Programowanie komponentowe – ćwiczenie 9
Cel ćwiczenia:
REST

1. Wykorzystujemy ćwiczenie 8

2. Dodajemy starter :
springdoc-openapi-starter-webmvc-ui

3. Utworzenie kontrolera w pakiecie .controllers

```
@RestController
@RequestMapping(value = "/rest")
@Validated
public class UserRestController {

	@PostMapping("/adduser")
	public ResponseEntity<Result> addUser(@Valid @RequestBody User user) {
		try {
			// TODO zapis User do bazy
			model.addAttribute("user", user);
			return ResponseEntity
				.status(HttpStatus.OK)
				.body(new Result()) // TODO zainicjować pola obiektu Result zainicjować
		}
		catch (Exception e) {
			return ResponseEntity
				.status(HttpStatus.FORBIDDEN)
				.body(new Result()) // TODO zainicjować pola obiektu Result zainicjować
		}
	}

	@DeleteMapping("/deleteuser/{name}")
	public ResponseEntity<Result> deleteUser(@PathVariable String name) {
		try {
			repository.deleteByName(name);
			return ResponseEntity
				.status(HttpStatus.OK)
				.body(new Result())
			// obiekt Result należy zainicjować
		}
		catch (Exception e) {
			return ResponseEntity
			.status(HttpStatus.FORBIDDEN)
			.body(new Result()) // obiekt Error mależy zainicjować
		}
	}

	// TODO dodać analogicznie usługi get, put.
	// TODO dodać obsługę wyjątków i zamienić w istniejących usługach (wykorzystać dowolną omówioną metodę)
```

}

4. Klasa Result

```
public class Result {
	private String status;
	private String errCode;
	private String errMsg;
	// TODO uzupełnić gettery, settery, builder, itp.
}
```

5. W miarę potrzeby rozbudować UserRepository

6. Uruchomienie aplikacji
Po uruchomieniu aplikacji należy z przeglądarki wywołać adres:
http://localhost:8080/swagger-ui.html
co spowoduje otwarcie strony UI z listą dostępnych usług.