزه، سامانه‌های مدیریت محتوا به همه مردم حتی افرادی با کمترین دانش فنی اجازه می‌دهند یک سایت کامل را طراحی کنند. پس تعجب نکنید که بیشتر سایت‌ها به لحاظ رابط کاربری شباهت زیادی به یکدیگر دارند و تنها نحوه چیدمان عناصر روی صفحات ممکن است متفاوت باشد. بخش عمده‌ای از سایت‌ها بر پایه وردپرس، دروپال و دیگر سامانه‌های مدیریت محتوا ساخته می‌شوند. در این مقاله نمی‌خواهیم به شما نشان دهیم که چگونه از یک سامانه مدیریت محتوا برای برنامه‌نویسی وب استفاده کنید، بلکه هدف آن است که به شما نشان دهیم که چگونه Node.js، MongoDB و Express را نصب‌کرده، یک صفحه وب را ایجاد کرده و آن را به پایگاه داده MongoDB متصل کنید. در این مقاله آموزشی با نحوه راه‌اندازی و به‌کارگیری Node.JS، پایگاه داده MongoDB و چهارچوب Express آشنا خواهید شد. برای دسترسی به کدهای کامل این راهنما به مخزن گیت‌ها به نشانی https://github.com/cwbuecheler/node-tutorial-for-frontend-devs مراجعه کنید.

این مقاله در دو بخش منتشر شده است. برای دیدن بخش دوم اینجا را کلیک کنید.

با یک جست‌وجوی ساده در فضای مجازی، میلیون‌ها مقاله آموزشی پیدا می‌کنید که ساخت برنامه Hello World با Node.JS را آموزش داده‌اند. این مقاله‌های آموزشی برای افرادی که هدف‌شان آشنایی با نحوه ساخت برنامه‌ها با Node.Js است، مناسب هستند. اما اگر تمایل دارید در کنار یادگیری Node.Js با جزییات بیشتری آشنا شوید، باید به سراغ منابع آموزشی بیشتری بروید. اما مشکلی در این میان وجود دارد. برخی از آموزش‌های سطح متوسط و پیشرفته فرض می‌کنند خواننده با اطلاعات مقدماتی آشنایی دارید، اما در 90 درصد موارد این‌گونه نیست. در نتیجه یک مقاله آموزشی بهتر است با رویکرد آموزش تدریجی نگارش شود تا برای بیشتر خوانندگان مفید باشد. مقاله‌ای که در حال مطالعه آن هستید بر پایه یک آموزش تدریجی آماده‌شده و سعی کرده نحوه کار با سه ابزار کاربردی موردنیاز توسعه‌دهندگان فرانت‌اند را آموزش دهد. در نتیجه آشنایی مقدماتی با جاوااسکریپت، CSS3 و HTML5 برای درک بهتر کدهای این مقاله مفید خواهد بود. در این مقاله یک برنامه کاربردی ایجاد می‌کنیم که به یک پایگاه داده متصل شده، محاوره‌هایی روی پایگاه داده انجام داده، نتایج به‌دست‌آمده را دستکاری کرده و تغییرات را روی پایگاه داده ذخیره خواهد کرد.

مطلب پیشنهادی

وب‌سایت سازهای موبایل
چگونه با گوشی همراه خود یک وب سایت بسازیم؟

بخش اول- نصب در 15 دقیقه

اگر در زمینه کار با ابزارهایی که اشاره شد، مبتدی هستید، باید مدت‌زمانی را صرف پیاده‌سازی و اجرای سه ابزار یاد شده کنید. فرآیند نصب و راه‌اندازی که روی ویندوز 10 انجام می‌شود، چندان پیچیده نیست و به‌سادگی انجام خواهد شد.

مرحله اول: نصب Node.Js

یک فرآیند ساده پیش‌رو دارید، به سایت Node.js.com رفته و روی دکمه سبز رنگ نصب که برای نسخه LTS (نسخه پایدار) ارائه‌شده کلیک کنید. فایل نصبی متناسب با معماری سیستم‌عامل دانلود می‌شود. (در زمان نگارش این مقاله دو نسخه پایدار 10.15.0 و جدیدترین نسخه 11.6.0 ارائه‌شده است.) فایل نصبی را اجرا کنید. مراحل نصب را با کلیک دکمه Next پشت سر بگذارید تا نصب کامل شود. سپس ابزار مدیریت بسته‌های نود موسوم به NPM را نصب کنید. این ابزار اجازه می‌دهد، پکیج‌های مختلف و کاربردی را به Node اضافه کنید. پنجره خط فرمان را باز کرده و با دستور cd به پوشه‌ای بروید که برنامه‌های تست در آن قرار خواهند گرفت. در این مقاله مسیر ما C:\ node است.

مرحله 2: نصب Express Generator

اکنون که Node اجرا شده باید مولفه‌هایی را که برای ایجاد وب‌سایت نیاز داریم، نصب کنیم. ابتدا باید چهارچوب Express را نصب کنیم. این چهارچوب نود را از یک برنامه زیرساختی جدا کرده و به ماهیتی تبدیل می‌کند که نقش یک وب سرور را بازی خواهد کرد. کار را با
Express-Generator که متفاوت از Express است، آغاز کنید. در حقیقت Express-Generator نقش یک شالوده را برای سایت‌هایی که با Express ایجاد می‌شوند، بازی می‌کند. در محیط خط فرمان و در مسیر C:\Node دستور زیر را وارد کنید:

npm install -g express-generator

Generator به شکل خودکار نصب خواهد شد. (شکل 1)

1□□□ □□□ Express Generator

مرحله سوم: ساخت یک پروژه Express

در این مقاله از Express و Jade استفاده می‌کنیم، اما از پیش‌پردازنده Stylus Css که بیشتر توسعه‌دهندگان از آن استفاده می‌کنند، استفاده نخواهیم کرد و به‌جای آن از CSS مستقیم استفاده خواهیم کرد. ما باید از Jade یا سایر موتورهای قالب‌ساز برای دسترسی به داده‌هایی که بر پایه Node/Express قرار دارند، استفاده کنیم. اگر اطلاعات کافی درباره اچ‌تی‌ام‌ال دارید، یادگیری Jade کار دشواری نیست. دقت کنید که تورفتگی کدها برای Jade اهمیت دارد و عدم توجه به آن‌ها باعث خراب شدن کدها می‌شود. Jade نسبت به تورفتگی‌ها، فاصله‌ها و Tabها حساس بوده و تورفتگی‌ها باید همگی قالب مشخصی داشته باشند. در پوشه C:\node یا هر پوشه‌ای که قرار است برنامه Node در آن ذخیره شود، رفته و فرمان زیر را اجرا کنید.

C:\node> express nodetest1

خروجی فرمان بالا همانند شکل 2 خواهد بود.

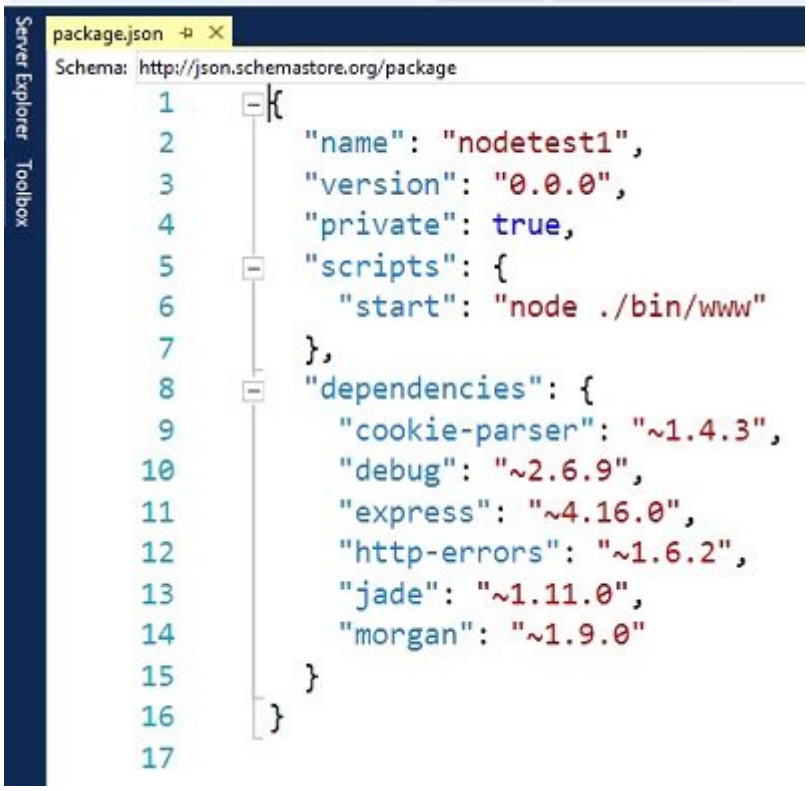
□□□2. □□□□□ □□ □□□□□ □□ Express

```
C:\Node>express nodetest1

   create : nodetest1\
   create : nodetest1\public\
   create : nodetest1\public\javascripts\
   create : nodetest1\public\images\
   create : nodetest1\public\stylesheets\
   create : nodetest1\public\stylesheets\style.css
   create : nodetest1\routes\
   create : nodetest1\routes\index.js
   create : nodetest1\routes\users.js
   create : nodetest1\views\
   create : nodetest1\views\error.jade
   create : nodetest1\views\index.jade
   create : nodetest1\views\layout.jade
   create : nodetest1\app.js
   create : nodetest1\package.json
   create : nodetest1\bin\
   create : nodetest1\bin\www

   change directory:
     > cd nodetest1

   install dependencies:
     > npm install

   run the app:
     > SET DEBUG=nodetest1:* & npm start


C:\Node>
```

مرحله چهارم: ویرایش وابستگی‌ها

یک‌سری ساختارهای پایه را ایجاد کردید، اما این ساختار هنوز تکمیل‌نشده است. به‌احتمال‌زیاد متوجه شده‌اید که روال Express-generator در پوشه Nodetest1 فایلی به نام package.json ایجاد کرده است. فایل فوق را در یک ویرایشگر متنی باز کنید. محتوای فایل شبیه شکل 3 خواهد بود.

□□□3. □□□□□ □□□□ □□□□ package.json

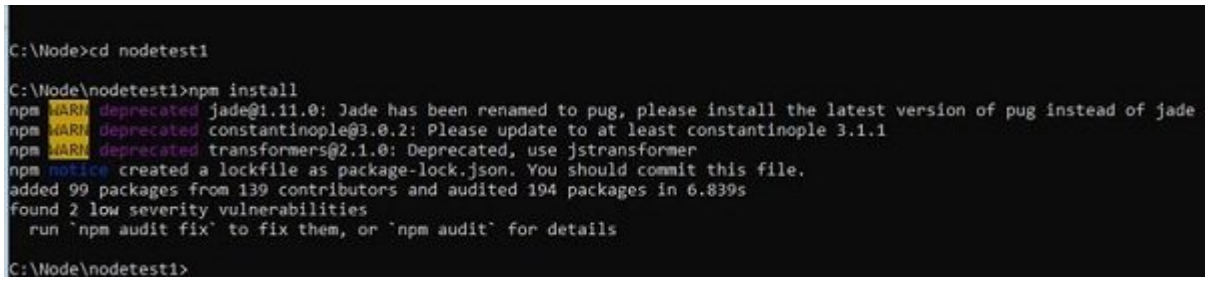


فایلی که مشاهده می‌کنید یک فایل اولیه JSON است که برنامه ما و وابستگی‌های مرتبط با آن را تشریح کرده است. اکنون باید در این فایل تغییراتی ایجاد کرده و مواردی را اضافه کنیم. برای این منظور باید فراخوانی MongoDB و Monk به این فایل اضافه شود. اکنون شی وابستگی‌ها باید همانند شکل 4 ویرایش شده باشد. توجه داشته باشید فایل درست JOSN را انتخاب کرده باشید، در غیر این صورت برنامه کاربردی شما نصب نخواهد شد. اطمینان حاصل کنید، کاماهای پس از هر خط به شکل درستی وارد شده باشند.

□□□4. □□ □□□□□□□□ □□□□□□□□ □□ □□□□ package.json

مرحله 5، نصب وابستگی‌ها

تا این مرحله وابستگی‌های موردنیاز را تعریف کرده‌ایم و آماده رفتن به مرحله بعدی هستیم. شماره نسخه‌ها برای این دو ماژول یکسان با جدیدترین به‌روزرسانی‌ها هستند، البته نسخه‌های جدید ابزار NPM به شکل مستمر جایگزین می‌شوند. دومرتبه به خط فرمان رفته، به پوشه nodetest1 رفته و فرمان C:\node\nodetest1>npm install را اجرا کنید. خروجی این فرمان باید شبیه شکل 5 باشد.



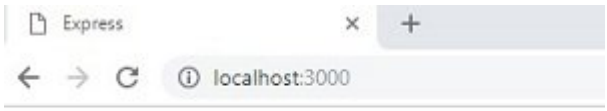
.5□□□
□□□
□□□□□□□
□□□
□□□□
□□□□
json

این فرمان(شکل5)، فایل ایجاد شده JSON را خوانده، اشیا و وابستگی‌های موجود در فایل JSON را روی کامپیوتر شما نصب می‌کند. وابستگی‌های نصب‌شده شامل Express می‌شوند. درست است که تولیدکننده را به شکل سراسری نصب کرده‌ایم، اما باید ماژول واقعی موجود در این شی خاص را نیز نصب کنیم. طی مراحل نصب ممکن است پیغام‌های هشداری را مشاهده کنید که یکی از هشدارها در ارتباط با تغییر نام Jade به Pug است. در این مقاله نیازی نیست به این هشدارها توجه کنید؛ چون مشکلی در کار ایجاد نمی‌کنند. زمانی‌که NPM اجرا می‌شود، پوشه‌ای به نام Node_Modules خواهید داشت که همه وابستگی‌های این راهنما درون آن قرار دارند. اکنون یک برنامه عملیاتی قابل اجرا را در اختیار داریم. حال باید وب‌سرور را آزمایش کنیم. در محیط خط فرمان دستور Npm Start را اجرا کنید. با فشار کلید اینتر خروجی شکل 6 را مشاهده خواهید کرد. (اگر پیغام امنیتی ویندوز را مشاهده کردید روی دکمه Allow access کلیک کنید.) اگر با اجرای فرمان فوق موفق شدید دو خط زیر را مشاهده کنید، به معنای آن است که همه‌چیز را درست نصب و اجرا کرده‌اید.

nodetest1@0.0.0 start C:\Node\nodetest1
node ./bin/www

.6□□□
□□□□□
□□□
□□□□
□□□□□
□□□□
Node
□□□□□□□
□□□ □
□□□□
□□□ □□□

اگر مرورگر وبی را باز کرده و آدرس http://localhost:3000 را در آن وارد کنید به صفحه خوشامدگویی Express خواهید رفت. (شکل 7)



شکل 7. صفحه نشان داده شده توسط Express

این صفحه نشان می‌دهد که وب‌سرور Node.Js و موتور Express در حال اجرا هستند و پیش‌پردازشگر اچ‌تی‌ام‌ال موسوم به Jade نیز نصب‌شده است.

بخش دوم ـ ساخت برنامه Hello World

محیط توسعه یکپارچه یا ویرایشگر حرفه‌ای مورد علاقه خود را باز کنید. برخی از توسعه‌دهندگان از ابزار Sublime Text استفاده می‌کنند. در ویرایشگر خود به پوشه nodetest1 رفته و فایل app.js را باز کنید. فایل فوق در برنامه ما نقش کلیدی دارد. همان‌گونه که در شکل 8 مشاهده می‌کنید، اطلاعات جالبی درون این فایل وجود دارد.

شکل 8. فایل app.js

فایل فوق گروهی از متغیرهای اولیه جاوااسکریپت را ایجاد کرده و این متغیرها را به یکسری بسته‌های ویژه، وابستگی‌ها، عملکرد نود و مسیرها متصل کرده است. این مسیرها در اصل نوعی کنترل‌کننده بوده و برای هدایت ترافیک استفاده می‌شوند. این متغیرها منطق خاصی از برنامه‌نویسی را شامل می‌شوند. البته می‌توانید معماری MVC را همراه با Express ایجاد کنید. (که مبحث فوق فراتر از آموزش ما است.) در این مرحله باید از مسیر کاربر صرف‌نظر کرده و روی بالاترین سطح از مسیر که به‌وسیله c:\node\nodetest1\routes\index.js کنترل می‌شود، متمرکز شویم. دستور var app=express(); در این فایل مهم است، زیرا اکسپرس را مقداردهی اولیه کرده و متغیر برنامه ما را به آن اختصاص می‌دهد. در ادامه از متغیر فوق برای پیکربندی یکسری خصلت‌های اکسپرس استفاده خواهیم کرد. مجموعه دستورات دیگر قرار گرفته در این فایل به برنامه ما می‌گویند که نماها (views) را باید کجا پیدا کرده، چه موتوری برای پردازش نماها استفاده‌شده و چند متد برای آماده‌سازی و اجرای ملزومات فراخوانی خواهند شد. دقت کنید که خط آخر به چهارچوب Express اعلام می‌دارد که اشیا ایستا را باید از پوشه /public/ ارائه کند، اما این ارائه باید به شکلی انجام شود که نشان دهد اشیا از بالاترین سطح ارائه شده‌اند.

```
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'jade');

app.use(logger('dev'));
app.use(express.json());
app.use(express.urlencoded({ extended: false }));
app.use(cookieParser());
app.use(express.static(path.join(__dirname, 'public')));

app.use('/', indexRouter);
app.use('/users', usersRouter);
```

برای مثال، پوشه و مسیر تصاویر ممکن است به شکل c:\node\nodetest1\public\images باشد، اما دسترسی به تصاویر و اشیا از طریق آدرس http://localhost:3000/images انجام خواهد شد. دو بلوک انتهایی این فایل که

در فهرست 1 مشاهده می‌کنید، حائز اهمیت هستند.

```
// catch 404 and forward to error handler
app.use(function(req, res, next) {
  next(createError(404));
});

// error handler
app.use(function(err, req, res, next) {
  // set locals, only providing error in development
  res.locals.message = err.message;
  res.locals.error = req.app.get('env') === 'development' ? err : {};

  // render the error page
  res.status(err.status || 500);
  res.render('error');
});
```

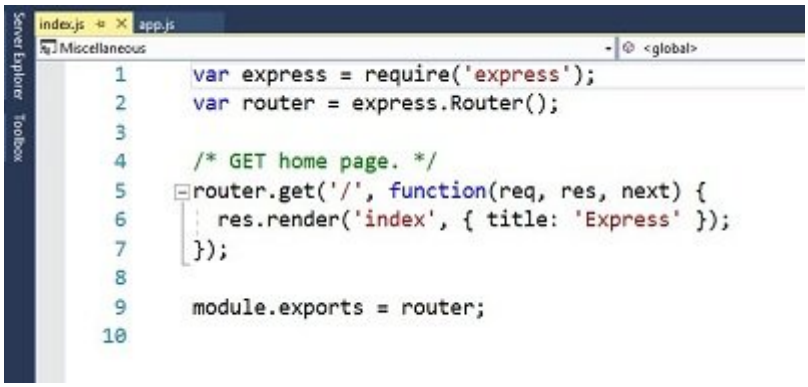
فهرست یک

این کدها به Express می‌گویند که درخواست‌های ارسال به آدرس http://localhost:3000 باید از مسیریاب index که ابتدای فایل تعریف شده‌اند، استفاده کنند و درخواست‌ها باید از طریق فایل مسیریاب users در آدرس http://localhost:3000/users پردازش شوند. در این قطعه کد نحوه پردازش خطاها نیز مشخص‌شده؛ آخرین خط این فایل module.exports=app; بخش زیربنایی Node است که برای اکسپورت کردن همه ماژول‌ها به یک شی استفاده می‌شود. برنامه اصلی ما نیز شی app خود را اکسپورت می‌کند.
تا این مرحله یکسری کارهای اولیه را انجام داده‌ایم. اما تمایلی نداریم در صفحه index برنامه عبارت ساده Hello World نشان داده شود، بلکه دوست داریم درباره مسیرها اطلاعات بیشتری به دست آورده و با عملکرد موتور Jade برای جمع‌آوری ملزومات بیشتر آشنا شویم. در ابتدا یک فرمان جدید app.use به فایل app.js اضافه می‌کنیم. در فایل فوق بخشی را که دو فرمان زیر در آن قرار دارد، پیدا کنید.

```
app.use('/', indexRouter);
app.use('/users', usersRouter);
```

این دو دستور به Express می‌گویند باید از چه فایل‌های مسیریابی استفاده کند. بهتر است فایل‌های مسیریابی مستقلی برای بخش‌های مختلف برنامه ایجاد کرد. برای مثال، فایل مسیر Users می‌تواند مشتمل بر اضافه، حذف و به‌روزرسانی کاربران باشد، درحالی‌که فایل مسیریابی جدیدی به نام Locations برای اضافه، حذف، ویرایش و نمایش داده‌های مکانی باشد. برای آن‌که کارها به شکل ساده‌ای انجام شود، باید همه کارها در مسیریاب index انجام شود. به عبارت دقیق‌تر باید از خط /users صرف‌نظر کرد. دقت کنید متغیر Routes از قبل در اکسپرس تعریف‌شده و به مسیریاب Index اشاره دارد. ما در نظر داریم متد Helloworld را به‌گونه‌ای به مسیریاب اضافه کنیم که به صفحه‌ای متفاوت از صفحه پیش‌فرض اشاره کند. در ویرایشگر خود پوشه Routes را باز کرده و فایل index.js را پیدا کنید. فایل فوق شبیه شکل9 خواهد بود.

□□□9. □□□□ index.js

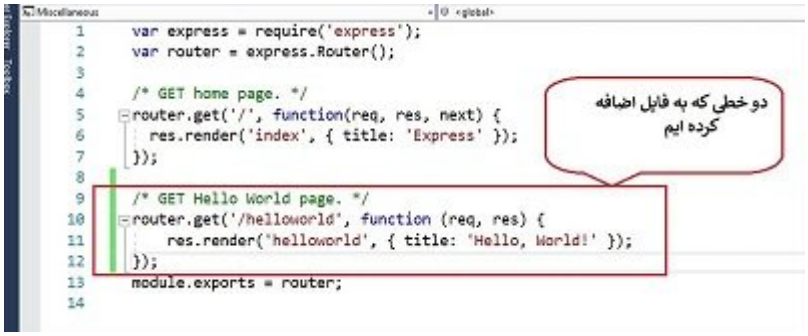


محتوای این فایل پیچیدگی خاصی ندارد. ما در نظر داریم عملکرد Express را تنظیم کرده و در ادامه متغیر Router را به متد مسیریاب Express ضمیمه کرده و در مرحله بعد زمانی که یک درخواست HTTP برای وب‌سایت ما ارسال می‌شود، از متد فوق استفاده کرده و در انتها تابع مسیریاب خود را به برنامه اکسپورت کنیم. تابع Get را می‌توان به‌سادگی به صفحه دیگری کپی کرده و همین کار را انجام دهیم. در انتهای این فایل و بالای خط module.exports دستورات زیر را وارد کنید:

GET Hello World page. */ */
router.get(‘/helloworld’, function(req, res) {
;({ ’! res.render(‘helloworld’, { title: ‘Hello, World
;({

اکنون فایل شما باید شبیه شکل 10 باشد.

□□□10. □□□□□□□ □□□□ index.js □□□□ □□□□□□ □□□□ □□□□ □□□□□□□



این تمام آن کاری است که باید برای مدیریت مسیریابی آدرس اینترنتی انجام می‌دادیم. با این حال، هنوز یک صفحه درست برای پاسخ‌گویی به این درخواست ایجاد نکرده‌ایم. برای این‌کار باید از Jade استفاده کنیم. به پوشه Views رفته و فایل Index.jade را باز کنید. پیش از انجام هر کاری، فایل فوق را با نام Helloworld.jade ذخیره کنید. دستورات درون این فایل به شرح زیر هستند:

extends layout

block content
{ h1= title p Welcome to #{title

این فایل سرراست بوده و از فایل Layout.jade به‌عنوان قالب استفاده کرده و آن را گسترش می‌دهد. درون بلوک content یک سرتیتر و یک پاراگراف وجود دارد. به متغیر Title که پیش از این در مسیر Index.js آن را تعیین کرده‌ایم، توجه کنید. ما نیازی نداریم در این کد تغییر خاصی بدهیم تا مسیریاب بتواند صفحه‌ای متفاوت از صفحه اصلی را نشان دهد، تنها به یک تغییر کوچک نیاز داریم. اکنون قطعه کد به شکل زیر خواهد بود:

extends layout

block content
h1= title
{ p Hello, World! Welcome to #{title

فایل را ذخیره کنید. اکنون به محیط خط فرمان بروید. اگر سرور همچنان در حال اجرا است، کلیدهای Ctrl+C را

برای متوقف کردن آن فشار داده و فرمان زیر را برای راه‌اندازی مجدد سرور اجرا کنید.

Npm start

مطلب پیشنهادی

معرفی کتاب:
نام کتاب: کدنویسی و توسعه وب ویژه افراد تازه‌کار

دقت کنید که تغییرات اعمال‌شده در قالب‌های Jade به راه‌اندازی مجدد سرور نیازی ندارند، در مقابل هر زمان یک فایل js شبیه به app.js یا فایل‌های مسیر تغییر می‌کنند، باید این کار را برای مشاهده تغییرات انجام دهید. اکنون که سرور مجدد راه‌اندازی شده به آدرس
http://localhost:3000/helloworld بروید. صفحه جدید همانند شکل 11 خواهد بود.

□□□11. □□□□ □□□□□□□□□□□□ *Express* □□ □□□□ □□□□□□ □□ □□ □□□□ □□□□□.

تا این مرحله موفق شده‌ایم مسیریاب خود را به شکلی تنظیم کنیم که بر مبنای آدرس‌های وارد شده، کاربران را به نماهای مختلف هدایت کند. در مرحله بعد باید یکسری مدل‌سازی انجام داده و پایگاه داده خود را ایجاد کنیم. در شماره آینده این مبحث را ادامه خواهیم داد.

تاریخ انتشار:
10 فروردین 1398

نشانی منبع:
https://www.shabakeh-mag.com/workshop/14821/%DA%86%DA%AF%D9%88%D9%86%D9%87-%D8%AF%D8%B1-60-%D8%AF%D9%82%DB%8C%D9%82%D9%87-%DB%8C%DA%A9-%D8%B5%D9%81%D8%AD%D9%87-%D9%88%D8%A8-%D8%A8%D8%B3%D8%A7%D8%B2%DB%8C%D9%85%D8%9F