Installation

Installation is very easy and only takes a few seconds.

Requirements

Please make sure your computer/server meets the following requirements:

- Node.js >= 9: Node.js is a server platform which runs JavaScript. Installation guide here.
- MongoDB: MongoDB is a powerful document store. Installation guide here.

Setup

Time to install Strapi!

```
npm install strapi@alpha -g
```

Note: if you encounter npm permissions issues, change the permissions to npm default directory.

It takes about 20 seconds with a good Internet connection. You can take a coffee ☕ if you have a slow one.

Having troubles during the installation? Check if someone already had the same issue https://github.com/strapi/strapi/issues. If not, you can post one, or ask for help https://strapi.io/support.

Check installation

Once completed, please check that the installation went well, by running:

```
strapi -v
```

That should print `3.0.0-alpha.x`.
Strapi is installed globally on your computer. Type `strapi` in your terminal you will have access to every available command lines.

- Install MongoDB >
- Install MongoDB Community Edition >
- Install MongoDB Community Edition on macOS

Install MongoDB Community Edition on macOS

On this page

Overview

Use this tutorial to install MongoDB Community Edition on macOS systems.

PLATFORM SUPPORT

MongoDB 3.6 is not tested on APFS, the new filesystem in macOS 10.13 and may encounter errors.

Starting in version 3.0, MongoDB only supports macOS versions 10.7 (Lion) and later on Intel x86-64.

You may download MongoDB Community Edition through either the MongoDB Download Center or the popular macOS package manager Homebrew.

NOTE

Starting in MongoDB 3.6, MongoDB binaries, `mongod` and `mongos`, bind to localhost by default. Previously, starting in MongoDB 2.6, only the binaries from the official MongoDB RPM (Red Hat, CentOS, Fedora Linux, and derivatives) and DEB (Debian, Ubuntu, and derivatives) packages bind to localhost by default. For more details, see Localhost Binding Compatibility Changes.

Install MongoDB Community Edition

NOTE

To install a different version of MongoDB, please refer to that version's documentation. For example, see version 3.4.

Install MongoDB Community Edition Manually

1

Download the binary files for the desired release of MongoDB.
Download the binaries from the MongoDB Download Center.

2

Extract the files from the downloaded archive.
For example, from a system shell, you can extract through the `tar` command:

copy
copied

```
tar -zxvf mongodb-osx-ssl-x86_64-3.6.4.tgz
```

3

Copy the extracted archive to the target directory.
Copy the extracted folder to the location from which MongoDB will run.

copy
copied

```
mkdir -p mongodb
cp -R -n mongodb-osx-ssl-x86_64-3.6.4/ mongodb
```

4

Ensure the location of the binaries is in the `PATH` variable.
The MongoDB binaries are in the `bin/` directory of the archive. To ensure that the binaries are in your `PATH`, you can modify your `PATH`.

For example, you can add the following line to your shell's `rc` file (e.g. `~/.bashrc`):

copy
copied

```
export PATH=<mongodb-install-directory>/bin:$PATH
```

Replace `<mongodb-install-directory>` with the path to the extracted MongoDB archive.

Install MongoDB Community Edition with Homebrew

Homebrew installs binary packages based on published "formulae." This section describes how to update `brew` to the latest packages and install MongoDB Community Edition. Homebrew requires some initial setup and configuration, which is beyond the scope of this document.

1

Update Homebrew's package database.
In a system shell, issue the following command:

copy

copied

```
brew update
```

2

Install MongoDB.

You can install MongoDB via `brew` with several different options. Use one of the following operations:

Install the MongoDB Binaries

To install the MongoDB binaries, issue the following command in a system shell:

copy
copied

```
brew install mongodb
```

Install the Latest Development Release of MongoDB

To install the latest development release for use in testing and development, issue the following command in a system shell:

copy
copied

```
brew install mongodb --devel
```

Run MongoDB

1

Create the data directory.

Before you start MongoDB for the first time, create the directory to which the `mongod` process will write data. By default, the `mongod` process uses the `/data/db` directory. If you create a directory other than this one, you must specify that directory in the `dbpath` option when starting the `mongod` process later in this procedure.

The following example command creates the default `/data/db` directory:

copy
copied

```
mkdir -p /data/db
```

2

Set permissions for the data directory.

Before running `mongod` for the first time, ensure that the user account running `mongod` has read and write permissions for the directory.

3

Run MongoDB.

To run MongoDB, run the `mongod` process at the system prompt. If necessary, specify the path of the `mongod` or the data directory. See the following examples.

Run without specifying paths
If your system `PATH` variable includes the location of the `mongod` binary and if you use the default data directory (i.e., `/data/db`), simply enter `mongod` at the system prompt:

copy
copied

```
mongod
```

Specify the path of the ***`mongod`***
If your `PATH` does not include the location of the `mongod` binary, enter the full path to the `mongod` binary at the system prompt:

copy
copied

```
<path to binary>/mongod
```

Specify the path of the data directory
If you do not use the default data directory (i.e., `/data/db`), specify the path to the data directory using the `--dbpath` option:

copy
copied

```
mongod --dbpath <path to data directory>
```

4

Verify that MongoDB has started successfully

Verify that MongoDB has started successfully by checking the process output for the following line:

copy
copied

```
[initandlisten] waiting for connections on port 27017
```

The output should be visible in the terminal or shell window.

You may see non-critical warnings in the process output. As long as you see the log line shown above, you can safely ignore these warnings during your initial evaluation of MongoDB.

5

Begin using MongoDB.

Start a `mongo` shell on the same host machine as the `mongod`. Use the `--host` command line option to specify the localhost address and port that the `mongod` listens on:

copy
copied

```
mongo --host 127.0.0.1:27017
```

Later, to stop MongoDB, press `Control+C` in the terminal where the `mongod` instance is running.

How to Install Node.js and NPM on a Mac

JavaScript is one of the most popular programming languages in the world. Because it's built into most web browsers, programmers and web designers can use JavaScript to add interactive features to websites that reach billions of people. But in the past couple of years, JavaScript has started to play a larger role outside of the browser, due in large part to Node.js.

Node.js is a tool for building fast network applications. It's known as a "JavaScript runtime environment" which simply means it lets you write JavaScript code that can run on your computer free of any web browser. Node.js is used to create fast web servers by companies like Walmart, eBay and Netflix.

But because Node.js can be used on your desktop computer, programmers have created useful Node-based tools that help with the process of building web sites. For example, Grunt is a popular tool used to automate common tasks like compiling Sass files to CSS, making JavaScript files smaller so they load in less time, and compressing images to smaller file size. While these tools run through the Node.js environment, you'll use another tool, NPM, to install them. NPM is what's called a "package manager." NPM makes installing a tool like Grunt as easy as `npm install -g grunt-cli`.

But before you can use Node.js or NPM you need to install them — while the NodeJS website includes an installer, there's a better way to install them on a Mac. In this article, I'll take you through the process of installing Node.js and NPM on a Mac using Homebrew. In another article, I'll show you Windows users how to install them on the Windows operating system.

Prerequisites

Before you install Node.js and NPM you'll first need to have some familiarity with the Mac Terminal application. Terminal lets you dig into the underbelly of the operating system and issue text commands to your computer. You'll need to use Terminal (or a similar application like iTerm) to not only install Node.js but also to use it and NPM.

Before you can install Node, you'll need to install two other applications. Fortunately, once you've got these on your machine, installing Node takes just a few minutes.

1. **XCode.** Apple's XCode development software is used to build Mac and iOS apps, but it also includes the tools you need to compile software for use on your Mac. XCode is free and you can find it in the Apple App Store.
2. **Homebrew**. Homebrew is a package manager for the Mac — it makes installing most open source sofware (like Node) as simple as writing `brew install node`. You can learn more about Homebrew at the Homebrew website. To install Homebrew just open Terminal and type `ruby -e "$(curl -fsSL`

`https://raw.githubusercontent.com/Homebrew/install/master/install)"`. You'll see messages in the Terminal explaining what you need to do to complete the installation process.

Why Homebrew?

Observant readers will notice in the screenshot above that there's an installer for NodeJS. You can download it directly from NodeJS.org. I recommend Homebrew over that installer for a few reasons:

1. When installing Node via the installer, you have to use the `sudo` command to make sure it installs properly (there is a workaround for this, but it's complicated). `sudo` lets the installer place files in areas of your file system that are only accessible to administrators. One nice thing about Homebrew is that it doesn't require access to administrator-only areas of your computer in order to install NodeJS (or any other package). This is a safer approach as it makes sure that any package you install with Homebrew can't wreak havoc on your computer.
2. After installing Node via the installer you have to add the path to the node executable to your system $PATH. This involves mucking around with your shell login file. For experienced Terminal users or Unix-people this isn't a big deal, but for those new to the command line that step can be daunting. Although the Homebrew method involves installing several different pieces of software, it's generally just a simple process of point-click-and-wait. It takes a bit longer this way but there's less room for error.
3. Homebrew is a great tool for web developers. First, it makes removing Node very easy (otherwise you have to crawl through your file system and delete a bunch of files manually). Second, it greatly simplfies the installation of other useful packages like Git, Ruby, or the very useful wget utility.

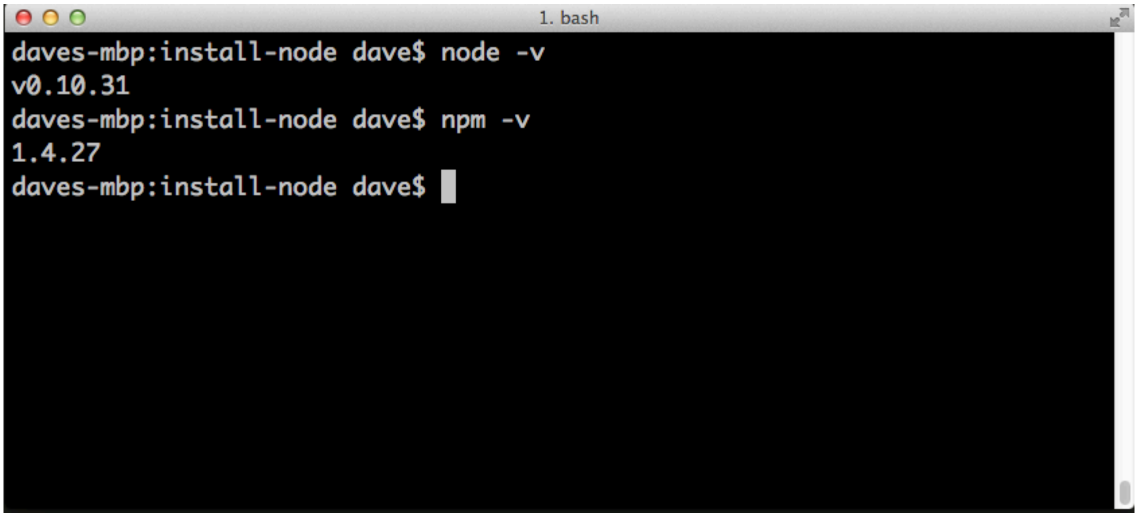
Installation

Installing Node.js and NPM is pretty straightforward using Homebrew. Homebrew handles downloading, unpacking and installing Node and NPM on your system. The whole process (after you have XCode and Homebrew installed) should only take you a few minutes.

1. **Open the Terminal app** and type `brew install node`.
2. **Sit back and wait.** Homebrew downloads some files and installs them. And that's it.

To make sure you have Node and NPM installed, run two simple commands to see what version of each is installed:

- To see if Node is installed, type `node -v` in Terminal. This should print the version number so you'll see something like this `v0.10.31`.
- To see if NPM is installed, type `npm -v` in Terminal. This should print the version number so you'll see something like this `1.4.27`



```
daves-mbp:install-node dave$ node -v
v0.10.31
daves-mbp:install-node dave$ npm -v
1.4.27
daves-mbp:install-node dave$ 
```

How to Update Node and NPM

New versions of Node and NPM come out frequently. You can use Homebrew to update the software it installs.

1. Make sure Homebrew has the latest version of the Node package. In Terminal type `brew update`
2. Upgrade Node: `brew upgrade node`

Quick start

This section explains how to handle Strapi for the first time, (check out our tutorial video).

Table of contents:

Create your first project

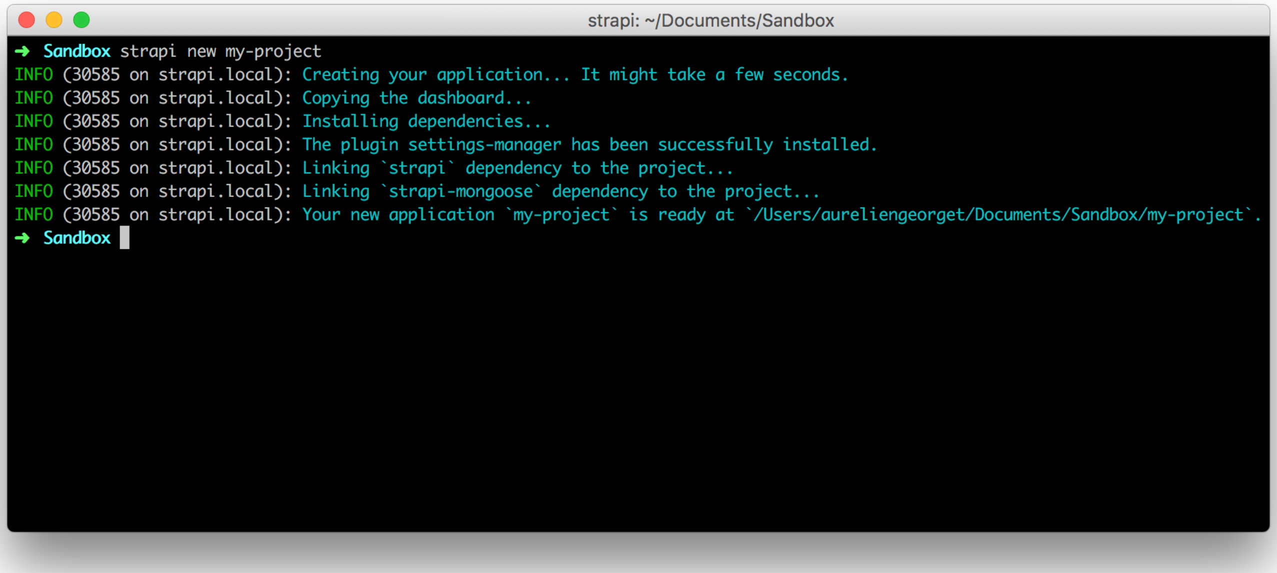
Creating your first project with Strapi is easy:

#1 — Open your terminal

Open your terminal in the directory you want to create your application in.

#2 — Run the following command line in your terminal:

```
strapi new my-project
```



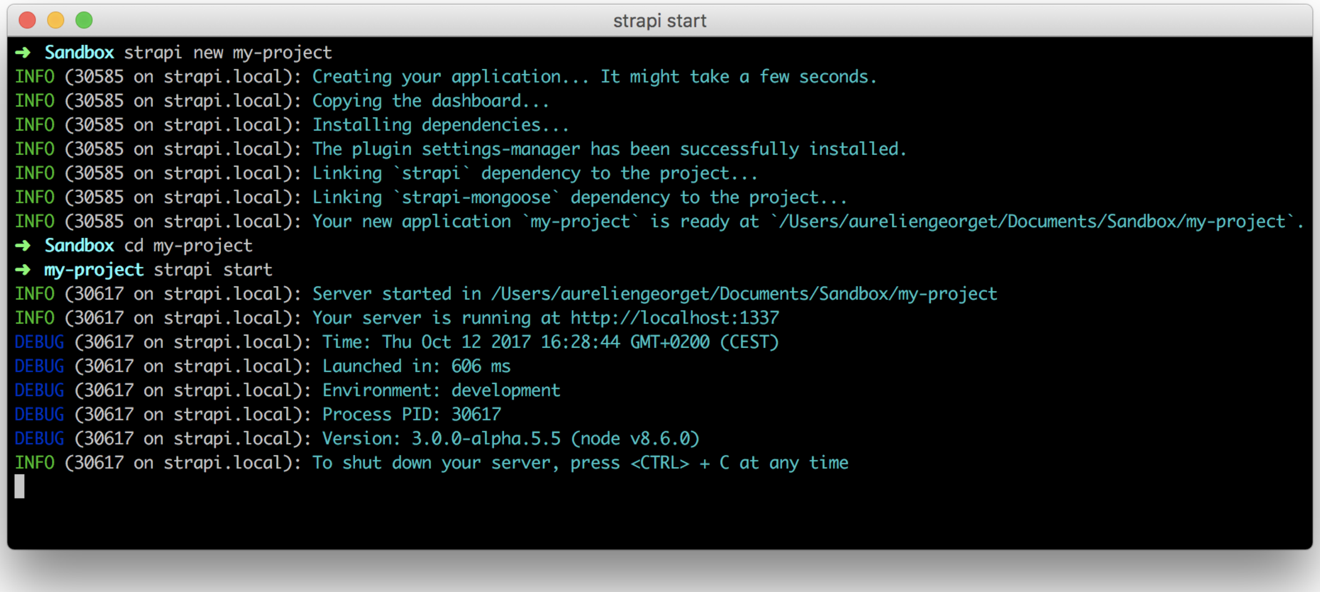
```
➜ Sandbox strapi new my-project
INFO (30585 on strapi.local): Creating your application... It might take a few seconds.
INFO (30585 on strapi.local): Copying the dashboard...
INFO (30585 on strapi.local): Installing dependencies...
INFO (30585 on strapi.local): The plugin settings-manager has been successfully installed.
INFO (30585 on strapi.local): Linking `strapi` dependency to the project...
INFO (30585 on strapi.local): Linking `strapi-mongoose` dependency to the project...
INFO (30585 on strapi.local): Your new application `my-project` is ready at `/Users/aureliengeorget/Documents/Sandbox/my-project`.
➜ Sandbox
```

This action creates a new folder named `my-project` with the entire files structure of a Strapi application.

#3 — Go to your project and launch the server:

In your terminal run the following commands:

```
cd my-project
strapi start
```



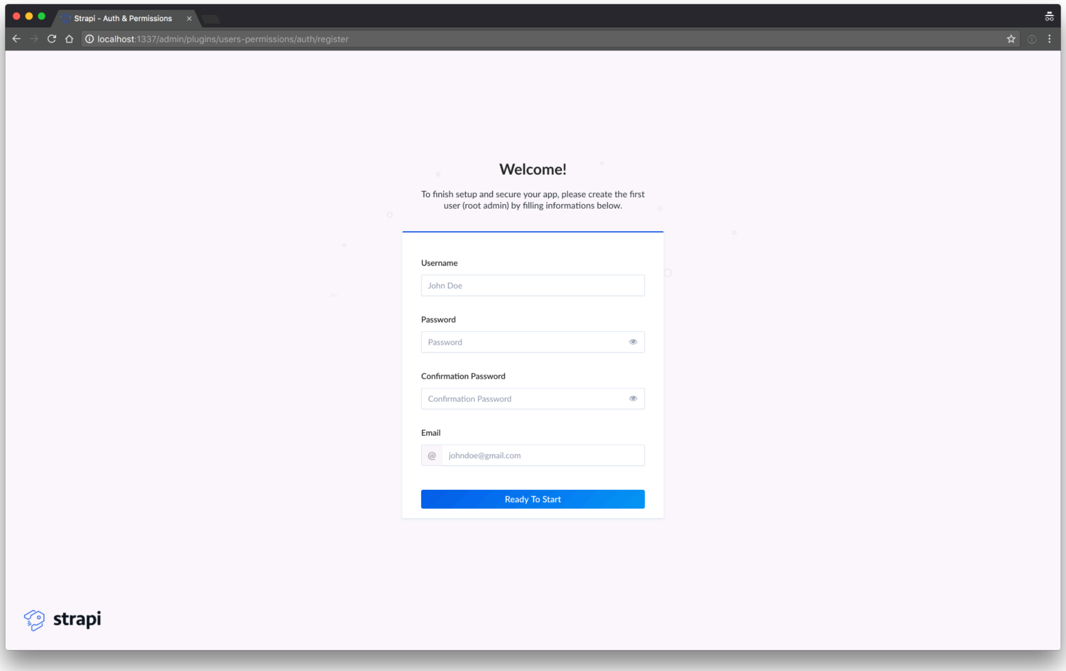
```
➜ Sandbox strapi new my-project
INFO (30585 on strapi.local): Creating your application... It might take a few seconds.
INFO (30585 on strapi.local): Copying the dashboard...
INFO (30585 on strapi.local): Installing dependencies...
INFO (30585 on strapi.local): The plugin settings-manager has been successfully installed.
INFO (30585 on strapi.local): Linking `strapi` dependency to the project...
INFO (30585 on strapi.local): Linking `strapi-mongoose` dependency to the project...
INFO (30585 on strapi.local): Your new application `my-project` is ready at `/Users/aureliengeorget/Documents/Sandbox/my-project`.
➜ Sandbox cd my-project
➜ my-project strapi start
INFO (30617 on strapi.local): Server started in /Users/aureliengeorget/Documents/Sandbox/my-project
INFO (30617 on strapi.local): Your server is running at http://localhost:1337
DEBUG (30617 on strapi.local): Time: Thu Oct 12 2017 16:28:44 GMT+0200 (CEST)
DEBUG (30617 on strapi.local): Launched in: 606 ms
DEBUG (30617 on strapi.local): Environment: development
DEBUG (30617 on strapi.local): Process PID: 30617
DEBUG (30617 on strapi.local): Version: 3.0.0-alpha.5.5 (node v8.6.0)
INFO (30617 on strapi.local): To shut down your server, press <CTRL> + C at any time
```

Now that your app is running let's see how to create your first user.

Create your first user

In order to use the admin panel and to consume your API you first need to register your first user. This process only happens once if you don't have any user table created and is made to create the `admin user` that has all the permissions granted for your API.
If your using MongoDB for your database you don't have to create your table manually (it's already handled by Strapi) otherwise you'll have to create your user table first.

To create your first user, start your server (`strapi start`) and go to : http://localhost:1337/admin.


Strapi - Auth & Permissions
localhost:1337/admin/plugins/users-permissions/auth/register
strapi

Welcome!

To finish setup and secure your app, please create the first user (root admin) by filling informations below.

Username

John Doe

Password

Password

Confirmation Password

Confirmation Password

Email

@ johndoe@gmail.com

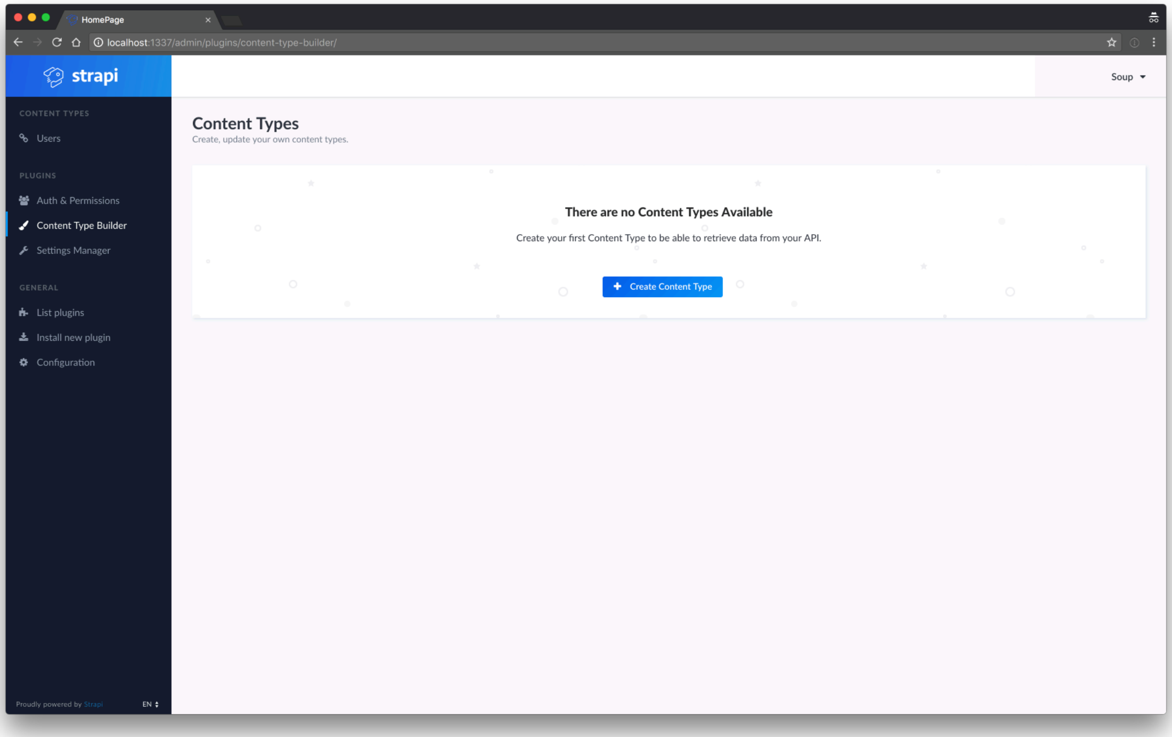
Ready To Start

Now that your first user is registered let's see how to create your first api.

Create your first API

To create your first API, start your server (`strapi start`) and go to : http://localhost:1337/admin.
At this point, your application is empty. To create your first API is to use the **Content Type Builder** plugin: a powerful UI to help you create an API in a few clicks. Let's take the example of an e-commerce API, which manages products.

#1 — Go to the **Content Type Builder** plugin.



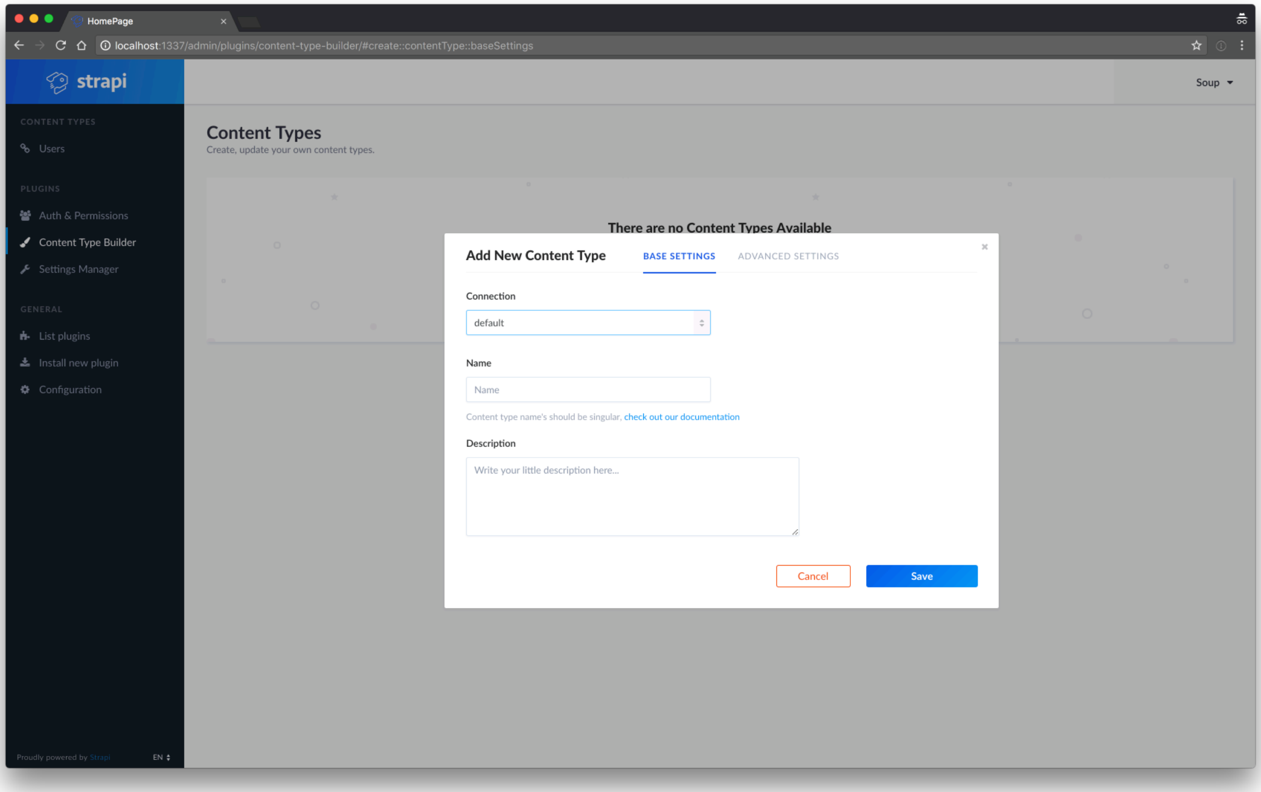
Content Types

Create, update your own content types.

There are no Content Types Available

Create your first Content Type to be able to retrieve data from your API.

#2 — Create a Content Type name `Product` and submit the form.



Content Types

Create, update your own content types.

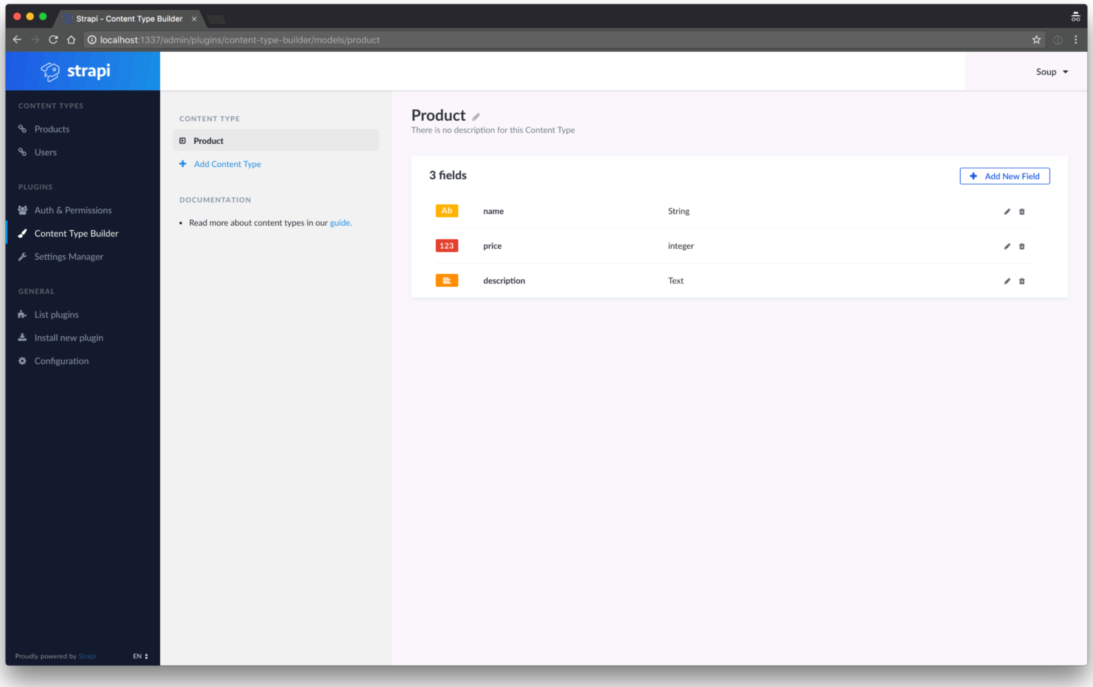
There are no Content Types Available

Connection: default

Content type name's should be singular, check out our documentation

#3 — Add three fields in this Content Type.

- A `string` field named `name`.
- A `text` field named `description`.
- A `number` field named `price` (with `float` as number format).



- Read more about content types in our guide.

Product

There is no description for this Content Type

3 fields

Ab	name	String
123	price	integer
≡	description	Text

#4 — Save. That's it!

Note: See the CLI documentation for informations about how to do it the hacker way.

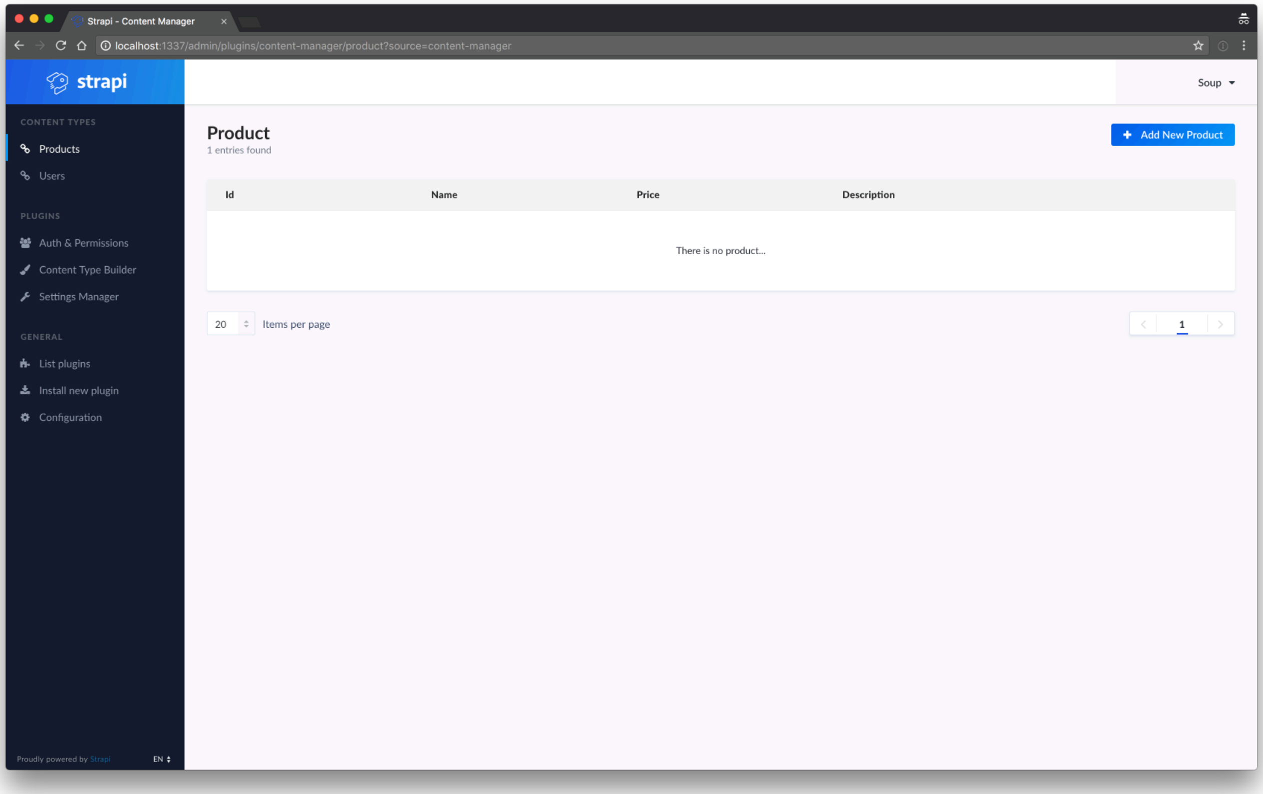
Files structure

A new directory has been created in the `./api` folder of your application which contains all the needed stuff for your `Product` Content Type: routes, controllers, services and models. Take a look at the API structure documentation for more informations.
Well done, you created your first API using Strapi!

Manage your data

After creating your first Content Type, it would be great to be able to create, edit or delete entries.

#1 — Go to the **Product list** by clicking on the link in the left menu (generated by the **Content Manager** plugin).

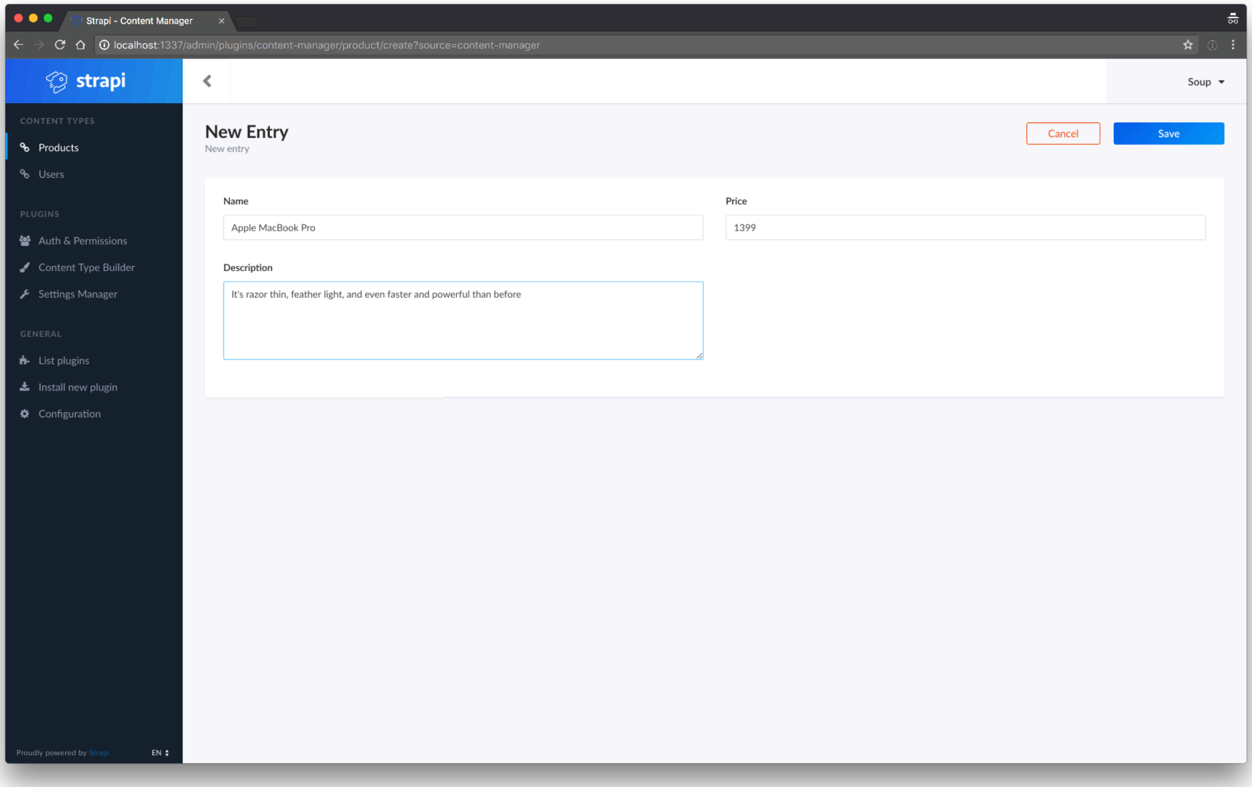


Product
1 entries found

Id	Name	Price	Description

There is no product...

#2 — Click on the button `Add New Product` and fill the form.



localhost:1337/admin/plugins/content-manager/product/create?source=content-manager

New Entry

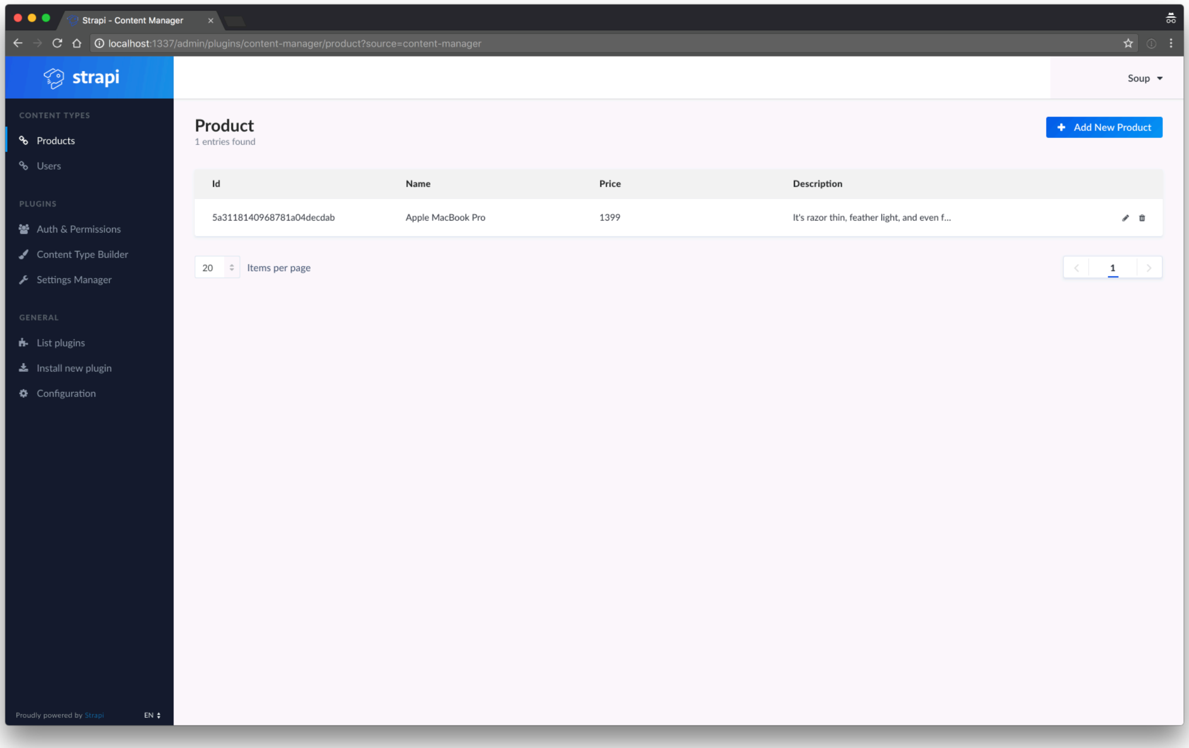
New entry

Name: Apple MacBook Pro

Price: 1399

Description: It's razor thin, feather light, and even faster and powerful than before

#3 — Save! You can edit or delete this entry by clicking on the icons at the right of the row.



Product

1 entries found

Id	Name	Price	Description
5a3118140968781a04decdab	Apple MacBook Pro	1399	It's razor thin, feather light, and even f...

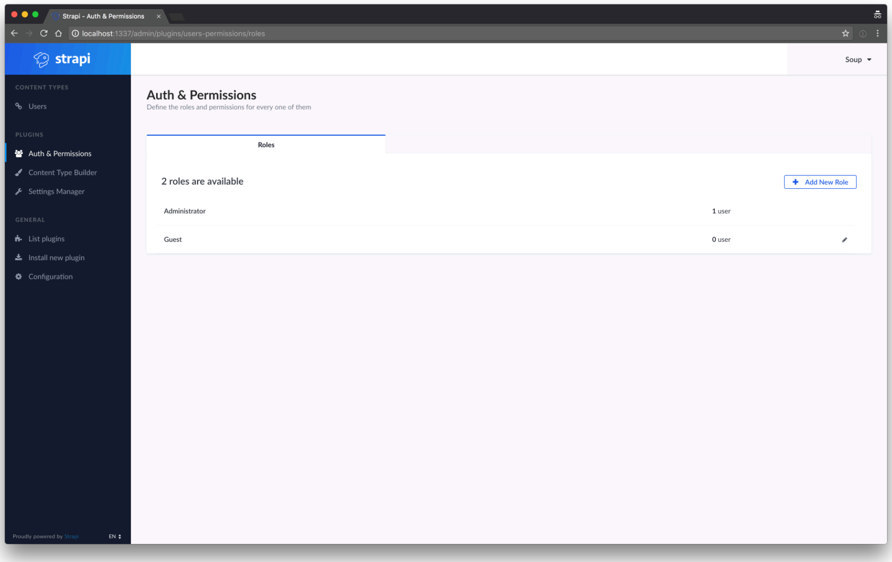
20 Items per page

1

Consume your API

Your API is now ready and contains data. At this point, you'll probably want to use this data in mobile or desktop applications. In order to do so, you'll need to allow access to other users (identified as 'Guest').

1 - Go to the **Auth & Permissions View** by clicking on **Auth & Permissions** link in the left menu and click on the **Guest Role** item.



Auth & Permissions

Define the roles and permissions for every one of them

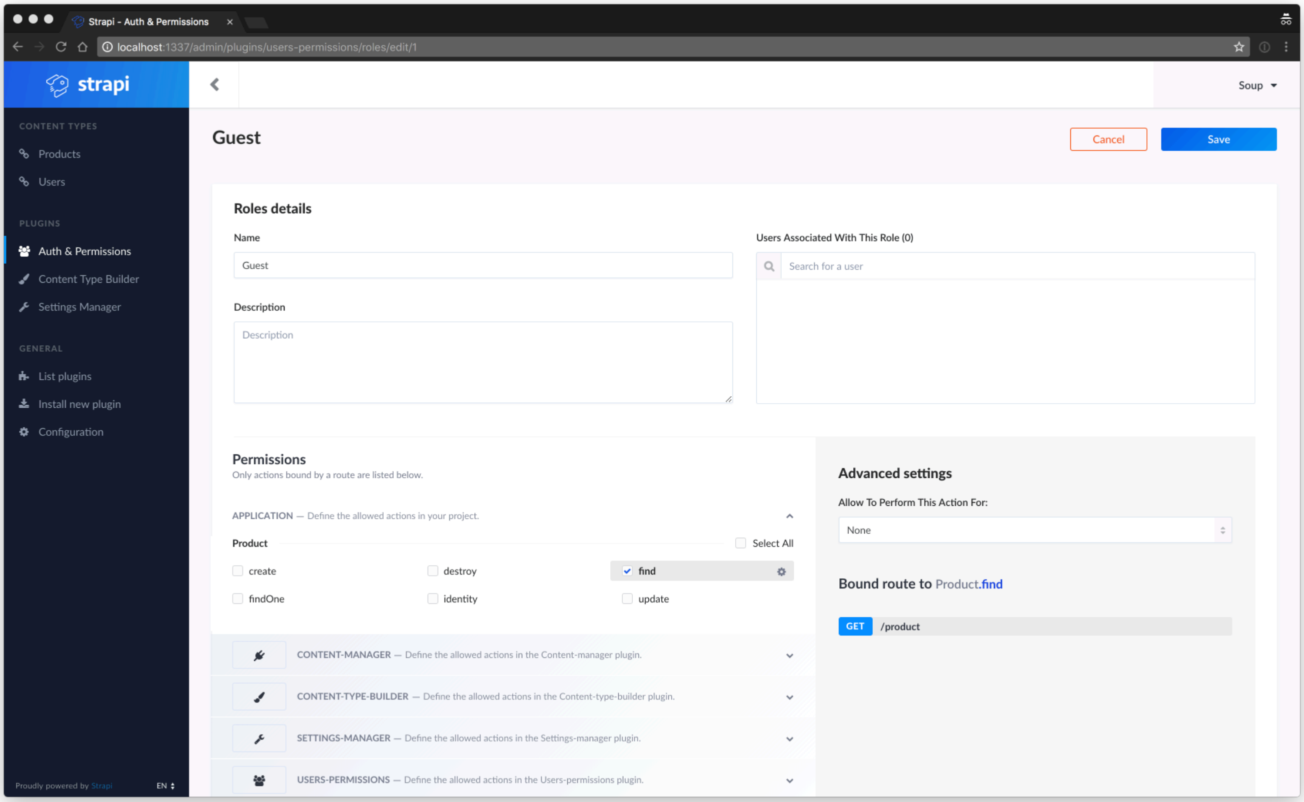
Roles

2 roles are available

Add New Role

Administrator	1 user
Guest	0 user

2 - Manage your APIs permissions in the **Permissions** section of the **Edit Guest Role view** by enabling or disabling specific actions.



Guest

Roles details

Name: Guest

Description: Description

Users Associated With This Role (0)

Search for a user

Permissions

Only actions bound by a route are listed below.

APPLICATION — Define the allowed actions in your project.

Product □ Select All

□ create □ destroy ☑ find

□ findOne □ identity □ update

CONTENT-MANAGER — Define the allowed actions in the Content-manager plugin.

CONTENT-TYPE-BUILDER — Define the allowed actions in the Content-type-builder plugin.

SETTINGS-MANAGER — Define the allowed actions in the Settings-manager plugin.

USERS-PERMISSIONS — Define the allowed actions in the Users-permissions plugin.

Advanced settings

Allow To Perform This Action For: None

Bound route to Product.find

GET /product

List entries (GET)

To retrieve the list of products, use the `GET /your-content-type` route.
Generated APIs provide a handy way to filter and order queries. In that way, ordering products by price is as easy as `GET http://localhost:1337/product?_sort=price:asc`. For more informations, read the filters documentation
Here is an example using jQuery.

```
$.ajax({
  type: 'GET',
  url: 'http://localhost:1337/product?_sort=price:asc', // Order by price.
  done: function(products) {
    console.log('Well done, here is the list of products: ', products);
  },
  fail: function(error) {
    console.log('An error occurred:', error);
  }
});
```

Get a specific entry (GET)

If you want to get a specific entry, add the `id` of the wanted product at the end of the url.

```
$.ajax({
  type: 'GET',
  url: 'http://localhost:1337/product/123', // Where `123` is the `id` of the product.
  done: function(product) {
    console.log('Well done, here is the product having the `id` 123: ', product);
  },
  fail: function(error) {
    console.log('An error occurred:', error);
  }
});
```

Create data (POST)

Use the `POST` route to create a new entry.
jQuery example:

```
$.ajax({
  type: 'POST',
  url: 'http://localhost:1337/product',
  data: {
    name: 'Cheese cake',
    description: 'Chocolate cheese cake with ice cream',
    price: 5
  },
  done: function(product) {
    console.log('Congrats, your product has been successfully created: ', product); //
Remember the product `id` for the next steps.
  },
  fail: function(error) {
    console.log('An error occurred:', error);
  }
});
```

Update data (PUT)

Use the `PUT` route to update an existing entry.
jQuery example:

```
$.ajax({
  type: 'PUT',
  url: 'http://localhost:1337/product/123', // Where `123` is the `id` of the product.
  data: {
    description: 'This is the new description'
  },
  done: function(product) {
    console.log('Congrats, your product has been successfully updated: ',
product.description);
  },
  fail: function(error) {
    console.log('An error occurred:', error);
  }
});
```

Delete data (DELETE)

Use the `DELETE` route to delete an existing entry.
jQuery example:

```
$.ajax({
  type: 'DELETE',
  url: 'http://localhost:1337/product/123', // Where `123` is the `id` of the product.
  done: function(product) {
    console.log('Congrats, your product has been successfully deleted: ', product);
  },
  fail: function(error) {
    console.log('An error occurred:', error);
  }
});
```

Congratulations! You successfully finished the Getting Started guide! Read the concepts to understand more advanced concepts.

Also, feel free to join the community thanks to the different channels listed in the community page: team members, contributors and developers will be happy to help you.

Before you go any further, there are some **requirements** to make sure the installation works. **Ensure these technologies are installed** on your computer.

- node

  at least v9.0.0I don't have Node.js

- npm

  at least v5.0.0I don't have npm

- mongoDB

  at least v3.4I don't have MongoDB

- Installation

  Open your terminal and run the following command.

  ```
  npm install strapi@alpha -g
  ```

  If you encounter npm permissions issues, change the permissions to npm default directory.

- Create your first project

  Use the built-in CLI to generate your first project. Run this command and wait until the project is created.

  ```
  strapi new myProject
  ```

  Then, the CLI will ask you to choose your database. We highly recommend to use MongoDB (better support)!

  ```
  Choose your main database:
  > MongoDB (highly recommended)
  > Postgres
  > MySQL
  ```

  ⚠️ Make sure MongoDB is running on your machine. Otherwise, the project creation will fail...

  Fill the database information. The default values should work if you have just installed MongoDB.

  ```
  Choose your main database: MongoDB (highly recommended)
  Database name: strapi
  Host: 127.0.0.1
  Port: 27017
  Username:
  Password:
  Authentication database:
  Enable SSL connection: false
  ```

  Next, enter in your project.

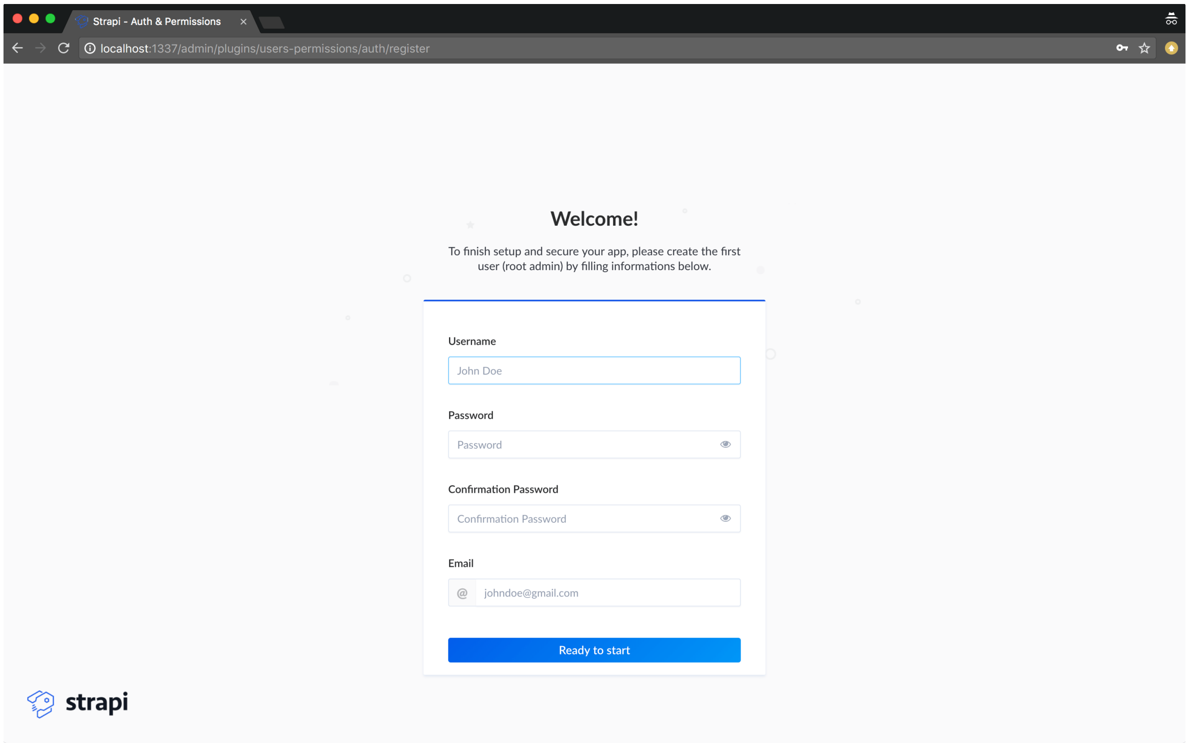
  ```
  cd myProject
  ```

- Launch the server

  Now, you have to use another command of the CLI to run the server. And you will be done!

  ```
  strapi start
  ```

Open your browser and **go to the administration panel** (http://localhost:1337/admin) to create the first user.



Strapi - Auth & Permissions
localhost:1337/admin/plugins/users-permissions/auth/register

Welcome!

To finish setup and secure your app, please create the first user (root admin) by filling informations below.

Username	John Doe
Password	Password
Confirmation Password	Confirmation Password
Email	@ johndoe@gmail.com

👏

Congratulations, you've finished the "Getting Started" tutorial.
Follow the instructions in the administration panel to get deeper into Strapi.