



## Proounce Solutions

# Java Spring Boot (Duration: 1 Hour)

### Objective

Build a Spring Boot REST API for a simple Employee Leave Management System. This task evaluates your backend development skills including REST API design, Spring Boot structure, validation, business logic, and clean coding practices.

### Requirements

#### 1. Entities

##### *Employee*

Field	Type	Validation
id	Long	Auto-generated
name	String	@NotBlank, minimum 3 characters
department	String	@NotBlank
email	String	@NotBlank, @Email, must be unique

##### *LeaveRequest*

Field	Type	Validation
id	Long	Auto-generated
employeeId	Long	Must exist in Employee table
startDate	LocalDate	@NotNull, must not be in the past
endDate	LocalDate	@NotNull, must be >=



		startDate
reason	String	@NotBlank, length between 10-200 chars
status	Enum	Default: PENDING, allowed values: PENDING, APPROVED, REJECTED



## 2. Endpoints

Method	Endpoint	Description
POST	/employees	Add a new employee
GET	/employees	List all employees
POST	/leaves	Create a new leave request (default status: PENDING)
PUT	/leaves/{id}/approve	Approve a leave request
PUT	/leaves/{id}/reject	Reject a leave request
GET	/leaves?employeeId={id}	List all leaves for a specific employee

### Business Rules (Validation Logic)

#### 1. Employee Validation:

- Email must be unique.
- Reject invalid email format or blank names.

#### 2. Leave Validation:

- startDate must not be before today.
- endDate must be  $\geq$  startDate.
- Reject if employee does not exist or overlapping approved leaves exist.

#### 3. Status Change Rules:

- Log approval/rejection actions to console.
- Disallow re-approval/rejection.

#### 4. API Behavior:

- Return 400 for validation errors, 404 for not found, 409 for conflicts.



### Technical Requirements

- Spring Boot (latest stable version)
- Spring Data JPA
- Database: H2 / MySQL / Oracle (any one)
- Java 17+
- Proper layered architecture (controller, service, repository, model, dto, exception)
- Use validation annotations and global exception handling.
- Return JSON error messages.



### Bonus Points

- Use DTOs for mapping.
- Custom exception classes.
- Unit tests using JUnit/Mockito.
- README with setup and examples.

### Expected Duration

~1 hour (Setup 15m, Logic 30m, Validation & Testing 15m)

### Submission / Demo

1. Push to GitHub or share zip.
2. Run using: `mvn spring-boot:run`
3. Demonstrate APIs via Postman.

### Evaluation Criteria

Category	Description
Spring Boot Setup	Project structure, controllers, configuration
Validation & Business Logic	Rules, error handling, input checks
Database Layer	Entity mapping, JPA queries
Code Quality	Layer separation, readability, naming
Error Handling	HTTP status codes and responses
Extras	DTOs, tests, documentation clarity