The website is an **Employee Management System** for a **Ministry of Education**, focused on validating employee data such as name, position, age, and annual leave, while ensuring consistency and correctness through validation rules.

1. **Validation of Student Data:**
   - **Name**: It should contain only letters.
   - **Date of Birth**: It should be in the past and logically consistent (e.g., a student cannot be 5 years old in university).
   - **ID Number**: It should comply with national or university standards.
   - **Email**: It should be in a valid format (e.g., example@domain.com).
2. **Validation of Teacher Data:**
   - **Name**: It should contain only letters.
   - **ID Number**: It should be valid and recognized.
   - **Academic Qualification**: The teacher should have a suitable academic qualification for the specialization.
   - **Experience**: There should be a specified number of years of teaching experience.
3. **Validation of Manager or Supervisor Data:**
   - **Name**: It should contain only letters.
   - **ID Number**: It should be valid and recognized.
   - **Academic Qualification**: The manager or supervisor should have a suitable academic qualification.
   - **Experience**: The manager or supervisor should have sufficient experience in management or academic supervision.

```
// Employee ID: Must not be null and should have at least 3 characters
@NotNull(message = "Can not be null")
@Size(min = 3, message = "ID must be more than 2 characters")
 id;



// Employee Name: Should contain only letters (no numbers) and should have at least 5 characters
@NotNull(message = "Can not be null")
@Size(min = 5, message = "Name must be more than 4 characters")
@Pattern(regexp = "^[a-zA-Z]+$", message = "Name must contain only letters and cannot contain numbers.")
name;



// Email: Should be a valid email format
@NotNull(message = "Can not be null")
@Email(message = "Must be a valid email format.")
 email;



// Phone Number: Should start with '05' and consist of exactly 10 digits
@NotNull(message = "Can not be null")
@Pattern(regexp = "^05\\d{8}$", message = "Phone number must start with '05' and consist of exactly 10 digits.")



// Age: Should be greater than 25
@NotNull(message = "Can not be null")
@Min(value = 25, message = "Age must be more than 25")
@Digits(integer = 3, fraction = 0, message = "Age must be a valid number.")
private int age;



// Position: Should be either 'supervisor' or 'coordinator'
@NotNull(message = "Can not be null")
@Pattern(regexp = "^(supervisor|coordinator)$", message = "Position must be either 'supervisor' or 'coordinator'
only.")
 position;
```

```
// onLeave: Initially set to false (assert that it's false when not on leave)
@AssertFalse(message = "onLeave must be initially set to false.")
 onLeave;



// Annual Leave: Should be a positive number (at least 1 day)
@NotNull(message = "AnnualLeave cannot be null.")
@Positive(message = "AnnualLeave must be a positive number.")
 annualLeave;



// Hire Date: Should be a past or present date and formatted correctly
@NotNull(message = "Can not be null")
@PastOrPresent(message = "HireDate should be a date in the present or the past.")
@JsonFormat(pattern = "yyyy-MM-dd HH:mm")  // Format date to yyyy-MM-dd HH:mm
 LocalDateTime hireDate;
```