Problem:

Consider these code fragments.

a. `end`
b. `get :index`
c. `assert_redirected_to car_path(assigns(:car))`
d. `assert_template :index`
e. `assert_template :new`
f. `assert_not_nil assigns(:cars)`
g. `get :new`
h. `test "should get index" do`
i. `post :create, car:{make:@car.make, model:@car.model, year: @car.year}`
j. `assert_response :success`

Using the above fragments, create a <u>functional test</u> for the “index” page of a car-themed web app. The test should make sure (1) that the HTTP response does not report an error, (2) that the correct ERB is rendered (index.html.erb), and (3) that the call to `Car.all` in the controller, which sets the `@cars` instance variable, does not fail and return nil. Note that your answer should use only 6 of the above fragments.

____

____

____

____

____

____

Solution:

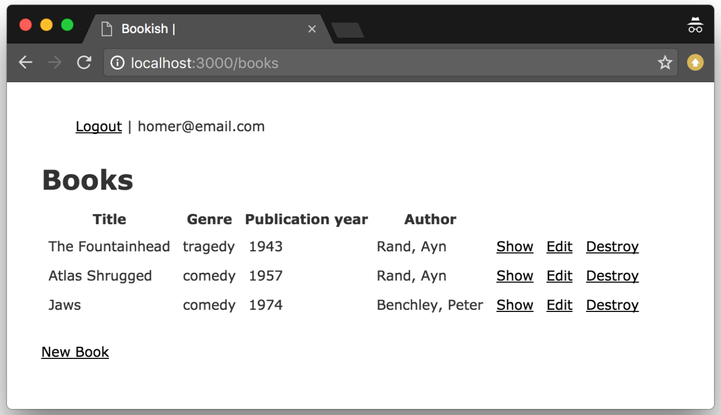
h

b

$$\left.\begin{array}{l} j \\ d \\ f \end{array}\right\} \text{order may vary}$$

a

Problem:


Bookish |

localhost:3000/books

Logout | homer@email.com

Books

Title	Genre	Publication year	Author			
The Fountainhead	tragedy	1943	Rand, Ayn	Show	Edit	Destroy
Atlas Shrugged	comedy	1957	Rand, Ayn	Show	Edit	Destroy
Jaws	comedy	1974	Benchley, Peter	Show	Edit	Destroy

New Book

Using the fragments below, create a functional test (class and method) for the "index" page of a book-themed web app (illustrated in the above figure). The test should do the following in this order (1) retrieve user fixture "one" and sign in the user, (2) simulate an HTTP request for the index page, (3) check that the HTTP response does not report an error, (4) check that the correct ERB is rendered (index.html.erb), and (5) check that the rendered HTML table contains a cell with everyone's favorite shark book. Some fragments may be used more than once in your solution. Some fragments may not be used at all.

```
a)  user = users(:one)
b)  include Devise::Test::IntegrationHelpers
c)  assert_response :error
d)  assert_response :success
e)  assert_select "h1", "Book"
f)  assert_select "td", "Jaws"
g)  sign_in user
h)  class BooksControllerTest < ActionDispatch::IntegrationTest
i)  assert_template :index
j)  assert_template :books
k)  test "should display books" do
l)  get books_url
m)  end
```

Solution:

h

b

K

a

g

l

d

i

f

m

m