**وثيقة التصميم لـ:**

Study Gate

فريق الثالث

إعداد:

|  |  |
| --- | --- |
| رقم القيد | الأسم |
| 2211845408 | عبد الرحمن جمعة الأسطى |
| 2211845977 | محمد بهلول زقيرة |
| 2211846663 | يوسف رجب عصمان |
| 2211846566 | هاشم مفتاح الجهيمي |
| 2211849800 | صهيب صلاح الدين كشير |
| 2211845398 | عبد الرحمن محمد أوحيده |

النسخة 1.0

ملاحظة:

**في حال لم يظهر تنسيق هذا المستند بالشكل الصحيح، يمكنك الاطلاع على النسخة الأصلية المعتمدة من خلال** [**الرابط التالي**](https://universityoftripoli-my.sharepoint.com/:w:/g/personal/a_whida_uot_edu_ly/EanFzJ800r9Cn5AJbR0uFUkBS9nkAKgaIn-vjTwZrixy3g?e=gKNkz7).

28/6/2025

# Introduction

هذه الوثيقة توفر التصميم التفصيلي لنظام "منصة StudyGate". الهدف من النظام هو تسهيل وإدارة جميع العمليات الأكاديمية إلكترونياً، مثل تسجيل الطلاب في المواد الدراسية، إدارة الجداول، رفع الدرجات، وتوفير تقارير الأداء. النظام مصمم لخدمة الطلاب، الأساتذة، والمسؤولين عبر واجهات مخصصة لكل منهم، بهدف استبدال النظام اليدوي القائم وزيادة الكفاءة. توضح هذه الوثيقة الهيكلية التي سيتبعها المطورون لبناء النظام، وتُعد المرجع الأساسي لكتابة الشيفرة البرمجية.

# Data design

هذا القسم يصف هيكل البيانات وقواعد البيانات المستخدمة في النظام.

## Database diagram (ERD)

|  |
| --- |
|  |
| Figure 1: ERD Diagram for the system |

يمكنك استخدام الصور الأخرى لتمثيل مخطط الكائن علاقة مثلا:

المخطط التالي يوضح الكيانات الرئيسية في النظام والعلاقات بينها.

|  |
| --- |
|  |
| Figure 2: ERD Diagram for the system |

## Data Dictionary

يصف هذا القاموس تفاصيل الجداول والحقول في قاعدة بيانات النظام.

**محمد بهلول زقيرة**

### Users Table

يُستخدم هذا الجدول لتخزين البيانات الأساسية لجميع مستخدمي النظام، سواء كانوا طلاباً، معلمين، أو إداريين.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| جدول رقم 1: جدول بيانات المستخدمين | | | | |
| Field Name | Description | Type | Length (bytes) | Notes |
| id | user id | int | 11 | primary key |
| name | User full name | varchar | 100 |  |
| email | user email | varchar | 100 |  |
| password | user password | varchar | 256 | encrypted |
| role | Identifies the user type | tinyint | 1 | 0-student, 1-lecturer, 2- admin |
| Cerate-at |  | TIMEST AMP |  |  |

#### Table Schema

public function up(): void

{

Schema::create('users', function (Blueprint $table) {

$table->id();

$table->string('name');

$table->string('email')->unique();

$table->string('password');

$table->tinyInteger('role');

$table->timestamps();

});

}

### عبدالرحمن الاسطى

### Subject Table

يحتوي على معلومات جميع المواد الدراسية المتاحة في المؤسسة التعليمية.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| جدول رقم 2: جدول بيانات المواد | | | | |
| Field Name | Description | Type | Length (bytes) | Notes |
| id | subject id | int | 11 | primary key |
| course code | رمز المادة الدراسي | varchar | 20 | Unique |
| name | اسم المادة الدراسي | varchar | 255 |  |
| credits | عدد الساعات المعتمدة | int | 2 |  |
| prerequisite\_id | متطلب سابق | int | 11 | Foreign key to courses table, nullable |

#### Table schema

public function up (): void

{

Schema:create('subjects', function (Blueprint $table) {

$table->id ();

$table->string ('name', 100)->nullable ();

$table->text('description')->nullable ();

});

}

**عبدالرحمن أوحيده**

### Subject Offers Table

يحتوي على معلومات جميع المواد الدراسية المتاحة في الفصل الدراسي الحالي.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| جدول رقم 3: جدول المواد المتوفرة | | | | |
| Field Name | Description | Type | Length (bytes) | Notes |
| id | subject id | int | 11 | primary key |
| subject\_id | رمز المادة الدراسي | int | 11 | Foreign key |
| teacher\_id | معرف المعلم المسؤول عن المادة | int | 11 | Foreign key |
| semester\_id | معرفف الفصل الدراسي الذي طرحت فيه المادة | Int | 11 | Foreign key |

#### Table schema

public function up(): void

{

Schema:create('subject\_offers', function (Blueprint $table) {

$table->id ();

$table->foreignId('subject\_id')->nullable()->constrained('subjects');

$table->foreignId('teacher\_id')->nullable()->constrained('users');

$table->foreignId('semester\_id')->nullable()->constrained('semesters');

});

}

**صهيب كشير**

### Class schedules Table

يحتوي على معلومات مواعيد و توقيت المحاضرات الدرسية

جدول رقم4 : جدول المحاضرات

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Description | Type | Length (bytes) | Notes |
| id | رقم فريد لكل جدول محاضرة. | |  | | --- | | INT | |  | | 11 | primary key |
| subject offer\_id | شير إلى العرض الدراسي المرتبط بهذه الحصة (مادة × فصل × عضو هيئة تدريس). | |  | | --- | | INT | |  | | 11 | Foreign Key |
| |  | | --- | | day of  \_week | |  | | ليوم الذي تُعقد فيه المحاضرة | |  | | --- | | ENUM | |  | |  |  |
| |  | | --- | | session | |  | | رقم الحصة في اليوم | |  | | --- | | ENUM | |  | |  |  |
| room | رقم أو اسم القاعة التي تُعقد فيها المحاضرة. | VARCHAR | 50 |  |

#### Table schema

public function up(): void { Schema::create('class\_schedules', function (Blueprint $table) { $table->id();

$table->foreignId('subject\_offer\_id')->constrained('subject\_offers');  
  
 $table->enum('day\_of\_week', [  
 'Saturday', 'Sunday', 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday'  
 ]);  
  
 $table->enum('session', [  
 'First', 'Second', 'Third', 'Fourth', 'Fifth'  
 ]);  
  
 $table->string('room', 50)->nullable();  
  
 $table->timestamps();  
});

### Exam schedules Table

يحتوي على معلومات مواعيد و توقيت المحاضرات الدرسية

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| جدول رقم 1: جدول الامتحانات | | | | |
| Field Name | Description | Type | Length (bytes) | Notes |
| id | رقم فريد لكل جدول محاضرة. | INT | 11 | primary key |
| subject offer\_id | شير إلى العرض الدراسي المرتبط بهذه الحصة (مادة × فصل × عضو هيئة تدريس). | INT | 11 |  |
| day of  \_week | ليوم الذي تُعقد فيه الامتحان | ENUM |  |  |
| exam\_date | تاريخ عقد الامتحان | date |  |  |
| session | فترة الامتحان (“الاولي “و “التانية” و “التالته “) | ENUM |  | encrypted |
| room | رقم أو اسم القاعة التي تُعقد فيها الامتحان. | VARCHAR | 50 | 0-student, 1-lecturer, 2- admin |

#### Table Schema

public function up (): void {Schema::create ('class\_schedules', function (Blueprint $table) { $table->id();

$table->foreignId('subject\_offer\_id')->constrained('subject\_offers');  
  
 $table->enum('day\_of\_week', [  
 'Saturday', 'Sunday', 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday'  
 ]);  
  
 $table->enum('session', [  
 'First', 'Second', 'Third', 'Fourth', 'Fifth'  
 ]);  
  
 $table->string('room', 50)->nullable();  
  
 $table->timestamps();  
});

### عبدالرحمن أوحيده

### Grade records

يستخدم لتسجيل درجات الطلاب في المواد التي سجّلوا فيها.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| جدول رقم 1: جدول بيانات المستخدمين | | | | |
| Field Name | Description | Type | Length (bytes) | Notes |
| id | المعرّف الرقمي لسجل الدرجة | int | 11 | primary key |
| enrollment\_id | معرّف عملية التسجيل المرتبطة بالدرجة | int | 11 | Foreign Key |
| grade | الدرجة التي حصل عليها الطالب | float |  |  |
| updated\_at | اخر وقت وتاريخ تم فيه تحديث الدرجة | timestamp |  | القيمة الافتراضية هي الوقت الحالي |

#### Table Schema

public function up(): void

{

Schema::create('grade\_records', function (Blueprint $table) {

$table->id();

$table->foreignId('enrollment\_id')->nullable()->constrained('enrollments');

$table->float('grade')->nullable();

$table->timestamp('updated\_at')->useCurrent();

});

}

### هاشم الجهيمي

### Enrollments Table

يسجل الطلاب في المواد المطروحة في فصل دراسي معين.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| جدول رقم 1: جدول بيانات المستخدمين | | | | |
| Field Name | Description | Type | Length (bytes) | Notes |
| id | المعرفف الرقمي لعملية التسجيل | int | 11 | primary key |
| student\_id | المعرف الرقمي لعملية التسجيل | int | 11 | Foreign Key |
| subject\_offer\_id | معرفف المادة المطروحة التي تم التسجيل فيها | int | 11 | Foreign Key |
| enrolled\_at | وقت وتاريخ تسجيل الطالب في المادة | int |  |  |

#### Table Schema

public function up (): void

{

Schema:create('enrollments'', function (Blueprint $table) {

$table->id ();

$table->foreignId('student\_id')->nullable()->constrained('users');

$table->foreignId('subject\_offer\_id')->nullable()->constrained('subject\_offers');

$table->timestamp('enrolled\_at')->useCurrent();

});

}

**يوسف عصمان**

### Semesters Table

يحتوي على معلومات حول الفصول الدراسية المختلفة، مثل تواريخ البدء والانتهاء.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| جدول رقم 1: جدول بيانات المستخدمين | | | | |
| Field Name | Description | Type | Length (bytes) | Notes |
| id | المعرفف الرقمي للفصل الدراسي | int | 11 | primary key |
| name | اسم الفصل الدراسي | varchar | 100 |  |
| is\_active | يحدد إذا كان الفصل الدراسي هو الحالي | tinyint | 100 | 0-active, 1-none, |
| start\_date | تاريخ بدء الفصل الدراسي | date | 256 |  |
| end\_date | تاريخ انتهاء الفصل الدراسي | date | 1 |  |
| enrollment\_open | يحدد إذا كان التسجيل مفتوحاً | tinyint |  | 0-active, 1-none, |

#### Table Schema

public function up(): void

{

Schema::create('users', function (Blueprint $table) {

$table->id();

$table->string('name');

$table->string('email')->unique();

$table->string('password');

$table->tinyInteger('role');

$table->timestamps();

});

}

# Architectural and component-level design

## 3.1 System Structure

تم اختيار نمط **"المتحكم-النموذج-العرض" (Model-View-Controller - MVC)**

ليكون الهيكل العام للنظام. هذا الاختيار يتماشى مع متطلبات النظام للجودة، خصوصاً.

**لقابلية الصيانة (Maintainability)** و**المرونة (Flexibility)**، حيث يسهل فصل منطق العمل عن واجهة المستخدم من إصلاح الأعطال وإضافة وظائف جديدة بسهولة في المستقبل.

* **Model:** مسؤول عن التعامل مع قاعدة بيانات MySQL وتمثيل البيانات (مثل User, Course) وتطبيق قوانين العمل (مثل منع تسجيل مادة دون متطلب سابق ).
* **View:** مسؤول عن عرض واجهات المستخدم (UI) التي يتفاعل معها الطالب أو الأستاذ أو المسؤول.
* **Controller:** يستقبل الطلبات من الـ View (مثل طلب تسجيل مادة) وينسق بين الـ Model والـ View لتنفيذ الطلب.

### 3.1.1 Architecture diagram

|  |
| --- |
|  |
| Figure 3: Architecture diagram |

عبدالرحمن أوحيدة

## 3.2 Description for Component “Enrollment Component”

هذا القسم يصف بالتفصيل المكون المسؤول عن وظيفة "تنزيل وإسقاط المواد".

### 3.2.1 Processing narrative (PSPEC)

هذا المكون مسؤول عن إدارة جميع العمليات المتعلقة بتسجيل الطلاب في المواد الدراسية وإسقاطها. من مسؤولياته:

* عرض المواد المتاحة للطالب حسب تخصصه وخطته الدراسية.
* التحقق من صحة عملية التسجيل عبر تطبيق قوانين العمل، مثل التحقق من عدم وجود تعارض في الجدول الزمني والتأكد من استيفاء المتطلبات السابقة.
* معالجة طلبات إسقاط المواد بعد موافقة الجهة المعنية (إن لزم الأمر).

### 3.2.2 Component interface description.

**Inputs**:

* student\_id: معرف الطالب الذي يقوم بالعملية.
* course\_id: معرف المادة المراد تسجيلها أو إسقاطها.
* action\_type: نوع العملية ('add' أو 'drop').

**Outputs**:

* Status: (نجاح/ فشل).
* message: رسالة نصية تصف نتيجة العملية (مثال: "تم تسجيل المادة بنجاح"، "يوجد تعارض في الجدول الزمني").

### 3.2.3 Component processing detail

algorithmic description:

* أستقبل طلب إضافة مادة مع student\_id ,course\_id.
* التحقق من أن الفصل الدراسي الحالي مفتوح ومفعّل. إذا لم يكن كذلك، إرجاع رسالة خطأ.
* التحقق من أن الطالب لم يتجاوز الحد الأقصى للساعات المسموح بها.
* التحقق من أن المادة المختارة ليس لها تعارض في الجدول الزمني مع المواد المسجلة مسبقاً للطالب. إذا وجد تعارض، إرجاع رسالة خطأ.
* التحقق من أن الطالب قد استوفى المتطلبات السابقة للمادة (إن وجدت).
* إذا نجحت جميع عمليات التحقق، قم بإنشاء سجل جديد في جدول Enrollment بالبيانات المدخلة.
* إرجاع رسالة نجاح.

**صهيب كشير**

## 3.3 Description for Component “class schedules”

سؤول عن إدارة وتخزين معلومات الجداول الزمنية للمحاضرات الأكاديمية.

### 3.1 Processing narrative (PSPEC)

تنظيم وحفظ الجداول الدراسية الخاصة بالمقررات الأكاديمية، مع ضمان عدم وجود تعارضات في المواعيد للطلاب أو أعضاء هيئة التدريس، وربط كل محاضرة بالمادة والأستاذ والقاعة واليوم والوقت المناسب.

### Component interface description.

**Inputs**:

* المعرف :`course\_id`
* المعلّم : `instructor\_id`
* القاعة : `classroom\_id`
* يوم الأسبوع : day\_of\_week
* وقت البداية : start\_time
* وقت النهاية : end\_time
* الفصل الدراسي : semester

**Outputs**:

* تأكيد نجاح العملية أو عرض أخطاء التعارض.
* تمكين عرض الجدول للطلاب والأساتذة حسب بياناتهم

### Component processing detail

algorithmic description:

**الخطوات:**

1. **التحقق من صحة البيانات**
   1. التأكد من إدخال جميع الحقول المطلوبة وصحة تنسيق الوقت.
   2. التأكد أن وقت البدء أصغر من وقت الانتهاء.
2. **التحقق من تعارض المعلم**
   1. هل يوجد جدول آخر لنفس المعلم في نفس اليوم وفي نفس الوقت؟
   2. إذا **نعم**، إرجاع خطأ "تعارض في جدول المعلم".
3. **التحقق من تعارض القاعة**
   1. هل القاعة محجوزة في نفس الوقت واليوم؟
   2. إذا **نعم**، إرجاع خطأ "القاعة مشغولة".
4. **(اختياري) التحقق من تعارض الطلاب**
   1. فحص الطلاب المسجلين في المادة إذا كان لديهم محاضرات أخرى في نفس الوقت.
   2. إذا **نعم**، إرجاع خطأ "تعارض في جدول الطالب".
5. **إدراج السجل في قاعدة البيانات**
   1. حفظ الجدول في جدول class\_schedules.
6. **إرجاع استجابة النجاح**

**الطالب : محمد بهلول زقيرة**

## 3.4 Description for Component “Login”

مسؤول عن التحقق من بيانات اعتماد المستخدمين وتمكينهم من الوصول إلى النظام بناءً على صلاحياتهم (طالب، أستاذ، مشرف...)

### 3.4.1 Processing narrative (PSPEC)

التحقق من صحة البريد الإلكتروني وكلمة المرور المُدخلة، وتحديد صلاحية المستخدم، ثم إنشاء جلسة دخول وتوجيهه إلى الواجهة المناسبة، أو إرجاع رسالة خطأ عند فشل التحقق

### 3.4.2 Component interface description.

**Inputs**:

* البريد الإلكتروني: email
* كلمة المرور: password

**Outputs**:

* نجاح تسجيل الدخول وإعادة التوجيه إلى الواجهة المناسبة
* رسالة خطأ في حال كانت بيانات الدخول غير صحيحة

### 3.4.3 Component processing detail

**الخطوات:**

**1.التحقق من أن جميع الحقول مطلوبة وغير فارغة**

**2.البحث عن المستخدم باستخدام البريد الإلكتروني**

**3.مقارنة كلمة المرور المدخلة مع الكلمة المخزنة**

**4.إذا كانت المطابقة صحيحة:**

1. **تحديد دور المستخدم**
2. **إنشاء جلسة دخول**
3. **توجيه المستخدم إلى الواجهة المناسبة حسب الدور**

**5.إذا كانت المطابقة غير صحيحة:**

1. **عرض رسالة "بيانات الدخول غير صحيحة"**
2. **منع الوصول للنظام**

## 3.5 Description for Component “User Management”

مسؤول عن إدارة حسابات المستخدمين داخل النظام، بما يشمل إضافة مستخدمين جدد، تعديل بياناتهم، تعيين الصلاحيات، وتفعيل أو تعطيل الحسابات. هذه الوحدة تستخدم عادة من قبل المسؤول أو المشرف

### 3.5.1 Processing narrative (PSPEC)

عند استدعاء هذا المكون، يتم تنفيذ عمليات على المستخدمين حسب الإجراء المطلوب (إضافة، تعديل، حذف، تفعيل، إلخ).  
 يتم التحقق من صلاحية المستخدم الذي يطلب العملية، ثم يتم تنفيذ التغيير المطلوب على بيانات المستخدم المستهدف، مع التأكد من صحة البيانات وإرجاع تأكيد أو رسالة خطأ

### 3.5.2 Component interface description

**المدخلات (Inputs):**

* معرف المستخدم (User ID)
* البيانات الشخصية (الاسم، البريد الإلكتروني، كلمة المرور، الدور)
* نوع العملية (إضافة، تعديل، حذف، تفعيل/تعطيل)
* صلاحيات المستخدم المنفّذ (مشرف، مدير)

**المخرجات (Outputs):**

* رسالة نجاح تؤكد العملية (مثل "تمت إضافة المستخدم بنجاح")
* أو رسالة خطأ توضح سبب الفشل (مثل "المستخدم موجود مسبقًا" أو "لا تملك صلاحية")

### 3.5.3 Component processing detail

**الخطوات:**

1. التحقق من صلاحيات المستخدم الذي يقوم بالعملية (مشرف أو مدير).
2. التحقق من صحة البيانات المُدخلة (مثلاً: البريد الإلكتروني بصيغة صحيحة، كلمة مرور قوية...).
3. تنفيذ العملية المطلوبة حسب نوعها:
   1. **إضافة مستخدم:**
      1. التأكد من أن البريد غير مستخدم مسبقًا
      2. حفظ البيانات الجديدة في قاعدة البيانات
   2. **تعديل بيانات مستخدم:**
      1. التأكد من أن المستخدم موجود
      2. تحديث البيانات المحددة فقط
   3. **تفعيل / تعطيل حساب:**
      1. تحديث حالة المستخدم في قاعدة البيانات
   4. **حذف مستخدم:**
      1. إزالة السجل أو وضعه كـ"محذوف" منطقيًا (soft delete)
4. إرجاع نتيجة العملية برسالة نجاح أو فشل

**الطالب عبد الرحمن الاسطي -2211845408**

### 3.6 Component Subject management

مكوّن **إدارة المواد** مسؤول عن تنفيذ جميع الوظائف المتعلقة بإدارة المواد الدراسية داخل النظام. وتشمل مسؤولياته ما يلي:

* إضافة مادة جديدة مع البيانات المرتبطة مثل (اسم المادة، الرمز، عدد الساعات، المتطلبات).
* تعديل بيانات مادة موجودة مسبقًا.
* حذف مادة من النظام.
* عرض قائمة المواد الحالية مع إمكانية البحث والتصفية.
* التأكد من تكامل البيانات مثل عدم تكرار رموز المواد وربط المتطلبات بشكل صحيح.
* التفاعل مع قاعدة البيانات لجلب وتحديث بيانات المواد.

هذا المكوّن يشكّل جزءًا أساسيًا في النظام الأكاديمي، ويُستخدم من قبل مسؤولي النظام أو الأقسام المختصة بإدارة المناهج الدراسية.

### 3.6.1 Processing narrative (PSPEC)

مكوّن "إدارة المواد" في Laravel مسؤول عن تنفيذ العمليات الأساسية على جدول المواد (courses) مثل:

* **إضافة مادة جديدة** باستخدام Controller وModel مع التحقق من صحة البيانات.
* **تعديل وحذف مادة** عبر واجهات CRUD.
* **عرض قائمة المواد** مع دعم التصفية والبحث.
* يستخدم Eloquent ORM للتعامل مع قاعدة البيانات.
* يتفاعل مع واجهات المستخدم (Blade) لعرض البيانات وإدخالها.
* يتكامل مع مكوّنات أخرى مثل الامتحانات عبر العلاقات (مثلاً: المادة تحتوي على امتحانات).

### 3.6.2 Component interface description

**المداخل (Inputs):**

* بيانات المادة من المستخدم عبر الفورم:
  + course\_name (اسم المادة)
  + course\_code (رمز المادة)
  + credit\_hours (عدد الساعات)
  + prerequisites (اختياري: مواد متطلبة)
* أوامر المستخدم مثل: إضافة، تعديل، حذف، بحث.

**المخرجات (Outputs):**

* قائمة المواد المستخرجة من قاعدة البيانات (باستخدام Eloquent).
* رسائل حالة (نجاح/فشل) تُعرض في الواجهة.
* إعادة توجيه (Redirect) حسب العملية: إلى قائمة المواد، أو إلى صفحة التعديل.
* استجابة JSON (عند استخدام AJAX أو API).

### 3.6.3 Component processing detail

* يبدأ تنفيذ المعالجة عندما يُرسل المستخدم طلبًا من خلال واجهة النظام (مثلاً: إضافة أو تعديل مادة).
* يتم توجيه الطلب إلى الـ Controller المناسب لمعالجته.
* يتم التحقق من صحة البيانات المدخلة (مثل التأكد من أن اسم المادة موجود وأن الرمز غير مكرر).
* بعد التحقق، يتم تمرير البيانات إلى النموذج (Model) الذي يتكفل بحفظها أو تعديلها في قاعدة البيانات.
* بعد تنفيذ العملية (إضافة، تعديل، حذف)، يتم إرسال نتيجة العملية إلى الواجهة لإظهار رسالة نجاح أو تنبيه.
* في حالة عرض قائمة المواد، يتم استرجاع البيانات من قاعدة البيانات وعرضها باستخدام الواجهة.
* يتكامل هذا المكوّن مع مكوّنات أخرى مثل "إدارة الامتحانات" في حالة الحاجة لربط المادة بامتحانات أو متطلبات.

**3.7 Component exam management**

### 3.7.1 Processing narrative (PSPEC)

مكوّن **إدارة الامتحانات** مسؤول عن تنظيم جميع العمليات المتعلقة بامتحانات المواد داخل النظام.  
 تشمل وظائفه الرئيسية:

* إضافة امتحان جديد مرتبط بمادة محددة.
* تعديل بيانات امتحان (مثل التاريخ، الوقت، المكان).
* حذف امتحان من قاعدة البيانات.
* عرض قائمة الامتحانات المجدولة.
* التحقق من عدم تعارض مواعيد الامتحانات.
* ربط كل امتحان بمادة دراسية باستخدام العلاقة المناسبة.

يُستخدم هذا المكوّن من قبل الإدارة الأكاديمية لتنظيم جداول الامتحانات وتحديثها.

### 3.7.2 Component interface description

**المدخلات (Inputs):**

* بيانات الامتحان: اسم المادة، التاريخ، الوقت، القاعة أو المكان.
* أوامر المستخدم: إضافة، تعديل، حذف، بحث عن امتحان.

**المخرجات (Outputs):**

* جدول الامتحانات المعروضة للمستخدم.
* رسائل حالة بعد تنفيذ العمليات (نجاح، فشل، خطأ في الإدخال).
* تنبيهات عند وجود تعارض في المواعيد.

### 3.7.3 Component processing detail

* يستقبل المستخدم طلبًا (مثل إضافة امتحان) من الواجهة.
* يتم توجيه الطلب إلى الـ Controller المناسب.
* تُجرى عملية التحقق من صحة البيانات (مثل التواريخ والمكان).
* تُرسل البيانات إلى النموذج (Model) لحفظها في جدول exams.
* في حالة التعديل أو الحذف، يتم تحديد الامتحان ومعالجته حسب الطلب.
* تعرض الواجهة الرسائل المناسبة وتُحدّث قائمة الامتحانات.
* المكوّن يدعم عرض الامتحانات بترتيب زمني وتصفيتها حسب المادة أو التاريخ.

**الطالب : يوسف عصمان**

**3.8 Component SemesterManagementComponent**

هذا القسم يصف بالتفصيل المكون المسؤول عن إدارة الفصول الدراسية.

### 3.8.1 Processing narrative (PSPEC)

هذا المكون مسؤول عن إدارة وظائف فتح فصل دراسي جديد والتحكم في الفصول الدراسية الحالية. من مسؤولياته:

* عرض قائمة الفصول الدراسية الحالية وحالتها (مفعلة / غير مفعلة).
* بدء فصل دراسي جديد بإدخال تاريخ البدء والانتهاء.
* تفعيل فصل معين وإلغاء تفعيل الآخرين تلقائيًا.
* فتح أو إغلاق باب التسجيل للطلاب

### 3.8.2 Component interface description

**Inputs:**

* semester\_name: اسم الفصل الدراسي الجديد.
* start\_date: تاريخ بداية الفصل.
* end\_date: تاريخ نهاية الفصل.
* is\_active: حالة التفعيل.
* enrollment\_open: حالة فتح باب التسجيل.

**Outputs:**

* Status: نجاح / فشل
* message: رسالة توضح النتيجة مثل "تم إنشاء الفصل الدراسي بنجاح" أو "يوجد فصل مفعّل حاليًا".

### 3.8.3 Component processing detail

**algorithmic description:**

* التحقق من وجود فصل دراسي آخر مفعّل.
* إذا لم يوجد، يقوم بإنشاء فصل جديد ويضبطه كمفعّل.
* تعطيل أي فصل دراسي آخر مفعّل مسبقًا.
* حفظ بيانات الفصل من اسم وتواريخ.
* تمكين أو تعطيل التسجيل حسب اختيار المسؤول.

**هاشم الجهيمي**

**3.9** Component Grade Management

هذا القسم يصف بالتفصيل المكون المسؤول عن وظائف "إدخال، معالجة، ونشر الدرجات" الأكاديمية.

### 3.9.1 Processing narrative (PSPEC)

هذا المكون يمكّن أعضاء هيئة التدريس من إدارة درجات الطلاب بشكل كامل ودقيق. تشمل مسؤولياته الأساسية ما يلي:

* **إدخال الدرجات:** توفير واجهة لأعضاء هيئة التدريس لإدخال أو تعديل درجات الطلاب لمختلف عناصر التقييم (مثل الاختبارات الفصلية، الواجبات، الاختبارات النهائية).
* **معالجة الدرجات:** حساب المجموع الكلي أو النهائي لدرجات الطالب في المادة تلقائيًا بناءً على أوزان عناصر التقييم المحددة مسبقًا في توصيف المادة.
* **التحقق من الصحة:** التأكد من أن الدرجات المدخلة تقع ضمن النطاق المسموح به (مثلاً، من 0 إلى 100) وتطبيق القواعد الخاصة بالدرجات.
* **نشر الدرجات:** السماح بنشر الدرجات النهائية للطلاب بعد اعتمادها رسميًا، لتصبح مرئية لهم عبر بواباتهم الخاصة.
* **التحكم في الوصول:** ضمان أن عضو هيئة التدريس لا يمكنه رصد أو تعديل درجات إلا للمواد المسندة إليه والطلاب المسجلين فيها.

### 3.9.2 Component interface description

**Inputs**:

* instructor\_id المعرف الفريد لعضو هيئة التدريس الذي يقوم بالعملية.
* section\_id المعرف الفريد للشعبة الدراسية.
* student\_id المعرف الفريد للطالب المراد رصد درجته.
* grade\_item مُعرف عنصر التقييم
* Score الدرجة الرقمية التي حصل عليها الطالب.
* action\_type نوع العملية

**Outputs**:

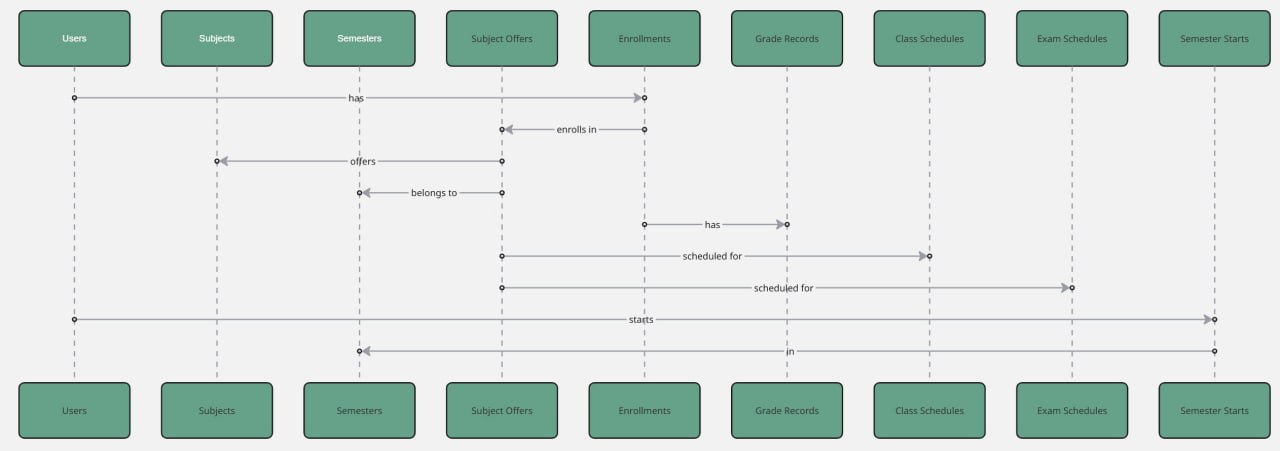
* Status حالة العملية، وتكون إحدى القيمتين: (نجاح) أو (فشل).
* Message رسالة نصية تصف نتيجة العملية (مثال: "تم رصد الدرجة بنجاح"، "عضو هيئة التدريس غير مصرح له بهذه العملية"، "الدرجة المدخلة غير صالحة").
* updated\_record (اختياري): كائن يحتوي على تفاصيل الدرجة المحدثة.

### 3.9.3 Component processing detail

algorithmic description:

* استقبال طلب رصد درجة مع المدخلات
* **التحقق من الصلاحيات (Authorization):** التأكد من أن instructor\_id هو بالفعل عضو هيئة التدريس المسند له تدريس الشعبة section\_id إذا لم يكن كذلك، يتم إرجاع رسالة خطأ "غير مصرح به".
* **التحقق من تسجيل الطالب:** التأكد من أن الطالب student\_id مسجل رسميًا في الشعبة section\_id.
* **التحقق من صلاحية الدرجة:** التأكد من أن قيمة score المدخلة تقع ضمن النطاق الصحيح (مثلاً، بين 0 ودرجة التقييم القصوى).
* **التحقق من فترة الرصد:** التأكد من أن عملية إدخال أو تعديل الدرجات مسموح بها حاليًا وفقًا للتقويم الأكاديمي.
* تنفيذ الرصد:
  + البحث عن سجل درجة موجود للطالب student\_id في عنصر التقييم grade\_item.
  + إذا كان السجل موجودًا، يتم تحديث (UPDATE) قيمة الدرجة.
  + إذا لم يكن السجل موجودًا، يتم إنشاء سجل جديد (INSERT) في جدول الدرجات (Grades Table).
* **تحديث المجموع (اختياري):** بعد كل عملية إدخال، يمكن إعادة حساب المجموع التراكمي للطالب في المادة.
* إرجاع رسالة نجاح.

### 3.3.1 Interaction Diagrams



# 4.0 User interface design

A description of the user interface design of the software is presented.

## 4.1 Description of the user interface

سيتم تصميم واجهة مستخدم بسيطة وواضحة (Usable) تركز على سهولة الوصول للوظائف. عند تسجيل الدخول، سيتم توجيه المستخدم إلى لوحة تحكم تعرض له المهام والإشعارات المتعلقة بدوره. على سبيل المثال، سيرى الطالب إشعاراً عند فتح فصل دراسي جديد

### 4.1.1 Screen images

|  |
| --- |
|  |
| Figure 4: صفحة إضافة المواد |
|  |
| Figure 5: عرض الفصول |
|  |
| Figure 6: الحدول الدراسي |

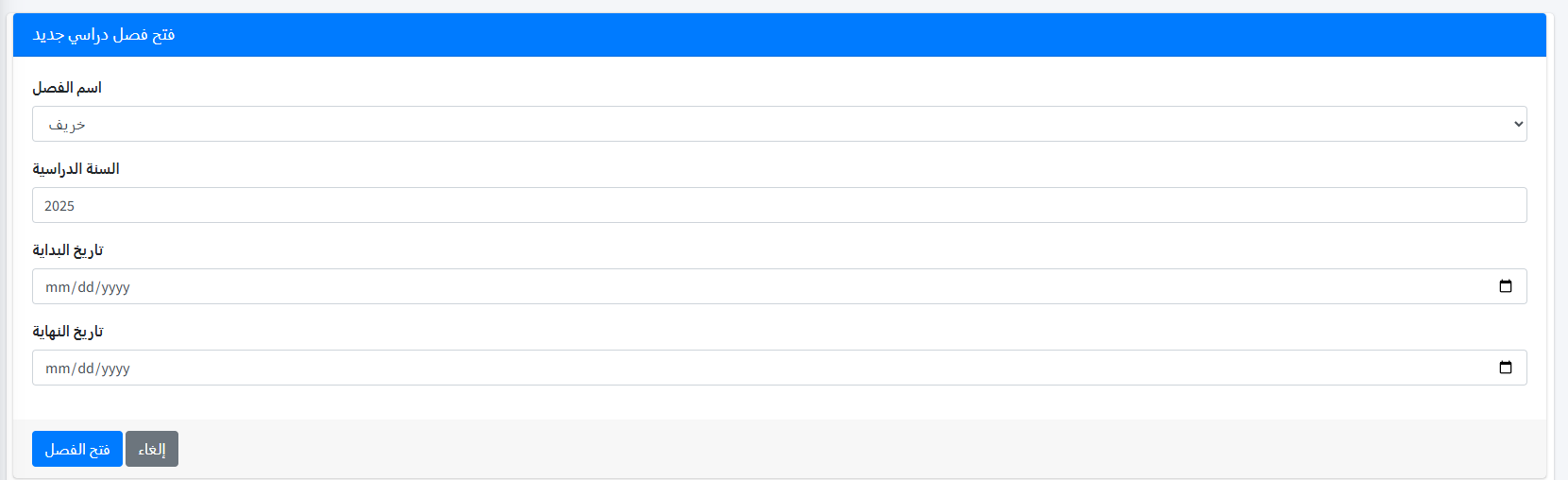


Figure 7:بدء سيسمتر جديد



Figure 8:عرض الطلبة الموجودين في النظام

## 4.2 Interface design rules

لضمان تجربة مستخدم ممتازة، سيتم اتباع القواعد الذهبية الثمانية لتصميم الواجهات، ومنها:

* **السعي نحو التناسق (Strive for consistency):** استخدام نفس الألوان والأيقونات والأزرار في جميع الشاشات.
* **تقديم تغذية راجعة مفيدة (Offer informative feedback):** عرض رسائل واضحة بعد كل عملية، مثل "تم الحفظ بنجاح"
* **صميم حوارات لإغلاق العمليات (Design dialog to yield closure):** وجود بداية ووسط ونهاية واضحة للمهام (مثل عملية التسجيل متعددة الخطوات).

## 4.3 Components available

GUI components available for implementation are noted.

## 4.4 UIDS description

The user interface development system is described. (You might not have one for your project.)

# 5.0 Login interface design

هذا الجزء يشرح الواجهة اللي راح يتفاعل معاها المستخدم. الهدف إنه يكون التصميم بسيط، واضح، وسهل الاستخدام

## 5.1 Description of the Login interface

واجهة تسجيل الدخول تم تصميمها بشكل بسيط. تنقسم الشاشة إلى قسمين:

الجهة اليسرى تحتوي على شعار أكاديمي بشكل قبعة التخرج مع خلفية زرقاء وشعار تحفيزي:  
 "Your smart gateway to academic success"

الجهة اليمنى فيها نموذج لتسجيل الدخول فيه:

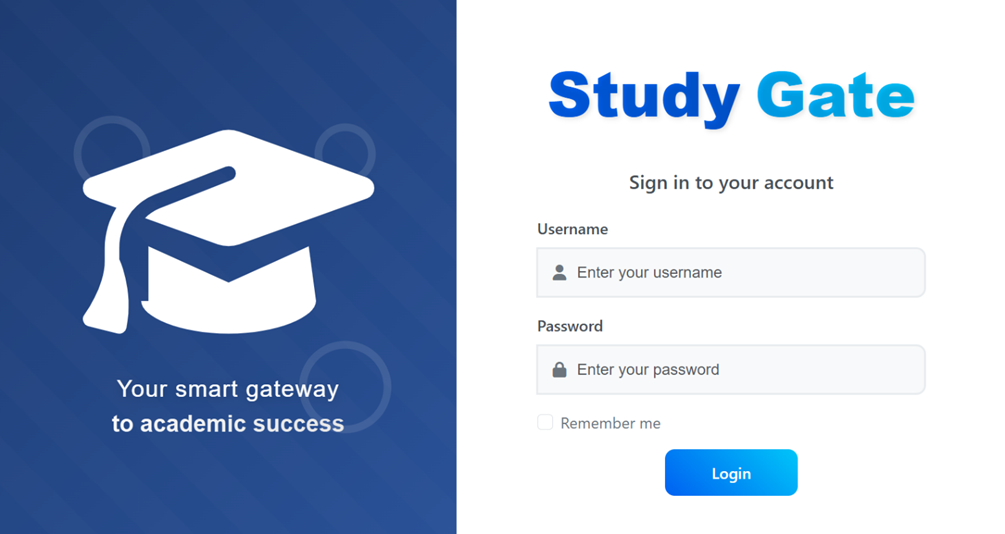
حقل اسم المستخدم

حقل كلمة المرور

خيار "تذكرني"

زر تسجيل الدخول بلون أزرق متدرج

### 5.1.1 Screen images



# 5.2 Interface design rules

تم اتباع المبادئ التالية في تصميم الواجهة

التناسق (Consistency)

إعطاء المستخدم تحكم واضح (User Control)

تقليل عبء الذاكرة (Reduce Memory Load)

وضوح الواجهة (Clarity of Interface)

التعامل مع الأخطاء بشكل سليم

توفير اختصارات للمستخدم المتقدم

استجابة فورية لأي إجراء من المستخدم

دعم قابلية الاسترجاع والتراجع

## 5.3 Components available

* Text Fields
* Password Fields
* Buttons (Login)
* Checkboxes (Remember Me)
* Icons داخل الحقول لتوضيح النوع (مثل قفل واسم مستخدم)

## 5.4 UIDS description

## HTML5 / CSS3

* **JavaScript / Bootstrap**
* التصميم متجاوب (Responsive) ويشتغل على الشاشات المختلفة

**الطالب : عبد الرحمن جمعة الاسطي**

# 6.0 subject management interface design

هذا القسم يصف تصميم الواجهة الخاصة بإدارة المواد في النظام، ويشمل وصف الشاشات، القواعد التصميمية، والمكوّنات المستخدمة لتوفير تجربة مستخدم واضحة ومنظمة.

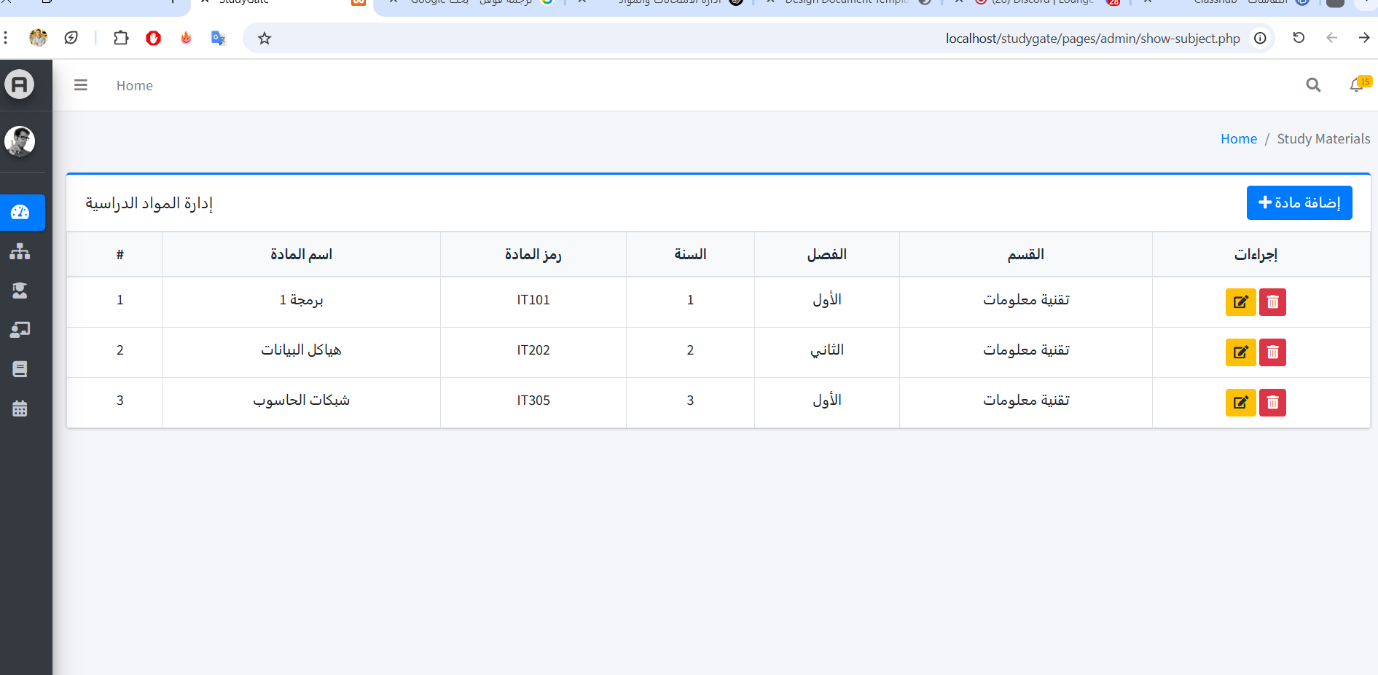
## 6.1 Description of the Subject management interface

واجهة إدارة المواد تمكّن المستخدم من تنفيذ عمليات متعلقة بالمواد الدراسية مثل الإضافة، التعديل، الحذف، والعرض.  
 تم تصميم الواجهة بطريقة بسيطة ومنظمة تضمن سهولة الاستخدام.

**الوظائف الرئيسية المتاحة في الواجهة:**

* عرض قائمة المواد (مع جدول يحتوي على الاسم، الرمز، عدد الساعات، المتطلبات).
* زر لإضافة مادة جديدة.
* زر تعديل لكل مادة.
* زر حذف.
* شريط بحث لتصفية المواد حسب الاسم أو الرمز.

### 6.1.1 Screen images



# 6.2 Interface design rules

* استخدام تصميم متناسق وألوان مريحة للعين.
* إبراز الأزرار الأساسية (إضافة، حفظ، تعديل) بلون واضح.
* استخدام جداول واضحة ذات رؤوس معبرة.
* تنبيهات واضحة عند حدوث خطأ أو نجاح العملية.
* الاستجابة السريعة (Responsive) مع إمكانية العمل على الهاتف والكمبيوتر.
* التأكد من أن جميع النماذج تحتوي على توجيهات Placeholder لتسهيل الإدخال.
* استخدام رموز (Icons) مع الأزرار لسهولة الفهم.

## 6.3 Components available

* **Form Elements:** حقول إدخال للنصوص، قوائم منسدلة، أزرار إرسال/إلغاء.
* **Data Table:** جدول لعرض قائمة المواد.
* **Modals:** نوافذ منبثقة لإضافة أو تعديل مادة بدون مغادرة الصفحة.
* **Alerts & Notifications:** رسائل فورية تُعرض بعد العمليات (نجاح، خطأ).
* **Search Bar:** حقل للبحث داخل جدول المواد.
* **Validation Feedback:** عرض ملاحظات الأخطاء تحت الحقول مباشرةً.

# 6.0 exam management interface design

يتناول هذا القسم تصميم واجهة المستخدم الخاصة بوظيفة إدارة الامتحانات، بما يشمل عرض وتعديل وإضافة جداول الامتحانات للمقررات المختلفة، مع التركيز على سهولة الاستخدام والوضوح

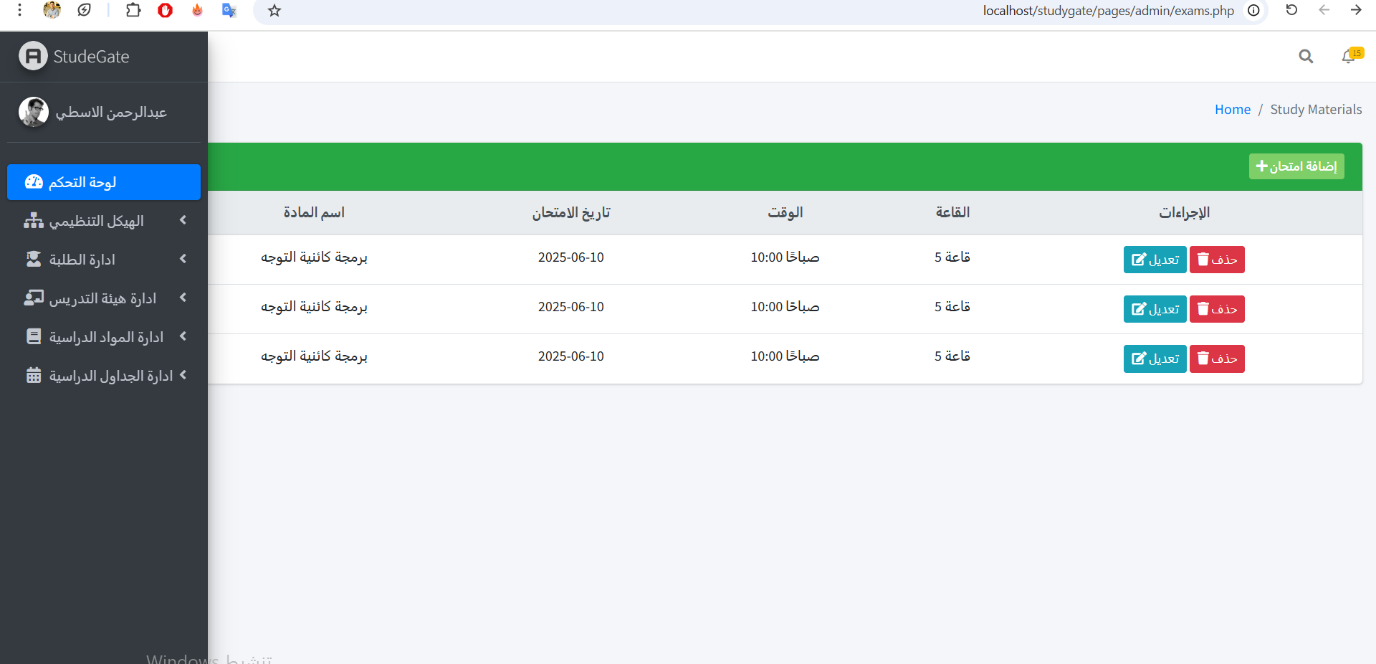
## 6.1 Description of the exam management interface

تم تصميم واجهة إدارة الامتحانات لتُمكّن المستخدم (مثل المشرف الأكاديمي) من تنفيذ المهام التالية:

* **عرض جدول الامتحانات**: يعرض قائمة بجميع الامتحانات المجدولة مع تفاصيل مثل اسم المادة، التاريخ، الوقت، والمكان.
* **إضافة امتحان جديد**: عبر نموذج يتضمن اختيار المادة، تحديد التاريخ والوقت والمكان.
* **تعديل امتحان موجود**: مع إمكانية تحديث البيانات بسهولة.
* **حذف امتحان**: من خلال زر مخصص بجانب كل صف.
* **بحث وتصفية**: لتمكين المستخدم من الوصول السريع إلى الامتحانات حسب المادة أو التاريخ.

واجهة الاستخدام تعتمد على الجداول والنماذج البسيطة، وتدعم الاستخدام على الحاسوب والأجهزة المحمولة.

### 6.1.1 Screen images





# 6.2 Interface design rules

* استخدام تصميم بسيط ومرتب يعرض البيانات بوضوح.
* تنظيم الحقول داخل النماذج (Form) حسب تسلسل الاستخدام المنطقي.
* استخدام تقويم لاختيار التاريخ وزر اختيار الوقت (Date & Time Picker).
* عرض رسائل تنبيه واضحة بعد كل عملية (نجاح، خطأ، حذف).
* توفير أزرار بألوان مختلفة حسب نوع العملية (إضافة - حفظ - إلغاء - حذف).
* اعتماد قابلية الاستخدام على الجوال (Responsive Design).
* إبراز الحقول المطلوبة برسوم أو نجمة \*.

## 6.3 Components available

* **جدول عرض الامتحانات**: يحتوي على الأعمدة (اسم المادة، التاريخ، الوقت، المكان، إجراءات).
* **نماذج الإدخال (Forms)**: لإدخال بيانات الامتحان (مادة، تاريخ، وقت، مكان).
* **أزرار الإجراءات**: إضافة، تعديل، حذف.
* **عناصر التحقق (Validation Messages)**: توضح الأخطاء عند إدخال بيانات غير صحيحة.
* **شريط البحث**: لتصفية الامتحانات حسب المادة أو التاريخ.
* **نوافذ منبثقة (Modals)**: لتعديل بيانات الامتحان دون مغادرة الصفحة.
* **رسائل تنبيه (Alerts)**: تظهر عند نجاح أو فشل العملية.

# 6.0 Testing Issues

Test strategy and preliminary test case specification are presented in this section.

## 6.1 Classes of tests

سيتم إجراء نوعين رئيسيين من الاختبارات:

* **Black-box Testing:** سيتم التركيز على اختبار وظائف النظام من وجهة نظر المستخدم دون النظر إلى الكود الداخلي. سيتم استخدام حالات اختبار القبول المذكورة في وثيقة SRS كمرجع أساسي، مثل اختبار محاولة تسجيل مادتين متعارضتين والتأكد من أن النظام يمنع ذلك ويعرض رسالة خطأ مناسبة.
* **White-box Testing:** سيتم اختبار المسارات المنطقية داخل الكود البرمجي. على سبيل المثال، اختبار الخوارزمية المسؤولة عن التحقق من تعارض الجداول للتأكد من أنها تغطي جميع الحالات الممكنة.

## 6.2 Expected software response

الاستجابة المتوقعة هي أن يجتاز النظام جميع حالات الاختبار المحددة في وثيقة SRS بنجاح. على سبيل المثال، عند محاولة الدخول ببيانات صحيحة بعد فتح فصل جديد، يجب أن يعرض النظام إشعاراً بذلك وظهور زر "بدء التسجيل".

## 6.3 Performance bounds

يجب أن يلتزم النظام بمتطلبات الأداء المحددة في وثيقة SRS، وهي:

* يجب أن يتم تحميل صفحات النظام في زمن لا يتجاوز 30 ثانية.
* يجب أن يتم تنفيذ أي عملية (مثل تسجيل مادة) خلال مدة لا تتجاوز دقيقة واحدة.
* يجب أن لا تقل نسبة توفر النظام عن 90% خلال فترات التشغيل الرسمية.

## 6.4 Identification of critical components

المكونات الحرجة التي تتطلب اختباراً مكثفاً هي:

* **مكون التسجيل (RegistrationComponent):** لأنه يعالج عمليات مالية وأكاديمية حساسة ويحتوي على منطق معقد (التحقق من التعارض والمتطلبات).
* **مكون المصادقة والتحكم في الوصول (Authentication & Authorization):** لأنه مسؤول عن أمن النظام ومنع الوصول غير المصرح به.

# 7.0 Appendices

يقدّم هذا القسم معلومات إضافية تدعم مواصفات التصميم البرمجي، بما في ذلك تتبع المتطلبات، مسائل التثبيت، مقاييس الجودة، وأي بيانات داعمة أخرى.

## 7.1 Requirements traceability matrix

|  |  |  |
| --- | --- | --- |
| المكوّنات المرتبطة(Class) | وصف المتطلب | رقم المتطلب |
| User | تسجيل الدخول للمستخدم | FR-01 |
| User | تسجيل مستخدم جديد وتحديد الصلاحيات | FR-02 |
| SemesterStart, Semester | بدء فصل دراسي جديد لطالب | FR-03 |
| Enrollment, SubjectOffer | تسجيل الطالب في مادة مقدّمة | FR-04 |
| |  | | --- | |  | | Subject, SubjectOffer, Semester | | عرض المواد المتاحة في فصل معين | FR-05 |
| ClassSchedule, SubjectOffer | جدولة الحصص الدراسية | FR-06 |
| GradeRecord, Enrollment | تسجيل درجات الطلاب | FR-07 |
| ExamSchedule, SubjectOffer | إدارة مواعيد الامتحانات | FR-08 |
| Subject | شروط المواد السابقة عند التسجيل | FR-09 |
| User, Enrollment, SemesterStart | |  | | --- | |  | | ربط الطالب بمستخدم النظام | | FR-10 |

## 7.2 Packaging and installation issues

يجب أخذ بعض الاعتبارات الخاصة بعين الاعتبار عند تغليف النظام وتثبيته، خصوصًا أنه مبني باستخدام إطار العمل Laravel، ويتطلب إعداد بيئة خادوم محلي (مثل XAMPP أو Laravel Valet). أهم الأمور المتعلقة بالتغليف والتثبيت تشمل:

* **تهيئة قاعدة البيانات:** يجب استيراد ملف قاعدة البيانات studygate.sql من خلال phpMyAdmin أو باستخدام أمر php artisan migrate إذا كانت الهجرة مفعلة.
* **إعداد ملف البيئة .env:** يتوجب تعديل ملف الإعداد .env ليحتوي على إعدادات الاتصال بقاعدة البيانات مثل اسم المستخدم وكلمة المرور واسم القاعدة.
* **تثبيت الحزم المطلوبة:** يجب تشغيل الأمر composer install لتثبيت جميع الاعتمادات (dependencies) الخاصة بـ Laravel.
* .
* **تهيئة البيانات التجريبية (اختياري):** يمكن استخدام php artisan db:seed لتعبئة القاعدة ببيانات تجريبية عند الحاجة.
* **تشغيل الخادوم المحلي:** يمكن تشغيل التطبيق محليًا باستخدام الأمر php artisan serve.

## 7.3 Design metrics to be used

المقاييس التالية ستُستخدم لتقييم جودة التصميم:

|  |  |
| --- | --- |
| الوصف | المقياس |
| قياس مدى تركّز الكلاس على وظيفة واحدة واضحة | Cohesion |
| مدى ارتباط كلاس بكلاسات أخرى – يُفضل أن يكون منخفضاً | Coupling |
| لحساب عدد المسارات المنطقية المحتملة في كل مكون | Cyclomatic Complexity |
| تقييم إمكانية إعادة استخدام المكونات في أجزاء أخرى من النظام | Reusability |
| قدرة التصميم على استيعاب وظائف جديدة مستقبلًا بدون تعديل كبير | Scalability Readiness |
| وضوح مسؤولية كل مكون / كلاس داخل النظام | Responsibility Clarity |

## 7.4 Supplementary information (as required)

يتضمن هذا القسم معلومات داعمة قد تكون مفيدة لفهم النظام أو تسهيل التعامل معه أثناء مراحل التطوير أو التشغيل. ومن أبرز هذه المعلومات:

* **أدوار المستخدمين:** النظام يدعم ثلاث أدوار رئيسية:
  + **المدير (Admin):** مسؤول عن بدء الفصل الدراسي، تخصيص المدرسين للمواد، متابعة الإحصائيات من خلال لوحة التحكم.
  + **المعلم (Teacher):** يمكنه رؤية المواد التي يدرّسها وجداول الصفوف المرتبطة بها.
  + **الطالب (Student):** يمكنه تسجيل الدخول واستعراض المواد التي تم تسجيله فيها ومواعيد الحصص والامتحانات.
* **تنظيم الفصول والامتحانات:** تم فصل جدول الحصص عن جدول الامتحانات لتسهيل إدارتها، مع استخدام الجلسات (Session) والأيام (Day of week) لتحديد الوقت بدلاً من التواريخ الدقيقة.
* **المرونة المستقبلية:** تم تصميم قاعدة البيانات والنظام بطريقة قابلة للتوسعة لاحقًا، مثل:
  + إمكانية إضافة إشعارات أو لوحة إعلانات.
  + دعم تقييمات الطلاب أو تقارير الحضور.
  + تطوير صلاحيات متقدمة أكثر لكل دور.
* **اللغة:** تم دعم اللغة العربية في جميع الواجهات، مع إمكانية دعم اللغة الإنجليزية لاحقًا باستخدام ميزة الترجمة (Laravel Localization).
* **توافق النظام:** يمكن تشغيل النظام على أي بيئة تدعم PHP وMySQL، مثل XAMPP أو WAMP أو استضافة Laravel رسمية.