# Thymeleaf and Validations - Exercise

## Functionality Overview

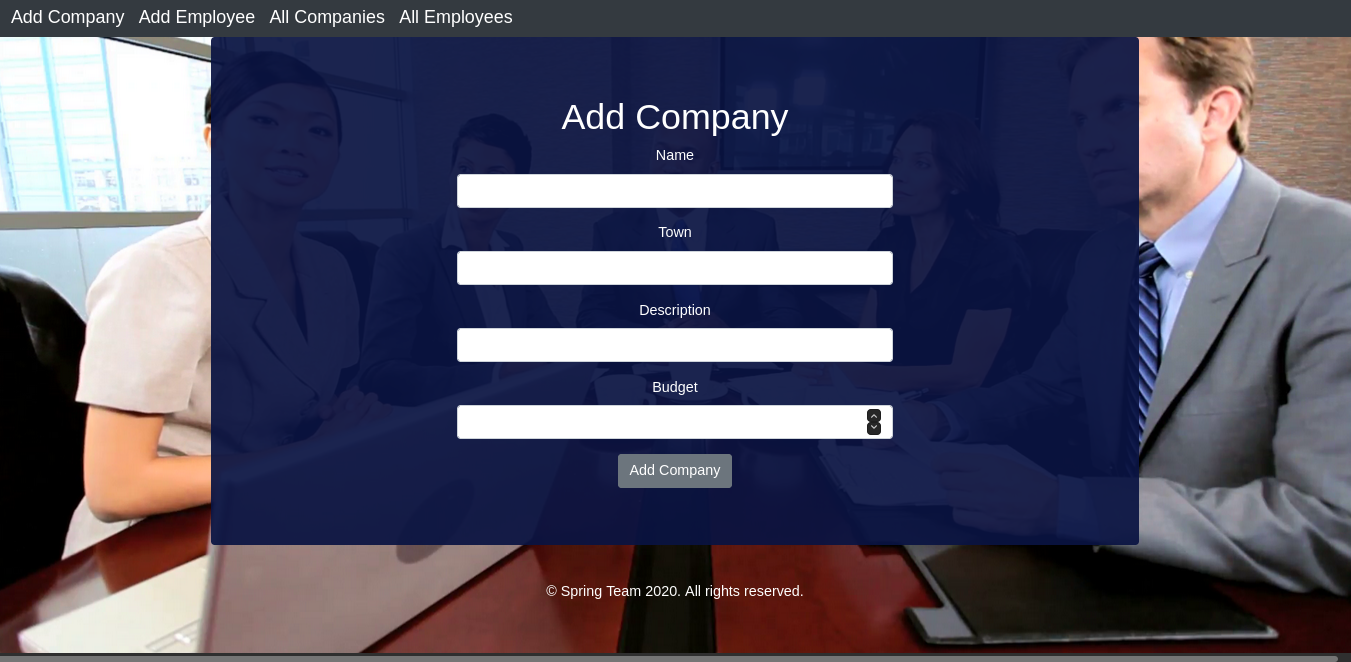
The application should be able to easily **accept** hard-formatted. The application is called – **LinkedOut**.

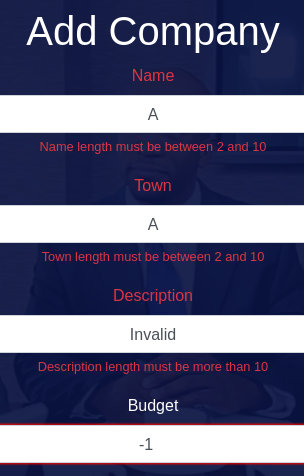
Look at the pictures below to see what must happen:

* **route("/")**



* **route("/companies/add")**





* **route("/employees/add")**



* **route("/companies/all")**





* **route("/employees/all")**





## Model Definition

There are 2 main models that the **LinkedOut database** application should contain in its functionality.

Design them in the **most appropriate** way, considering the following **data constraints**:

### Company

* id – a **char sequence**
* **budget –** a **number.** (must be a positive number). Cannot be **null**.
* **description –** a **very long char sequence.** Cannot be **null**. Must be **at least 10 characters.**
* name – a **char sequence.** Must be **unique** and cannot be **null**. Must be **between 2 and 10 characters**.
* **town –** a **char sequence.** Cannot be **null**. Must be **between 2 and 10 characters**.

### Employee

* id – a **char sequence**
* **birthdate –** a **date.** Cannot be **null**.
* **educationLevel –** a **char sequence.** Cannot be **null**.
* firstName – a **char sequence.** Cannot be **null.** Must be at least **2 characters.** Cannot be **null**.
* **jobTitle –** a **char sequence.** Cannot be **null**.
* lastName – a **char sequence.** Cannot be **null.** Must be at least **2 characters.** Cannot be **null**.
* **salary –** a **number** (must be a positive number). Cannot be **null.**

**NOTE**: Name the entities and their class members, **exactly** in the **format stated** above. Do not name them in snake case with the dashes, of course.

#### Relationships

Your partners gave you a little hint about the more complex relationships in the database, so that you can implement it correctly.

One Employee may be in only one Compnay, and one Compnay may have many Employees.

