The first argument to both methods is a route (pattern) that will be matched. The path() method uses angle brackets to define parts of a URL that will be captured and passed through to the view function as named arguments. The re\_path() function uses a flexible pattern matching approach known as a regular expression. We'll talk about these in a later article!

The second argument is another function that will be called when the pattern is matched. The notation views.book\_detail indicates that the function is called book\_detail() and can be found in a module called views (i.e. inside a file named views.py)

The code snippet shows what the HTML template called by the render() function in the previous section might look like. This template has been written under the assumption that it will have access to a list variable called youngest\_teams when it is rendered (this is contained in the context variable inside the render() function above). Inside the HTML skeleton we have an expression that first checks if the youngest\_teams variable exists, and then iterates it in a for loop. On each iteration the template displays each team's team\_name value in an [<li>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/li) element.

Forms: HTML Forms are used to collect user data for processing on the server. Django simplifies form creation, validation, and processing.

User authentication and permissions: Django includes a robust user authentication and permission system that has been built with security in mind.

Caching: Creating content dynamically is much more computationally intensive (and slow) than serving static content. Django provides flexible caching so that you can store all or part of a rendered page so that it doesn't get re-rendered except when necessary.

Administration site: The Django administration site is included by default when you create an app using the basic skeleton. It makes it trivially easy to provide an admin page for site administrators to create, edit, and view any data models in your site.

Serializing data: Django makes it easy to serialize and serve your data as XML or JSON. This can be useful when creating a web service (a website that purely serves data to be consumed by other applications or sites, and doesn't display anything itself), or when creating a website in which the client-side code handles all the rendering of data.