

Thymeleaf - Add Employee



Add Employee - DEMO

Employee Directory

Add Employee

First Name	Last Name	Email
Leslie	Andrews	leslie@luv2code.com
Emma	Baumgarten	emma@luv2code.com
Avani	Gupta	avani@luv2code.com
Yuri	Petrov	yuri@luv2code.com
Juan	Vega	juan@luv2code.com

Add Employee

Step-By-Step

Add Employee

Step-By-Step

1. New **Add Employee** button for list-employees.html



The screenshot shows a user interface titled "Employee Directory". At the top left is a blue "Add Employee" button. Below it is a table with three columns: "First Name", "Last Name", and "Email". The table contains five rows of data:

First Name	Last Name	Email
Leslie	Andrews	leslie@luv2code.com
Emma	Baumgarten	emma@luv2code.com
Avani	Gupta	avani@luv2code.com
Yuri	Petrov	yuri@luv2code.com
Juan	Vega	juan@luv2code.com

Add Employee

Step-By-Step

1. New **Add Employee** button for list-employees.html
2. Create HTML form for new employee



Save Employee

First name

Last name

Email

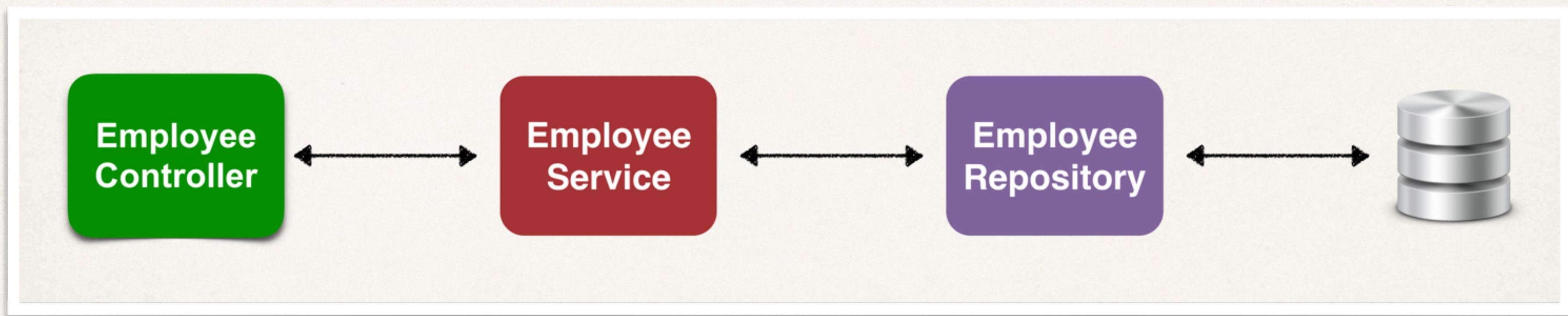
Save

[Back to Employees List](#)

Add Employee

Step-By-Step

1. New **Add Employee** button for list-employees.html
2. Create HTML form for new employee
3. Process form data to save employee



Step 1: New "Add Employee" button

Step 1: New "Add Employee" button

- Add Employee button will href link to
 - request mapping **/employees/showFormForAdd**

Step 1: New "Add Employee" button

- Add Employee button will href link to
 - request mapping **/employees/showFormForAdd**

```
<a th:href="@{/employees/showFormForAdd}">  
    Add Employee  
</a>
```

Add Employee

Step 1: New "Add Employee" button

- Add Employee button
- request mapping

@ symbol
Reference context path of your application
(app root)

```
<a th:href="@{/employees/showFormForAdd}">  
    Add Employee  
</a>
```

Add Employee

Step 1: New "Add Employee" button

- Add Employee button will href link to
 - request mapping **/employees/showFormForAdd**

```
<a th:href="@{/employees/showFormForAdd}">  
    Add Employee  
</a>
```

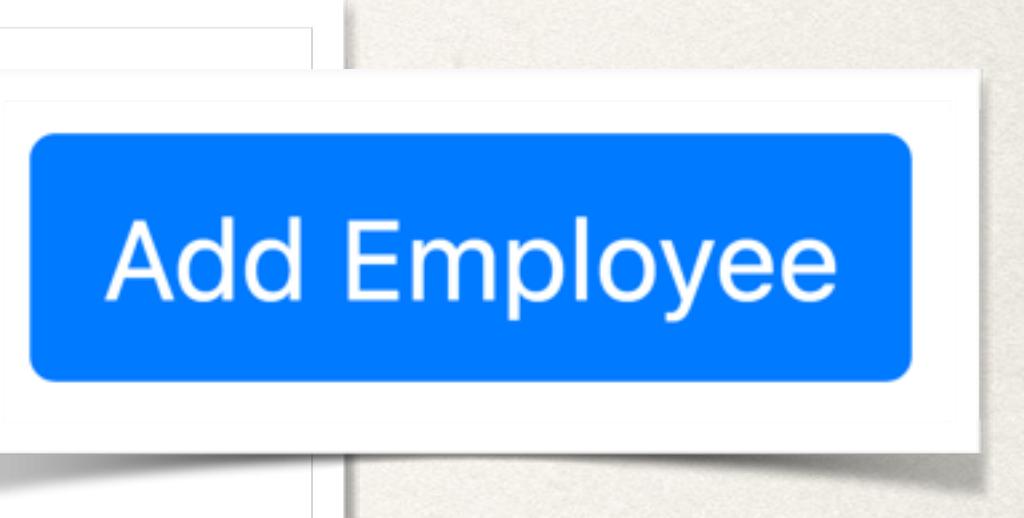
Add Employee

Apply Bootstrap styles

Step 1: New "Add Employee" button

- Add Employee button will href link to
 - request mapping **/employees/showFormForAdd**

```
<a th:href="@{/employees/showFormForAdd}"  
    class="btn btn-primary btn-sm mb-3">  
    Add Employee  
</a>
```



Add Employee

Apply Bootstrap styles

Step 1: New "Add Employee" button

- Add Employee button will href link to
 - request mapping **/employees/showFormForAdd**

```
<a th:href="@{/employees/showFormForAdd}"  
    class="btn btn-primary btn-sm mb-3">  
    Add Employee  
</a>
```

Apply Bootstrap styles

Button
Button Primary
Button Small
Margin Bottom, 3 pixels

Add Employee

Step 1: New "Add Employee" button

- Add Employee button will href link to
 - request Docs on Bootstrap styles: www.getbootstrap.com

```
<a th:href="@{/employees/showFormForAdd}"  
    class="btn btn-primary btn-sm mb-3">  
    Add Employee  
</a>
```

Apply Bootstrap styles

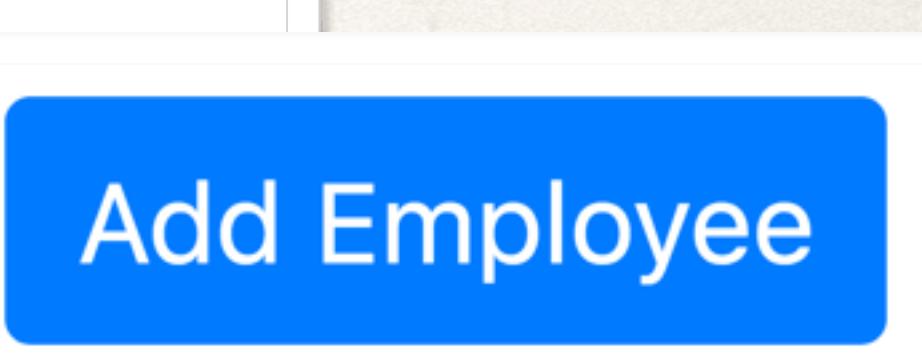
Button
Button Primary
Button Small
Margin Bottom, 3 pixels

Add Employee

Step 1: New "Add Employee" button

- Add Employee button will href link to
 - request mapping **/employees/showFormForAdd**

```
<a th:href="@{/employees/showFormForAdd}"  
    class="btn btn-primary btn-sm mb-3">  
    Add Employee  
</a>
```



Step 1: New "Add Employee" button

- Add Employee button will href link to request mapping `/employees/add`
- request mapping `/employees/add`

TODO:

Add controller request mapping for
`/employees/showFormForAdd`

```
<a th:href="@{/employees/showFormForAdd}"  
    class="btn btn-primary btn-sm mb-3">  
    Add Employee  
</a>
```

Add Employee

Showing Form

Showing Form

In your Spring Controller

Showing Form

In your Spring Controller

- Before you show the form, you must add a *model attribute*

Showing Form

In your Spring Controller

- Before you show the form, you must add a *model attribute*
- This is an object that will hold form data for the *data binding*

Controller code to show form

Controller code to show form

```
@Controller  
@RequestMapping("/employees")  
public class EmployeeController {
```

Controller code to show form

```
@Controller  
@RequestMapping("/employees")  
public class EmployeeController {  
  
    @GetMapping("/showFormForAdd")  
    public String showFormForAdd(Model theModel) {
```

Controller code to show form

```
@Controller  
@RequestMapping("/employees")  
public class EmployeeController {  
  
    @GetMapping("/showFormForAdd")  
    public String showFormForAdd(Model theModel) {  
  
        // create model attribute to bind form data  
        Employee theEmployee = new Employee();
```

Controller code to show form

```
@Controller  
@RequestMapping("/employees")  
public class EmployeeController {  
  
    @GetMapping("/showFormForAdd")  
    public String showFormForAdd(Model theModel) {  
  
        // create model attribute to bind form data  
        Employee theEmployee = new Employee();  
  
        theModel.addAttribute("employee", theEmployee);  
    }  
}
```

Controller code to show form

```
@Controller  
@RequestMapping("/employees")  
public class EmployeeController {  
  
    @GetMapping("/showFormForAdd")  
    public String showFormForAdd(Model theModel) {  
  
        // create model attribute to bind form data  
        Employee theEmployee = new Employee();  
  
        theModel.addAttribute("employee", theEmployee);  
    }  
}
```

Our Thymleaf template will access this data for binding form data

Controller code to show form

```
@Controller  
@RequestMapping("/employees")  
public class EmployeeController {  
  
    @GetMapping("/showFormForAdd")  
    public String showFormForAdd(Model theModel) {  
  
        // create model attribute to bind form data  
        Employee theEmployee = new Employee();  
  
        theModel.addAttribute("employee", theEmployee);  
  
        return "employees/employee-form";  
    }  
  
    ...  
}
```

Our Thymleaf template will access this data for binding form data

Controller code to show form

```
@Controller  
@RequestMapping("/employees")  
public class EmployeeController {  
  
    @GetMapping("/showFormForAdd")  
    public String showFormForAdd(Model theModel) {  
  
        // create model attribute to bind form data  
        Employee theEmployee = new Employee();  
  
        theModel.addAttribute("employee", theEmployee);  
  
        return "employees/employee-form";  
    }  
    ...  
}
```

Our Thymleaf template will access this data for binding form data

src/main/resources/templates/employees/employee-form.html

Thymeleaf and Spring MVC Data Binding

Thymeleaf and Spring MVC Data Binding

- Thymeleaf has special expressions for binding Spring MVC form data

Thymeleaf and Spring MVC Data Binding

- Thymeleaf has special expressions for binding Spring MVC form data
- Automatically setting / retrieving data from a Java object

Thymeleaf Expressions

Thymeleaf Expressions

- Thymeleaf expressions can help you build the HTML form :-)

Thymeleaf Expressions

- Thymeleaf expressions can help you build the HTML form :-)

Expression	Description
th:action	Location to send form data

Thymeleaf Expressions

- Thymeleaf expressions can help you build the HTML form :-)

Expression	Description
th:action	Location to send form data
th:object	Reference to model attribute

Thymeleaf Expressions

- Thymeleaf expressions can help you build the HTML form :-)

Expression	Description
th:action	Location to send form data
th:object	Reference to model attribute
th:field	Bind input field to a property on model attribute

Thymeleaf Expressions

- Thymeleaf expressions can help you build the HTML form :-)

Expression	Description
th:action	Location to send form data
th:object	Reference to model attribute
th:field	Bind input field to a property on model attribute
<i>more</i>	See - www.luv2code.com/thymeleaf-create-form

Step 2: Create HTML form for new employee

Step 2: Create HTML form for new employee

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
</form>
```

Step 2: Create HTML form for new employee

Empty place holder
Thymeleaf will handle real work

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
</form>
```

Step 2: Create HTML form for new employee

Empty place holder
Thymeleaf will handle real work

Real work
Send form data to
/employees/save

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
</form>
```

Step 2: Create HTML form for new employee

Empty place holder
Thymeleaf will handle real work

Real work
Send form data to
`/employees/save`

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
</form>
```

Our model attribute

```
theModel.addAttribute("employee", theEmployee);
```

Step 2: Create HTML form for new employee

Step 2: Create HTML form for new employee

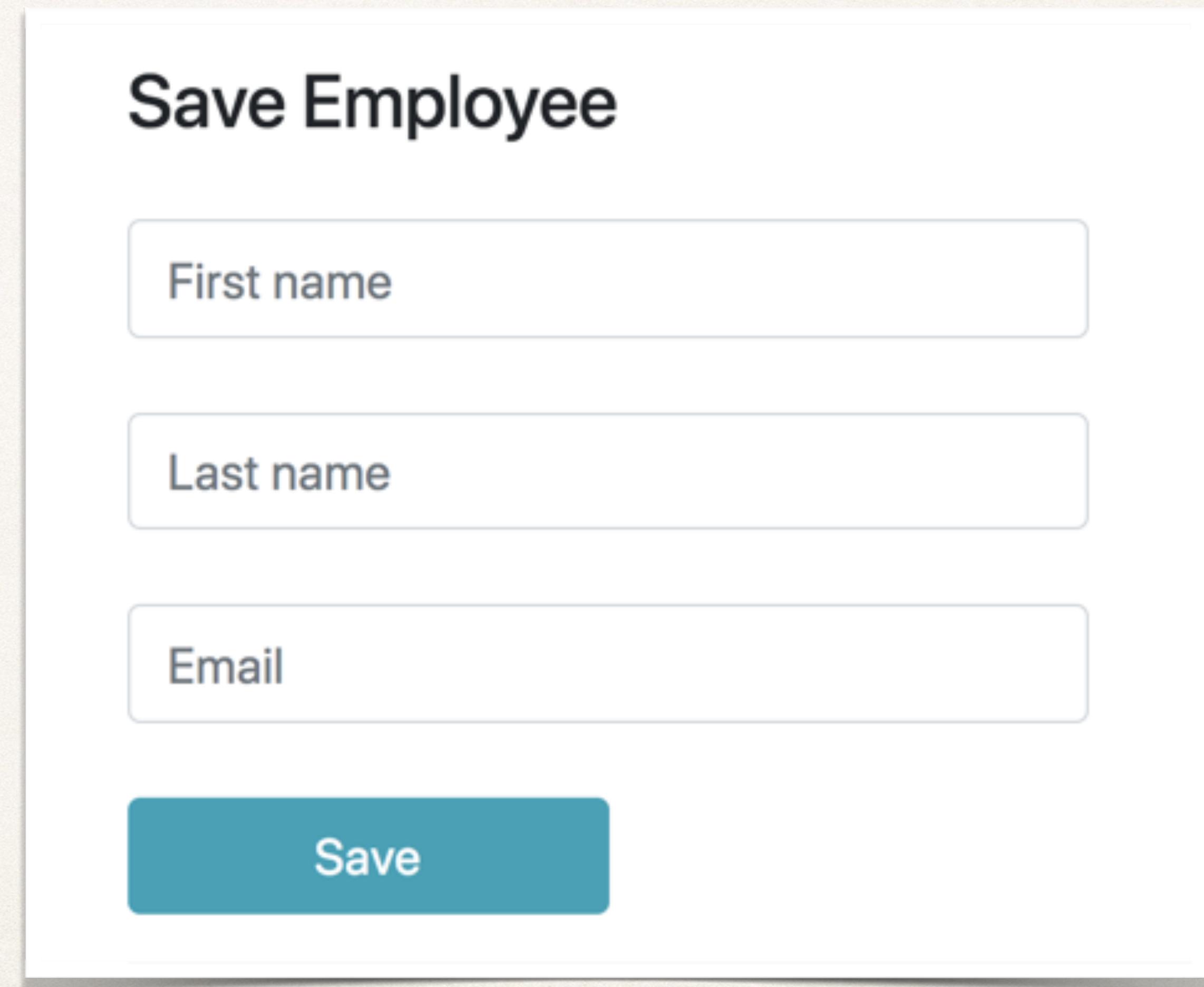
Save Employee

First name

Last name

Email

Save



Step 2: Create HTML form for new employee

Save Employee

First name

Last name

Email

Save

Step 2: Create HTML form for new employee

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">
```

Save Employee

First name

Last name

Email

Save

Step 2: Create HTML form for new employee

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
<input type="text" th:field="*{firstName}" placeholder="First name">
```

Save Employee

First name

Last name

Email

Save

Step 2: Create HTML form for new employee

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
<input type="text" th:field="*{firstName}" placeholder="First name">
```

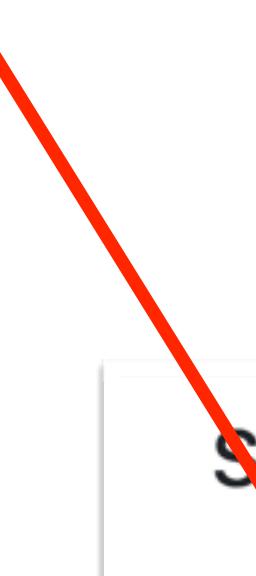
Save Employee

First name

Last name

Email

Save



Step 2: Create HTML form for new employee

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
<input type="text" th:field="*{firstName}" placeholder="First name">
```

* { ... }

Selects property on referenced
th:object

Save Employee

First name

Last name

Email

Save

Step 2: Create HTML form for new employee

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
<input type="text" th:field="*{firstName}" placeholder="First name">
```

* { ... }

Selects property on referenced
th:object

Save Employee

First name

Last name

Email

Save

Step 2: Create HTML form for new employee

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
<input type="text" th:field="*{firstName}" placeholder="First name">  
  
<input type="text" th:field="*{lastName}" placeholder="Last name">  
  
<input type="text" th:field="*{email}" placeholder="Email">
```

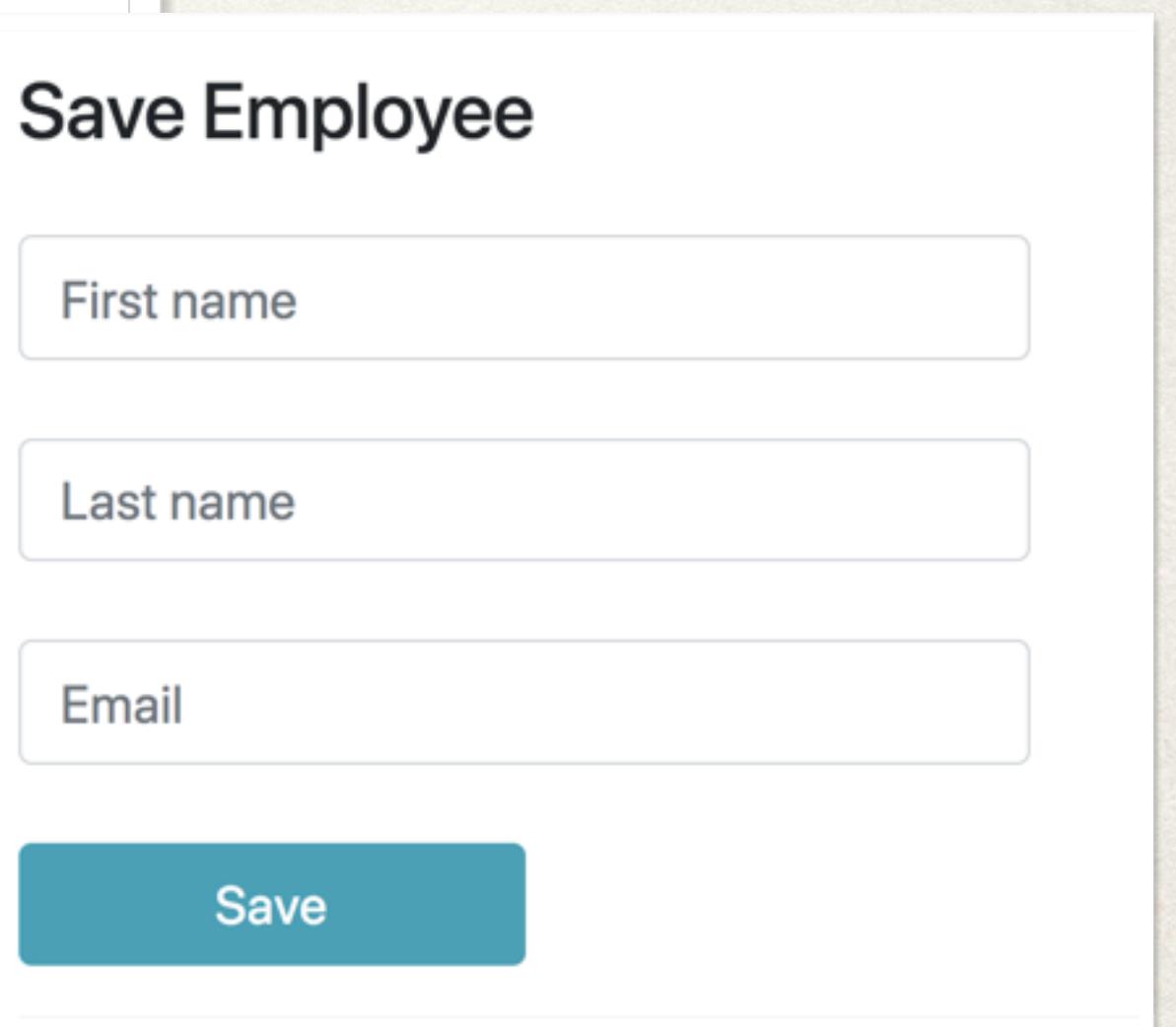
Save Employee

First name

Last name

Email

Save



Step 2: Create HTML form for new employee

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
<input type="text" th:field="*{firstName}" placeholder="First name">  
  
<input type="text" th:field="*{lastName}" placeholder="Last name">  
  
<input type="text" th:field="*{email}" placeholder="Email">  
  
<button type="submit">Save</button>  
  
</form>
```

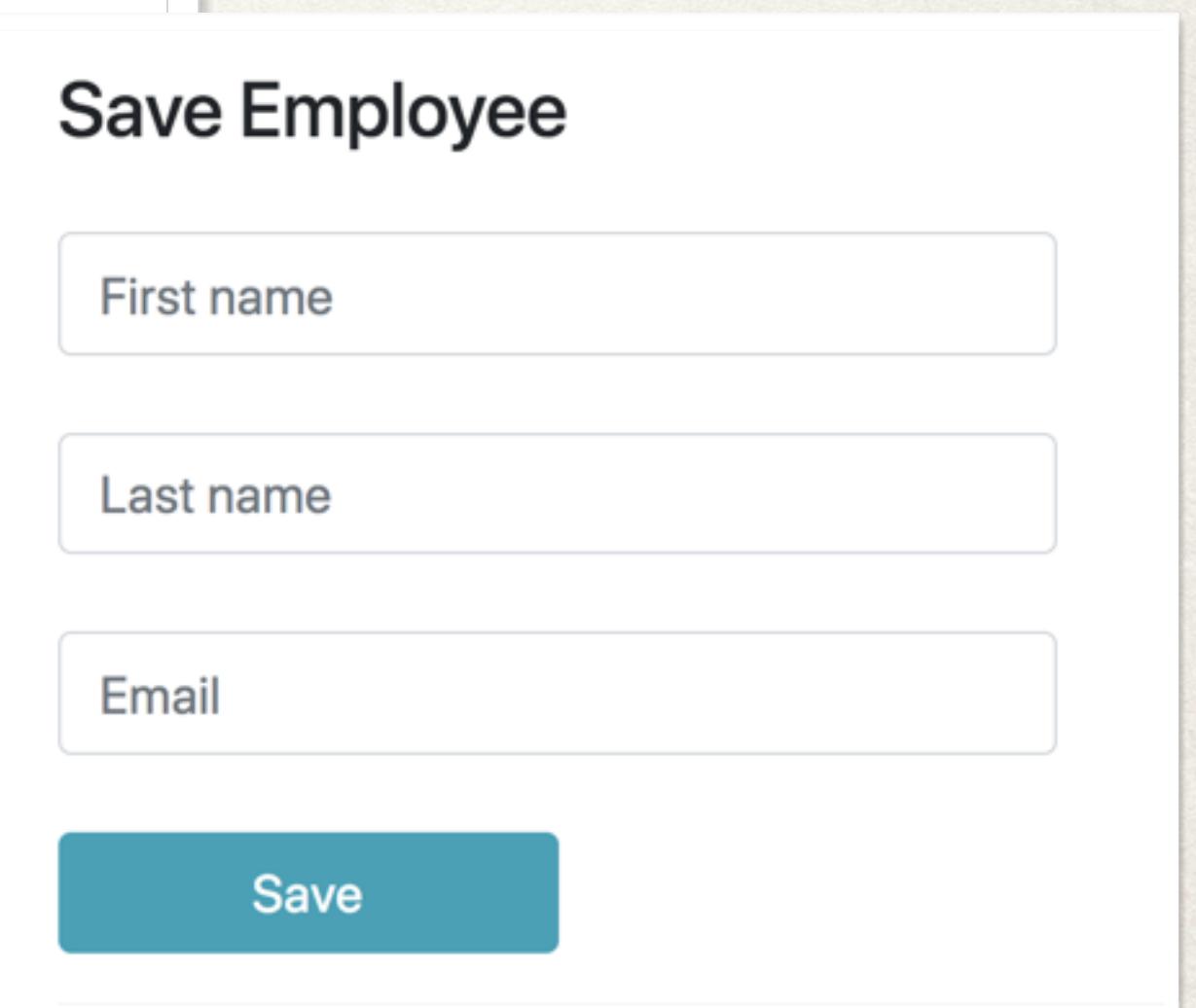
Save Employee

First name

Last name

Email

Save



Step 2: Create HTML form for new employee

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
  <input type="text" th:field="*{firstName}" placeholder="First name">  
  
  <input type="text" th:field="*{lastName}" placeholder="Last name">  
  
  <input type="text" th:field="*{email}" placeholder="Email">  
  
  <button type="submit">Save</button>  
  
</form>
```

Step 2: Create HTML form for new employee

1

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
  <input type="text" th:field="*{firstName}" placeholder="First name">  
  
  <input type="text" th:field="*{lastName}" placeholder="Last name">  
  
  <input type="text" th:field="*{email}" placeholder="Email">  
  
  <button type="submit">Save</button>  
  
</form>
```

When form is **loaded**,
will call:

employee.getFirstName()

...

employee.getLastName

Step 2: Create HTML form for new employee

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
  <input type="text" th:field="*{firstName}" placeholder="First name">  
  
  <input type="text" th:field="*{lastName}" placeholder="Last name">  
  
  <input type="text" th:field="*{email}" placeholder="Email">  
  
  <button type="submit">Save</button>  
</form>
```

1

When form is **loaded**,
will call:

employee.getFirstName()

...
employee.getLastName

2

When form is **submitted**,
will call:

employee.setFirstName(...)

...
employee.setLastName(...)

Step 2: Create HTML form for new employee

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
  <input type="text" th:field="*{firstName}" placeholder="First name">  
  
  <input type="text" th:field="*{lastName}" placeholder="Last name">  
  
  <input type="text" th:field="*{email}" placeholder="Email">  
  
  <button type="submit">Save</button>  
  
</form>
```

Step 2: Create HTML form for new employee

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
<input type="text" th:field="*{firstName}" placeholder="First name">  
  
<input type="text" th:field="*{lastName}" placeholder="Last name">  
  
<input type="text" th:field="*{email}" placeholder="Email">  
  
<button type="submit">Save</button>  
  
</form>
```

Save Employee

First name

Last name

Email

Save

Step 2: Create HTML form for new employee

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
<input type="text" th:field="*{firstName}" placeholder="First name">  
  
<input type="text" th:field="*{lastName}" placeholder="Last name">  
  
<input type="text" th:field="*{email}" placeholder="Email">  
  
<button type="submit">Save</button>  
  
</form>
```



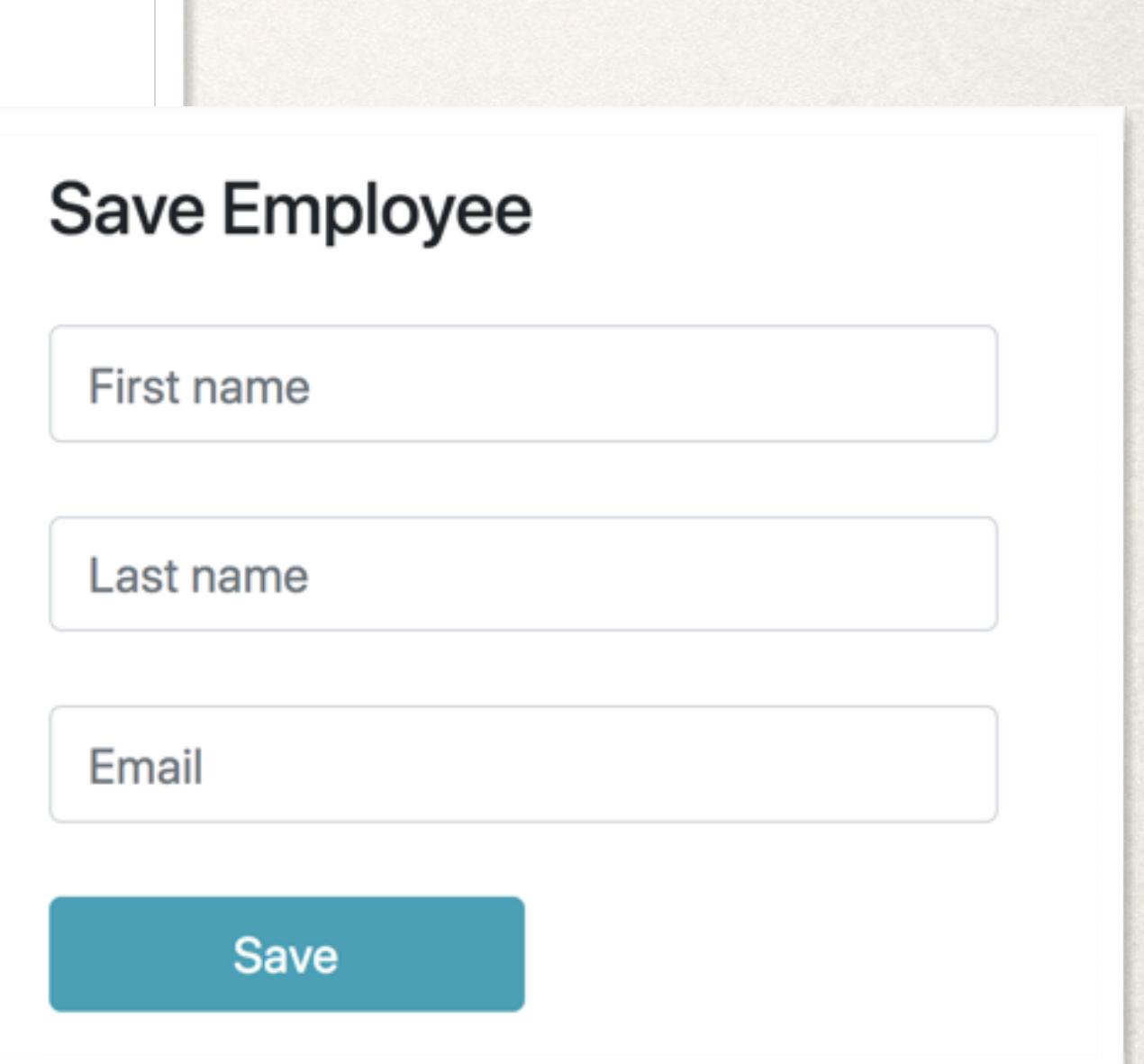
Save Employee

First name

Last name

Email

Save

A screenshot of a web browser window titled "Save Employee". It contains three text input fields: "First name", "Last name", and "Email", each with a placeholder text. Below the inputs is a teal-colored "Save" button.

Step 2: Create HTML form for new employee

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
<input type="text" th:field="*{firstName}" placeholder="First name"  
       class="form-control mb-4 col-4">  
  
</form>
```

Apply Bootstrap styles

Save Employee

First name

Last name

Email

Save

Step 2: Create HTML form for new employee

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
<input type="text" th:field="*{firstName}" placeholder="First name"  
       class="form-control mb-4 col-4">  
  
</form>
```

Apply Bootstrap styles

Save Employee

First name

Last name

Email

Save

Step 2: Create HTML form for new employee

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
<input type="text" th:field="*{firstName}" placeholder="First name"  
       class="form-control mb-4 col-4">
```

Apply Bootstrap styles

Form control
Margin Bottom, 4 pixels
Column Span 4

```
</form>
```

Save Employee

First name

Last name

Email

Save

Step 2: Create HTML form for new employee

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
<input type="text" th:field="*{firstName}" placeholder="First name"  
       class="form-control mb-4 col-4">  
  
</form>
```

Save Employee

First name

Last name

Email

Save

Step 2: Create HTML form for new employee

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
<input type="text" th:field="*{firstName}" placeholder="First name"  
       class="form-control mb-4 col-4">  
  
<input type="text" th:field="*{lastName}" placeholder="Last name"  
       class="form-control mb-4 col-4">  
  
<input type="text" th:field="*{email}" placeholder="Email"  
       class="form-control mb-4 col-4">  
  
<button type="submit" class="btn btn-info col-2">Save</button>  
  
</form>
```

Apply Bootstrap styles

Button
Button Info
Column Span 2

Save Employee

First name

Last name

Email

Save

Step 2: Create HTML form for new employee

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
<input type="text" th:field="*{firstName}" placeholder="First name"  
       class="form-control mb-4 col-4">  
  
<input type="text" th:field="*{lastName}" placeholder="Last name"  
       class="form-control mb-4 col-4">  
  
<input type="text" th:field="*{email}" placeholder="Email"  
       class="form-control mb-4 col-4">  
  
<button type="submit" class="btn btn-info col-2">Save</button>  
  
</form>
```

Save Employee

First name

Last name

Email

Save

Step 2: Create HTML form for new employee

```
<form action="#" th:action="@{/employees/save}"  
      th:object="${employee}" method="POST">  
  
<input type="text" th:field="*{firstName}" placeholder="First name"  
       class="form-control mb-4 col-4">  
  
<input type="text" th:field="*{lastName}" placeholder="Last name"  
       class="form-control mb-4 col-4">  
  
<input type="text" th:field="*{email}" placeholder="Email address"  
       class="form-control mb-4 col-4">  
  
<button type="submit" class="btn btn-info col-2">Save</button>  
  
</form>
```

TODO:

Add controller request mapping for
`/employees/save`

Save Employee

First name

Last name

Email

Save

Step 3: Process form data to save employee

Step 3: Process form data to save employee

```
@Controller  
@RequestMapping("/employees")  
public class EmployeeController {
```

Step 3: Process form data to save employee

```
@Controller  
@RequestMapping("/employees")  
public class EmployeeController {  
  
    private EmployeeService employeeService;  
  
    public EmployeeController(EmployeeService theEmployeeService) {  
        employeeService = theEmployeeService;  
    }  
}
```

Constructor injection

Step 3: Process form data to save employee

```
@Controller  
@RequestMapping("/e  
public class Employ  
  
    private EmployeeService employeeService;  
  
    public EmployeeController(EmployeeService theEmployeeService) {  
        employeeService = theEmployeeService;  
    }
```

Since only one constructor
@Autowired is optional

Constructor injection

Step 3: Process form data to save employee

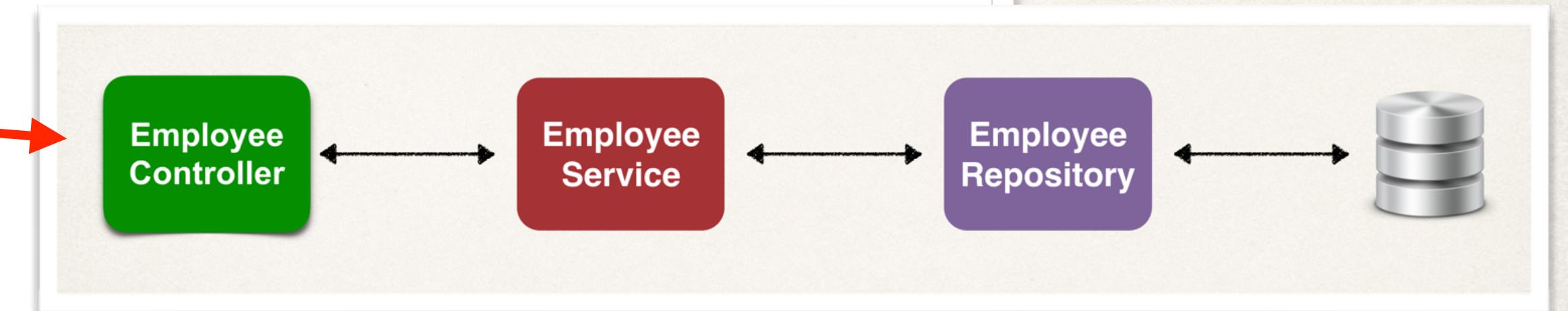
```
@Controller  
@RequestMapping("/employees")  
public class EmployeeController {  
  
    private EmployeeService employeeService;  
  
    public EmployeeController(EmployeeService theEmployeeService) {  
        employeeService = theEmployeeService;  
    }  
  
    @PostMapping("/save")  
    public String saveEmployee(@ModelAttribute("employee") Employee theEmployee) {
```

Step 3: Process form data to save employee

```
@Controller  
@RequestMapping("/employees")  
public class EmployeeController {  
  
    private EmployeeService emp...  
  
    public EmployeeController(E...  
        employeeService = theEmp...  
    }  
  
    @PostMapping("/save")  
    public String saveEmployee(@ModelAttribute("employee") Employee theEmployee) {  
  
        <form action="#" th:action="@{/employees/save}"  
              th:object="${employee}" method="POST">  
    }  
}
```

Step 3: Process form data to save employee

```
@Controller  
@RequestMapping("/employees")  
public class EmployeeController {  
  
    private EmployeeService employeeService;  
  
    public EmployeeController(EmployeeService theEmployeeService) {  
        employeeService = theEmployeeService;  
    }  
  
    @PostMapping("/save")  
    public String saveEmployee(@ModelAttribute("employee") Employee theEmployee) {  
  
        // save the employee  
        employeeService.save(theEmployee);  
    }  
}
```



Step 3: Process form data to save employee

```
@Controller  
@RequestMapping("/employees")  
public class EmployeeController {  
  
    private EmployeeService employeeService;  
  
    public EmployeeController(EmployeeService theEmployeeService) {  
        employeeService = theEmployeeService;  
    }  
  
    @PostMapping("/save")  
    public String saveEmployee(@ModelAttribute("employee") Employee theEmployee) {  
  
        // save the employee  
        employeeService.save(theEmployee);  
  
        // use a redirect to prevent duplicate submissions  
        return "redirect:/employees/list";  
    }  
    ...  
}
```

Redirect to request mapping
`/employees/list`

Step 3: Process form data to save employee

```
@Controller  
@RequestMapping("/employees")  
public class EmployeeController {  
  
    private EmployeeService employeeService;  
  
    public EmployeeController(EmployeeService theEmployeeService) {  
        employeeService = theEmployeeService;  
    }  
  
    @PostMapping("/save")  
    public String saveEmployee(@ModelAttribute("employee") Employee theEmployee) {  
  
        // save the employee  
        employeeService.save(theEmployee);  
  
        // use a redirect to prevent duplicate submissions  
        return "redirect:/employees/list";  
    }  
    ...  
}
```

Redirect to request mapping
`/employees/list`

"Post/Redirect/Get" pattern

For more info see
www.luv2code.com/post-redirect-get