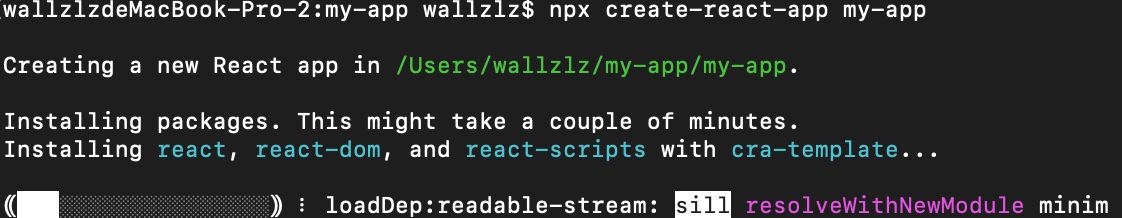
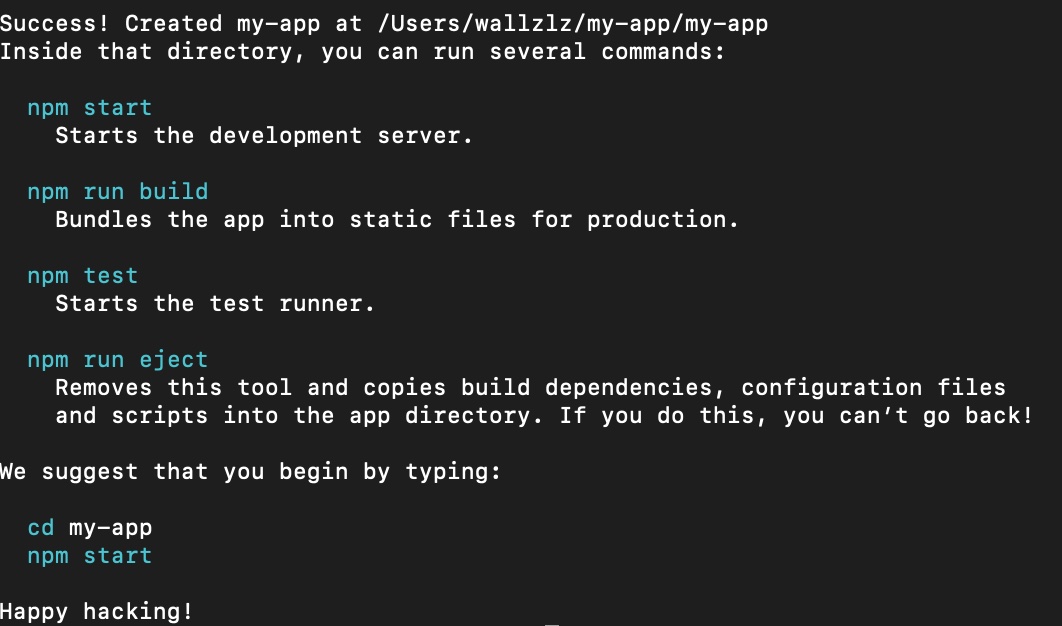
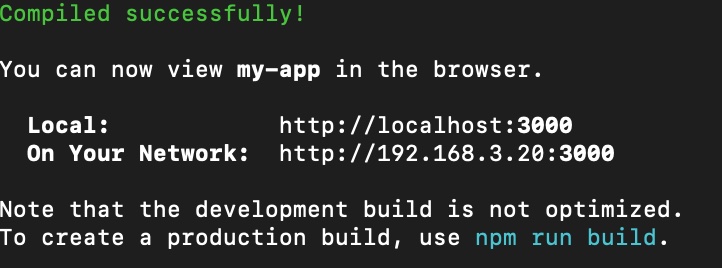
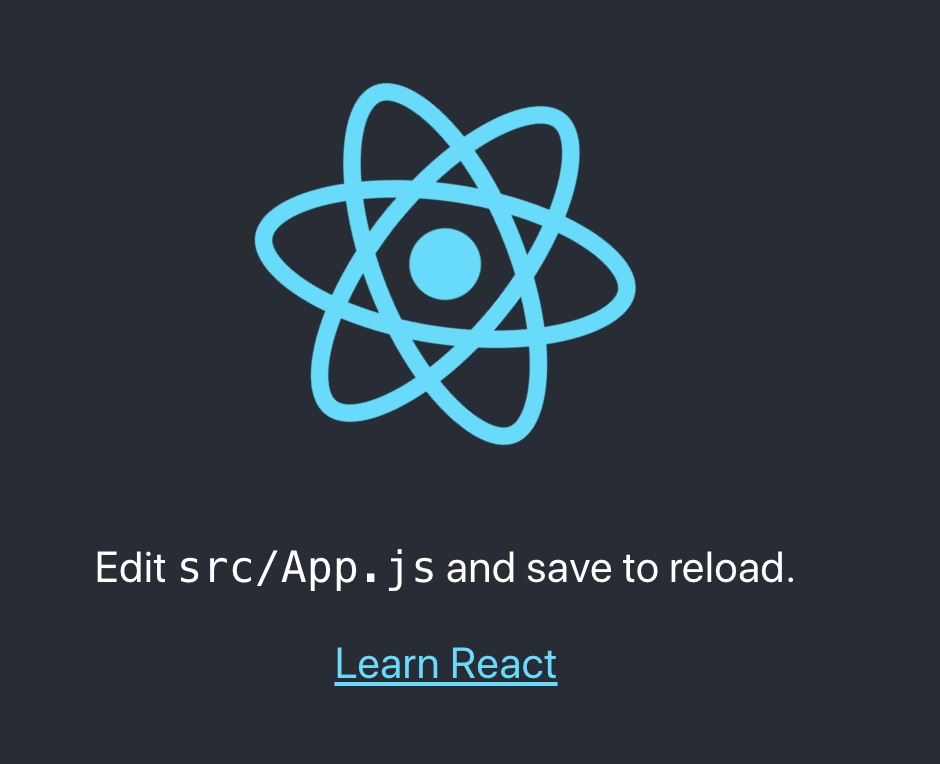
第一次练习

首先按照教程在终端用npm create-react-app my-app创建了文件夹，再用cd my-app打开文件夹，最后npm start弹出网页。

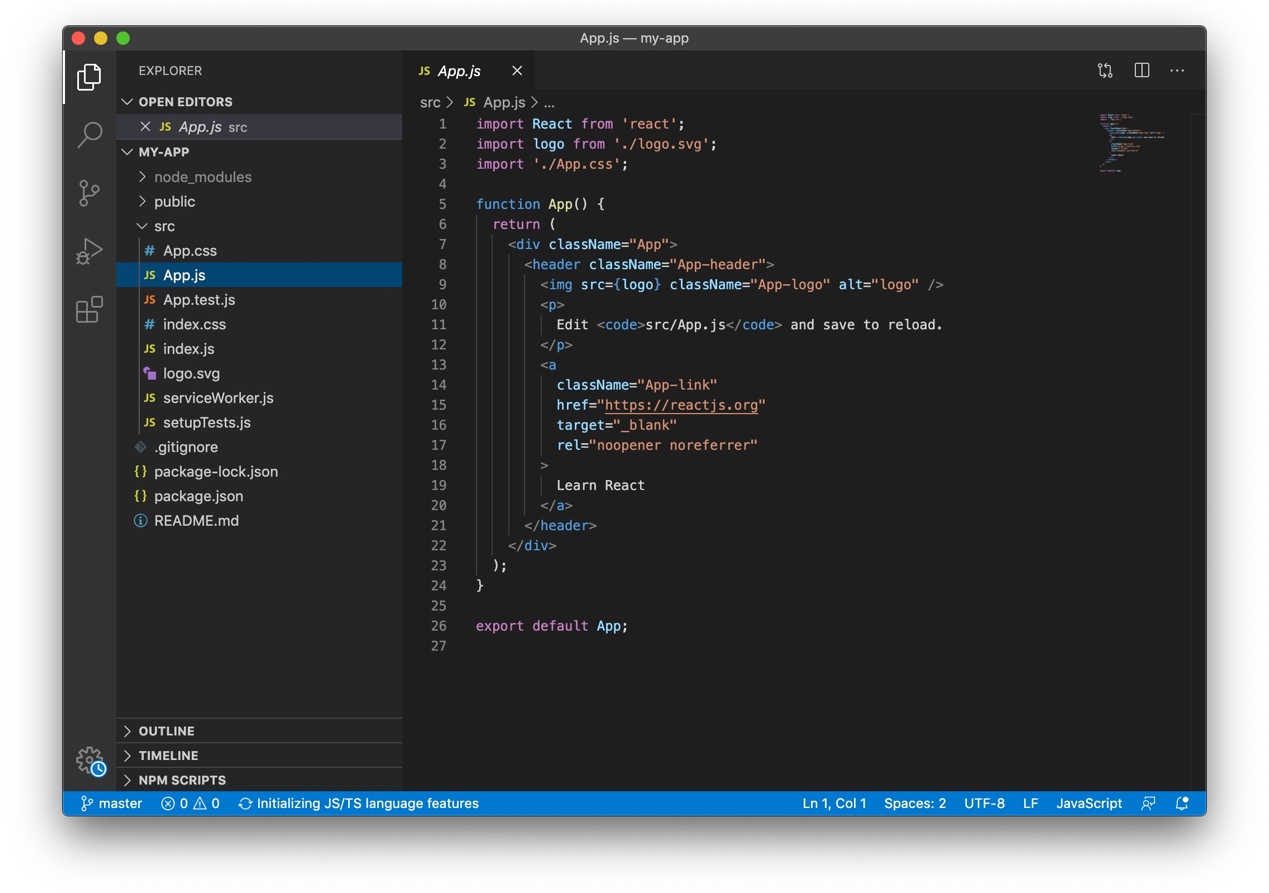




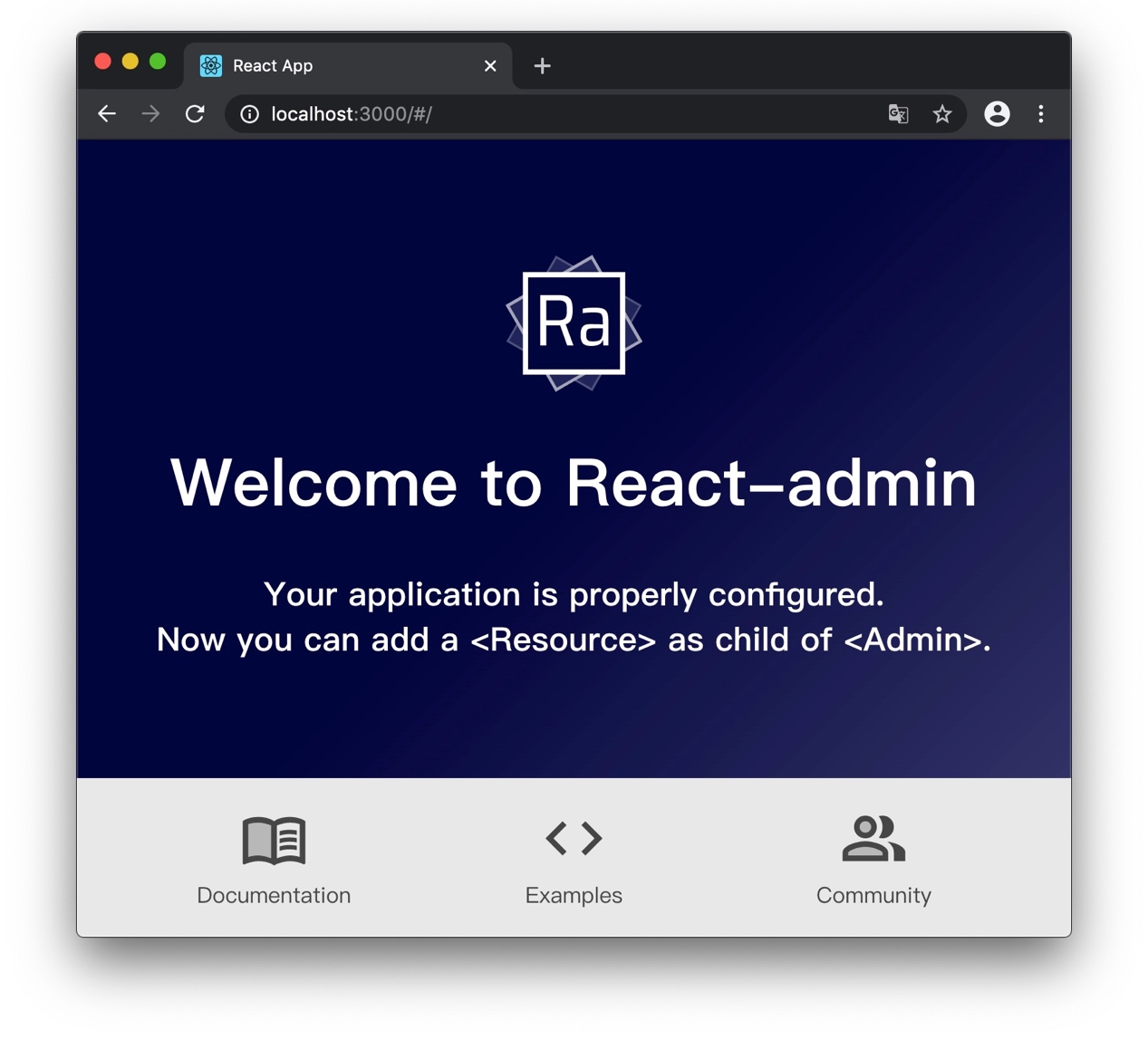
自动弹出网页的画面：



用visual studio code打开my-app的文件夹，文件夹下的文件如下：



然后根据教程中的代码修改src文件中的App.js，进入用户界面，结果如下：

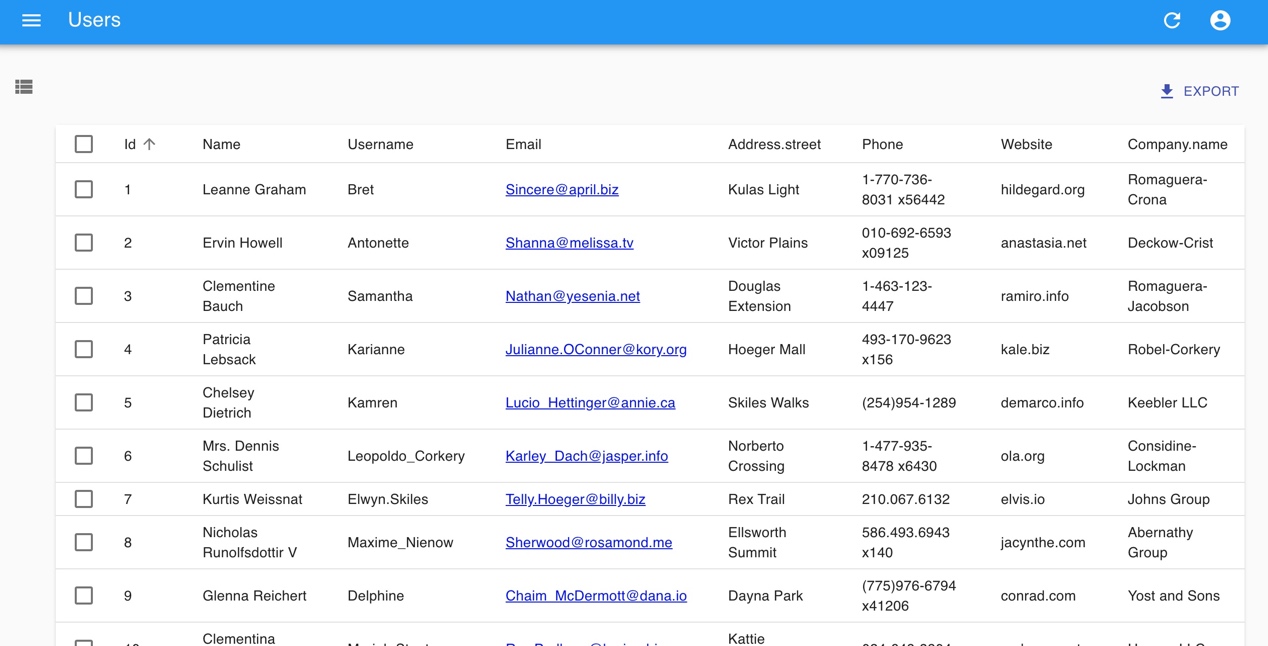


下面开始添加网页的内容。

1. 将API端点映射到资源

组件Admin需要一个或多个子组件Resource，每个Resource都将一个name映射到API中的一个端点。修改App.js以增加一个名为users的Resource。

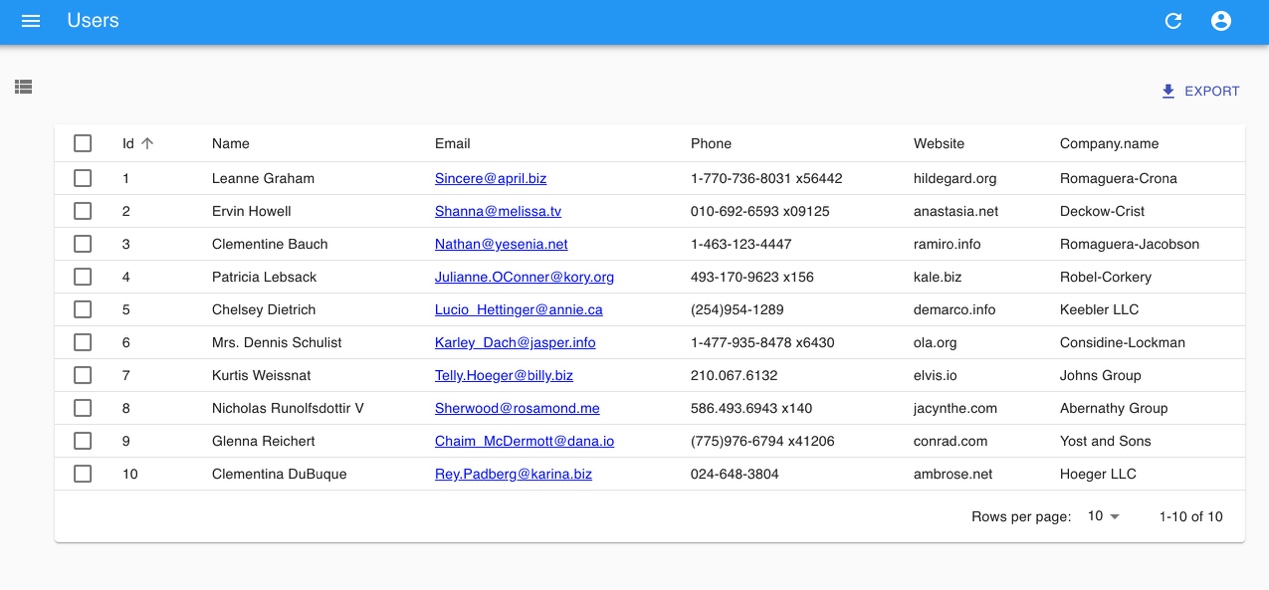
当查看浏览器中开发工具里的network选项，可以发现应用程序获取了<https://jsonplaceholder.typicode.com/users>中的URL，然后使用该结果来构建Datagrid。



该列表已经具有以下的功能：可以通过单击列标题重新排序，也可以通过使用底部的控件更改页面。

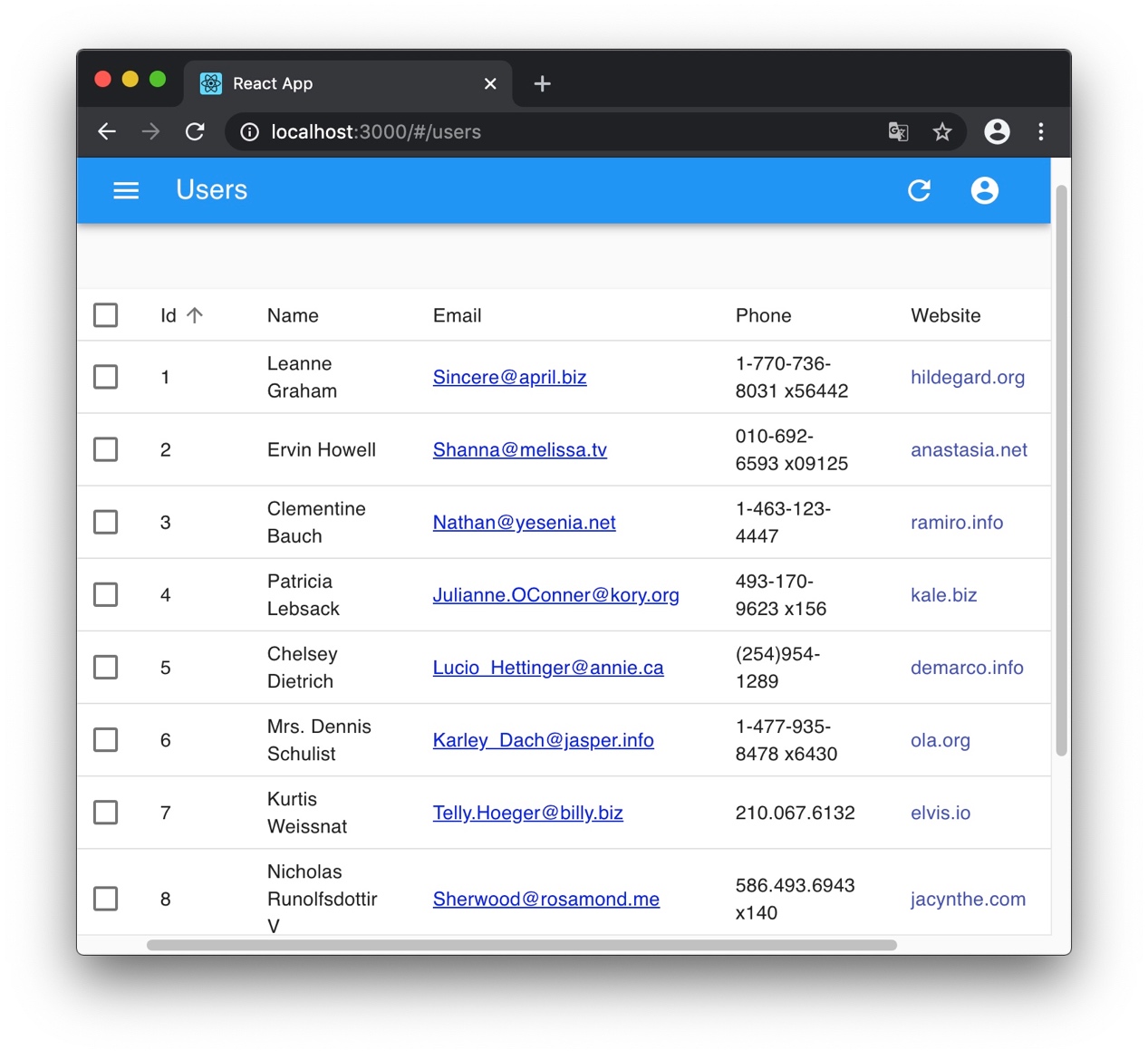
1. 选择列

ListGuesser不适合在生产中使用，它只是一种快速引导管理员的方法。因此，需要使用自定义React组件替换用户资源中的ListGuesser组件。

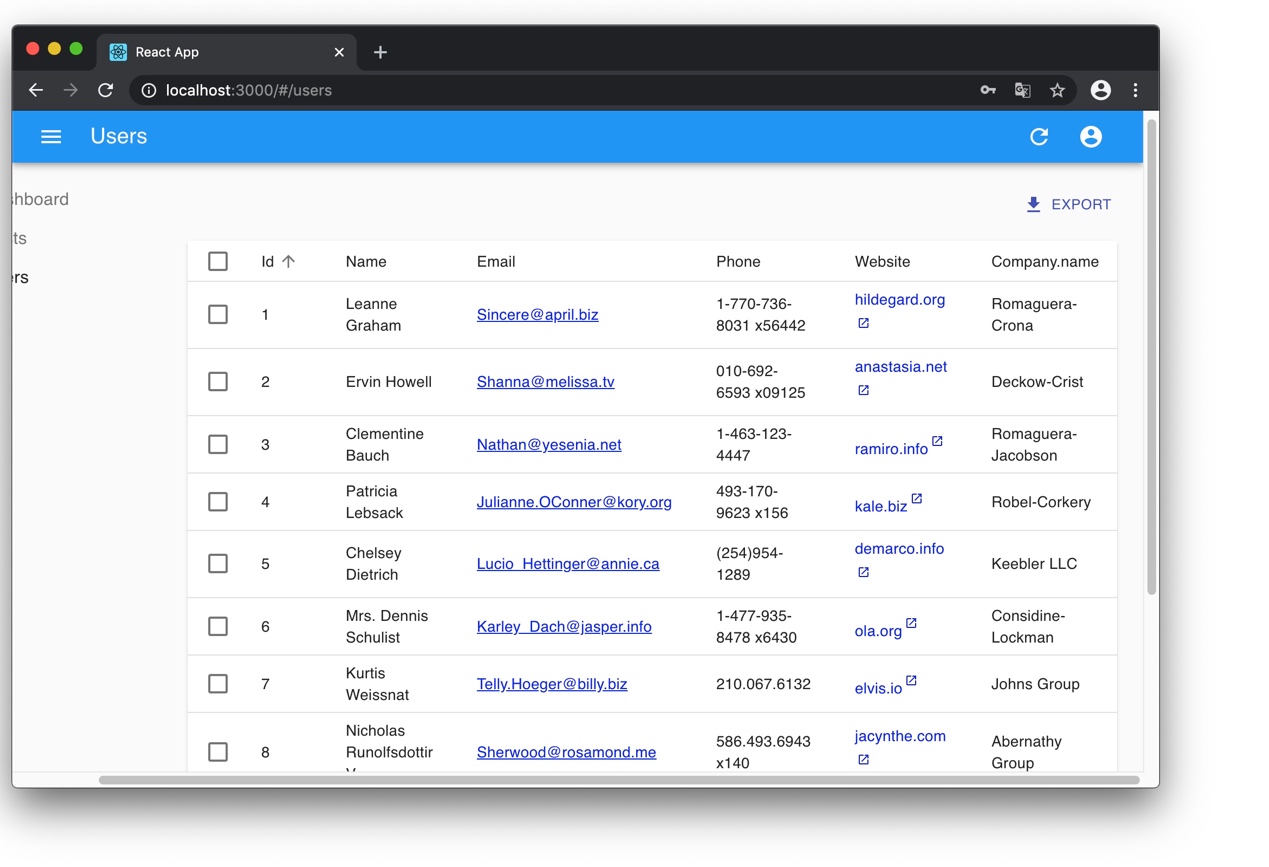
创建新文件users.js，然后修改App.js，用新组件去替代ListGuesser。

1. 使用fields类型

在React -admin中，fields是简单的React组件。在运行时，它们接收从API获取的记录。

我们可以用自己的组件替换任何react-admin组件，也就是说，如果某个react-admin组件不能完全满足我们的需要，我们可以轻松地将它与自己的版本进行交换。

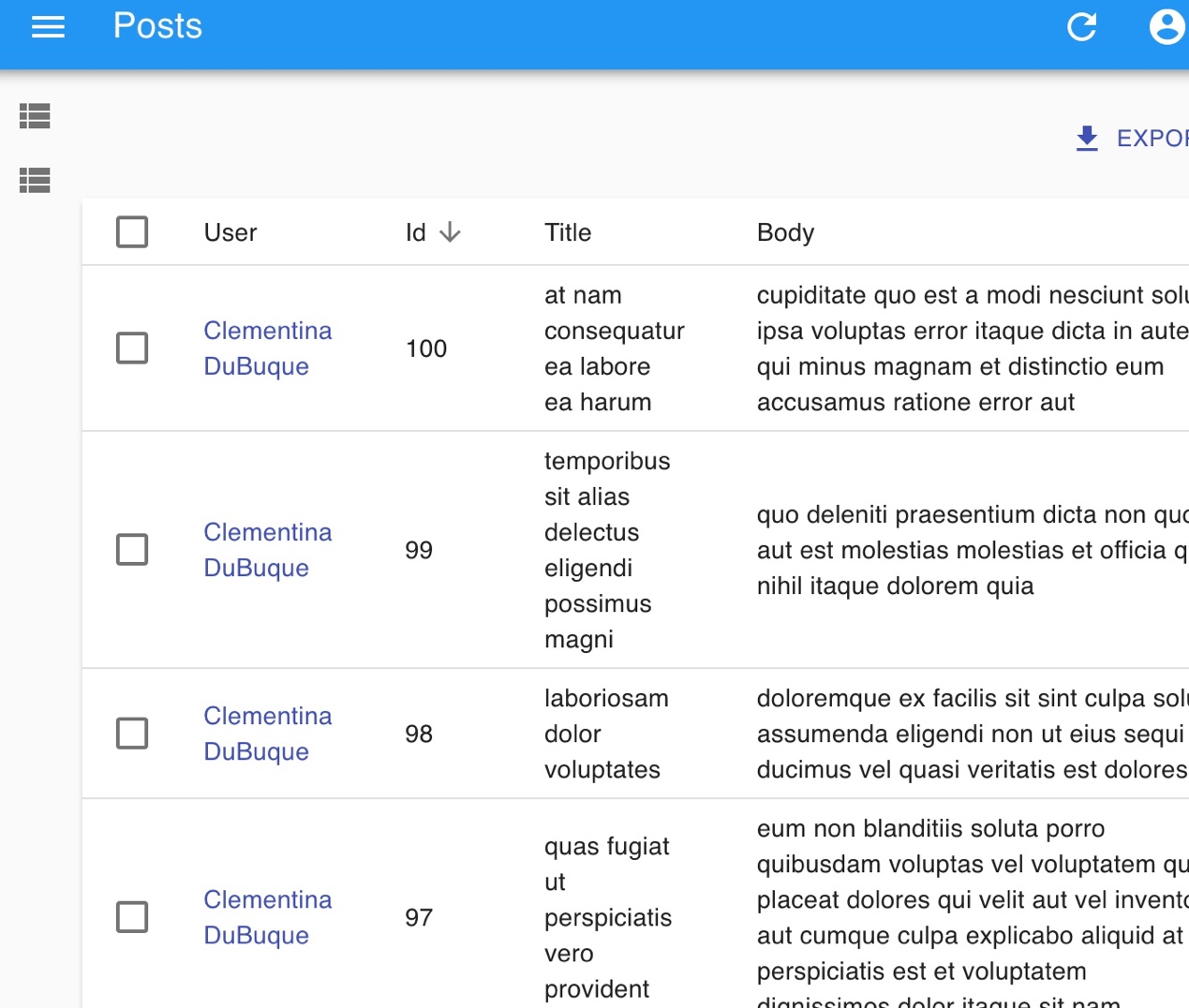
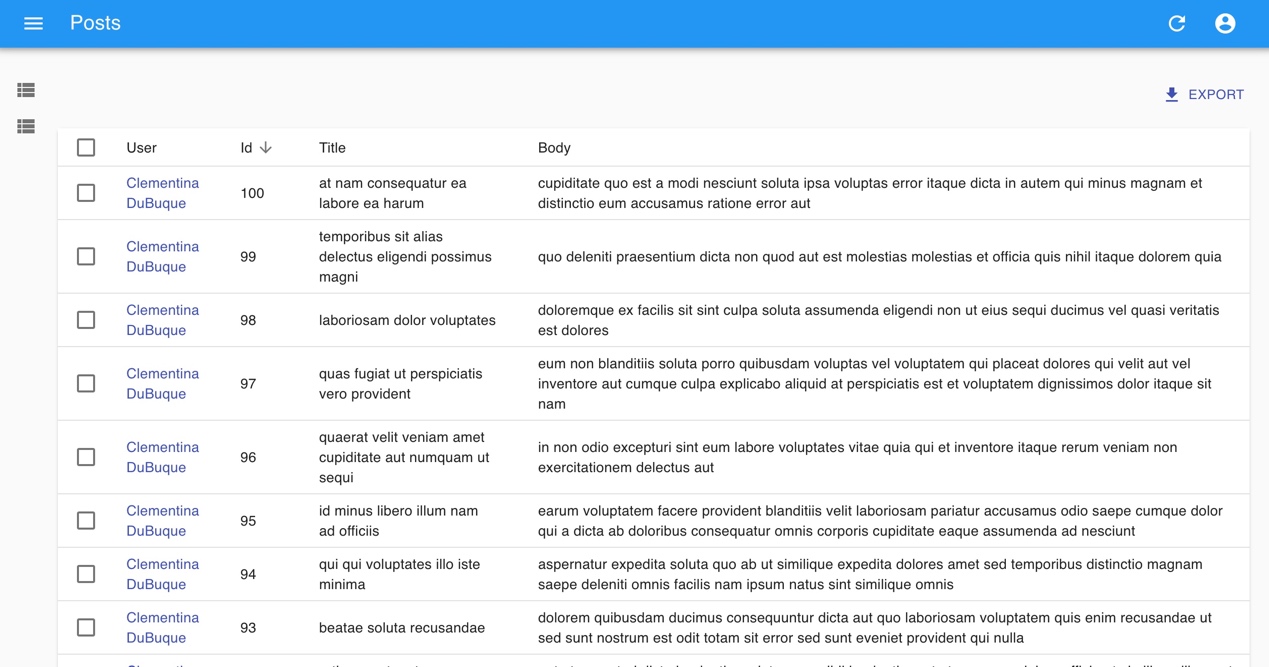
1. 自定义风格

在JSS中，要将使用CSS属性名JS变量的样式传递给组件，可以使用makeStyles来构建一个React钩子。钩子将为这些样式创建新的类名，并在classes对象中返回新的类名。然后，就可以在className中使用这些名称。

1. 处理关系

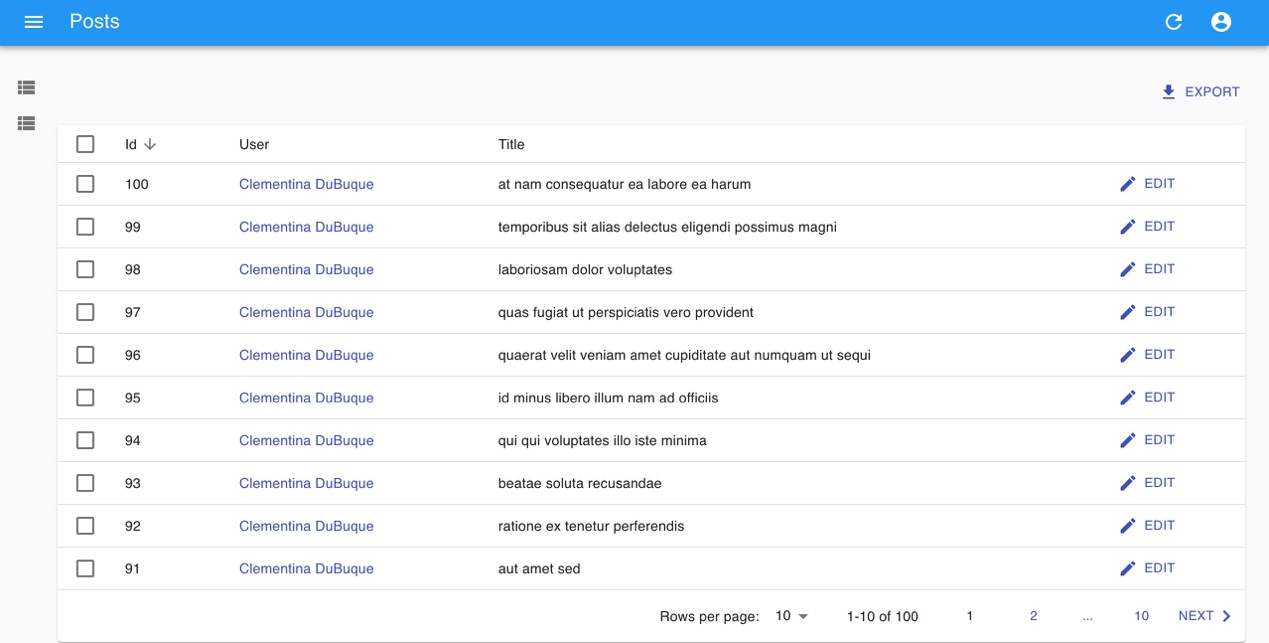
在JSONPlaceholder中，每个post都包含一个userId字段，它指向一个用户。

post列表显示每行上的用户名。



1. 添加创建和编辑功能

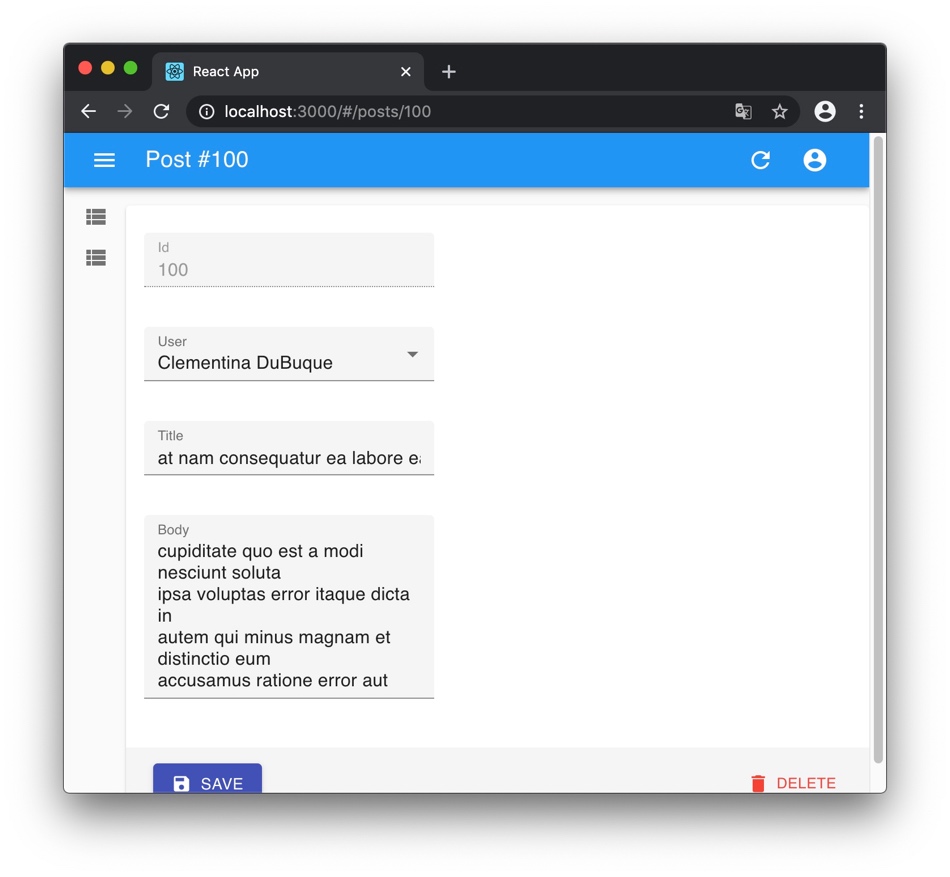
管理界面不只是显示远程数据，它还应该允许编辑记录。



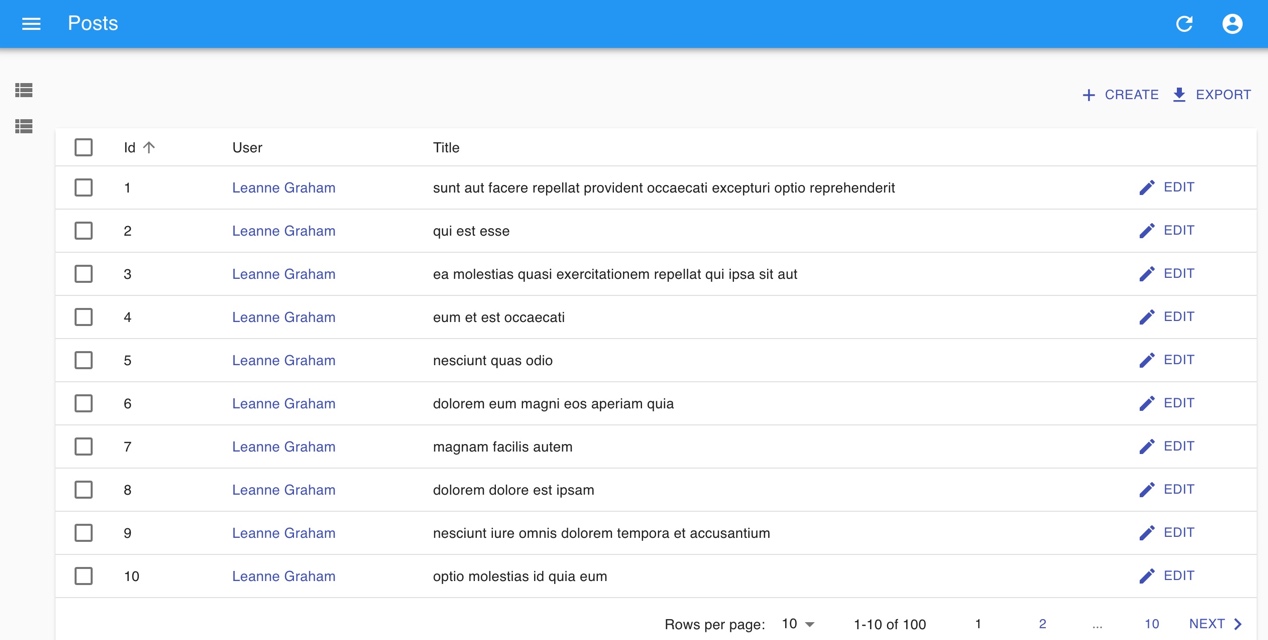
用户只需单击edit按钮就可以显示编辑页面。

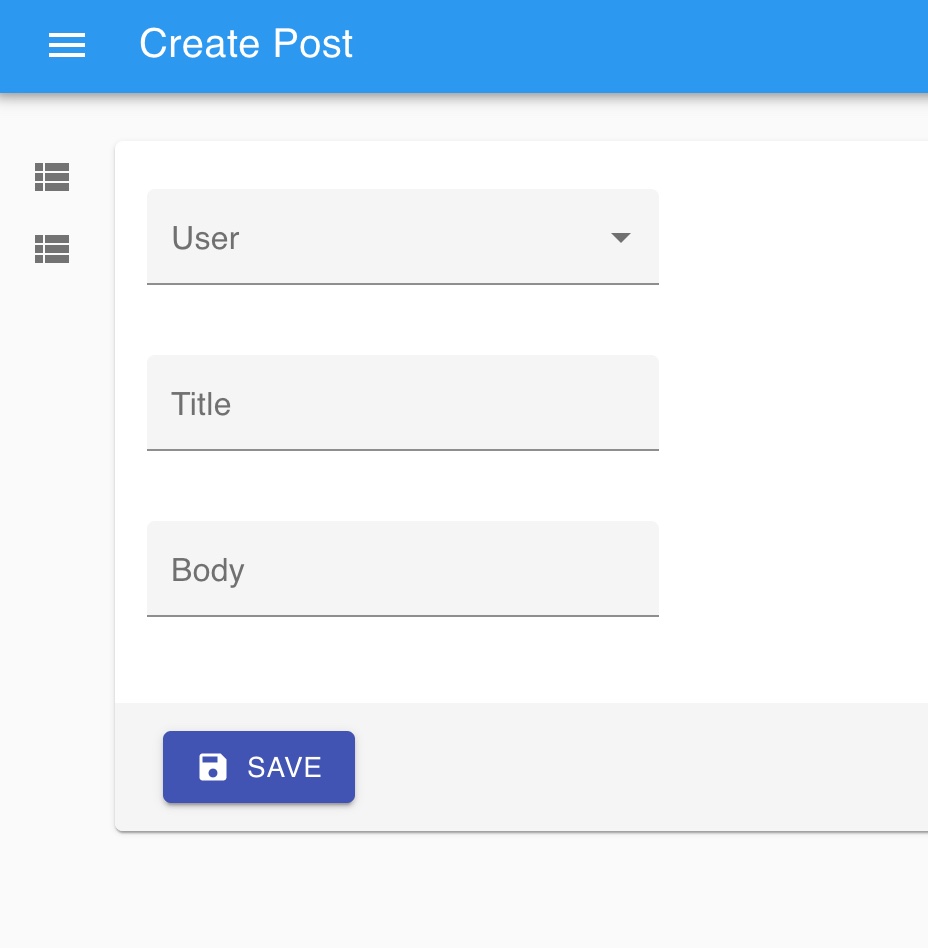


现在，调整PostEdit组件来禁止修改id，再将用户id修改为用户名，并使body能够输入更长的文本。



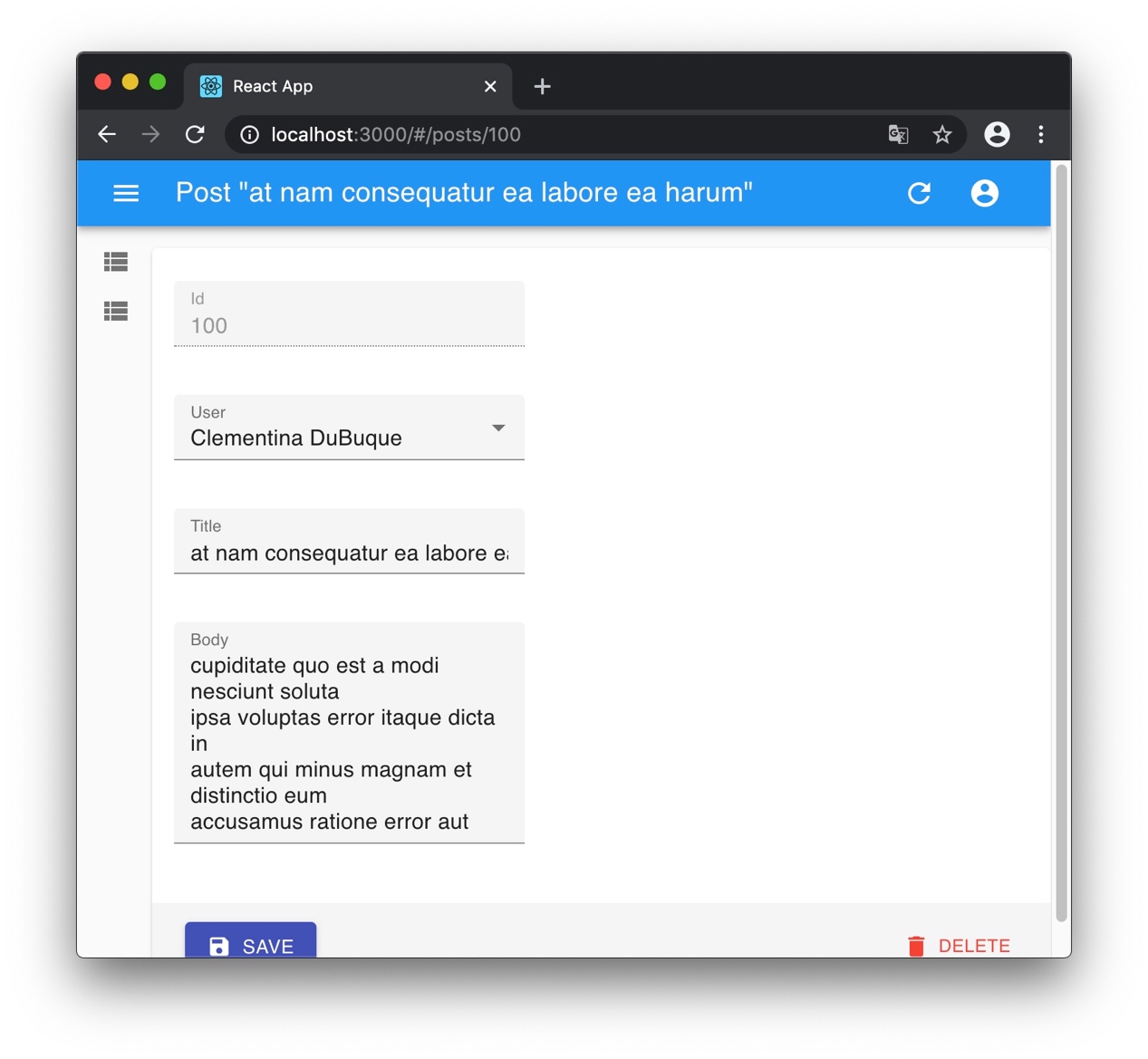
增加创建功能，如右上角create所示：



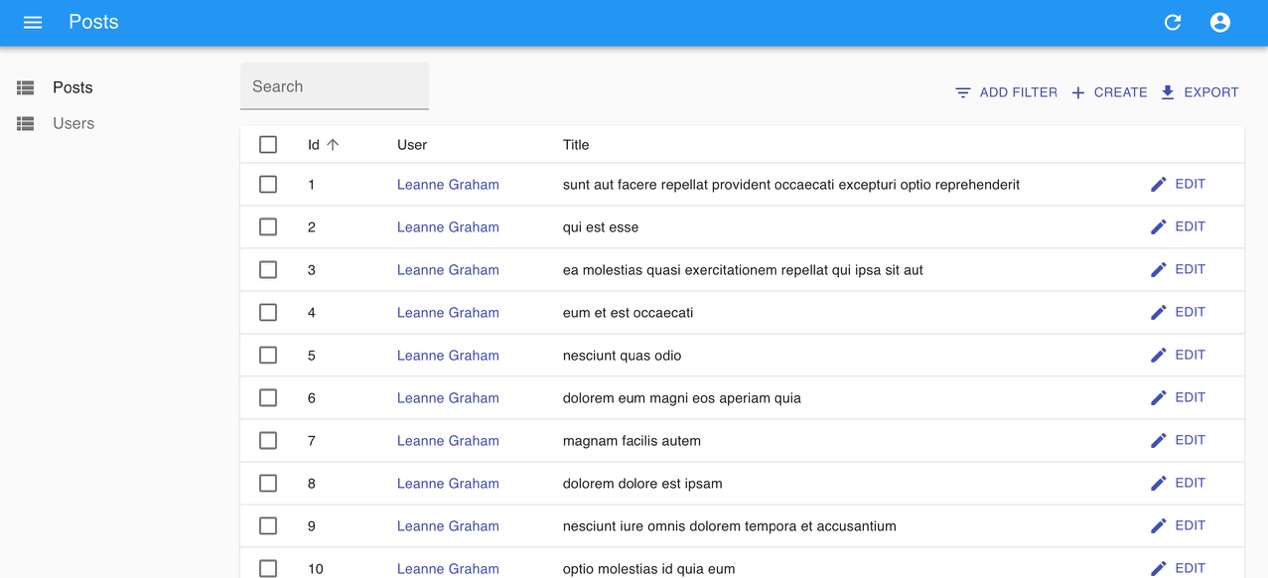
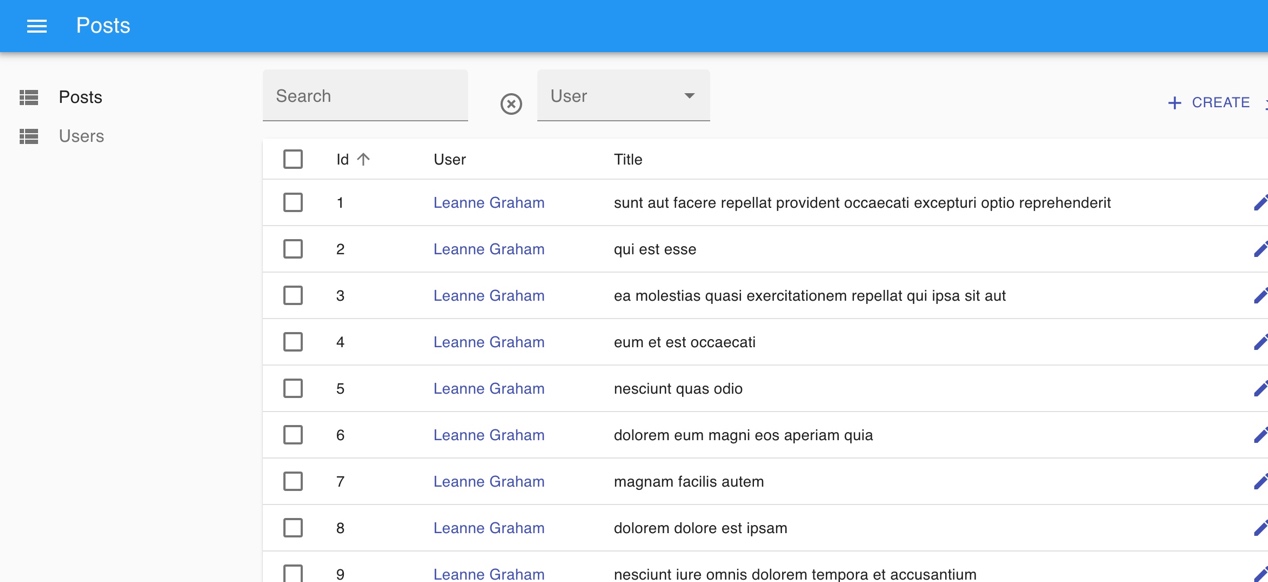


1. 动态显示与撤销

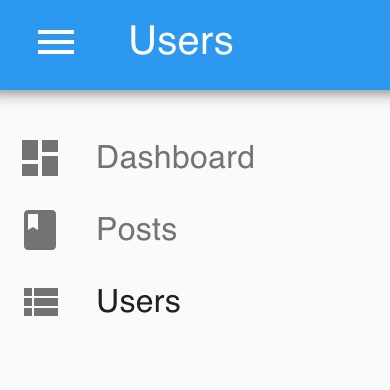
post编辑页面有一个小问题:它使用post id作为主标题(显示在顶部栏中的文本)，可以用标题组件自定义视图标题:



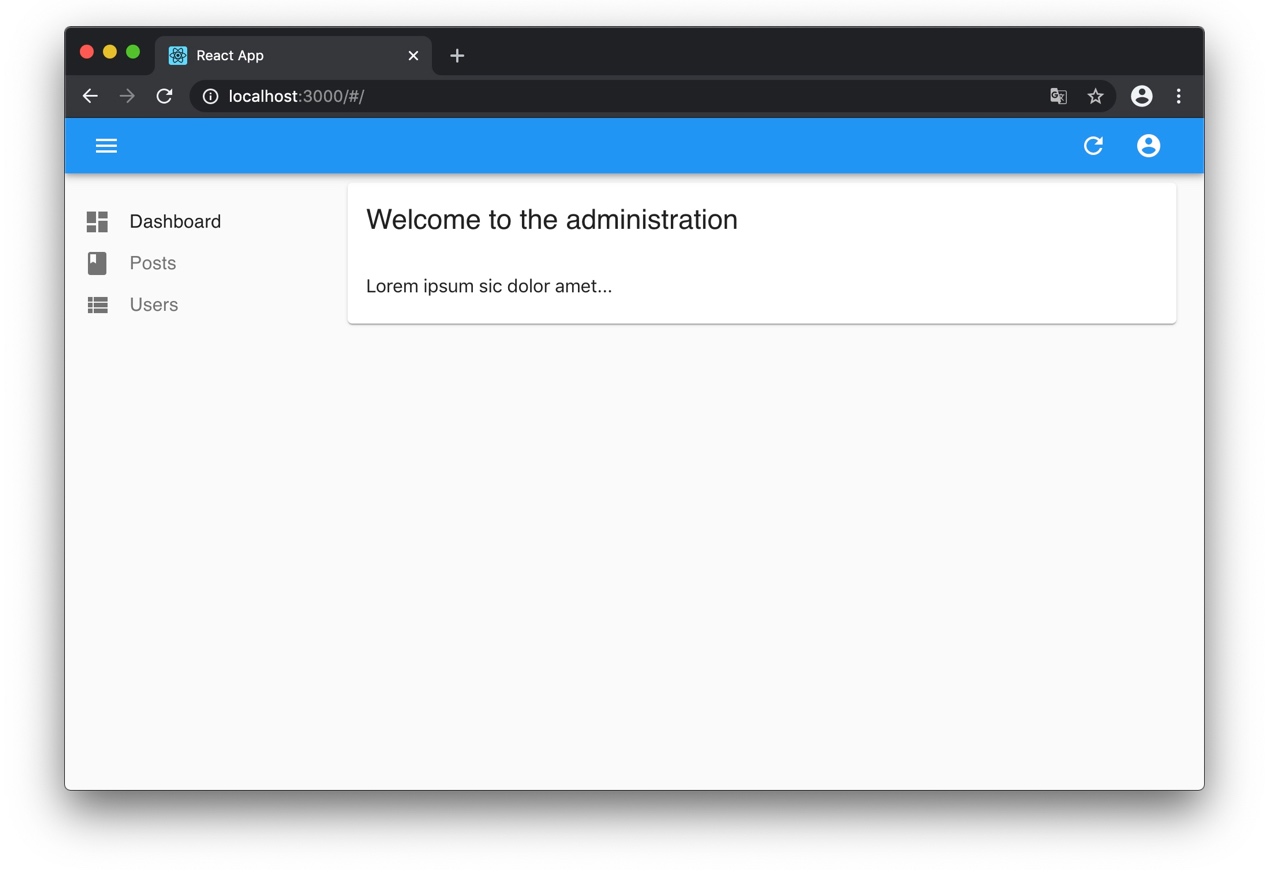
1. 向列表中添加搜索和过滤器



1. 自定义菜单图标

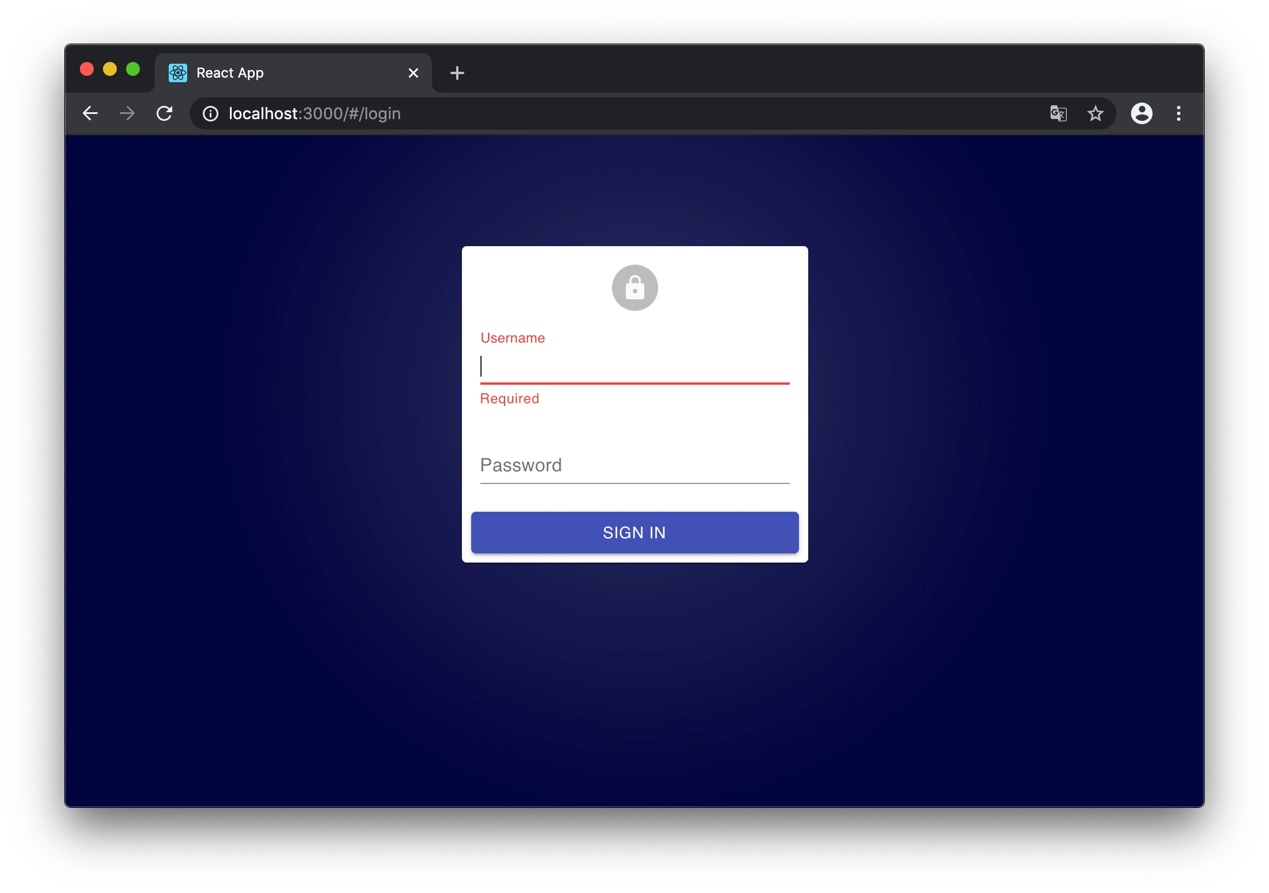
侧边栏菜单为文章和用户显示了相同的图标，我们可以自定义菜单图标。

1. 使用自定义主页

默认情况下，react-admin将第一个资源元素的列表页面显示为主页。如果想要显示自定义组件，可以将其传递到Admin的指示板道具中。

1. 添加登录页面

大多数管理应用程序都需要身份验证。React-admin可以在显示页面之前检查用户凭证，并在REST API返回403错误代码时重定向到登录页面。



1. 支持移动设备

试着调整浏览器的大小，看看侧边栏是如何在小屏幕上切换的。

但一个响应式布局并不足以使一个应用程序响应。Datagrid组件在桌面上运行良好，但并不适用于移动设备。如果必须在移动设备上使用，那么必须为小屏幕提供一个替代组件。

