Education System

Student and staff Data Validation

@NotNull(message = "name should not be empty ")
// يجب ان يكون اسم الطالب او الموظف غير فارغ حتى يستطيع التسجيل في النظام وفي حالة كان فارغ سوف تظهر رسالة الخطأ

@Size(min = 4,max = 10 ,message = "name should ber mor than 4 and less than 10")
// يجب ان يكون عدد احرف اسم الطالب اوالموظف اكبر من 4 واقل من10

@Pattern(regexp = "^[a-zA-Z]+$")
// استخدمت Pattren للاسم حتى يقبل فقط الحروف الابجدية
private String name;

@NotNull(message = "id should not be empty ")
// يجب ان يكون Id الطالب او الموظف غير فارغ حتى يستطيع التسجيل في النظام وفي حالة كان فارغ سوف تظهر رسالة الخطأ

@Size(min = 10,max = 10,message = "id length should be 10")
// قمت بتحديد طول الid
private String id;

@Email (message = "invalid email")
// @Email من اجل ان تقبل الايميل المدخل عن طريق الطالب او الموظف بالشكل الصحيح وفي حال كتابة الايميل بغير صيغة الايميل سوف تظهر رسالة خطا
private String email;

@Pattern(regexp = "^05\\d{8}$", message = "Invalid phone number format")

// Explanation of the phone pattern:
//05: Ensures that the phone number starts with "05".
//\d{8}: Matches exactly 8 digits after "05".

```
    private String phone;
@NotNull(message = "BirthDate should be not null")

    @JsonFormat(pattern = "yyyy-MM-dd")

من اجل ان يكون تاريخ الميلاد بهذا الشكل    //
    @Past

حتى يكون تاريخ الميلاد المدخل في الماضي    //

    private LocalDate dateBrith;

    @Positive
يجيب ان يكون مستوى الطالب رقم موجب    //

    private int level;

@NotNull(message = " qualifications should be not null")
يجب ان لايكون المؤهل فارغ//

@Pattern(regexp = "^(Bachelor |Diploma)$")
مؤهل الموظف يجب ان يكون Bachelor او Diploma\\
 private String qualifications;
```

@NotNull(message = "password should not be null")

@Pattern(regexp = "^(?=.*[a-z])(?=.*[A-Z])(?=.*\\\\d)(?=.*[@$!%*?&])[A-Za-z\\\\d@$!%*?&]{8,}$\n")

//Explanation of the password pattern:
//
//(?=.*[a-z]): Positive lookahead assertion for at least one lowercase letter.
//(?=.*[A-Z]): Positive lookahead assertion for at least one uppercase letter.
//(?=.*\d): Positive lookahead assertion for at least one digit.
//(?=.*[@$!%*?&]): Positive lookahead assertion for at least one special character (you can customize the set of special characters).
//[A-Za-z\d@$!%*?&]{8,}: Allows only uppercase letters, lowercase letters, digits, and the specified special characters. The {8,} ensures a minimum length of 8 characters.

private String password;

// يمكن أن تؤدي المعلومات الشخصية غير الصحيحة إلى مشكلات في الاتصال وخطأ في التعرف على الهوية.