Let’s build a To Do Manager with the basic CRUD features:

1. As a user, I can list tasks
2. As a user, I can view the details of a task
3. As a user, I can add a new task
4. As a user, I can edit a task (mark as completed / update title & details)
5. As a user, I can remove a task

**Specs**

**1 - Model**

Generate the Task model through the right rails generator. It should have at least the following columns:

* title, as a string
* details, as a text
* completed, as a boolean (default: false)

**2 - Controller**

Generate an empty (no actions) TasksController with the right rails generator.

For this exercise, **do not use resources** in your config/routes.rb. The goal of this exercise is to re-build the regular CRUD from scratch.

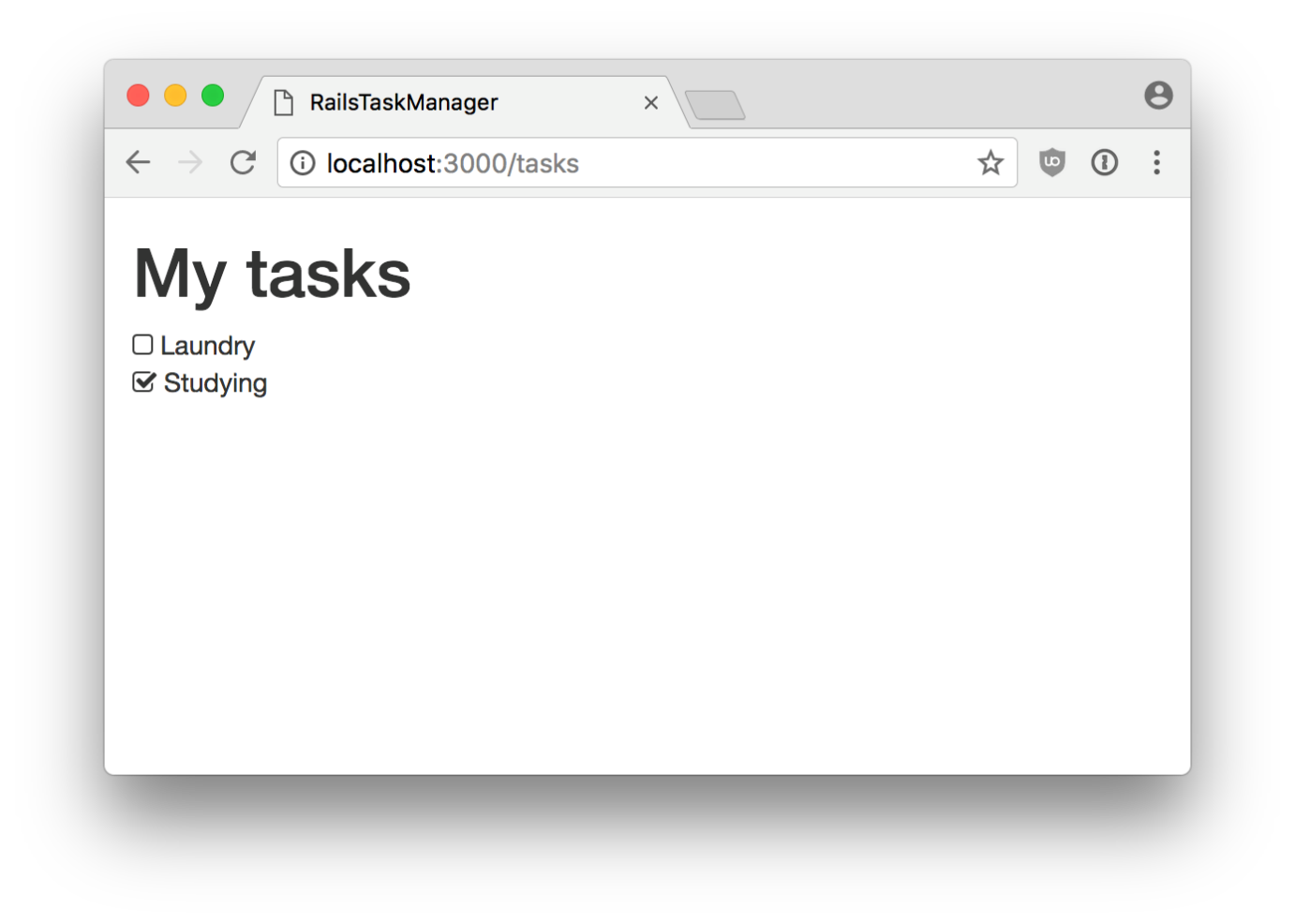
**3 - As a user, I can list tasks**

First, add a new route to list the tasks, following the convention from the lecture. Then, add a controller action and its view. This action should fetch **all** tasks, and a view should loop over these to display them, like in the screenshot below.

To test your view, you need some tasks in the database! To create some, run a rails console in another terminal tab and then run:

*Task*.create title: 'Laundry', details: 'Do not mix colors!'

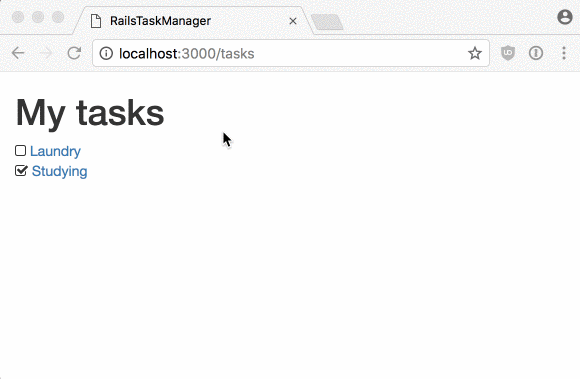
*Task*.create title: 'Studying', details: 'A lot of flashcards to do', completed: true



**4 - As a user, I can view the details of a task**

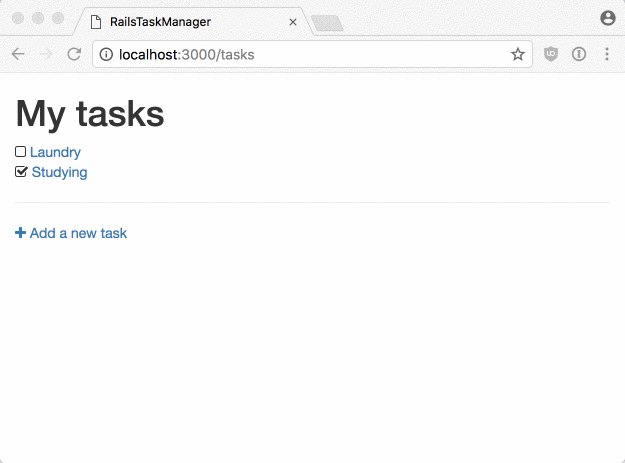
We now have a list of tasks, and we would like to click on the task title and navigate to a new page, displaying the details of this task. Following the conventions from the lecture, add a new route, a new controller action and a new view. This action should **find** a specific task, thanks to its id, directly from params.

Update the index.html.erb view with the link\_to helper to build the links.



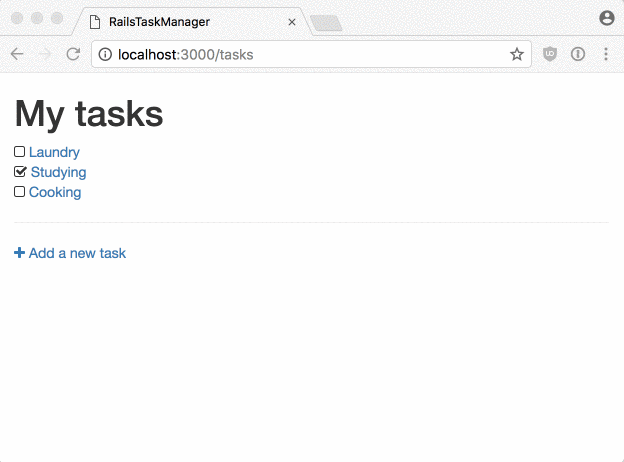
**5 - As a user, I can add a new task**

Following the lecture conventions, add two routes to handle the creation of a task. One route is there to display the Task form, and another one is there to handle the POST request generated when submitting this form. Try to use directly the form\_for helper in your view.



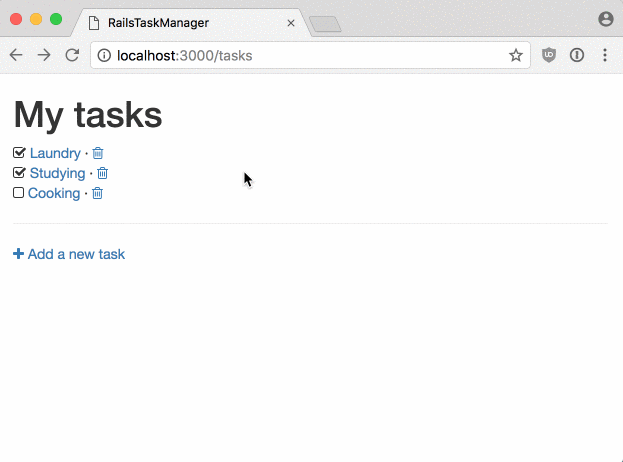
**6 - As a user, I can edit a task**

We want to be able to edit a task, changing its title, its details and especially **marking it as completed**. Following the lecture conventions, add the two routes you need for that feature. Implement the controller actions, and the views.



**7 - As a user, I can remove a task**

Last feature, we want to be able to destroy tasks directly from the list. A JavaScript confirmation could be handy.



**8 - Refactoring (Optional)**

Have a critical look at your code and introduce the following refactoring:

1. Use resources in your config/routes.rb
2. Use a before\_action in the TasksController
3. Should we DRY a bit the new and edit views? How can we handle the fact that the new form should **not** display “Completed”? ([hint](http://api.rubyonrails.org/classes/ActiveRecord/Persistence.html#method-i-new_record-3F))