✅ Case Study Title: Employee Info API using Spring Boot AutoConfiguration

/employeeapp/src/main/java/com/example/employeeapp/model/Employee.java

```
package com.example.employeeapp.model;
public class Employee {
	private int id;
	private String name;
	private String email;
	private String department;

	public Employee() {}
	public Employee(int id, String name, String email, String department) {
		this.id = id;
		this.name = name;
		this.email = email;
		this.department = department;
	}
	public int getId() {
		return id;
	}
	public void setId(int id) {
		this.id = id;
	}
	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
	public String getEmail() {
		return email;
	}
	public void setEmail(String email) {
		this.email = email;
	}
	public String getDepartment() {
		return department;
	}
	public void setDepartment(String department) {
		this.department = department;
	}
}
```

/employeeapp/src/main/java/com/example/employeeapp/controller/EmployeeController.java

```
package com.example.employeeapp.controller;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import com.example.employeeapp.model.Employee;
import com.example.employeeapp.service.EmployeeService;
@Controller
@RequestMapping("/employees") //main route for employees
public class EmployeeController {

	@Autowired
	private EmployeeService service;

	@GetMapping //default route
	public String listEmployees(Model model) {
		model.addAttribute("employees", service.getAll());
		return "index"; //html template page
	}

	@GetMapping("/add") //get form to post
	public String showAddForm(Model model) {
		model.addAttribute("employees", new Employee());
		return "add"; //html template page
	}

	@PostMapping("/add") //route for post request
	public String addEmployee(@ModelAttribute Employee emp) {
		service.add(emp);
		return "redirect:/employees";
	}

	@GetMapping("/edit/{id}")
	public String showEditForm(@PathVariable int id, Model model) {
		model.addAttribute("employees", service.getById(id));
		return "edit";
	}

	@PostMapping("/update")
	public String updateEmployee(@ModelAttribute Employee emp) {
		service.update(emp);
```

```java
		return "redirect:/employees";
	}

	@GetMapping("/delete/{id}")
	public String deleteEmployee(@PathVariable int id) {
		service.delete(id);
		return "redirect:/employees";
	}
}
```

2. Spring Boot – Actuators

🎯 Case Study: Monitoring an Inventory System

Application.properties

```properties
spring.application.name=actuatordemo
#Spring Boot Actuator exposes production-ready features like health checks, metrics, beans, and custom endpoints.
management.endpoints.web.exposure.include=health,beans,metrics,env

management.endpoints.web.base-path=/actuator

management.endpoint.health.show-details=always
```