بسم الله الرحمن الرحيم

جامعة بوليتكنك فلسطين

كلية تكنولوجيا المعلومات وهندسة الحاسوب

دائرة علم الحاسوب

مشروع تخرج بعنوان

College Exam Scheduling Assistant

مساعد جدولة امتحانات الكلية

إعداد الطلاب

عبد العفو امجد الزغير **فخرالدين ابراهيم تلاحمه**

بإشراف الدكتور

هاشم هشام التميمي

تم تنفيذ هذا المشروع كجزء من متطلبات الحصول على درجة البكالوريوس في تخصص علم الحاسوب، بهدف تطبيق المفاهيم النظرية والعملية المكتسبة خلال فترة الدراسة على مشروع عملي يخدم البيئة الأكاديمية.

اهداء

إلى والدينا

بفصاحة القلب وبكل احترام وتقدير، نتوجه إليكما برسالة تحمل في طياتها أعظم معاني الامتنان والعرفان. إن الكلمات، مهما بلغت من البلاغة، تعجز عن وصف ما يختلج في قلوبنا من حب ومودة لكما. كنتما لنا الركيزة التي نعتمد عليها في أوقات الشدة، والسند الذي يرفعنا عند التعثر، والشمعة التي أضاءت لنا الطريق وسط ظلمات الحياة.

كنتما مصدر الإلهام الأول لنا، وغرستما فينا قيم الصبر والاجتهاد، فكان هذا الإنجاز ثمرة لدعمكما المستمر ودعواتكما الصادقة. لا يسعنا إلا أن نبقى ممتنين لجهودكما العظيمة، ونسأل الله أن يجزيكما عنا خير الجزاء ويبارك في أعماركما.

نهدي هذا العمل المتواضع إليكما، فهو بفضلكما أولاً وأخيرًا.

...

إلى أصدقائنا

بكل احترام وتقدير، نرفع لكم تحية ملؤها الود والاعتزاز. كنتم وما زلتم الأصدقاء الأوفياء والرفاق المخلصين، الذين جعلوا من رحلتنا أكثر بهجة وسهولة. لقد كنتم العون والسند في كل لحظة، الصخرة الصلبة التي استندنا إليها لعبور تحديات الحياة.

شكرًا لكم على كل لحظة شاركتمونا فيها، وعلى دعمكم الذي لم يتوقف، وتضحياتكم الجليلة التي لا تُنسى. حضوركم في حياتنا كان ولا يزال نعمة عظيمة، ونجاحنا اليوم هو انعكاس لروح الجماعة التي شاركتمونا إياها.

شكر وتقدير

إلى أساتذتنا الكرام، نتقدم بأسمى عبارات الشكر والتقدير على ما بذلتموه من جهود عظيمة خلال مسيرتنا الدراسية. لقد كنتم قدوةً لنا ومصدر إلهام، وكان لجهودكم دورٌ كبيرٌ في صياغة ملامح مستقبلنا الأكاديمي والمهني.

نتوجه بخالص الشكر والتقدير إلى الدكتور **هاشم هشام التميمي**، الذي كان مثالاً يحتذى به في التفاني والإرشاد القيم. لقد أثرى عقولنا بمعرفته وعلمه، وكان دائم الدعم لنا في كل خطوة على طريق النجاح.

كما نتقدم بخالص الشكر إلى الأستاذ **معتصم الحيح**، الذي كان شريكًا فاعلًا في عملية إعداد النظام، حيث قدّم لنا رؤى وإرشادات قيمة خلال مراحل تطوير المشروع، وساهم في تيسير العمل بشكل كبير من خلال خبرته ودعمه المستمر.

لا يسعنا إلا أن نعبر عن امتناننا العميق لصبركم وحرصكم على تهيئة بيئة تعليمية مشجعة، مليئة بالتحفيز والتقدير، مما جعل سنوات الدراسة تجربة غنية ومثمرة.

نود أن نتقدم بالشكر أيضًا إلى **جامعة بوليتكنك فلسطين**، على ما قدمته من مرافق تعليمية وخدمات ساهمت بشكل كبير في تمكيننا من تحقيق أهدافنا الأكاديمية.

نسأل الله أن يوفقكم ويجزيكم خير الجزاء، وأن يستمر عطاؤكم في خدمة العلم والتعليم، لما فيه الخير والتطور.

الملخص

هذا المشروع يهدف إلى تسهيل عملية تنظيم امتحانات كلية تكنولوجيا المعلومات وهندسة الحاسوب في جامعة بوليتكنك فلسطين. ببساطة، النظام يجعل إعداد الجداول أسهل وأسرع، حيث يمكن للمستخدمين تحميل بيانات الطلاب من ملفات Excel وإدارة القاعات والمراقبين بسهولة. كما يوفر واجهة مرنة تعتمد على السحب والإفلات لتنسيق الجداول بدون تعقيد.

تم تطوير النظام بالتنسيق مع إدارة كلية تكنولوجيا المعلومات وهندسة الحاسوب في جامعة بوليتكنك فلسطين لضمان تلبية احتياجاتها الفعلية. خلال التنفيذ، تم عقد اجتماعات لمناقشة التحديات مثل التعارضات في المواعيد وتوزيع الامتحانات بشكل عادل. هذه النقاشات ساعدت على تحسين النظام ليكون أكثر دقة وملاءمة للواقع الأكاديمي.

أحد أبرز إنجازات النظام هو تقليل الوقت والجهد في إعداد الجداول، وتقليل الأخطاء الناتجة عن العمليات اليدوية. كما ساهم في تحسين التواصل بين المعنيين وجعل إدارة الامتحانات أكثر تنظيمًا وسلاسة، مما يرفع من كفاءة العملية التعليمية ككل.

Abstract

This project focuses on simplifying and improving the scheduling of exams at the College of Information Technology and Computer Engineering at Palestine Polytechnic University. It provides an easy way to organize exams by allowing users to upload student data via Excel files and manage exam rooms and invigilators efficiently. With a user-friendly, drag-and-drop interface, the system makes scheduling faster and less complicated.

Developed in collaboration with the college administration, the system was designed to meet real needs. Meetings with the College of Information Technology and Computer Engineering at Palestine Polytechnic University Registrar helped address challenges like scheduling conflicts and fair exam distribution, ensuring the system aligns with practical requirements.

The system has made a noticeable impact by reducing the time and effort needed for scheduling, minimizing human errors, and improving communication among stakeholders. It offers a more organized and efficient way to manage exams, making the process smoother and more reliable.

قائمة المحتويات:

الملخص 4
Abstract 5
1. المقدمة 9
1.1 مقدمة 9
1.2 الدافع 9
1.3 نطاق المشروع 10
1.4 بدائل المشروع 10
1.5 أهداف المشروع 11
1.6 أهمية المشروع 12
1.7 المنهجية المتبعة 12
2. تحليل وتصميم النظام 13
2.1 المقدمة 13
2.2 المتطلبات الوظيفية 13
2.3 المتطلبات غير الوظيفية 14
2.4 Use Case Diagram 15
2.5 Activity Diagram 16
2.6 تحليل المتطلبات الوظيفية 24
2.7 Sequence Diagram 28
3. قاعدة البيانات والتطوير التقني 29
3.1 تفاصيل تقنية النظام 29
3.2 تفاصيل قاعدة البيانات 30
3.3 ER Diagram 38
4. تطوير النظام 39
4.1 Model Architectural 39
4.2 خطوات التطوير 40
5. اختبار النظام 58
5.1 اختبارات الوحدة unit testing 58
5.2 اختبارات التكامل Integration Testing 59
5.3 اختبارات الأداء Performance testing 60
5.4 اختبارات التفاعل مع المستخدمين User Interaction Testing 61
5.5 توسعة نطاق الاختبارات 62
5.6 التحديات أثناء الاختبارات 62
5.7 الخلاصة 63
6. النتائج والتوصيات 64
6.1 التحديات 64
6.2 النتائج 64
6.3 التوصيات 65
6.4 الخاتمة 65
7. المراجع 66

قائمة الجداول:

جدول1: مقارنة بي انظمة الجدولة 10
جدول2: رفع بيانات الطلاب 23
جدول3: إدخال بيانات المراقبين 23
جدول4: كشف التعارضات 24
جدول5: تعيين القاعات 24
جدول6: جدولة الامتحانات 25
جدول7: تقارير القاعات 25
جدول8: التقارير النهائية 26
جدول9: توسع النظام ليشمل جميع كليات الجامعة 26
جدول10: Student 30
جدول11: Enrollment 30
جدول12: Course 31
جدول13: Exam 31
جدول14: Schedule 32
جدول15: AdminAccount 32
جدول16: Invigilator 33
جدول17: Hall 33
جدول18: InstructorHall 34
جدول19: course_mappings 34
جدول20: course_mapping_relations 35
جدول21: availability_days 35
جدول22: availability_slots 36

قائمة الأشكال:

الشكل1: Use Case Diagram 15
الشكل2: عملية رفع بيانات الطلاب 16
الشكل3: عملية إدارة المراقبين 17
الشكل4: عملية جدولة الامتحانات 19
الشكل5: عملية الكشف عن التعارضات 20
الشكل6: عملية إدارة حساب المشرف 21
الشكل7: عملية إدارة القاعات 22
الشكل8: عملية توليد التقارير 23
الشكل9: Sequence Diagram 28
الشكل10: ER Diagram 38
الشكل11: Model Architectural 39
الشكل12: واجهة تسجيل الدخول ((Login 48
الشكل13: واجهة إدارة القاعات ((HALLS 49
الشكل14: تحديد القاعات والمراقبين للامتحانات 50
الشكل15: عرض تقارير القاعات والمراقبين 50
الشكل16: واجهة إدارة الامتحانات ((EXAM 51
الشكل17: رفع بيانات الطلاب للامتحانات 51
الشكل18: جدولة الامتحانات باستخدام خاصية السحب والإفلات 51
الشكل19: الية عرض التعارضات في الامتحانات 52
الشكل20: واجهة النماذج ((FORMS 52
الشكل21: إضافة المراقبين وتحديث بياناتهم 53
الشكل22: إضافة القاعات الجديدة 53
الشكل23: إضافة مساقات الجديدة 54
الشكل24: تعريف المساقات التي لها اكثر من مسمى 54
الشكل25: واجهة الطلاب ((STUDENTS 54
الشكل26: البحث عن الطلاب 55
الشكل27: واجهة التعارضات ((CONFLICTS 55
الشكل28: واجهة عرض جدول التعارضات ((CONFLICTS TABLE VIEW 56
الشكل29: واجهة إدارة المستخدمين ((USERS 56
الشكل30: إضافة مستخدم جديد 57
الشكل31: رسالة الخطأ 58
الشكل32: تقرير مفصل عن التعارضات 58
الشكل33: التكامل بين الواجهة الأمامية والخلفية 59
الشكل34: التعارضات اثناء الجدولة 59
الشكل35: تقرير الطلاب داخل القاعات 60
الشكل36: تقرير التعارضات لدى الطلاب 60
الشكل37: الوقت المستغرق لرفع ملف ال Excel 60
الشكل38: الوقت المستغرق لجدولة الامتحانت 61
الشكل39: الامتحانت المجدولة بشكل تلقائي 61
الشكل40: التنقل بين الصفحات 61
الشكل41: رسالة خطأ 62
الشكل42: جدول الامتحانات مع التعارضات 65
الشكل43: جدول الامتحانات بعد معالجة التعارضات 65
الشكل44: ميزة السحب والافلات 65

1. المقدمة

1.1 مقدمة

في ظل التطور التكنولوجي السريع والتحول نحو الأتمتة في العمليات الإدارية والتعليمية، أصبحت الحاجة ماسة إلى نظم فعّالة لإدارة جداول الامتحانات بما يضمن تحسين الكفاءة وتقليل الأخطاء البشرية. ولحل هذه المشكلة، تتوفر طريقتان رئيسيتان: الأولى تعتمد على أنظمة الجدولة الآلية بالكامل مثل UniTime، والثانية تقوم على نظام محوسب يدمج بين الأتمتة والتدخل البشري، وهي الطريقة التي اعتمدناها في هذا المشروع.

أنظمة الجدولة الآلية بالكامل تعتمد على خوارزميات متقدمة تقوم بإعداد الجدول دون تدخل الإنسان. ورغم مزاياها التي تشمل السرعة وتقليل الأخطاء، إلا أن لها عيوبًا مثل غياب المرونة، حيث قد تفشل في مراعاة الظروف الاستثنائية أو المتطلبات الخاصة التي يمكن للبشر التعامل معها بسهولة. بالإضافة إلى ذلك، غالبًا ما تكون هذه الأنظمة مكلفة ومعقدة في الإعداد والصيانة، مما يجعلها غير مثالية للمؤسسات ذات الموارد المحدودة أو تلك التي تحتاج إلى تخصيص أكبر.

في المقابل، الطريقة الثانية التي تعتمد على نظام محوسب يدمج بين الأتمتة والتدخل البشري أثبتت كفاءتها ومرونتها العالية، خاصةً في بيئات العمل التي تتطلب قرارات دقيقة تستند إلى الخبرة البشرية. في مشروعنا، طورنا نظامًا يوفر أدوات تفاعلية مثل رفع بيانات الطلاب وإدخال بيانات القاعات والمراقبين، واستخدام واجهات سهلة بخاصية السحب والإفلات لجدولة الامتحانات. كما أضفنا ميزة الجدولة الآلية التي يمكن استخدامها لإنشاء جداول امتحانات بشكل تلقائي عند الحاجة، مما يوفر خيارات متعددة لتلبية متطلبات الكلية.

أحد أهم مميزات هذا النهج هو المرونة التي يوفرها. يتيح التدخل البشري التعامل مع الحالات الاستثنائية والاحتياجات الخاصة التي قد تواجه الكلية، مثل ترتيب الامتحانات بناءً على متطلبات محددة أو التعامل مع تغييرات طارئة.

من خلال الجمع بين الأتمتة والتدخل البشري، ساهم هذا النظام في تقليل الوقت والجهد المطلوبين لإعداد الجداول، مع تقليل الأخطاء البشرية المرتبطة بالعمليات اليدوية. كما أن واجهاته التفاعلية سهلت الاستخدام، مما جعلها متاحة لجميع الفئات دون الحاجة إلى تدريب مكثف. هذا النهج لا يعزز فقط كفاءة العمل، بل يوفر أيضًا تجربة أفضل للمسؤولين عن الجدولة، مما يجعل إدارة جداول الامتحانات أكثر سلاسة ودقة.

1.2 الدافع

تواجه كلية تكنولوجيا المعلومات وهندسة الحاسوب في جامعة بوليتكنك فلسطين العديد من التحديات في إدارة جداول الامتحانات، مما يستدعي تطوير نظام متكامل لتجاوز العقبات الحالية. من أبرز هذه التحديات تعارض المواعيد بين الامتحانات، والذي يؤدي إلى إرباك في سير العملية الامتحانية وإرباك الطلاب. بالإضافة إلى ذلك، تبرز صعوبة تخصيص القاعات بما يتناسب مع السعة المطلوبة لكل امتحان، مما يسبب ازدحامًا أو هدرًا في استخدام الموارد المتاحة.

تعتمد الكلية حاليًا على أدوات تقليدية مثل النظام الورقي أو نظام Caphub، التي تفتقر إلى خاصية التفاعل التلقائي وحل المشكلات بكفاءة، مما يجعل العملية مرهقة وتستغرق وقتًا طويلاً. كما أن غياب نظام يمكنه التكيف مع التغيرات المفاجئة يجعل من الصعب تحديث الجداول بسرعة في حال ظهور ظروف طارئة.

بناءً على هذه التحديات، جاءت الحاجة إلى تصميم نظام ذكي قادر على تحسين إدارة جداول الامتحانات، من خلال تقديم حلول فعالة ومتكاملة تلبي احتياجات الكلية وتعزز من كفاءة العمليات.

1.3 نطاق المشروع

يستهدف هذا المشروع كلية تكنولوجيا المعلومات وهندسة الحاسوب في جامعة بوليتكنك فلسطين، حيث يهدف إلى تحسين إدارة جداول الامتحانات بشكل شامل ومتكامل. يركز النظام على تلبية احتياجات الكلية الحالية، مع تصميم مرن يتيح إمكانية توسيع نطاقه ليشمل كليات أخرى داخل الجامعة، بل وحتى تطبيقه في مؤسسات تعليمية خارج الجامعة. هذا التوجه يعكس رؤية المشروع لتقديم حل قابل للتطبيق على نطاق أوسع، مع الحفاظ على كفاءة الأداء وسهولة الاستخدام.

1.4 بدائل المشروع

النظام الحالي (Caphub):

Caphub هو النظام الحالي الذي تستخدمه الجامعة لجدولة الامتحانات، ويُعد حلاً أساسيًا يُسهم في تسهيل بعض جوانب العملية. ومع ذلك، لا يتم الاعتماد عليه بشكل كامل، حيث يعاني من غياب التكامل مع أنظمة البيانات الأخرى، مما يتطلب إدخال المعلومات يدويًا، وهو أمر يستهلك الوقت ويزيد من احتمالية الأخطاء. كما يفتقر النظام إلى آلية تلقائية لكشف التعارضات بين مواعيد الامتحانات أو توزيع القاعات والمراقبين، مما يدفع المستخدمين إلى معالجة هذه الجوانب يدويًا باستخدام الورقة والقلم.

إضافة إلى ذلك، لا يوفر **Caphub** جميع الأدوات التي تحتاجها الكلية لضمان تنظيم دقيق ومرن للامتحانات، ما يجعل العملية أقل كفاءة ويُبقي بعض المهام خارج نطاق الأتمتة. واجهته التقليدية والافتقار إلى المرونة يحدان من قدرته على التكيف مع التغييرات أو احتياجات الجدولة الطارئة، مما يعزز الحاجة إلى نظام أكثر تكاملًا وكفاءة، وهو ما يقدمه مشروعنا بشكل شامل، عبر ميزات مبتكرة تعالج القيود الحالية وتجعل الجدولة أكثر دقة وكفاءة، وأبرزها:

- **واجهة نظام تفاعلية:** يتميز مشروعنا بواجهة مستخدم متطورة وتفاعلية تسهل عملية الجدولة وتجعلها أكثر مرونة وسهولة في الاستخدام.
- **سهولة كشف التعارضات:** يوفر النظام إمكانية كشف التعارضات بسرعة وسهولة، مما يقلل من احتمالية وقوع أخطاء.
- **حل التعارضات بفعالية:** يتميز النظام بقدرة عالية على حل التعارضات باستخدام خاصية السحب والإفلات، مما يسهل عملية التعديل والتحسين.
- **تكامل البيانات:** يمكن تحميل جميع البيانات المتوفرة في مكان واحد، مما يسهل إدارة العملية ويضمن توافر جميع المعلومات اللازمة لإعداد الجداول بشكل دقيق.

(Microsoft Excel):

تعتبر برامج الأوفيس من الأدوات الشائعة التي تستخدم في إدارة الجداول الزمنية، مثل برنامج Excel. ومع ذلك، تفتقر هذه البرامج إلى القدرة على اكتشاف التعارضات بين مواعيد الامتحانات بشكل تلقائي، مما يتطلب جهدًا يدويًا كبيرًا للتحقق من صحة الجدول. بالإضافة إلى ذلك، تصبح هذه البرامج غير عملية مع زيادة حجم البيانات وتعقيد الجداول.

الجدولة الورقية:

تعتبر الجدولة الورقية تقليدية وتستغرق وقتًا وجهدًا كبيرين. إنها عرضة للأخطاء البشرية مثل تعارض المواعيد، وتوزيع غير متساوٍ للمراقبين والقاعات. إضافة إلى ذلك، فإن إجراء التعديلات في الجداول الورقية بعد إعدادها يصبح معقدًا للغاية، مما يزيد من مخاطر الأخطاء وضياع الوقت.

جدول1: مقارنة بي انظمة الجدولة

المعيار	Microsoft Excel	الجدولة الورقية	النظام المقترح
سهولة الاستخدام	سهل الاستخدام لكنه يتطلب خبرة في الصيغ	صعبة وتستهلك وقتًا في التعديل	واجهة تفاعلية وسهلة الاستخدام
إمكانية اكتشاف التعارضات	غير قادر على اكتشاف التعارضات تلقائيًا	غير قادرة على اكتشاف التعارضات	يكتشف التعارضات تلقائيًا باستخدام الخوارزميات
إجراء التعديلات	معقد عند وجود العديد من التعديلات	معقدة جدًا وصعبة التعديل بعد الإعداد	مرن ويمكن تعديل الجداول بسهولة
الدقة وتجنب الأخطاء	عرضة للأخطاء البشرية عند إدخال البيانات يدويًا	عرضة للأخطاء البشرية بشكل كبير	يقلل الأخطاء البشرية عبر التحقق الذكي
الوقت والجهد المطلوب	يتطلب جهدًا يدويًا كبيرًا للتحقق من الجدول	تتطلب وقتًا وجهدًا كبيرين	يوفر الوقت والجهد من خلال الأتمتة
التعامل مع الجداول الكبيرة	غير عملي مع زيادة حجم البيانات	غير مناسبة للجداول الكبيرة	مصمم للتعامل مع الجداول الكبيرة بفعالية
إصدار التقارير والتحليلات	يدعم التقارير لكنه يحتاج إلى إعداد يدوي	لا توفر تقارير أو تحليلات إلا يدويًا	يصدر تقارير فورية تساعد في اتخاذ القرارات

1.5 أهداف المشروع

يهدف المشروع إلى تصميم وتطوير نظام متكامل يساهم في أتمتة إعداد وإدارة جداول الامتحانات بطريقة دقيقة وفعّالة. يسعى النظام إلى تحسين تجربة المستخدمين، خاصة المسؤولين، من خلال توفير واجهات مستخدم تفاعلية وسهلة الاستخدام. كما يركز المشروع على تقليل الأخطاء والتعارضات التي قد تنجم عن الجدولة اليدوية، مما يسهم في تعزيز كفاءة العملية الامتحانية.

بالإضافة إلى ذلك، يهدف النظام إلى تقديم تقارير فورية تساعد في اتخاذ قرارات إدارية مستنيرة، مع دعم حل التعارضات بطريقة مرنة. يمكن للنظام التعامل مع هذه التعارضات تلقائيًا باستخدام خوارزميات ذكية أو السماح بحلها يدويًا وفقًا لأولويات الإدارة، مما يضمن تلبية احتياجات الكلية وتحسين كفاءة الجدولة.

1.6 أهمية المشروع

يسهم المشروع بشكل كبير في تحسين كفاءة إدارة الامتحانات من خلال أتمتة العمليات وتقليل الاعتماد على الأساليب اليدوية، مما يقلل من الأخطاء البشرية باستخدام تقنيات ذكية للتحقق التلقائي من التعارضات. كما يدعم المشروع الشفافية والتنسيق بين الإدارات المختلفة في الكلية، مما يسهل التعاون والتنظيم بين المسؤولين عن الجدولة والمراقبين.

علاوة على ذلك، يوفر النظام الوقت والجهد من خلال تسريع عمليات الجدولة وتوفير أدوات دقيقة لتخصيص القاعات والمراقبين. كما يُقدم النظام تجربة أكثر تنظيماً من خلال إشعارات مخصصة تُبقي الطلاب والمراقبين على اطلاع مستمر بالتحديثات، مما يضمن استمرارية العمليات وانسيابيتها بشكل فعال.

1.7 المنهجية المتبعة

تم اختيار منهجية **Agile Development** لتطوير النظام نظرًا لمرونتها وقدرتها على التكيف مع المتطلبات المتغيرة للمستخدمين. تتيح هذه المنهجية تحسين مخرجات المشروع بشكل مستمر وفقًا للتغذية الراجعة من الجهات المستفيدة. تتبع المنهجية مجموعة من الخطوات المنهجية التي تضمن تحقيق الأهداف بكفاءة:

جمع وتحليل المتطلبات

تبدأ المنهجية بجمع المتطلبات من المسؤولين والجهات ذات الصلة، مع التركيز على فهم التحديات والاحتياجات بدقة لضمان تلبيتها في النظام.

تصميم النظام

يتم إعداد تصاميم مبدئية للنظام تتضمن تخطيطًا واضحًا للوظائف الرئيسية والمكونات المختلفة، مع مراعاة توفير تجربة مستخدم سهلة ومتكاملة.

تطوير النظام على مراحل (Sprints)

يتم تقسيم العمل إلى دورات تطوير قصيرة (Sprints)، حيث يتم التركيز على تطوير وظائف محددة في كل دورة، مما يتيح تحسين النظام تدريجيًا.

اختبار كل مرحلة

تخضع كل وظيفة أو ميزة جديدة للاختبار مباشرة بعد تطويرها، لضمان خلوها من الأخطاء وفعاليتها في أداء المهام المطلوبة.

تنفيذ التعديلات بناءً على التغذية الراجعة

يتم الحصول على ملاحظات المستخدمين للنظام بعد كل دورة تطوير، مما يتيح إجراء التعديلات والتحسينات اللازمة لضمان أن النظام يلبي توقعاتهم بشكل كامل.

تُعد هذه المنهجية مثالية لتطوير نظام إدارة جداول الامتحانات، حيث تسهم في تحقيق توازن بين المرونة والجودة، مما يضمن تقديم نظام فعّال وقابل للتطوير المستقبلي.

2. تحليل وتصميم النظام

2.1 المقدمة

يركز هذا الفصل على تقديم تحليل شامل للنظام المكتمل، من خلال استعراض المتطلبات الوظيفية وغير الوظيفية التي بُني عليها. بالإضافة إلى ذلك، يتضمن الفصل مجموعة من الرسوم التوضيحية التي تُبرز البنية الأساسية للنظام وطريقة عمله، مما يُساعد في توضيح كيفية تصميمه لتحقيق أهدافه وتلبية احتياجات الكلية بشكل فعال.

2.2 المتطلبات الوظيفية

1. **رفع بيانات الطلاب**

يسمح النظام لمسؤول النظام برفع بيانات الطلاب على شكل ملف Excel يحتوي على أسماء الطلاب، أرقامهم الجامعية، والمقررات الدراسية المسجلة. لضمان التوافق مع النظام، يجب أن تكون أسماء الأعمدة في الملف محددة ومتفق عليها مسبقًا. بمجرد رفع الملف، يتم استيراد البيانات تلقائيًا إلى قاعدة بيانات البرنامج، مما يُسهّل تنظيم الجداول الأكاديمية وجدولة الامتحانات بدقة وسلاسة.

2. **إدخال بيانات المراقبين**

يمكن إدخال بيانات المراقبين يدويًا، أوقات توفرهم، وأي معلومات إضافية متعلقة بالإشراف على الامتحانات. يوفر هذا النظام وسيلة مريحة لمسؤول النظام لإدارة المراقبين وتوزيعهم على القاعات بشكل مناسب، مع مراعاة أوقات توافرهم ومتطلبات الجدولة.

3. **كشف التعارضات**

يتولى النظام الكشف التلقائي عن التعارضات في جدول الامتحانات. تشمل هذه التعارضات تزامن امتحانين لطالب واحد أو تخصيص قاعة لأكثر من امتحان في نفس الوقت. بمجرد اكتشاف تعارض، يقوم النظام بتنبيه المسؤول لتعديل الجدول وإصلاح المشكلات قبل اعتماد النسخة النهائية، مما يضمن دقة وجدولة منظمة وخالية من الأخطاء.

4. **تعيين القاعات**

يتيح النظام لمسؤول الجدولة تخصيص القاعات الدراسية بطريقة مرنة وفعّالة بناءً على عدد الطلاب المسجلين وسعة القاعات المتوفرة. يعتمد هذا التخصيص على البيانات المدخلة، حيث يتم توزيع الطلاب بشكل منظم لضمان الاستفادة المثلى من الموارد المتاحة. يسهم النظام في تقليل التكدس داخل القاعات من خلال تحقيق التوازن بين السعة الفعلية للقاعة وعدد الطلاب، مما يضمن توفير بيئة مريحة وملائمة لإجراء الامتحانات.

5. **جدولة الامتحانات**

يدعم النظام جدولة الامتحانات من خلال واجهة مستخدم متطورة تعتمد على خاصية السحب والإفلات، مما يمنح المستخدمين مرونة وسهولة في تحديد مواعيد الامتحانات. كما يدمج النظام ميزة كشف التعارضات مباشرة أثناء الجدولة، مما يُمكّن المستخدم من تعديل الجدول بشكل ديناميكي وسريع عند الحاجة.

6. **تقارير القاعات**

يوفر النظام إمكانية إنشاء تقارير مفصلة حول استغلال القاعات، تتضمن بيانات دقيقة عن الحجز والتوافر. تساعد هذه التقارير المسؤولين على تحليل استخدام الموارد الأكاديمية وتحسين استراتيجيات تخصيص القاعات، مما يُسهم في كفاءة الإدارة الأكاديمية.

7. **التقارير النهائية**

يُمكّن النظام المسؤولين من إنشاء تقارير شاملة تلخص جميع تفاصيل الجدول الزمني للامتحانات، توزيع القاعات، والمراقبين. تُعد هذه التقارير أداة أساسية للمراجعات الإدارية والتخطيط المستقبلي، مما يدعم تحسين العمليات الأكاديمية في الفصول الدراسية القادمة.

8. **توسع النظام ليشمل جميع كليات الجامعة**

تم توسيع نطاق النظام ليشمل كافة كليات الجامعة بفضل مرونته وكفاءته العالية. يتيح هذا التوسع تحسين عمليات الجدولة بشكل شامل لجميع الكليات، مع ضمان إدارة دقيقة لتنظيم الامتحانات بما يتوافق مع احتياجات كل كلية على حدة، مما يساهم في تعزيز كفاءة العمل على مستوى الجامعة.

2.3 المتطلبات غير الوظيفية

1. **الأمان**

يضمن النظام مستوى عالٍ من الأمان من خلال تشفير البيانات الحساسة مثل بيانات الطلاب والمراقبين، مع تحديد الصلاحيات بحيث تكون مقتصرة على مسؤول النظام فقط. يهدف ذلك إلى حماية المعلومات من الوصول غير المصرح به وضمان سرية البيانات.

2. **المرونة**

يدعم النظام التشغيل على مجموعة متنوعة من الأجهزة والشاشات، بما في ذلك الهواتف الذكية وأجهزة الحاسوب اللوحية وأجهزة الحاسوب المكتبية. يتميز بواجهة مستخدم متجاوبة (Responsive UI)، مما يضمن تجربة مستخدم سلسة ومتكاملة بغض النظر عن نوع الجهاز أو المتصفح المستخدم.

3. **الأداء**

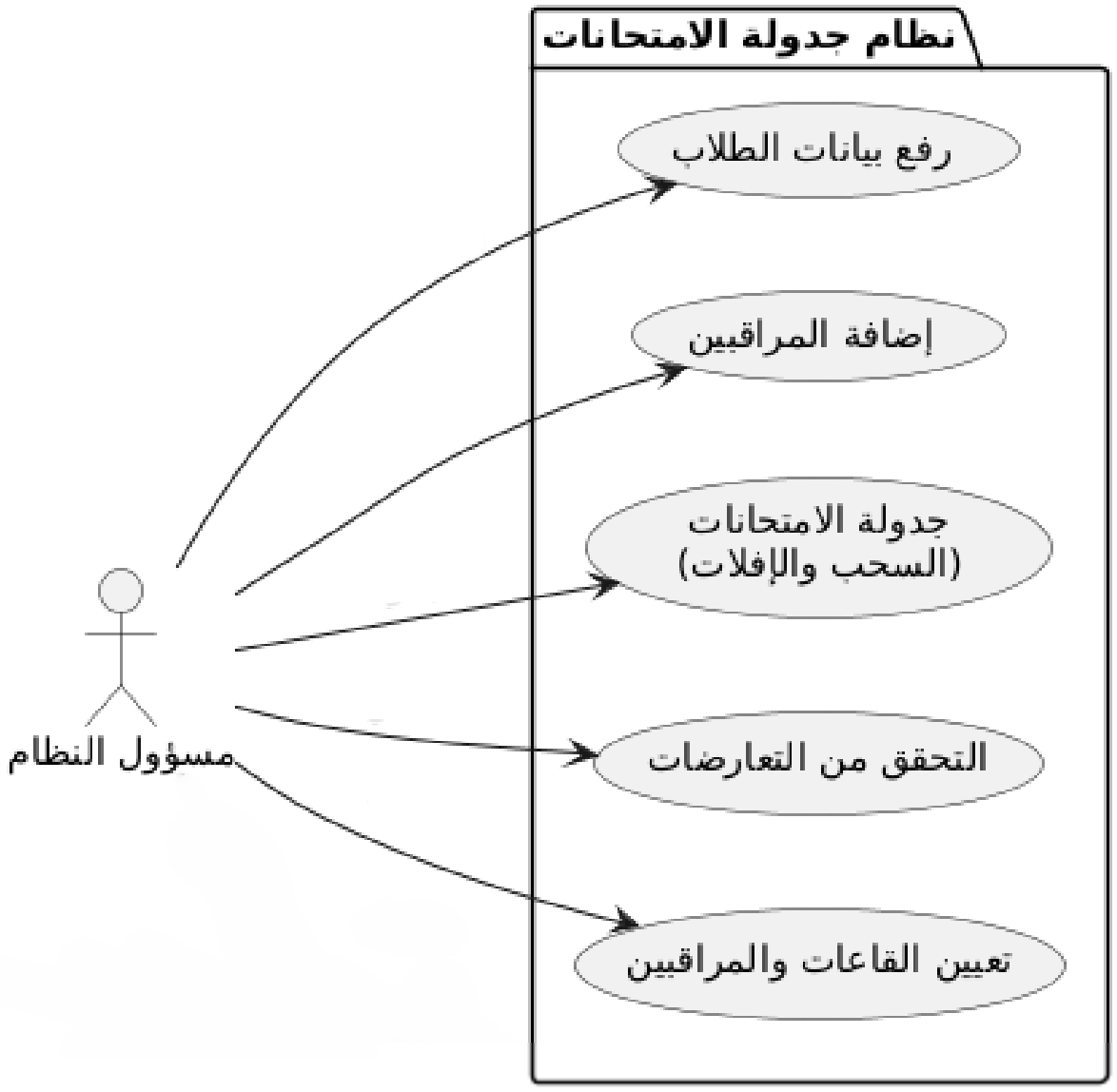
تم تحسين النظام لضمان سرعة استجابته حتى عند معالجة بيانات كبيرة الحجم. تشمل التحسينات تقنيات متقدمة لمعالجة البيانات وخوارزميات فعالة لتقليل زمن المعالجة، مما يوفر تجربة استخدام سريعة وفعّالة.

4. **التوافر**

يتم تصميم النظام ليعمل بشكل مستمر دون انقطاع، مع الاعتماد على بنية تقنية تدعم الاستمرارية وتقلل من احتمالات التوقف. يضمن ذلك تلبية احتياجات المستخدمين بشكل دائم، خصوصًا خلال فترات الامتحانات الحرجة.

Use Case Diagram 2.4

يشرح **الشكل 1** حالات الاستخدام الرئيسية للنظام، بما في ذلك التفاعلات بين مسؤول النظام والوظائف المختلفة مثل رفع بيانات الطلاب، إضافة المراقبين، جدولة الامتحانات، والتحقق من التعارضات. الهدف هو توضيح كيفية استخدام النظام لتحقيق المهام المطلوبة



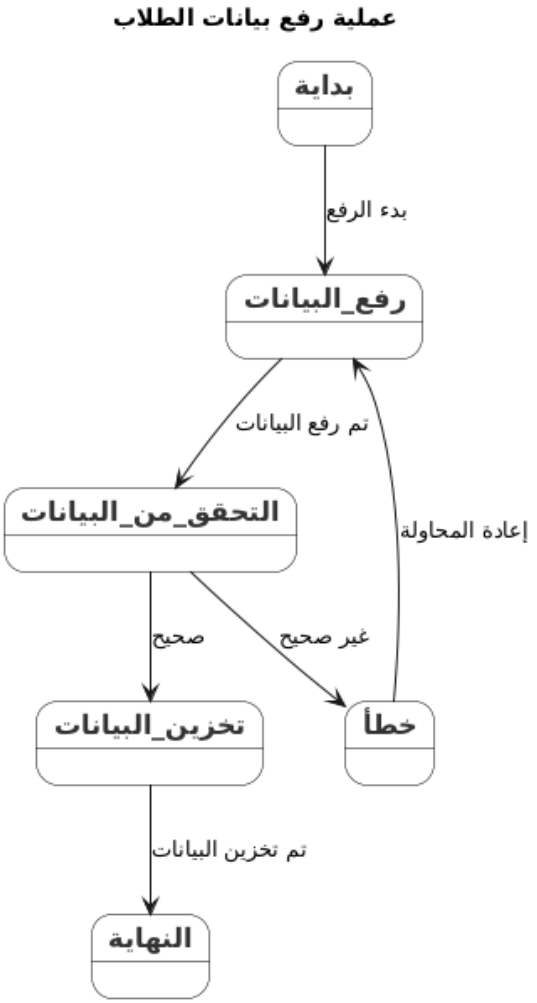
الشكل1: Use Case Diagram

Activity Diagram 2.5

تمثل مخططات النشاط (Activity Diagram) أحد الأدوات الأساسية لفهم سلوك النظام وتصميمه. تهدف هذه المخططات إلى توضيح الحالات المختلفة التي يمر بها النظام وكيفية الانتقال بينها بناءً على الأحداث والعمليات التي تحدث أثناء تشغيل النظام.
يتم تمثيل كل حالة على شكل عقدة في الرسم، بينما توضح الأسهم العمليات أو الأحداث التي تؤدي إلى انتقال النظام من حالة إلى أخرى. يساعد هذا الأسلوب في تقديم نظرة واضحة حول ديناميكية النظام وسلوكه في التعامل مع السيناريوهات المختلفة.

1. عملية رفع بيانات الطلاب

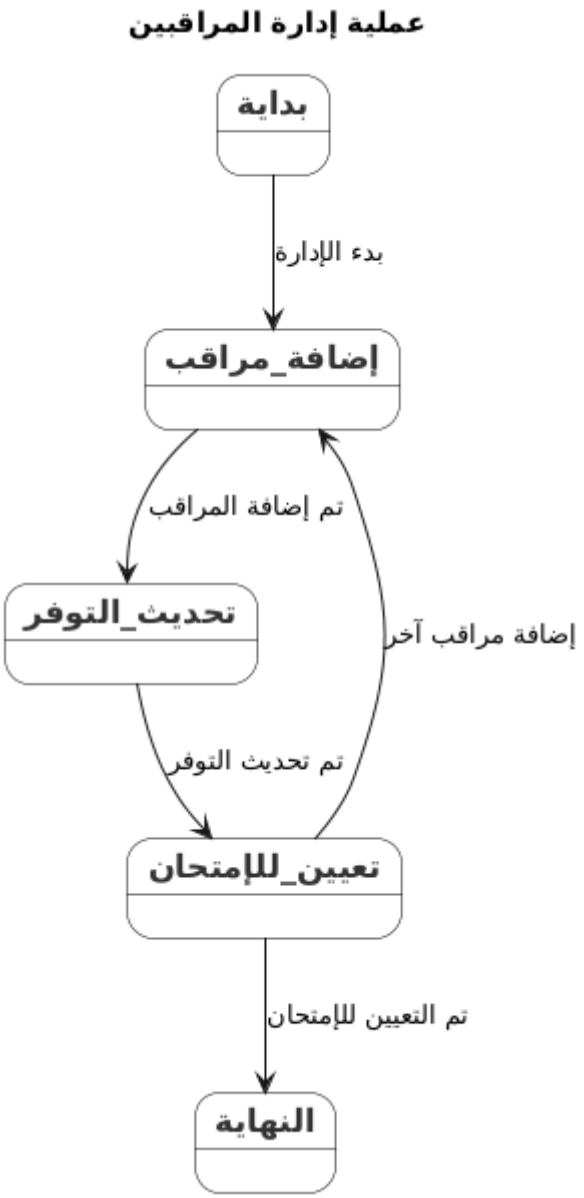
الشكل 2 يُظهر سير العمل في رفع بيانات الطلاب من البداية إلى النهاية، بما في ذلك التحقق من صحة البيانات وتخزينها، بالإضافة إلى التعامل مع أي أخطاء قد تظهر خلال العملية.



الشكل2: عملية رفع بيانات الطلاب

2. عملية إدارة المراقبين

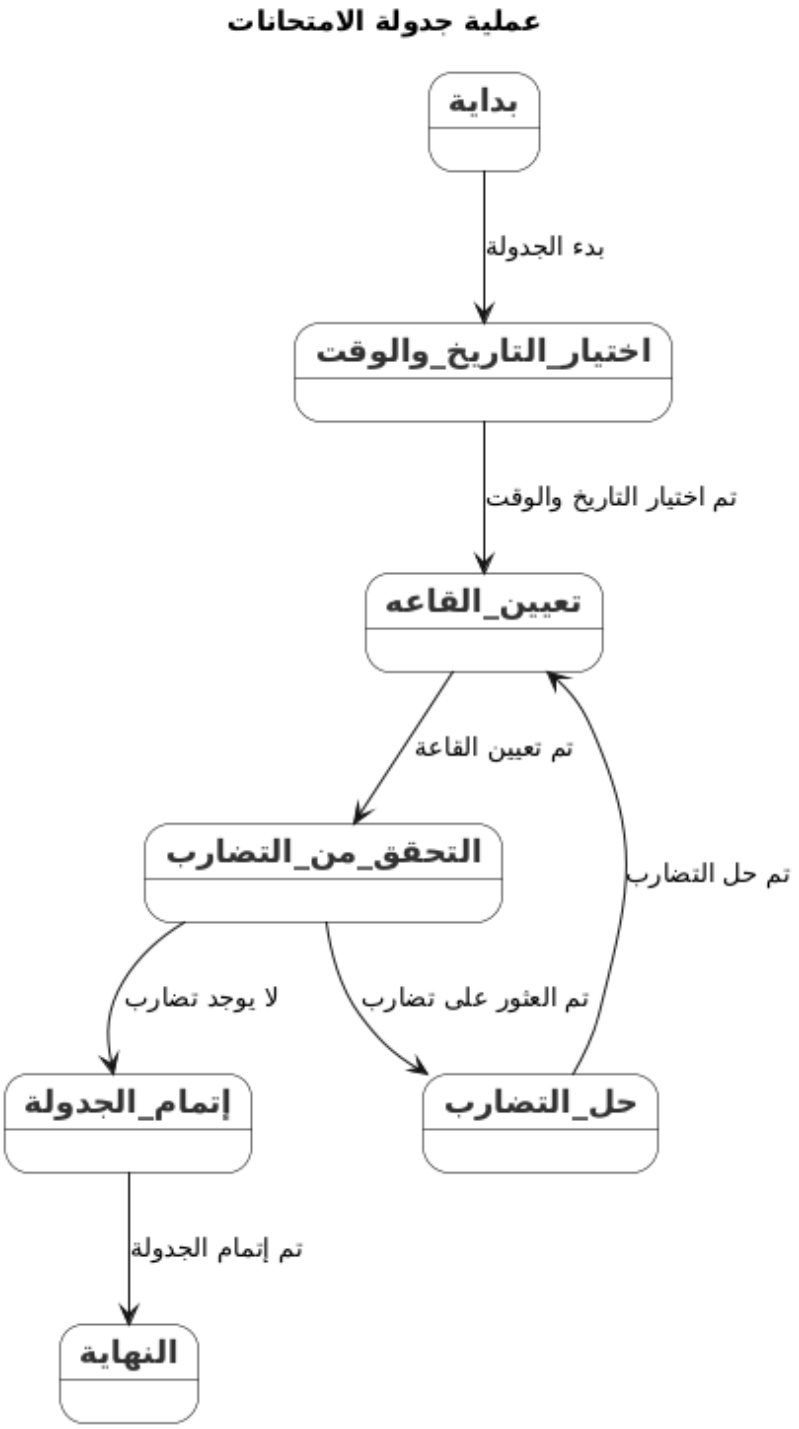
يمثل **الشكل 3** الإجراءات المتبعة لإدارة المراقبين، بدءًا من إضافتهم إلى النظام، تحديث توفرهم، وتعيينهم للامتحانات، مع إمكانية إضافة مراقبين جدد عند الحاجة.



الشكل3: عملية إدارة المراقبين

3. عملية جدولة الامتحانات

الشكل 4 يعرض كيفية تنظيم وجدولة الامتحانات، بدءًا من تحديد التاريخ والوقت، تعيين القاعات، والتحقق من تضارب الجداول، وصولاً إلى حل هذه التضاربات وإتمام الجدولة.



الشكل4: عملية جدولة الامتحانات

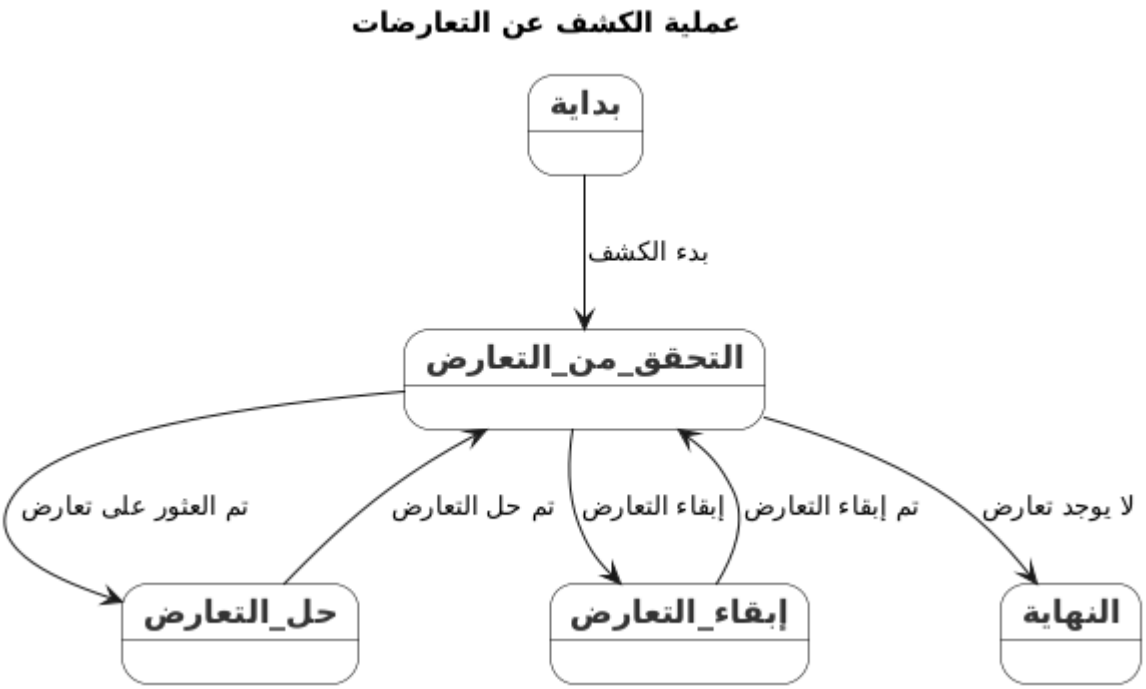
4. عملية الكشف عن التعارضات

آلية الكشف عن التعارضات: النظام يركز على كشف الطلاب الذين لديهم أكثر من امتحان مقرر في نفس الوقت باستخدام آلية دقيقة وفعّالة. تبدأ العملية بالتحقق من صحة المدخلات مثل تاريخ ووقت الامتحان، لضمان أن البيانات المدخلة كاملة وصحيحة. بعدها، يتم تنفيذ استعلام على قاعدة البيانات لتحديد الطلاب المسجلين في أكثر من امتحان في نفس الموعد.

يعمل النظام على تصنيف البيانات باستخدام معرفات الطلاب (`studentID`) مع استخدام دالة العدّ (`COUNT`) لتحديد التعارضات. في حال وجود تعارضات، يظهر النظام نقطة حمراء بجانب التاريخ والوقت المتأثرين. عند وضع المستخدم مؤشر الماوس فوق النقطة، تظهر تفاصيل التعارضات بشكل فوري، متضمنة أسماء الطلاب وجدول الامتحانات المتعارضة. هذه الميزة تتيح للمسؤول الاطلاع على التفاصيل بدقة لاتخاذ القرارات المناسبة.

أما إذا لم يتم العثور على أي تعارضات، فلا يتم عرض أي إشارات أو رسائل، مما يعني أن الجدول خالٍ من المشاكل وجاهز للاعتماد. هذه الطريقة البصرية المبسطة تسهّل التعرف على التعارضات وتوفر تجربة استخدام مريحة وفعّالة.

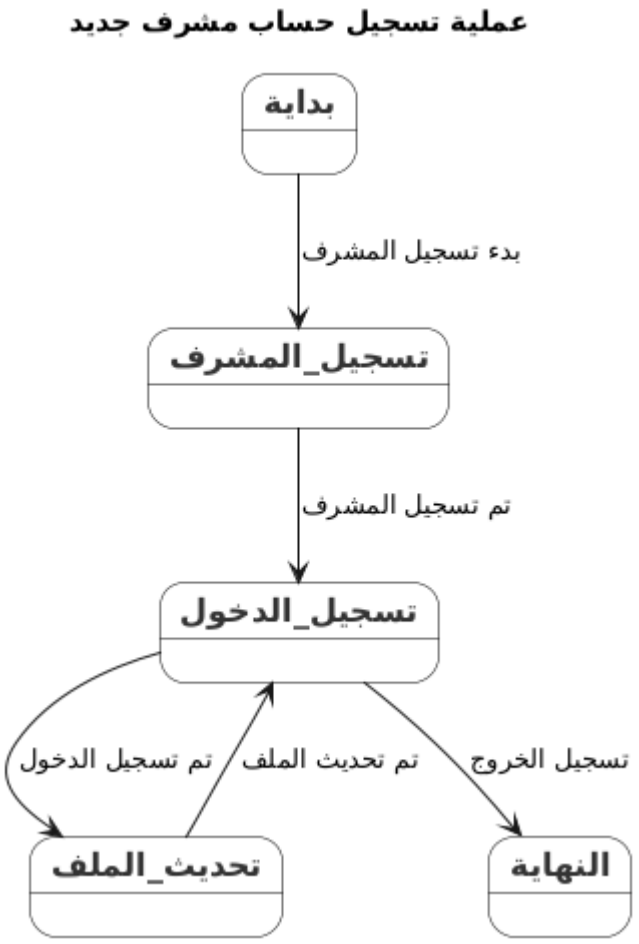
حيث يوضح **الشكل 5** آلية الكشف عن التعارضات المحتملة في جدول الامتحانات، وتوفير الحلول المناسبة لتجاوز هذه التعارضات، مع إمكانية إعادة التحقق من الوضع بعد المعالجة.



الشكل5: عملية الكشف عن التعارضات

5. عملية إدارة حساب المشرف

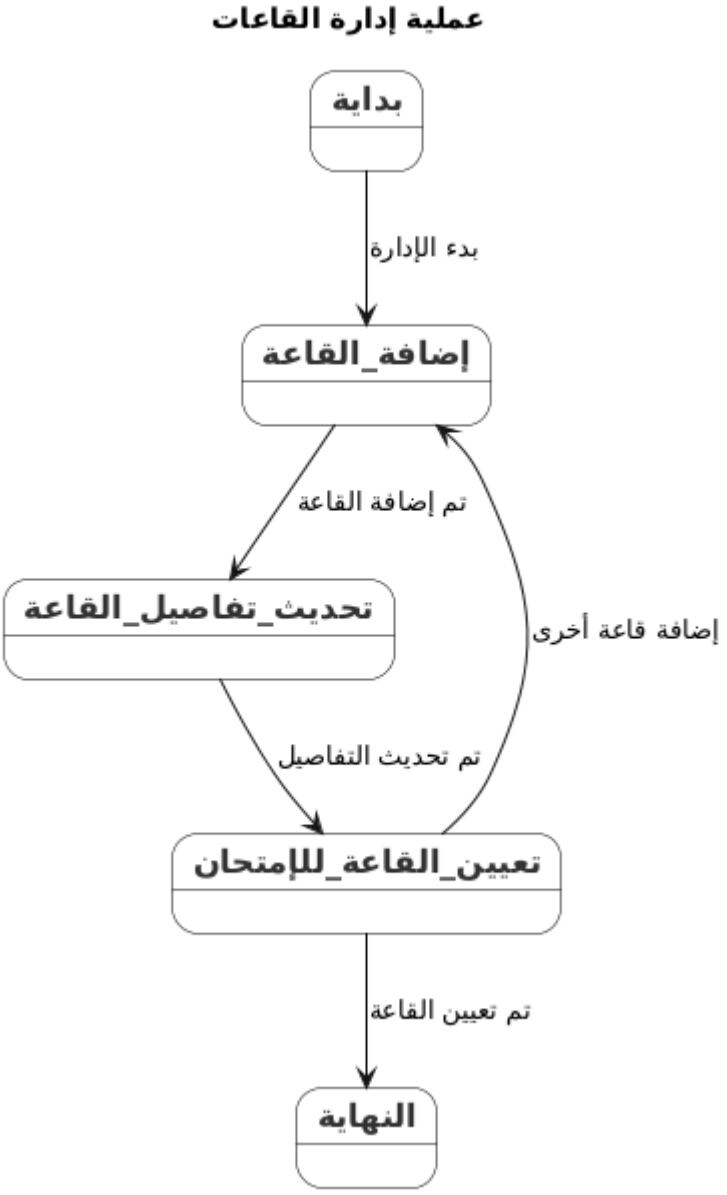
الشكل 6 يوضح كيفية إدارة حسابات المشرفين، بدءًا من عملية التسجيل، تسجيل الدخول، تحديث الملف الشخصي، وصولاً إلى تسجيل الخروج، مع ضمان تسلسل الإجراءات بشكل سليم.



الشكل6: عملية إدارة حساب المشرف

6. عملية إدارة القاعات

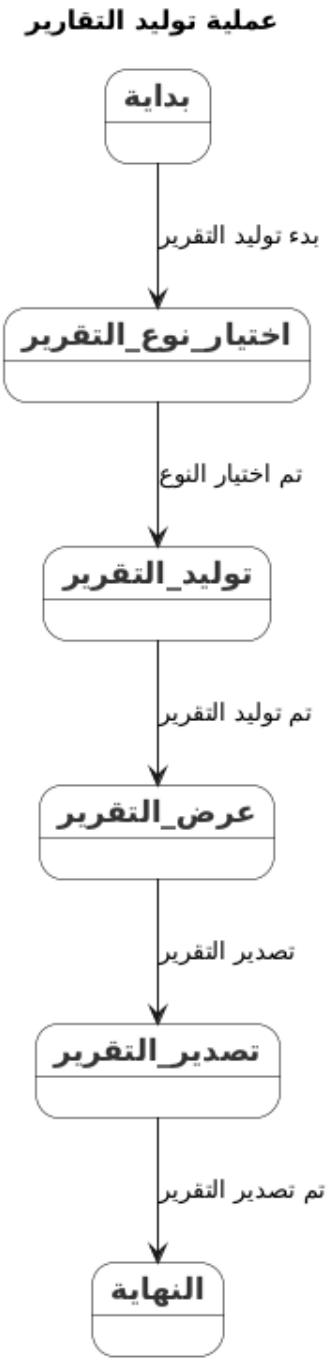
الشكل 7 يمثل كيفية إدارة القاعات في النظام، بما في ذلك إضافة قاعات جديدة، تحديث تفاصيل القاعات، وتعيين القاعات للامتحانات، مع إمكانية إضافة قاعات أخرى حسب الحاجة.



الشكل7: عملية إدارة القاعات

7. عملية توليد التقارير

الشكل 8 يُظهر كيفية توليد التقارير، بدءًا من اختيار نوع التقرير المطلوب، ثم توليده وعرضه، وأخيرًا تصديره بشكل مناسب، لضمان الحصول على التقرير بالشكل الذي يناسب المتطلبات.



الشكل8: عملية توليد التقارير

2.6 تحليل المتطلبات الوظيفية

جدول2: رفع بيانات الطلاب

وصف الوظيفة	يسمح للمسؤول برفع ملفات Excel/CSV تحتوي على بيانات الطلاب.
المستخدم	مسؤول النظام (Admin)
المدخلات	ملف Excel/CSV يحتوي على بيانات الطلاب.
العملية	التحقق من صحة البيانات، تخزينها في قاعدة البيانات.
المخرجات	بيانات الطلاب محفوظة في النظام، تقرير بالأخطاء إن وجدت.
الشروط والقيود	يجب أن يكون الملف بتنسيق Excel أو CSV.
الأداء المتوقع	أقل من 3 ثوانٍ عند تحميل 5000 سجل.

جدول3: إدخال بيانات المراقبين

وصف الوظيفة	إدخال بيانات المراقبين يدويًا أو عبر ملف Excel.
المستخدم	مسؤول النظام (Admin)
المدخلات	اسم المراقب وأوقات التفرغ.
العملية	إدخال يدوي أو رفع ملف، التحقق من عدم التكرار، تخزين المعلومات.
المخرجات	قائمة محدثة بالمراقبين وأوقات التفرغ.
الشروط والقيود	لا يمكن إضافة نفس المراقب مرتين.

جدول4: كشف التعارضات

وصف الوظيفة	تحليل جدول الامتحانات واكتشاف التعارضات المحتملة.
المستخدم	مسؤول النظام، إدارة الكلية
المدخلات	جدول الامتحانات، بيانات الطلاب، بيانات القاعات.
العملية	مقارنة مواعيد الامتحانات، التحقق من قدرة القاعات، تحليل أوقات المراقبين.
المخرجات	تقرير مفصل بالتعارضات مع حلول مقترحة.
الشروط والقيود	لا يمكن للطالب تقديم أكثر من امتحان واحد في نفس الوقت.

جدول5: تعيين القاعات

وصف الوظيفة	مطابقة القاعات مع عدد الطلاب المسجلين.
المستخدم	مسؤول النظام
المدخلات	بيانات القاعات وعدد الطلاب لكل امتحان.
العملية	مطابقة القاعات وفقًا للسعة والتوافر، توزيع الطلاب.
المخرجات	قائمة القاعات المخصصة، تقرير السعة المتبقية.
الشروط والقيود	لا يمكن تجاوز سعة القاعة.

جدول6: جدولة الامتحانات

وصف الوظيفة	جدولة الامتحانات باستخدام واجهة السحب والإفلات.
المستخدم	مسؤول النظام
المدخلات	بيانات الامتحانات، القاعات، والمراقبين.
العملية	استخدام السحب والإفلات، التحقق من التعارضات.
المخرجات	جدول امتحانات جاهز، إشعارات التعارض.
الشروط والقيود	يجب تعيين مراقب لكل قاعة.

جدول7: تقارير القاعات

وصف الوظيفة	عرض تقارير توزيع القاعات والطلاب غير المخصصين لقاعات.
المستخدم	إدارة الكلية، المسؤولون الأكاديميون
المدخلات	بيانات القاعات، جدول الامتحانات.
العملية	تحليل القاعات المشغولة، تحديد الطلاب غير المخصصين.
المخرجات	تقرير إداري حول القاعات الممتلئة والطلاب غير المخصصين.
الشروط والقيود	يجب أن يحتوي التقرير على بيانات محدثة.

جدول8: التقارير النهائية

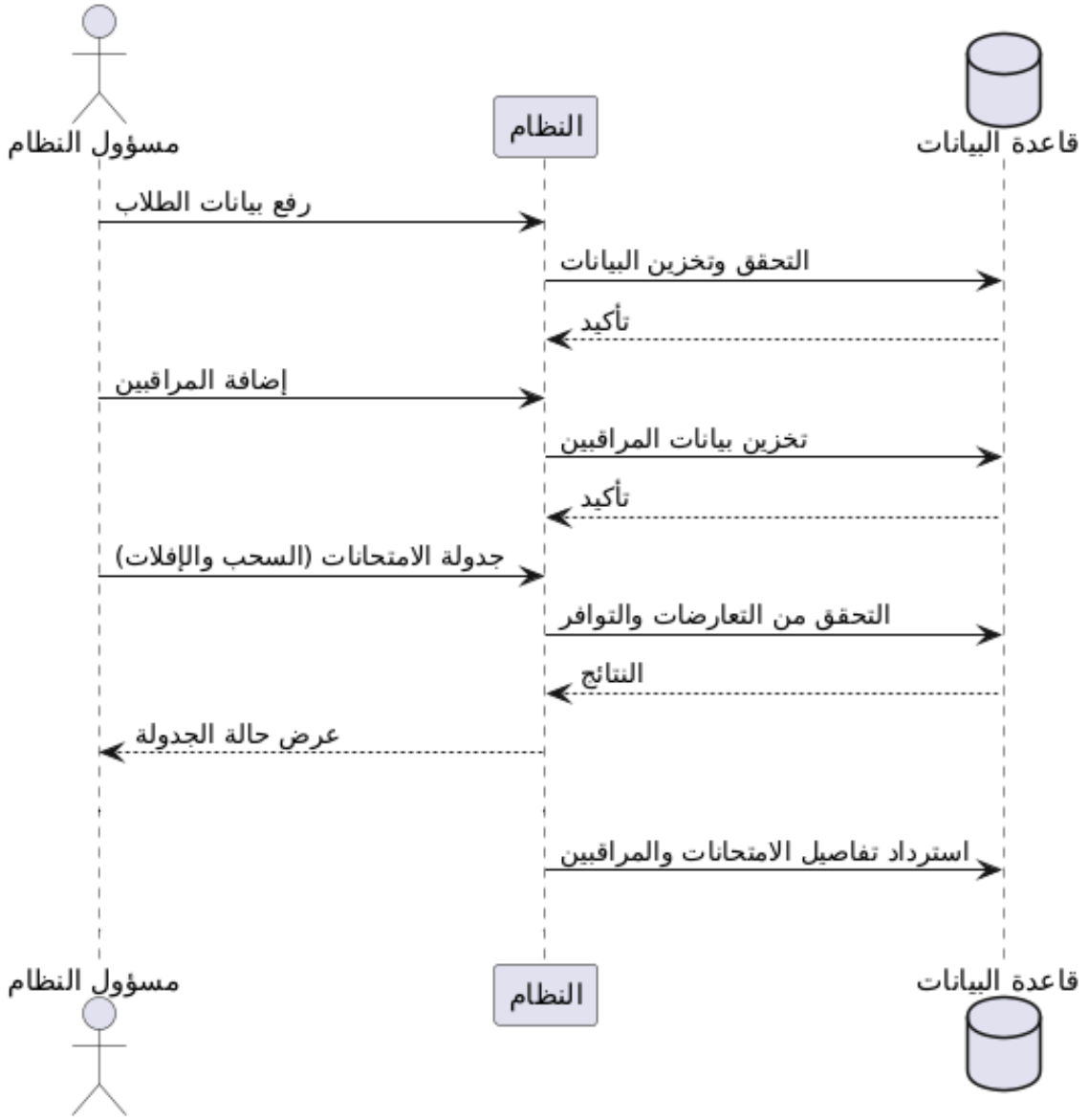
وصف الوظيفة	ملخص شامل لجداول الامتحانات والمراقبين والتوزيع النهائي للقاعات.
المستخدم	إدارة الكلية، المسؤولون الأكاديميون
المدخلات	بيانات الجداول، الامتحانات، والمراقبين.
العملية	إنشاء تقرير نهائي، تصديره إلى ملفات قابلة للطباعة.
المخرجات	تقرير مفصل قابل للطباعة.
الشروط والقيود	يجب أن يتضمن التقرير جميع البيانات المهمة.

جدول9: توسع النظام ليشمل جميع كليات الجامعة

وصف الوظيفة	إدارة جداول الامتحانات لجميع كليات الجامعة.
المستخدم	إدارة الجامعة، مسؤولو الكليات
المدخلات	بيانات الطلاب، القاعات، والمراقبين الخاصة بكل كلية.
العملية	تخصيص بيانات كل كلية بشكل مستقل، ضمان التنظيم الصحيح.
المخرجات	نظام موحد لجميع الكليات.
الشروط والقيود	يجب تخصيص بيانات كل كلية بشكل مستقل.

Sequence Diagram 2.7

الشكل 9 يمثل تسلسل العمليات بين المسؤول والنظام وقاعدة البيانات. يوضح كيفية تنفيذ الأنشطة الرئيسية مثل رفع بيانات الطلاب، إضافة المراقبين، جدولة الامتحانات، التحقق من التعارضات، وإرسال الإشعارات. الهدف هو توفير فهم واضح لتدفق البيانات والتفاعل داخل النظام.



الشكل9: Sequence Diagram

3. قاعدة البيانات والتطوير التقني

3.1 تفاصيل تقنية النظام

1. الواجهة الأمامية (Frontend):

تم تصميم الواجهة الأمامية للنظام باستخدام **Angular Framework**، الذي يُعد من أفضل الأطر لتطوير تطبيقات الويب التفاعلية. يوفر Angular العديد من المميزات التي تُسهم في بناء نظام قوي وسهل الاستخدام، مع التركيز على الأداء وتجربة المستخدم.

1. **تصميم واجهة مستخدم تفاعلية وسهلة الاستخدام**
تعتمد الواجهة الأمامية على مكونات **Angular Material** لتوفير تصميم عصري وبديهي يضمن سهولة التفاعل مع النظام للمستخدمين النهائيين. تم الاهتمام بجعل كل عناصر الواجهة مرنة وسهلة التنقل.
2. **إدارة الحالة**
يتم استخدام خدمات **Angular Services** لإدارة حالة التطبيق بشكل مركزي، مما يضمن تزامن البيانات بين الواجهة الأمامية والواجهة الخلفية. هذه الطريقة تقلل من تعقيد النظام وتجعل تجربة المستخدم سهلة.
3. **الأداء والاستجابة**
تم تحسين الأداء من خلال استخدام تقنيات مثل **Lazy Loading** لتحميل الأجزاء الضرورية فقط من التطبيق عند الحاجة. يساعد هذا النهج في تسريع تحميل الصفحات وتحسين الاستجابة للمستخدمين.
4. **قابلية الصيانة والتوسع**
يعتمد Angular على بنية معيارية تسهل تقسيم التطبيق إلى مكونات صغيرة ومستقلة. هذه البنية تتيح إجراء التعديلات أو إضافة ميزات جديدة دون التأثير على باقي أجزاء النظام، مما يجعل التطبيق قابلاً للصيانة والتوسع في المستقبل.
5. **ميزات إضافية**
   - تفاعلية عالية تعزز تجربة المستخدم.
   - دعم تقنيات حديثة لضمان التوافق مع مختلف الأجهزة والمتصفحات.

باستخدام Angular، تضمن الواجهة الأمامية للنظام تجربة مستخدم استثنائية، مع التركيز على الأداء والمرونة، مما يساعد على تحقيق أهداف النظام بكفاءة.

2. الواجهة الخلفية (Backend):

تم تطوير الواجهة الخلفية للنظام باستخدام **Node.js** مع إطار العمل **Express.js** لتوفير بيئة خفيفة وفعالة لإدارة الطلبات وتنفيذ العمليات. يركز تصميم الواجهة الخلفية على الأداء، الأمان، وقابلية التوسع لتلبية احتياجات النظام الحالي والمستقبلية.

1. **إدارة البيانات**
تعتمد الواجهة الخلفية على واجهات برمجية (APIs) لتنفيذ العمليات الأساسية على قاعدة البيانات، مثل الإضافة والتحديث والاستعلام والحذف (CRUD). هذا النهج يضمن تفاعلاً سلسًا بين الواجهة الأمامية والبيانات المخزنة.

2. **الاستجابة السريعة**

باستخدام خصائص **Node.js** لمعالجة العمليات غير المتزامنة (**Asynchronous Operations**)، يتم ضمان استجابة النظام السريعة حتى في ظل زيادة الطلبات أو البيانات الضخمة. يسمح هذا النهج بتنفيذ مهام متعددة دون تأخير أو تأثير على أداء النظام.

3. **الأمان**

لضمان حماية البيانات، يتم استخدام تقنيات المصادقة مثل **JSON Web Tokens (JWT)**، التي توفر طريقة آمنة ومشفرة للتحقق من هوية المستخدمين. يضمن ذلك أن العمليات والبيانات داخل النظام محمية ضد الوصول غير المصرح به.

4. **التوسع**

تم تصميم البنية الخلفية بطريقة تدعم **التوسع الأفقي**، مما يسمح بإضافة موارد جديدة (مثل خوادم إضافية) بسهولة حسب احتياجات النظام. هذه البنية تجعل النظام قابلاً للنمو دون التأثير على الأداء.

5. **ميزات إضافية**

- إدارة فعالة للطلبات بفضل Express.js.
- دعم كامل للتعامل مع البيانات الكبيرة من خلال تحسين مسارات المعالجة.
- تصميم يدعم قابلية التكامل مع أنظمة أخرى مستقبلاً.

تمثل الواجهة الخلفية العمود الفقري للنظام، حيث تضمن الأداء العالي، الأمان، والمرونة، مما يسهم في تقديم نظام قوي يلبي تطلعات المستخدمين بشكل كامل.

3.2 تفاصيل قاعدة البيانات

1. قاعدة البيانات (Database)

تم اختيار **MySQL** كنظام إدارة قاعدة البيانات لتميزها كأحد أشهر أنظمة إدارة قواعد البيانات العلائقية، حيث توفر أداءً عالياً، موثوقية كبيرة، وسهولة التكامل مع **Node.js**. يضمن هذا الاختيار تلبية المتطلبات الوظيفية للنظام بكفاءة واستقرار.

1. **تصميم قاعدة البيانات**

صُممت قاعدة البيانات بشكل يضمن تخزين البيانات بطريقة منظمة وفعّالة. تم الاعتماد على هيكلة واضحة باستخدام الجداول والفهارس، مما يساعد على تحسين سرعة الوصول إلى البيانات وتسهيل إدارتها. تم الأخذ في الاعتبار تلبية احتياجات النظام الحالية وإمكانية التوسع المستقبلي.

2. **إدارة العلاقات**

تُتيح MySQL إدارة العلاقات بين الجداول بشكل مرن ومنظم، مما يعزز من تكامل البيانات ودقتها. يتيح هذا الربط بين البيانات المختلفة مثل الطلاب، الامتحانات، القاعات، والمراقبين، مما يدعم العمليات المعقدة داخل النظام بسلاسة.

3. **الأمان والنسخ الاحتياطي**

تم تطبيق أفضل ممارسات الأمان لحماية البيانات، مثل التشفير وحماية الوصول باستخدام سياسات محددة للمستخدمين. كما تم إعداد آليات النسخ الاحتياطي بشكل دوري لضمان استعادة البيانات بسهولة في حالة حدوث أي خلل أو فقدان للبيانات.

4. **التكامل مع Node.js**

بفضل التكامل السلس بين MySQL وNode.js، يتم ضمان استجابة النظام السريعة عند التعامل مع البيانات الكبيرة. كما يدعم هذا التكامل المرونة في تنفيذ العمليات والتكيف مع متطلبات المستخدمين المتغيرة.

2. الجداول الرئيسية:

1. جدول الطلاب

يُستخدم جدول الطلاب لتخزين بيانات الطلبة بشكل منظم، حيث يحتوي على معرّف فريد (studentId) لكل طالب لضمان عدم التكرار. بالإضافة إلى ذلك، يتم تسجيل اسم الطالب وبريده الإلكتروني، مع تمييز حقل البريد الإلكتروني (email) بخاصية التفرد لضمان دقة البيانات وتسهيل عمليات التواصل الأكاديمي.

جدول10: Student

الحقل	النوع	الطول	الخصائص	Null	الوصف
studentId	int	INT(11)	PK	NOT NULL	معرف الطالب الفريد
name	string	50	-	NOT NULL	اسم الطالب
email	string	100	UNIQUE	NOT NULL	البريد الإلكتروني للطالب

2. جدول التسجيل

يُدير جدول التسجيل العلاقة بين الطلاب والمقررات الدراسية التي يتم تسجيلهم فيها. يحتوي الجدول على معرّف فريد (enrollmentId) لكل عملية تسجيل، ويرتبط بالطلاب عبر حقل studentId وبالمقررات عبر حقل courseId كمفاتيح خارجية، مما يضمن إدارة دقيقة ومنظمة للبيانات.

جدول11: Enrollment

الحقل	النوع	الطول	الخصائص	Null	الوصف
enrollmentId	int	INT(11)	PK	NOT NULL	معرف التسجيل الفريد
studentId	int	INT(11)	FK	NOT NULL	معرف الطالب المرتبط بالتسجيل
courseId	int	INT(11)	FK	NOT NULL	معرف المقرر المرتبط بالتسجيل

3. جدول المساقات

يُعد جدول المساقات المسؤول عن تخزين بيانات المواد الدراسية المتاحة في النظام. يحتوي الجدول على معرّف فريد لكل مساق (courseId)، مما يضمن عدم التكرار بين المساقات. بالإضافة إلى ذلك، يتم تسجيل اسم المساق (courseName) واسم الكلية التي ينتمي إليها المساق (College)، حيث يتم تخصيص حقلين منفصلين لكل منهما بسعة تصل إلى 100 حرف، مما يتيح تسجيل أسماء واضحة وكاملة.

جدول12: Course

الحقل	النوع	الطول	الخصائص	Null	الوصف
courseId	int	INT(11)	PK	NOT NULL	معرف المقرر الفريد
courseName	string	100	-	NOT NULL	اسم المقرر الدراسي
College	string	100	-	NOT NULL	اسم الكلية

4. جدول الامتحانات

جدول الامتحانات يُستخدم لتخزين بيانات الامتحانات المجدولة في النظام. يحتوي الجدول على معرّف فريد لكل امتحان (examId) لضمان تمييز كل امتحان عن الآخر. كما يشمل اسم الامتحان (examName) وحقلين إضافيين لتسجيل تاريخ الامتحان (date) ووقته (time)، مما يضمن إدارة دقيقة للجدولة الزمنية للامتحانات.

جدول13: Exam

الحقل	النوع	الطول	الخصائص	Null	الوصف
examId	int	INT(11)	PK	NOT NULL	معرف الامتحان الفريد
examName	string	100	-	NOT NULL	اسم الامتحان
date	Date	-	-	NOT NULL	تاريخ الامتحان
time	Time	-	-	NOT NULL	وقت الامتحان

5. جدول المواعيد

يمثل جدول المواعيد أحد المكونات الأساسية للنظام، حيث يُستخدم لتنظيم ربط الامتحانات بالمواعيد الزمنية المحددة. يحتوي الجدول على معرّف فريد لكل موعد (scheduleId) لضمان تمييز كل عملية جدولة عن الأخرى. كما يشمل حقل examId الذي يربط المواعيد ببيانات الامتحانات المجدولة، مما يضمن تكاملاً دقيقًا وتنظيمًا فعالًا لعملية الجدولة.

جدول14: Schedule

الحقل	النوع	الطول	الخصائص	Null	الوصف
scheduleId	int	INT(11)	PK	NOT NULL	معرف جدول المواعيد الفريد
examId	int	INT(11)	FK	NOT NULL	معرف الامتحان المدرج في الجدول

6. جدول حسابات المسؤولين

يُستخدم جدول حسابات المسؤولين لتخزين بيانات مسؤولي النظام بشكل آمن ومنظم. يحتوي الجدول على معرّف فريد لكل مسؤول لضمان عدم تكرار الحسابات، ويتضمن اسم المستخدم والبريد الإلكتروني بخاصية التفرد لضمان دقة البيانات. تُشفر كلمات المرور باستخدام خوارزمية **Blowfish**، التي توفر مستوى عالٍ من الأمان وحماية البيانات الحساسة. كما يشمل الجدول اسم الكلية المرتبطة بكل مسؤول، مما يتيح تخصيص الصلاحيات والمهام بناءً على احتياجات الكلية، مما يعزز تنظيم العمليات الإدارية وكفاءتها.

جدول15: AdminAccount

الحقل	النوع	الطول	الخصائص	Null	الوصف
adminId	int	INT(11)	PK	NOT NULL	معرف المسؤول الفريد
username	string	50	UNIQUE	NOT NULL	اسم المستخدم
password	string	255	ENCRYPTED	NOT NULL	كلمة المرور
email	string	100	UNIQUE	NOT NULL	البريد الإلكتروني للمسؤول
College	string	100	-	NOT NULL	اسم الكلية

7. جدول المراقبين

يمثل جدول المراقبين العنصر الأساسي لتخزين بيانات المراقبين المسؤولين عن الإشراف على الامتحانات. يحتوي الجدول على معرّف فريد لكل مراقب (InvigilatorId) لضمان تمييز كل مراقب عن الآخر. كما يشمل الجدول اسم المراقب (InvigilatorName) الذي يمكن أن يصل طوله إلى 100 حرف لتوفير معلومات دقيقة. بالإضافة إلى ذلك، يحتوي الجدول على حقل availability لتحديد توفر المراقب (متاح/غير متاح)، مما يساعد في تنظيم الجدولة وتوزيع المهام بسهولة.

جدول16: Invigilator

الحقل	النوع	الطول	الخصائص	Null	الوصف
InvigilatorId	int	INT(11)	PK	NOT NULL	معرف المراقب الفريد
InvigilatorName	string	100	-	NOT NULL	اسم المراقب
availability	string	20	-	NULL	توفر المراقب (متاح/غير متاح)

8. جدول القاعات

يُستخدم جدول القاعات لتخزين البيانات المتعلقة بالقاعات المتاحة لعقد الامتحانات. يحتوي الجدول على معرّف فريد لكل قاعة (hallId) لضمان عدم تكرار السجلات. بالإضافة إلى ذلك، يتضمن الجدول حقل capacity الذي يُحدد السعة القصوى للقاعة، وحقل location الذي يصف موقع القاعة بدقة. هذا التصميم يساهم في تنظيم استخدام القاعات وضمان توزيعها بشكل مثالي بما يتناسب مع عدد الطلاب والامتحانات المجدولة.

جدول17: Hall

الحقل	النوع	الطول	الخصائص	Null	الوصف
hallId	int	INT(11)	PK	NOT NULL	معرف القاعة الفريد
capacity	string	10	-	NULL	سعة القاعة
location	string	100	-	NULL	موقع القاعة

9. جدول توزيع المراقبين على القاعات

يُستخدم جدول توزيع المراقبين على القاعات لتنظيم عملية إشراف المراقبين على الامتحانات داخل القاعات. يحتوي الجدول على معرّف فريد لكل عملية توزيع لضمان تتبع دقيق للسجلات. يربط الجدول بين الامتحانات و القاعات والمراقبين من خلال الحقول المرتبطة، مما يضمن إدارة منظمة وفعّالة لعملية الإشراف. يساهم هذا الجدول في تحقيق توزيع عادل للمراقبين على القاعات، مما يعزز من كفاءة وسلاسة سير الامتحانات.

جدول18: InstructorHall

الحقل	النوع	الطول	الخصائص	Null	الوصف
instructoronhallId	int	INT(11)	PK	NOT NULL	معرف تسجيل المدرس في القاعة الفريد
examId	int	INT(11)	FK	NOT NULL	معرف الامتحان المرتبط
hallId	int	INT(11)	FK	NOT NULL	معرف القاعة
instructorId	int	INT(11)	FK	NOT NULL	معرف المدرس المسؤول

10. جدول المجموعات الدراسية

يُستخدم جدول المجموعات الدراسية لتنظيم المساقات ضمن مجموعات مترابطة. يحتوي الجدول على معرّف فريد لكل مجموعة ووصف للمجموعة، مما يتيح للمسؤولين تصنيف المساقات بناءً على علاقتها ببعضها. يساهم هذا الجدول في تبسيط عمليات الجدولة والربط بين المساقات المشتركة.

جدول19: course_mappings

الحقل	النوع	الطول	الخصائص	Null	الوصف
mappingId	int	INT(11)	PK	NOT NULL	معرف فريد للمجموعة
courseId	int	INT(11)	FK	NOT NULL	معرف المساق المرتبط بالمجموعة

11. جدول علاقات المساقات الدراسية

يُدير جدول علاقات المساقات الدراسية العلاقات بين المساقات الدراسية والمجموعات التي تنتمي إليها. يحتوي على معرّف فريد لكل علاقة (relationId) ويربط بين المجموعات (mappingId) والمساقات (courseId). يساعد هذا الجدول في تحسين إدارة البيانات المرتبطة بالمساقات.

جدول20: course_mapping_relations

الحقل	النوع	الطول	الخصائص	Null	الوصف
relationId	int	INT(11)	PK	NOT NULL	معرف العلاقة
mappingId	int	INT(11)	FK	NOT NULL	معرف المجموعة المرتبطة
relatedCourseId	int	INT(11)	FK	NOT NULL	معرف المساق المرتبط بالعلاقة

12. جدول أيام التوفر

يُستخدم جدول أيام التوفر لتسجيل الأيام المتاحة للمراقبين. يحتوي على معرّف اليوم (dayId) ويربطه بمعرّف المراقب (invigilatorId). من خلال هذا الجدول، يمكن جدولة المراقبين وفقًا لأيام توفرهم، مما يضمن تحسين توزيع العمل.

جدول21: availability_days

الحقل	النوع	الطول	الخصائص	Null	الوصف
dayId	int	INT(11)	PK	NOT NULL	معرف اليوم
invigilatorId	int	INT(11)	FK	NOT NULL	معرف المراقب المرتبط
day	string	20	-	NOT NULL	اليوم المتاح

13. جدول فترات التوفر الزمنية (availability_slots)

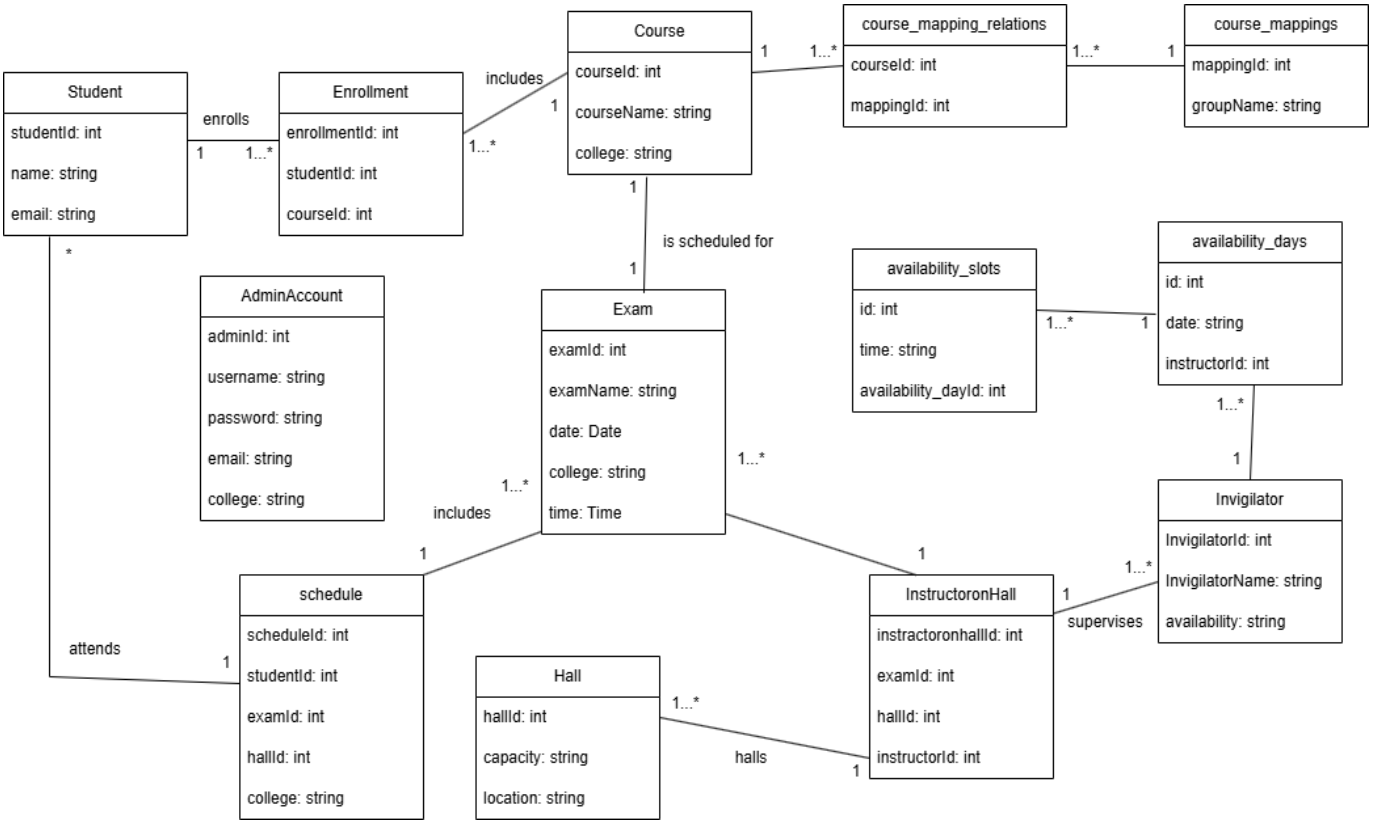
يُحدد جدول فترات التوفر الزمنية الفترات الزمنية المتاحة للمراقبين في الأيام المسجلة في جدول أيام التوفر. يحتوي على معرّف فريد لكل فترة (slotId)، وقت البداية (startTime)، ووقت النهاية (endTime)، ويرتبط بالمراقب (invigilatorId). يسهم الجدول في توزيع المهام بدقة بناءً على الفترات الزمنية المتاحة.

جدول22: availability_slots

الحقل	النوع	الطول	الخصائص	Null	الوصف
slotId	int	INT(11)	PK	NOT NULL	معرف الفترة
invigilatorId	int	INT(11)	FK	NOT NULL	معرف المراقب المرتبط
startTime	Time	-	-	NOT NULL	وقت بدء الفترة
endTime	Time	-	-	NOT NULL	وقت انتهاء الفترة

ER Diagram 3.3

الشكل 10 يوضح العلاقة بين الكيانات الرئيسية في النظام مثل الطلاب، الامتحانات، القاعات، والمراقبين. يبرز كيفية ارتباط هذه الكيانات ببعضها من خلال العلاقات مثل تسجيل الطلاب في الامتحانات، تعيين القاعات، وجدولة المراقبين. الهدف هو تقديم تصور شامل للبنية البياناتية للنظام.

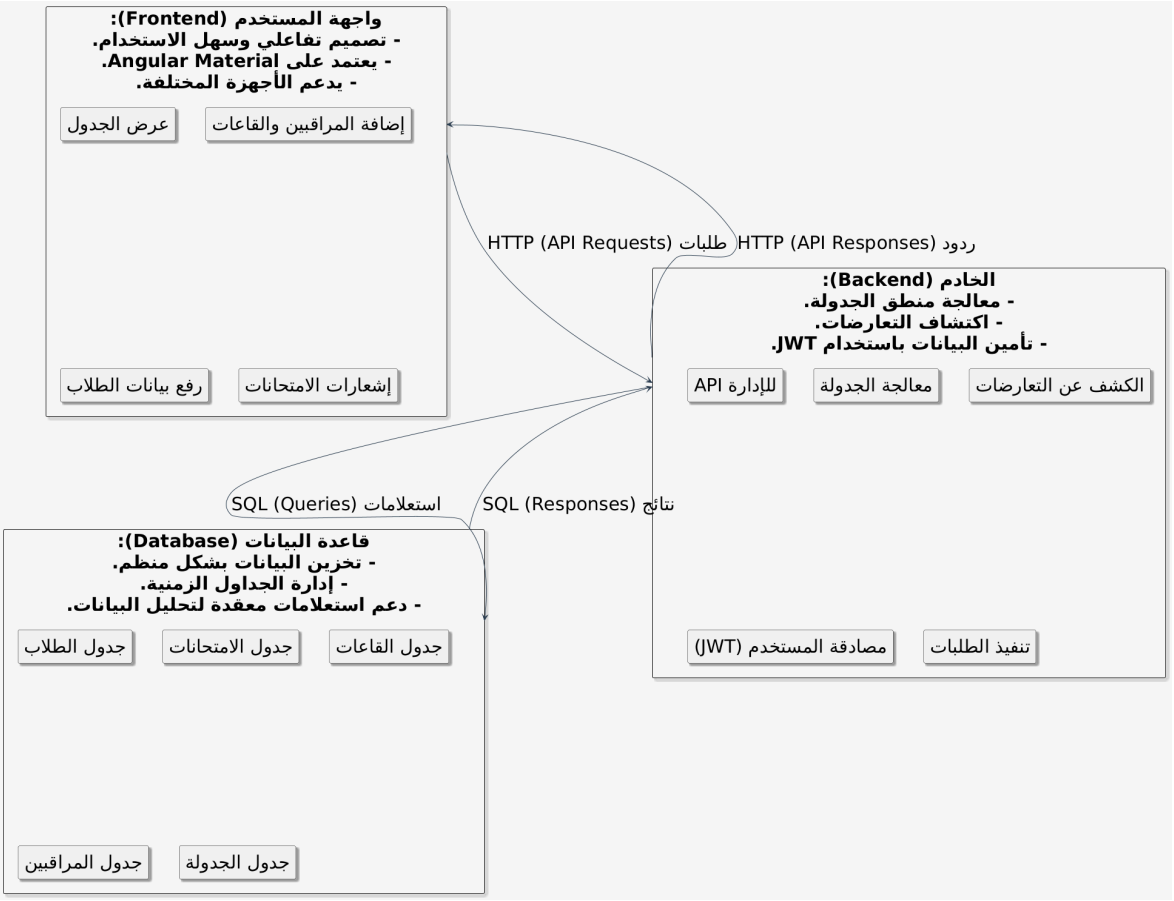


الشكل10: ER Diagram

4. تطوير النظام

Model Architectural 4.1

الشكل 11 يصف الهيكل العام للنظام وكيفية تفاعل مكوناته المختلفة، بما في ذلك الواجهة الأمامية (Frontend) والواجهة الخلفية (Backend) وقاعدة البيانات. الهدف هو توفير رؤية شاملة لكيفية توزيع العمليات والبيانات بين هذه المكونات لضمان الكفاءة والتكامل.



الشكل11: Model Architectural

4.2 خطوات التطوير

1. تطوير واجهات المستخدم

تم استخدام Angular Framework لتصميم واجهات تفاعلية تلبي احتياجات المستخدمين، مع التركيز على البساطة والفاعلية. تميزت الواجهات بميزات متقدمة مثل خاصية السحب والإفلات التي تُسهّل إعداد الجداول. بالإضافة إلى ذلك، تم تحسين تجربة المستخدم عبر تصميم متجاوب يدعم مختلف الأجهزة، مما يضمن سهولة الوصول إلى النظام واستخدامه في أي وقت ومن أي مكان. كما تم التركيز على جعل الواجهات واضحة وسهلة التنقل حتى للمستخدمين الذين ليست لديهم خبرة تقنية، مما ساهم في رفع كفاءة العمل وتقليل وقت التعلم.

2. إنشاء خوادم API

تم تطوير واجهات برمجية باستخدام Node.js وExpress.js لتوفير بيئة خفيفة وآمنة لتنفيذ العمليات. تتيح هذه الواجهات رفع ومعالجة بيانات الطلاب، إدارة القاعات والمراقبين، والكشف عن التعارضات بشكل مرن وفعّال. تم التركيز على تحسين الأداء عبر العمليات غير المتزامنة (Asynchronous Operations) لتقليل زمن الاستجابة حتى مع زيادة حجم البيانات.

تم أيضًا تعزيز الأمان باستخدام تقنيات مثل JWT لتأمين عمليات المصادقة، مما يحمي بيانات المستخدمين من الوصول غير المصرح به. بالإضافة إلى ذلك، تم تصميم الواجهات البرمجية لتكون مرنة وقابلة للتوسع، مما يتيح تطوير ميزات إضافية بسهولة مستقبلًا.

3. تصميم قاعدة البيانات

تم تصميم قاعدة البيانات باستخدام MySQL لتكون مرنة ومترابطة لدعم كافة العمليات المختلفة في النظام. تمت مراعاة الكفاءة العالية في إدارة العلاقات بين الجداول وسرعة استرجاع البيانات لضمان أداء مميز حتى مع زيادة حجم البيانات.

كما تم إنشاء هيكلية متكاملة تجمع بين البيانات الأساسية مثل بيانات الطلاب، القاعات، المراقبين، والامتحانات بطريقة مترابطة، مما يُسهل إدارة الموارد وتحسين سير العمل. تعكس قاعدة البيانات التصميم المدروس للنظام، حيث تم تحسينها لتوفير تجربة سلسة للمستخدمين وضمان دقة البيانات.

4. السودوكود: العمليات الأساسية للنظام

يتناول هذا القسم السودوكود المستخلص مباشرة من الكود البرمجي الفعلي للنظام، مما يعكس المنطق الذي تم تطويره لتنفيذ العمليات الأساسية. الهدف من تقديم السودوكود هو تبسيط فهم بنية النظام وآلية عمله، مما يساهم في تحسين عملية التطوير واختبار المكونات المختلفة. يعرض السودوكود العمليات الرئيسية مثل رفع بيانات الطلاب، إدارة القاعات والمراقبين، اكتشاف التعارضات، وجدولة الامتحانات، والتي تُشكل جوهر عمل النظام.

- **رفع بيانات الطلاب:**

في هذا الجزء من السودوكود، يتم معالجة رفع بيانات الطلاب من ملف Excel باستخدام مكتبات مثل Express و Multer و XLSX. حيث يقوم الكود بقراءة الملف المرفوع وتحويله إلى بيانات قابلة للمعالجة، ليتم استخراج معلومات الطلاب منه.

```
Function: handleFileUpload
Input: req, res
Output: Success response or error message
Begin
    Import necessary modules: express, multer, XLSX, and models
    Create an Express router
    Configure multer for file uploads using memory storage
    Define POST route for file uploads
        If no file is uploaded
            Return 400 error response with message "No file uploaded"
        Read file buffer and load it into a workbook
        Extract the first sheet from the workbook
        Convert the sheet into JSON format
        Fetch all course IDs from Course model where college matches
req.query.college
        Create a set of course IDs for validation
        Filter the JSON data
            Keep rows where:
                The third column is a number
                The course ID exists in the set of course IDs
            Map the valid rows into objects with fields: student number, name,
and course number
        Process the filtered data in batches
            For each batch:
                For each enrollment in the batch:
                    Extract student number, name, and course number
                    If student with the given student number doesn't exist
                        Create a new student record with the extracted name and
student number
                    Associate the student with the course based on course number
        Create exam records for courses with enrolled students
        Delete students who are not enrolled in any courses
        Send success response with message and current date
End
```

- **ادخال بيانات المراقبين:**

هذا الجزء من السودوكود يتعامل مع إدخال بيانات المراقبين الجدد، حيث يتحقق من وجود البريد الإلكتروني، ثم ينشئ المراقب ويربطه بالأيام والأوقات المتاحة.

```
Function: handlePostRequest
Input: req, res
Output: Success response or error message
Begin
```

```
   Try
      Extract name, email, and days from req.body
      Check if an instructor with the provided email already exists
         If true
            Return 400 error with message "البريد الإلكتروني مستخدم من قبل"
      Create a new instructor record using the extracted name and email
      For each day in the provided days
         Create an AvailabilityDays record with the date
         Link it to the new instructor using instructorID
         For each time in the day's time slots
            Create an AvailabilityTime record with the time
            Link it to the associated AvailabilityDays record
      Return 201 success response with message "تمت إضافة المراقب
{newInstructor.name}"
   Catch any error
      Return 400 error response with the error message
End
```

- **تعارضات:**

هذا الجزء من السودوكود يتعامل مع التحقق من تعارضات مواعيد الطلاب، حيث يتحقق من وجود طلاب لديهم مواعيد متعددة لنفس اليوم والوقت في جدول الامتحانات، ثم يقوم بحساب عدد هؤلاء الطلاب وإرجاع النتيجة.

```
Function: conflictStudent
Input: date, time
Output: Count of students with conflicting schedules or an error message
Begin
   If date is null or empty
      Return error message "Please provide a valid date"
   Try
      Query the database to find students with more than one schedule for
the specified date and time
         Select studentID
         From Schedule
         Where Exam.date equals the given date
               and Exam.time equals the given time
         Group by studentID
         Having count(scheduleID) greater than 1
      Count the resulting students
      Return the count
   Catch any error
      Log the error details
      Return error message "An error occurred while fetching data"
End
```

- **تحديد القاعات:**

هذا الجزء من السودوكود يتعامل مع تحديد القاعات للامتحانات، حيث يتم البحث عن الامتحان باستخدام معرّف الامتحان، ثم تحديد عدد الطلاب المرتبطين به. يتم فحص القاعات المحددة للتأكد من قدرتها على استيعاب الطلاب، وتحديث سجلات الجدولة للقاعات المناسبة بناءً على سعة القاعة وعدد الطلاب.

```
Function: POST /select-hall/:examID
Input: examID, selectedHalls
Output: Success response or error message
Begin
    Try
        Find the exam associated with the provided examID
        If the exam does not exist
            Return 404 error with message "Exam not found"
        Retrieve the studentCount for the specified exam
        For each hall in selectedHalls
            Get the hallCapacity for the current hall
            Find existing schedules in the hall
            For each existing schedule in the hall
                Retrieve the associated exam
                If dates and times match and the hall is full
                    Break loop
                Else if the hall has remaining capacity
                    Adjust hallCapacity accordingly
            If studentCount is less than or equal to hallCapacity
                Update the Schedule records for the exam:
                    Assign the current hallID and limit to studentCount
                Assign instructors from the hall to the exam
                Break loop
            Else
                Update the Schedule records for the exam:
                    Assign the current hallID and limit to hallCapacity
                Assign instructors from the hall to the exam
                Subtract hallCapacity from studentCount
        Send a 200 OK response
    Catch any error
        Return a 400 Bad Request response with the error message
End
```

- **جدولة الامتحانات:**

هذا الجزء من السودوكود يتعامل مع تحديث مواعيد الامتحانات، حيث يتم تعديل التاريخ والوقت إذا تم توفيرهما، وتحديث سجلات الطلاب والامتحانات المرتبطة. في حال حذف التاريخ والوقت، يتم إزالة السجلات

```
Function: PUT /update/:id
Input: id, req.body (date, time)
Output: Success response or error message
Begin
    Try
        Find exam by examID
```

```
    If exam does not exist
        Return 404 error with message "No exam found with the given ID."
    If date or time is provided
        Format date to "YYYY-MM-DD"
        Update the exam with the formatted date and provided time
        If the current course has course_mappings
            For each mapping in course_mappings
                For each relatedCourse in mapping
                    If relatedCourse is not the original course
                        Find relatedExam by relatedCourse.courseID
                        If relatedExam exists
                            Update relatedExam with the formatted date and time
                            Find students enrolled in relatedCourse
                            For each student
                                Create Schedule for the student and relatedExam
                            EndFor
                        EndIf
                EndFor
            EndFor
        EndIf
        Find students enrolled in the original course
        For each student
            Create Schedule for the student and the exam
        EndFor
    Else
        Delete InstructorOnHall assignments for the exam
        Delete Schedules for the exam
        Update the exam with date and time set to null
        If the current course has course_mappings
            For each mapping in course_mappings
                For each relatedCourse in mapping
                    If relatedCourse is not the original course
                        Find relatedExam by relatedCourse.courseID
                        If relatedExam exists
                            Delete InstructorOnHall assignments for relatedExam
                            Delete Schedules for relatedExam
                            Update relatedExam with date and time set to null
                        EndIf
                EndFor
            EndFor
        EndIf
    EndIf
    Send 200 OK response with success message
  Catch any error
    Log the error
    Send 500 Internal Server Error with the error message
End
```

- **التقارير:**

هذا الجزء من السودوكود يتعلق بإنشاء تقارير مواعيد الامتحانات. في التقرير الأول، يتم التحقق من البيانات المدخلة مثل اسم القاعة وموعد الامتحان، ثم إنشاء تقرير PDF يحتوي على تفاصيل الطلاب والمراقبين. في التقرير الثاني، يتم إنشاء تقرير يشمل جميع القاعات مع التفاصيل ذاتها لكل قاعة في الموعد المحدد.

```
Function: POST /generate-hall-report
Input: hallName, examDate, examTime
Output: PDF report or error message
Begin
   If hallName, examDate, or examTime is not provided
      Return 400 error with message "All data required: hallName,
examDate, examTime"
   EndIf
   Format examDate and examTime
   Find hall by hallName
   If hall does not exist
      Return 404 error with message "Hall not found"
   EndIf
   Find exams scheduled at the given examDate and examTime
   If no exams are found
      Return 404 error with message "No exams at this date and time"
   EndIf
   Find schedules associated with the hall and the exams
   If no schedules are found
      Return 404 error with message "No schedules for this hall and time"
   EndIf
   Find instructors assigned to the exams and the hall
   Create a PDF document containing:
      Header: "Hall Report for [hallName]"
      Subheaders: date, time, instructor names
      Table: student signatures, exam names, student names, student IDs
   Save the PDF document to a temporary file
   Send the PDF document to the client
   Delete the temporary PDF file after sending
End
```

```
2.
Function: POST /generate-all-halls-report
Input: examDate, examTime
Output: Consolidated PDF report for all halls or error message
Begin
   If examDate or examTime is not provided
      Return 400 error with message "All data required: examDate,
examTime"
   EndIf
   Format examDate and examTime
   Find all halls
```

```
    If no halls are found
        Return 404 error with message "No halls available"
    EndIf
    Find exams scheduled at the given examDate and examTime
    If no exams are found
        Return 404 error with message "No exams at this date and time"
    EndIf
    Find schedules for the exams across all halls
    If no schedules are found
        Return 404 error with message "No schedules for this time"
    EndIf
    Find instructors assigned to these exams for each hall
    Create a PDF document containing:
        For each hall
            Add a section with:
                Header: "Hall Report for [hallName]"
                Subheaders: date, time, instructor names
                Table: student signatures, exam names, student names, student
IDs
        EndFor
    Save the PDF document to a temporary file
    Send the PDF document to the client
    Delete the temporary PDF file after sending
End
```

- **تسجيل مشرف جديد:**

هذا الجزء من السودوكود يتعلق بتسجيل مشرف جديد، حيث يتم التحقق من وجود بريد إلكتروني مستخدم مسبقًا، وفي حال عدم وجوده، يتم إنشاء مشرف جديد مع تفاصيله.

```
Function: POST /register
Input: username, email, password, college
Output: Success response with admin details or error message
Begin
    Extract username, email, password, and college from the request body
    Check if an admin with the given email already exists
        If an admin exists
            Return 400 error with message "Email already in use"
        Else
            Create a new admin record with:
                - Username
                - Email
                - Password (securely hashed)
                - College
            Return 201 success response with message "Admin created
successfully" and the created admin's details
End
```

- **تسجيل الدخول:**

هذا الجزء من السودوكود يتعامل مع تسجيل دخول المشرف، حيث يتم التحقق من صحة البريد الإلكتروني وكلمة المرور، وفي حال النجاح يتم إنشاء رمز توثيق (JWT) للمشرف.

```
Function: POST /login
Input: email (string), password (string)
Output: HTTP response with JWT token and admin details on success, or
error message on failure
Begin
   Extract `email` and `password` from the request body
   Search for an admin record with the provided `email`
      If no admin is found
         Return 400 error response with message "Email or password
incorrect"
      Otherwise
         Compare the provided `password` with the stored hashed password
            If the passwords match
               Generate a JWT containing the following payload:
                  - Admin ID
                  - Email
                  - College
               Set the JWT in the HTTP response as a secure, HTTP-only
cookie
               Return 200 success response with:
                  - Message: "Login successful"
                  - Email: The admin's email address
                  - College: The admin's associated college
                  - Token: The generated JWT
            Otherwise
               Return 400 error response with message "Email or password
incorrect"
End
```

- **التحقق من التوثيق باستخدام JWT:**

هذا الجزء من السودوكود يتعامل مع التحقق من توثيق المستخدم عبر JWT، حيث يتم التحقق من وجود الرمز في الكوكيز، وإذا كان صالحًا يتم التحقق من صحته وإرفاقه مع لطلب، وفي حال عدم التوثيق أو وجود رمز غير صالح يتم إرسال رسالة خطأ.

```
Function: authenticate
Input: HTTP request with cookies containing a JWT
Output: Proceed to the next middleware or return an authentication error
Begin
   Retrieve the JWT from the request cookies
   If no token is found
      Return 401 error with message "Authentication required"
```

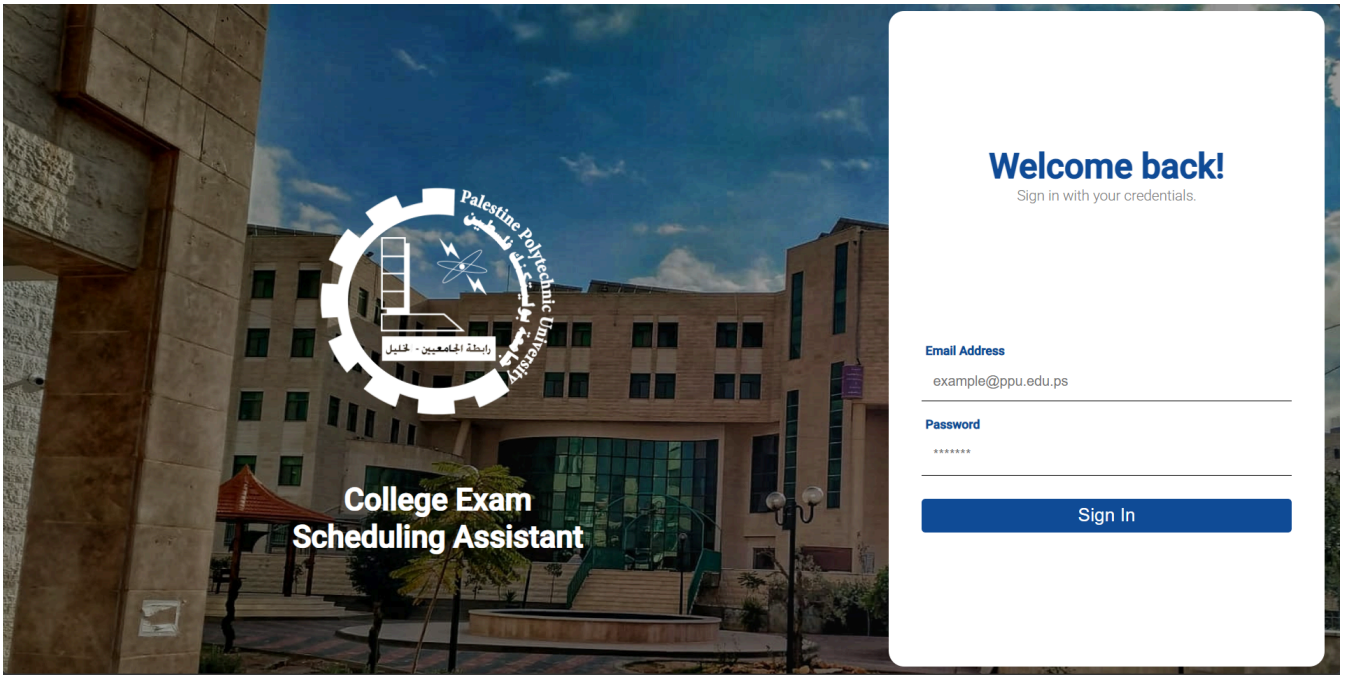
```
    Else
        Verify the token using the JWT secret
            If verification is successful
                Attach the decoded token data to the request object
                Proceed to the next middleware
            Else
                Return 401 error with message "Token invalid"
End
```

5. تصميم الواجهات

تم تصميم واجهات النظام بعناية فائقة لتلبية احتياجات المستخدمين مع التركيز على سهولة الاستخدام والفعالية. تعتمد الواجهات على تقنيات حديثة وواجهة مستخدم تفاعلية لضمان تجربة سلسة ومتكاملة. تشمل الواجهات الأساسية ما يلي:

1. واجهة تسجيل الدخول (Login)

يعرض **الشكل 12** واجهة تسجيل الدخول وهي البوابة الأولى للنظام، حيث توفر مستوى عالٍ من الأمان عبر المصادقة على المستخدمين. تتيح إدخال اسم المستخدم وكلمة المرور. تعتمد على تقنيات أمان مثل JWT لضمان حماية البيانات ومنع الوصول غير المصرح به.

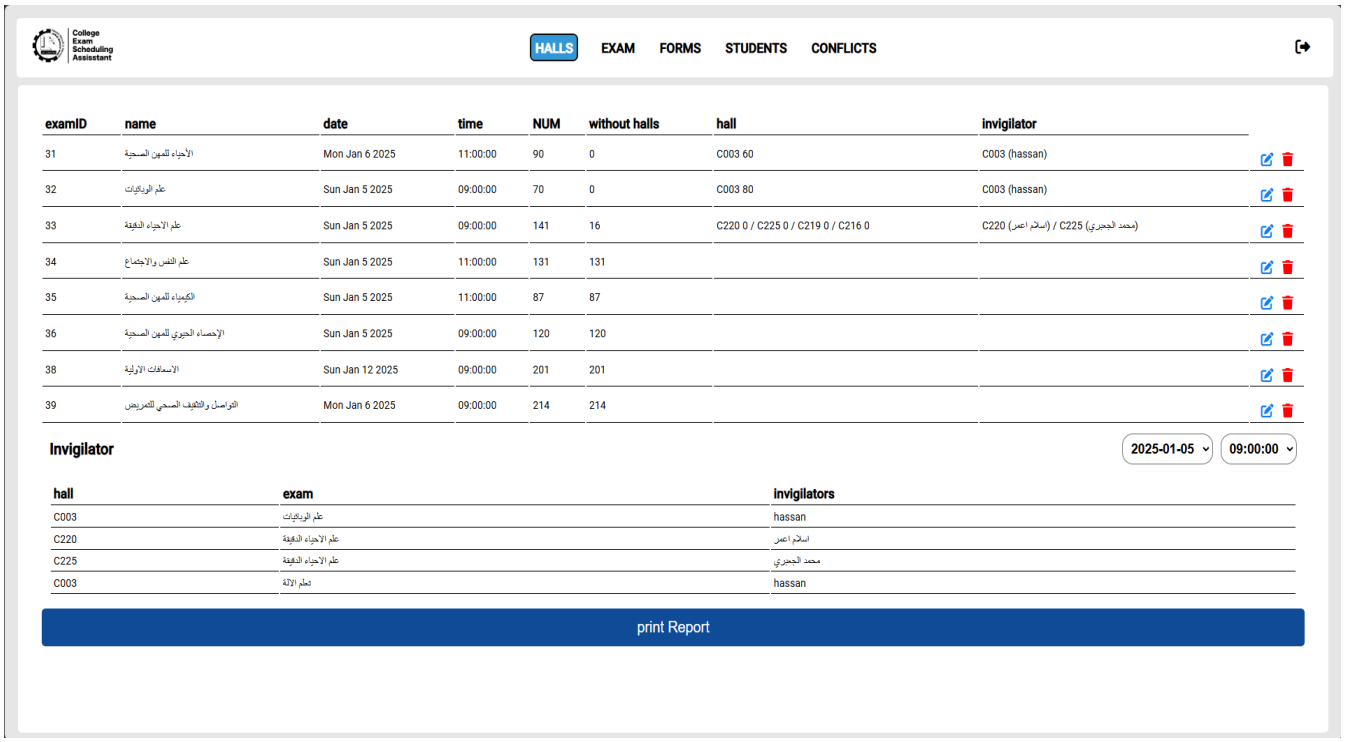


الشكل12: واجهة تسجيل الدخول (Login)

2. واجهة إدارة القاعات (HALLS)

واجهة إدارة القاعات هي إحدى الواجهات الرئيسية للنظام، حيث تُستخدم لتحديد القاعات المناسبة لكل امتحان وتنظيم عملية الإشراف عليها من خلال تعيين المراقبين. تركز الواجهة على دمج القاعات مع الامتحانات بطريقة تفاعلية تسهم في تسهيل عملية الجدولة وضمان استخدام الموارد المكانية والبشرية بكفاءة.

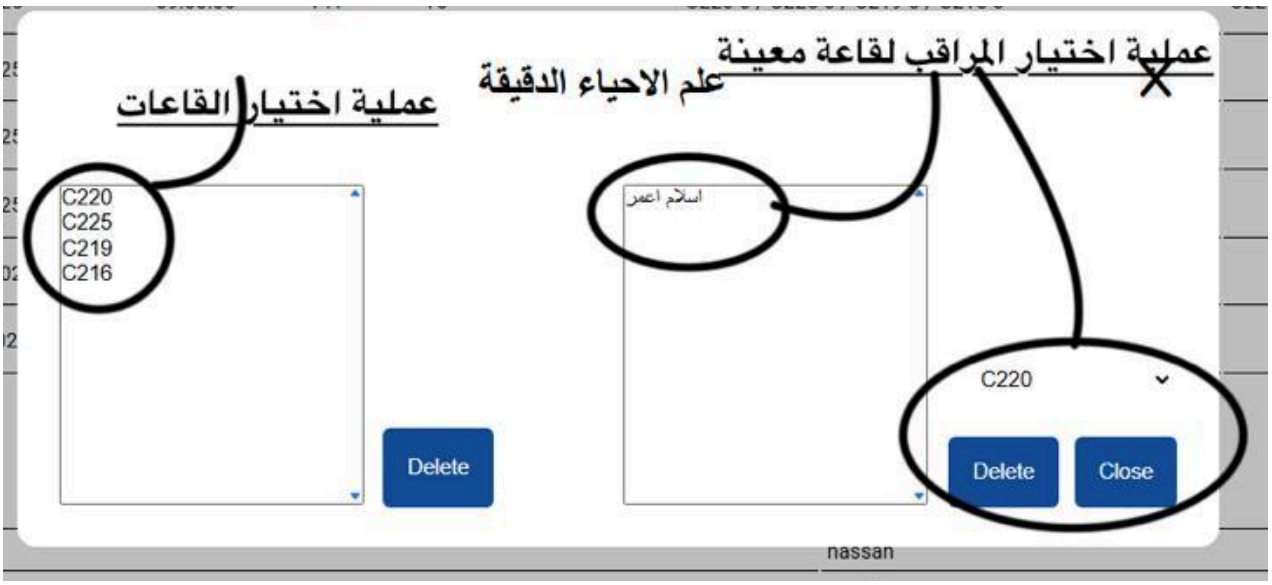
يستعرض **الشكل 13** التصميم العام لواجهة إدارة القاعات، التي تعرض جميع الخيارات والوظائف المتاحة لتخصيص القاعات والمراقبين.



الشكل13: واجهة إدارة القاعات (HALLS)

تحديد القاعات والمراقبين للامتحانات

تتيح هذه الميزة للمستخدم اختيار القاعات المناسبة والمراقبين لكل امتحان بناءً على سعة القاعة وعدد الطلاب المسجلين، مع عرض تفاصيل القاعات المتاحة. كما يمكن للمستخدم تعديل القاعات المخصصة عند الحاجة. **الشكل 14** يوضح طريقة تحديد القاعات.



الشكل14: تحديد القاعات والمراقبين للامتحانات

عرض تقارير القاعات والمراقبين

تتيح هذه الميزة للمستخدم استعراض تقارير شاملة حول القاعات والمراقبين، مثل توزيعهم وجدولة الامتحانات. يُظهر **الشكل 15** التفاصيل المتعلقة بتوزيع الموارد لضمان تنظيم مثالي للامتحانات.

Invigilator

hall	exam	invigilators
C003	علم الوبائيات	hassan
C220	علم الاحياء الدقيقة	اسلام اعمر
C225	علم الاحياء الدقيقة	محمد الجبيري
C003	تعلم الالة	hassan

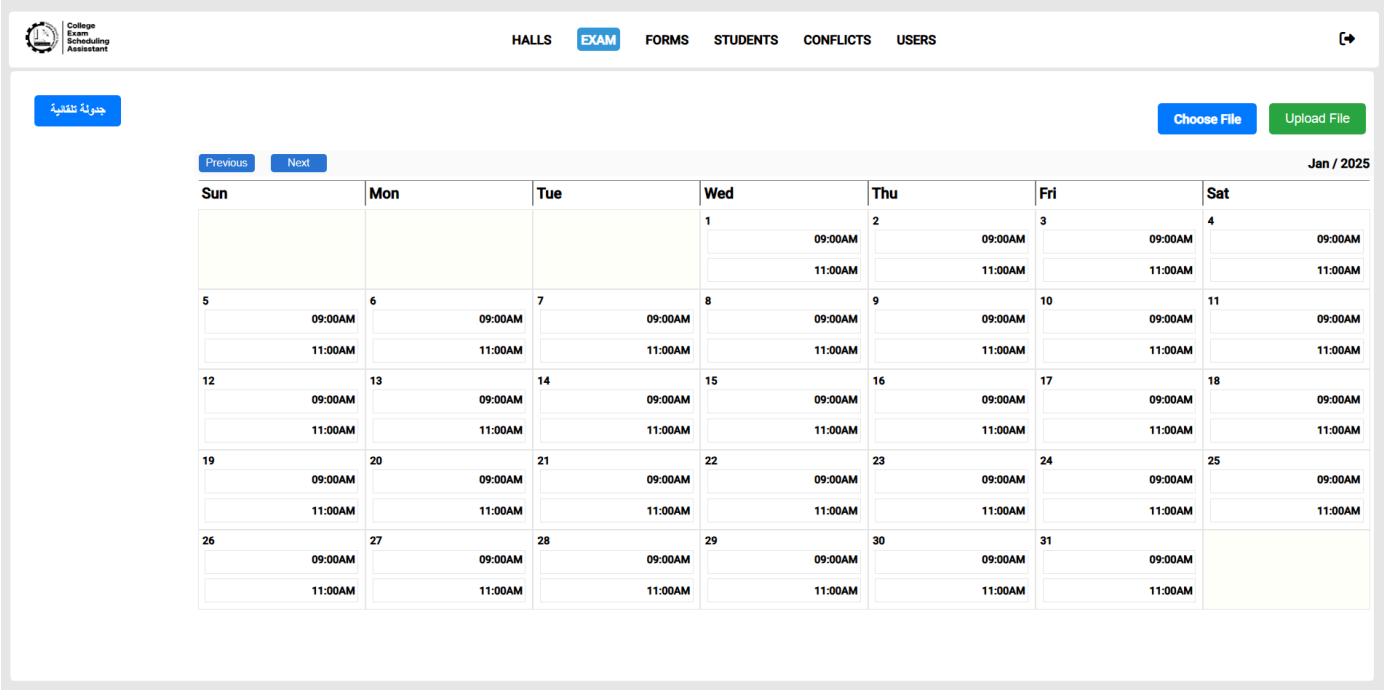
print Report

الشكل15: عرض تقارير القاعات والمراقبين

3. واجهة إدارة الامتحانات (EXAM)

واجهة إدارة الامتحانات هي إحدى الواجهات الرئيسية في النظام، حيث تتيح للمستخدمين رفع بيانات الطلاب وجدولة الامتحانات بطريقة مرنة وسهلة. تجمع الواجهة بين أدوات التفاعل الحديثة مثل السحب والإفلات وميزات كشف التعارضات لضمان تنظيم دقيق وجدولة خالية من الأخطاء.

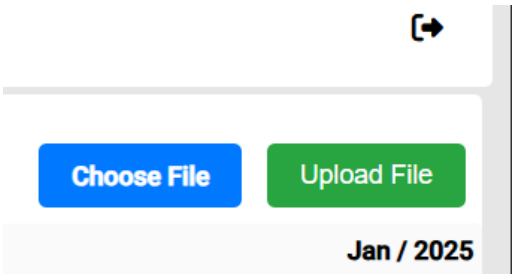
يستعرض **الشكل 16** التصميم العام لواجهة إدارة الامتحانات، الذي يُظهر جميع الوظائف الأساسية مثل رفع الملفات وجدولة المواعيد.



الشكل16: واجهة إدارة الامتحانات (EXAM)

رفع بيانات الطلاب للامتحانات

تمكن هذه الميزة المستخدم من اختيار ملف ال Excel الذي يحتوي على بيانات الطلاب مثل أسمائهم، أرقامهم الجامعية، والمساقات المسجلين بها من ثم رفعه. ثم يتم معالجة البيانات مباشرة داخل النظام مع التأكد من صحتها. يُوضح **الشكل 17** عملية رفع البيانات.



الشكل17: رفع بيانات الطلاب للامتحانات

جدولة الامتحانات باستخدام خاصية السحب والإفلات

تُوفر هذه الميزة تقويمًا تفاعليًا يمكن من خلاله تحديد مواعيد الامتحانات وإجراء تعديلات بسهولة باستخدام السحب والإفلات. يُظهر **الشكل 18** طريقة استخدام خاصية السحب والإفلات في الجدولة.

الشكل18: جدولة الامتحانات باستخدام خاصية السحب والإفلات

الية عرض التعارضات في الامتحانات

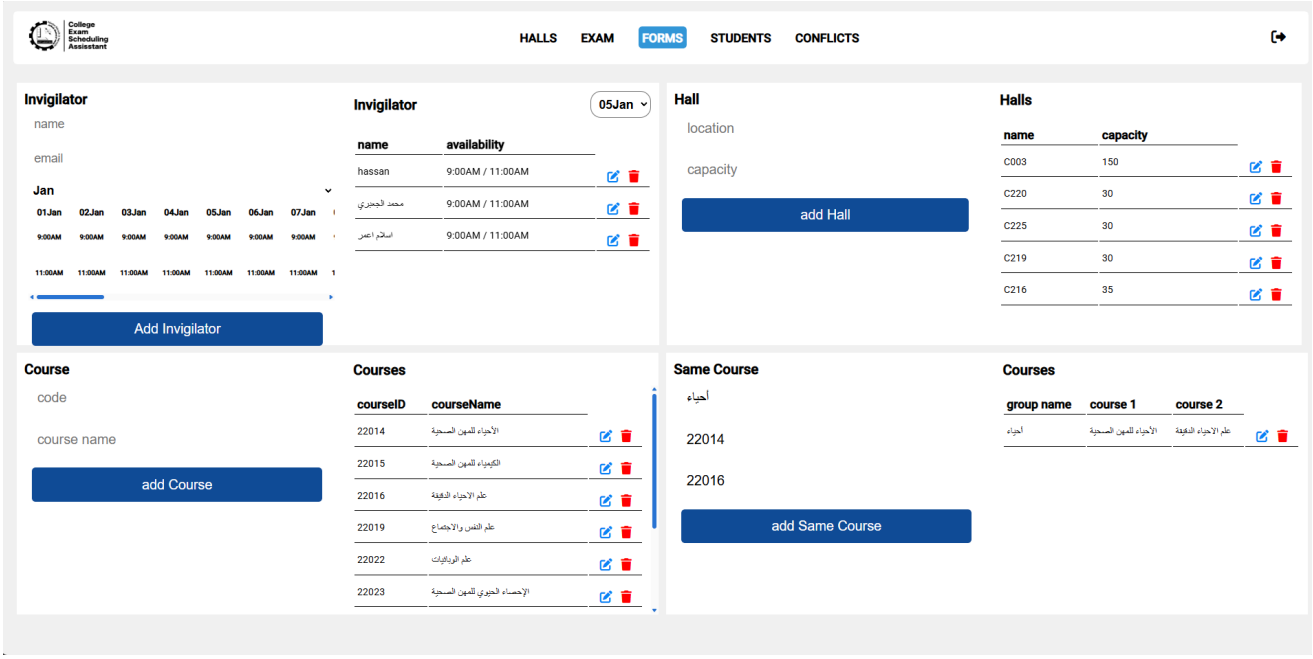
تُساعد هذه الميزة على اكتشاف التعارضات مثل تداخل مواعيد الامتحانات. تُعرض التنبيهات فور اكتشاف التعارضات مع عددها. يُبرز **الشكل 19** آلية عرض التعارضات.

الشكل19: الية عرض التعارضات في الامتحانات

4. واجهة النماذج (FORMS)

واجهة النماذج هي جزء أساسي من النظام، حيث تُستخدم لإضافة وإدارة البيانات التي قد لا تكون موجودة في الملفات المرفوعة مسبقًا. تركز الواجهة على تقديم أدوات مرنة تُسهل إضافة المراقبين، القاعات، والدورات الجديدة، مما يعزز من مرونة النظام وقدرته على التكيف مع التحديثات المستمرة.

يستعرض **الشكل 20** التصميم العام لواجهة النماذج، والذي يُظهر الخيارات المختلفة لإدارة البيانات الإضافية.



الشكل20: واجهة النماذج (FORMS)

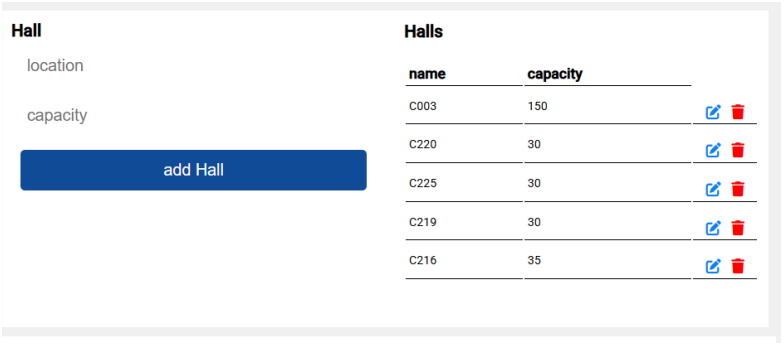
إضافة المراقبين وتحديث بياناتهم

تُتيح هذه الميزة إدخال بيانات المراقبين الجدد، مثل الأسماء وأوقات التوافر. يمكن أيضًا تعديل بيانات المراقبين الحاليين لتتناسب مع التحديثات المطلوبة. **الشكل 21** يُوضح كيفية إضافة أو تعديل المراقبين.

الشكل21: إضافة المراقبين وتحديث بياناتهم

إضافة القاعات الجديدة

تُوفر هذه الميزة إمكانية إدخال قاعات جديدة غير مُدرجة مسبقًا مع تحديد السعة والموقع. كما تُتيح تعديل خصائص القاعات أو حذفها عند الحاجة. يُبرز **الشكل 22** عملية إضافة القاعات الجديدة.



الشكل22: إضافة القاعات الجديدة

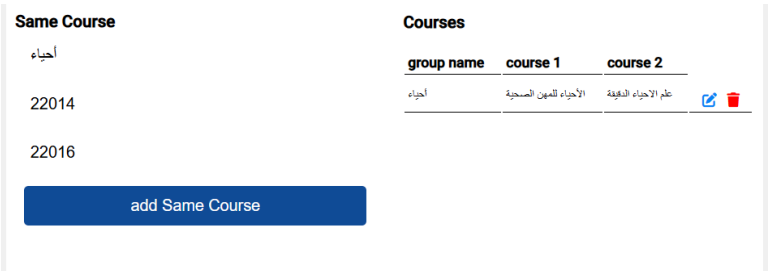
إضافة مساقات الجديدة

تُستخدم هذه الميزة لتعريف مساقات جديدة لم تكن مشمولة في البيانات المرفوعة، مع إدخال تفاصيل مثل اسم المساق والرمز الخاص به. **الشكل 23** يُظهر آلية إضافة المساقات.

الشكل23: إضافة مساقات الجديدة

تعريف المساقات التي لها اكثر من مسمى

تمكن هذه الميزة من تعريف المساقات التي تُعتبر متطابقة، مثل المواد المشتركة بين الأقسام المختلفة، مما يُسهم في تسهيل الجدولة. يُوضح **الشكل 24** كيفية تحديد المساقات التي لها اكثر من مسمى.

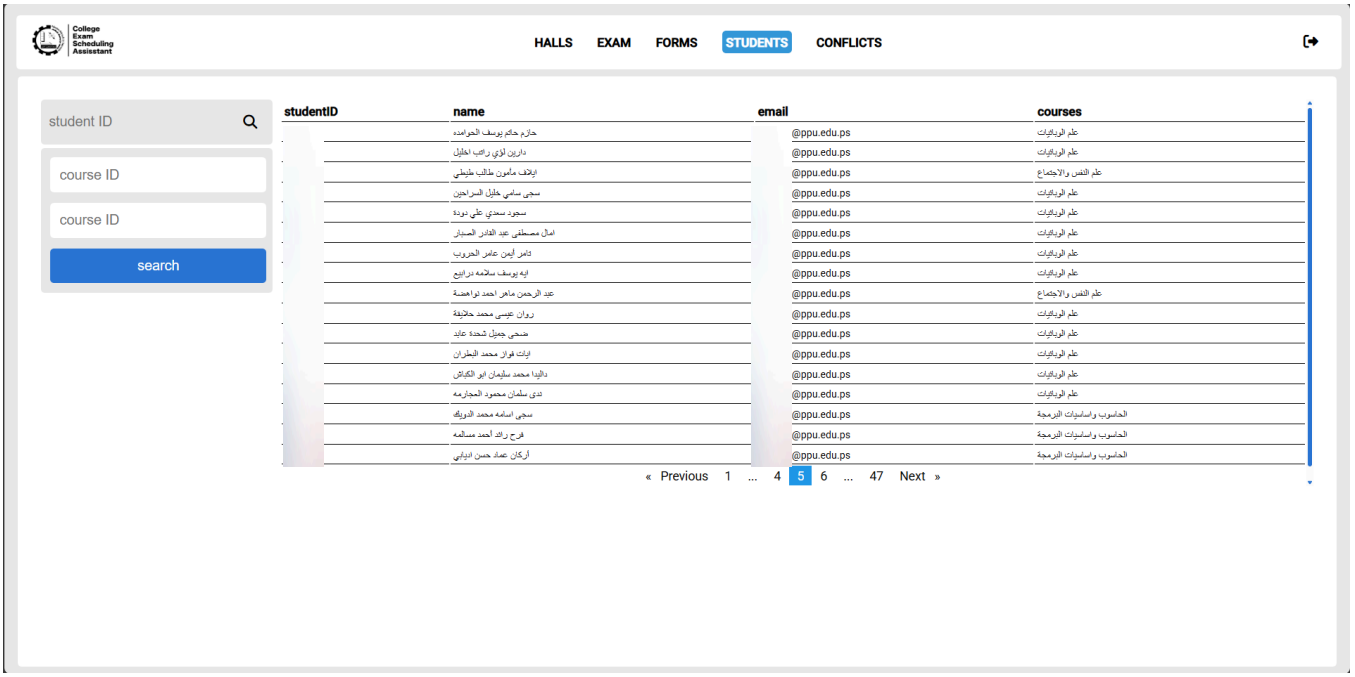


الشكل24: تعريف المساقات التي لها اكثر من مسمى

5. واجهة الطلاب (STUDENTS)

واجهة الطلاب تُعتبر من المكونات الأساسية للنظام، حيث تتيح عرض بيانات الطلاب بطريقة منظمة وفعّالة. تُوفر هذه الواجهة أدوات مرنة لعرض بيانات الطلاب، مع إمكانية البحث حسب معايير محددة، مما يُسهم في تحسين دقة البيانات وسهولة الوصول إليها.

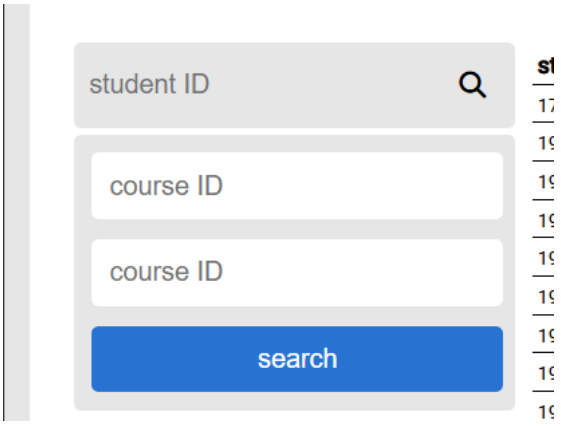
يستعرض **الشكل 25** التصميم العام لواجهة الطلاب، التي تعرض قائمة بالطلاب المسجلين في النظام مع التفاصيل المرتبطة بكل طالب.



الشكل25: واجهة الطلاب (STUDENTS)

البحث عن الطلاب

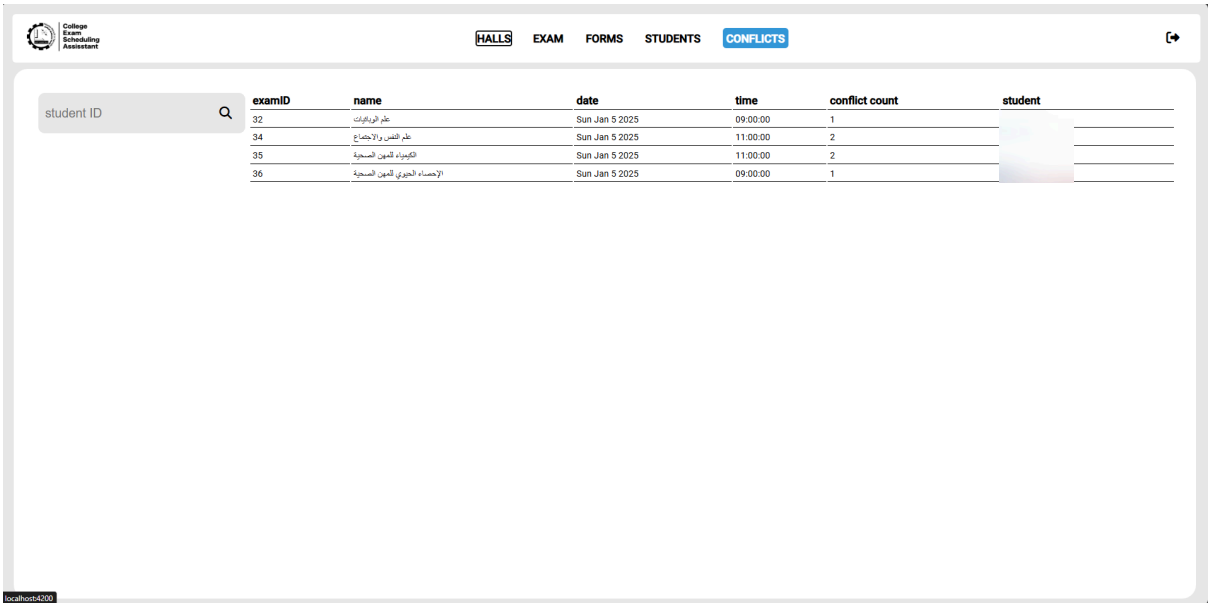
تُوفر الواجهة أداة بحث تفاعلية تمكّن المسؤول من البحث عن الطلاب باستخدام معايير مثل الرقم الجامعي، أو رقم المساق، وتتيح ايضًا عملية المقارنة بين مساقين مما يسهل الوصول إلى المعلومات المطلوبة بسرعة. **الشكل 26** يُوضح عملية البحث عن الطلاب.



الشكل26: البحث عن الطلاب

6. واجهة التعارضات (CONFLICTS)

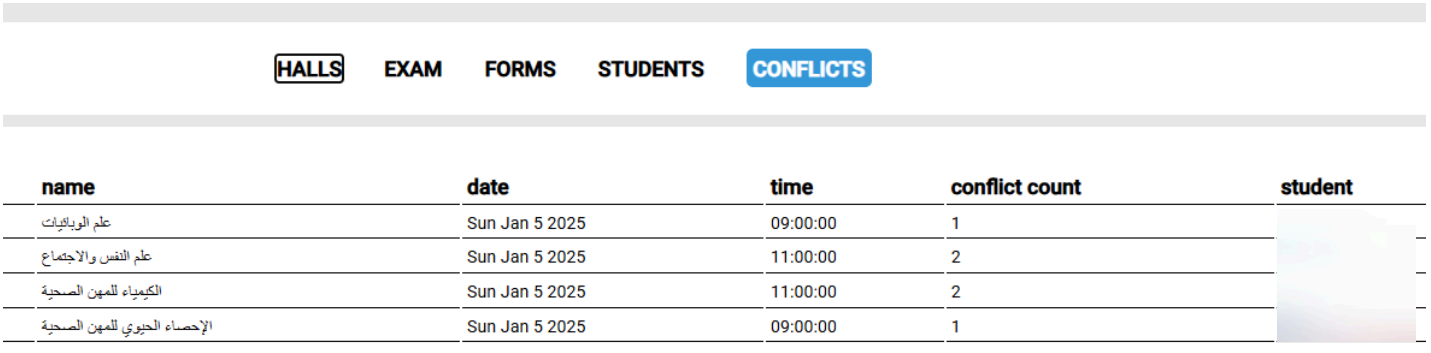
واجهة التعارضات هي أحد المكونات الرئيسية للنظام، حيث تُستخدم لاكتشاف التي قد تنشأ أثناء عملية جدولة الامتحانات. تُساعد هذه الواجهة المسؤولين في التعرف على التعارضات، مثل تداخل المواعيد لعدد من الطلاب في أكثر من امتحان.
يستعرض **الشكل 27** التصميم العام لواجهة التعارضات، التي تعرض قائمة بالتعارضات المكتشفة في الجدولة.



الشكل27: واجهة التعارضات (CONFLICTS)

واجهة عرض جدول التعارضات (CONFLICTS TABLE VIEW)

يُظهر **الشكل 28** تصميم جدول عرض التعارضات في النظام. الجدول يُبرز المشكلات المكتشفة أثناء عملية الجدولة، مع تفاصيل شاملة تسهل على المسؤولين معالجة الأخطاء بسرعة ودقة.

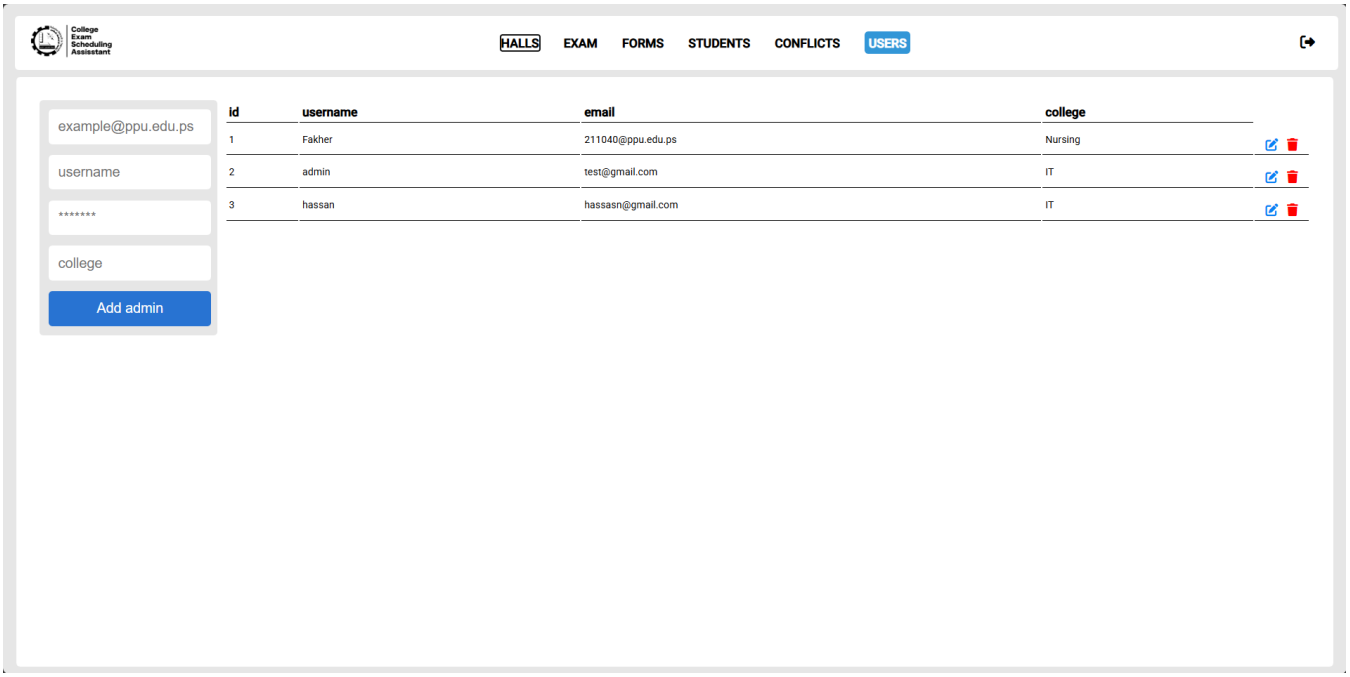


الشكل28: واجهة عرض جدول التعارضات (CONFLICTS TABLE VIEW)

7. واجهة إدارة المستخدمين (USERS)

واجهة إدارة المستخدمين تُعد أداة أساسية تُظهر فقط للمسؤولين الإداريين (Admin) لضمان أمان النظام وتنظيم الوصول إليه. تُتيح هذه الواجهة التحكم الكامل في إضافة المستخدمين الجدد، تعديل بياناتهم، أو حذف الحسابات عند الحاجة. يقتصر الوصول لهذه الواجهة على المسؤولين المعنيين بصلاحيات الإدارة لضمان حماية النظام من التعديلات غير المصرح بها.

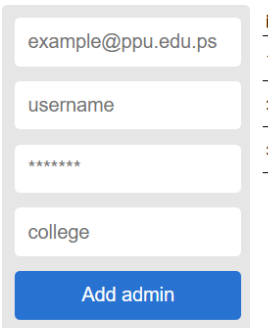
يستعرض **الشكل 29** التصميم العام لواجهة إدارة المستخدمين، التي تعرض قائمة بجميع الحسابات المسجلة مع التفاصيل المرتبطة بكل مستخدم.



الشكل29: واجهة إدارة المستخدمين (USERS)

إضافة مستخدم جديد

تُتيح هذه الميزة للمسؤول الإداري إنشاء حسابات جديدة للمستخدمين من خلال إدخال بيانات مثل اسم المستخدم، البريد الإلكتروني، كلمة المرور، والكلية التي ينتمي إليها المستخدم. بناءً على الكلية المُحددة، يتم تخصيص الصلاحيات المناسبة تلقائيًا لضمان تنظيم الأدوار والمهام داخل النظام بدقة وفعالية. يُوضح **الشكل 30** آلية إضافة مستخدم جديد من خلال واجهة مبسطة وسهلة الاستخدام.



الشكل30: إضافة مستخدم جديد

5. اختبار النظام

اختبارات النظام هي مرحلة حاسمة لضمان أن المشروع يلبي جميع المتطلبات بكفاءة ودقة. تم تنفيذ مجموعة متكاملة من الاختبارات التي ركزت على الجوانب الوظيفية وغير الوظيفية للنظام. تمثل هذه الاختبارات عملية شاملة تهدف إلى كشف أي نقاط ضعف وتحسين أداء النظام ليكون جاهزًا للتطبيق العملي في بيئات العمل الواقعية. في هذا الجزء، سنتوسع أكثر في شرح الاختبارات والنتائج التي تم تحقيقها.

5.1 اختبارات الوحدة unit testing

تركز اختبارات الوحدة على كل وظيفة من وظائف النظام بشكل منفصل لضمان أن المكونات الفردية تعمل بدقة ووفقًا للتوقعات. من بين الاختبارات التي تم تنفيذها:

- **رفع بيانات الطلاب:**

تم تصميم النظام لقبول ملفات **Excel فقط** بمختلف الأحجام والتنسيقات، حيث يتم التحقق من صحة البيانات بشكل دقيق قبل معالجتها. إذا كان ملف Excel يحتوي على أخطاء، مثل أسماء أعمدة غير متوافقة أو بيانات ناقصة، يقوم النظام برفض الملف وعرض رسالة خطأ. هذا يضمن إدخال بيانات صحيحة ومتوافقة مع معايير النظام. تم اختبار هذه العملية باستخدام ملفات تحتوي على بيانات صحيحة وأخرى تتضمن أخطاء، وأظهر النظام كفاءة في التعامل مع الملفات الصحيحة وتنبيه المستخدم بالملفات التي تحتوي على أخطاء أو تنسيقات غير متوافقة.

```
} catch (error) {
  console.error("Upload error:", error);
  res.status(500).json({
    message: "حدث خطأ أثناء معالجة الملف",
    error: error.message,
  });
}
```

الشكل31: رسالة الخطأ

- **كشف التعارضات:**

تم اختبار كشف التعارضات باستخدام سيناريوهات متعددة، تضمنت جداول امتحانات معقدة حيث كان الطلاب مسجلين في أكثر من امتحان بنفس الوقت. أظهرت النتائج أن النظام يتمتع بدقة عالية في رصد هذه الحالات.

examID	name	date	time	conflict count	student
32	علم الوبائيات	Sun Jan 5 2025	09:00:00	1	
34	علم النفس والاجتماع	Sun Jan 5 2025	11:00:00	2	
35	الكيمياء للمهن الصحية	Sun Jan 5 2025	11:00:00	2	
36	الإحصاء الحيوي للمهن الصحية	Sun Jan 5 2025	09:00:00	1	

الشكل32: تقرير مفصل عن التعارضات

5.2 اختبارات التكامل Integration Testing

اختبارات التكامل تهدف إلى التحقق من كيفية عمل المكونات المختلفة معًا لضمان تجربة سلسة للمستخدم.

- **التكامل بين الواجهة الأمامية والخلفية**:
في هذا الاختبار، تم إدخال بيانات عبر الواجهة الأمامية للتحقق من معالجتها في الخادم وتخزينها بشكل صحيح في قاعدة البيانات. أظهرت النتائج تدفقًا دقيقًا وسلسًا للبيانات بين المكونات.

الواجهات الخلفية

```
app.use(
  cors({
    origin: "http://localhost:4200",
    credentials: true,
  })
);
```

الواجهات الامامية

```
{
  "/api": {
    "target": "http://localhost:3000",
    "secure": false,
    "changeOrigin": true,
    "logLevel": "debug"
  }
}
```

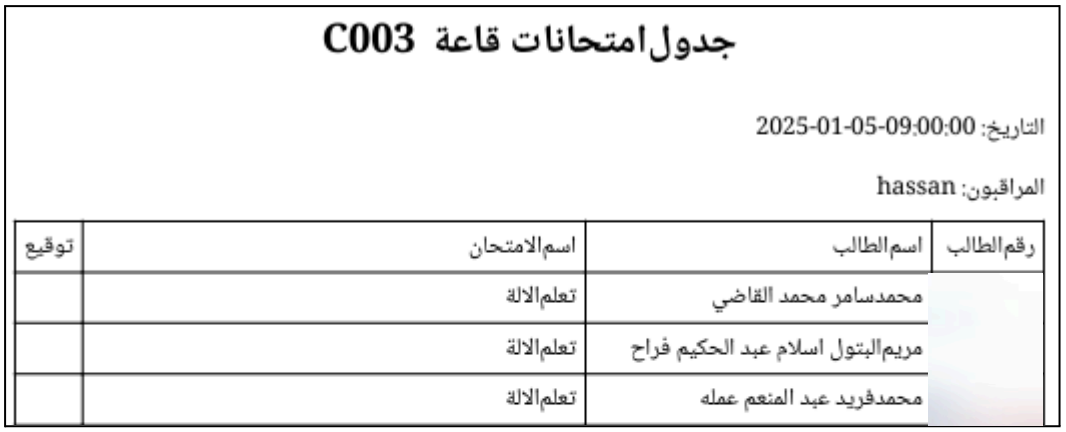
الشكل33: التكامل بين الواجهة الأمامية والخلفية

- **عرض النتائج في الوقت الفعلي**:
تم اختبار عرض نتائج كشف التعارضات في الواجهة الأمامية بشكل فوري. أظهر النظام قدرة على توفير معلومات دقيقة، مع عرض نقاط التحذير (مثل النقاط الحمراء) التي تحتوي على التفاصيل عند تمرير مؤشر الماوس فوقها.

الشكل34: التعارضات اثناء الجدولة

- **توليد التقارير:**

تم فحص قدرة النظام على جمع البيانات من الوحدات المختلفة لإنشاء تقارير نهائية شاملة، كما هو موضح في **الشكل 35 والشكل 36** كجزء من وحدة التقارير. أظهرت التقارير الناتجة توافقًا تامًا مع البيانات المخزنة، حيث تضمنت جميع التفاصيل المطلوبة بشكل دقيق وشامل.



جدول امتحانات قاعة C003

التاريخ: 2025-01-05-09:00:00

المراقبون: hassan

رقم الطالب	اسم الطالب	اسم الامتحان	توقيع
	محمدسامر محمد القاضي	تعلم الالة	
	مريم البتول اسلام عبد الحكيم فراح	تعلم الالة	
	محمدفريد عبد المنعم عمله	تعلم الالة	

الشكل35: تقرير الطلاب داخل القاعات

جدول الطلاب المتعارضين

التاريخ: 2025-01-05 00:00:00

الوقت: 09:00:00

الطالب رقم	اسم الطالب	الامتحانات المتعارضة
	مهندمحمود موسى طقاطقة	وأمن المعلومات (C003) -برمجة الكيانات (C003) أخلاقيات الحاسوب
	ايه زياد عبد العزيز العواوده	وأمن المعلومات (C003) -برمجة الكيانات (C003) أخلاقيات الحاسوب

الشكل36: تقرير التعارضات لدى الطلاب

5.3 اختبارات الأداء Performance testing

اختبارات الأداء تمثل جزءًا مهمًا في قياس قدرة النظام على التعامل مع بيانات ضخمة وضغط العمل.

- **اختبار الاداء في رفع البيانات**

أثناء اختبار رفع ملفات Excel تحتوي على 30,000 سطر من البيانات، استغرق النظام حوالي ثانيتين لمعالجة الملف بالكامل. تضمنت هذه العملية التحقق من البيانات، تصنيف الطلاب حسب الكلية، وتخزين البيانات بشكل سليم.

الشكل37: الوقت المستغرق لرفع ملف ال Excel

- **اختبار الاداء اثناء الجدولة**

تمت محاكاة النظام لجدولة امتحانات أكثر من 900 طالب موزعين على كليتين، واستغرقت العملية قرابة 4 ثواني فقط، حيث أظهر النظام كفاءة عالية واستقرارًا في معالجة البيانات دون أي تعطل، مما يثبت قدرته على التعامل مع أحمال كبيرة بفعالية.

```
executionTime: 0:04.108 (m:ss.mmm)
```

الشكل38: الوقت المستغرق لجدولة الامتحانات

الشكل39: الامتحانات المجدولة بشكل تلقائي

- **اختبار استمرارية الأداء**

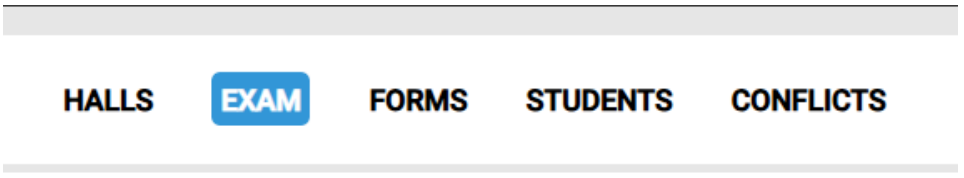
تم تشغيل النظام لمدة 24 ساعة متواصلة للتأكد من استقراره على المدى الطويل. أظهرت النتائج أن النظام قادر على التعامل مع الاستخدام المكثف دون أي تراجع في الأداء أو ظهور أعطال.

5.4 اختبارات التفاعل مع المستخدمين User Interaction Testing

اختبارات التفاعل تهدف إلى قياس سهولة استخدام النظام وفعاليته بالنسبة للمستخدمين.

- **اختبار واجهة المستخدم:**

تم اختبار الواجهة لضمان سهولة التنقل بين الوظائف المختلفة. أظهرت النتائج أن واجهة المستخدم سهلة الاستخدام حتى للمستخدمين الذين ليس لديهم خبرة تقنية.



الشكل40: التنقل بين الصفحات

- **التعامل مع الأخطاء:**

عند إدخال بيانات غير صحيحة، أظهر النظام رسائل خطأ واضحة وسهلة الفهم، مما ساعد المستخدمين على تصحيح الأخطاء بسرعة.

يظهر **الشكل 41** رسالة خطأ في حال أقدم المستخدم على رفع الملف قبل اختياره

الشكل41: رسالة خطأ

5.5 توسعة نطاق الاختبارات

بالإضافة إلى الاختبارات الأساسية، تم توسيع نطاق الاختبارات لتغطية سيناريوهات إضافية قد تظهر أثناء الاستخدام الفعلي للنظام. الهدف من هذه التوسعة هو ضمان جاهزية النظام لمواجهة أي حالة غير متوقعة بكفاءة.

- **اختبارات الإدخال غير المتوقع:**

تم إدخال بيانات تحتوي على رموز خاصة وأحرف غير معتادة للتحقق من استقرار النظام عند التعامل مع مثل هذه الحالات. أثبت النظام كفاءته في معالجة البيانات الغريبة دون فشل، مع توفير رسائل خطأ دقيقة عند الحاجة.

- **اختبارات الحمل طويل المدى:**

تم تشغيل النظام لفترات ممتدة تحت ضغط بيانات متزايد للتحقق من استقراره مع مرور الوقت. أظهرت النتائج أن النظام قادر على الحفاظ على أدائه دون أي انخفاض في الكفاءة، مما يعكس استقرار التصميم وقوة البنية البرمجية.

5.6 التحديات أثناء الاختبارات

أثناء مرحلة الاختبار، ظهرت عدة تحديات تقنية تتعلق بأداء النظام وتعامل الخوارزميات مع السيناريوهات المختلفة. تم التعامل مع هذه التحديات بنجاح لضمان تحسين أداء النظام وكفاءته.

- **التحدي في معالجة البيانات الكبيرة:**

خلال رفع ملف Excel يحتوي على حوالي 30,000 سطر، لوحظت تأخيرات في زمن المعالجة نتيجة لحجم البيانات الضخم. تم تحسين الخوارزميات المستخدمة لمعالجة البيانات، مما أدى إلى تقليل زمن المعالجة إلى **3 دقائق فقط**، مع ضمان دقة النتائج واستقرار الأداء.

- **التعامل مع تعارضات غير متوقعة:**

أثناء الاختبارات المحاكية، ظهرت حالات تعارض جديدة لم تكن مبرمجة مسبقًا، مثل تداخل غير متوقع في المواعيد بين الطلاب والقاعات. تم تعديل النظام لتعزيز مرونة الخوارزميات، مما ساعد على التعامل مع هذه الحالات بشكل ديناميكي وتلقائي.

5.7 الخلاصة

بعد إجراء اختبارات شاملة وموسعة، أثبت النظام كفاءته وقدرته على تلبية المتطلبات الأكاديمية بدقة وسرعة. أظهر النظام مرونة كبيرة في معالجة البيانات، كشف التعارضات، وتوليد التقارير. كما نجح في تحسين عملية الجدولة التقليدية، مما جعله أداة عملية يمكن الاعتماد عليها في بيئات العمل الأكاديمية.

هذه النتائج تؤكد أن النظام ليس فقط جاهزًا للتطبيق العملي، بل أيضًا يمثل حلًا موثوقًا ومستدامًا لتحسين إدارة جداول الامتحانات وتقليل الأخطاء.

6. النتائج والتوصيات

6.1 التحديات

1. التحديات التقنية

1. غياب API جاهز للتكامل مع نظام الجامعة:

أحد أبرز التحديات التقنية كان غياب واجهة برمجية جاهزة (API) للتكامل مع نظام الجامعة. نتيجة لذلك، اضطر الفريق إلى إيجاد حلول بديلة مثل استخدام ملفات Excel تحتوي على بيانات الطلاب والمقررات. هذه العملية تطلبت تحليل البيانات يدويًا لاستخراج المعلومات المطلوبة، مما أدى إلى استهلاك وقت إضافي أثناء إدخال البيانات وزيادة الجهد المبذول.

2. التعامل مع البيانات الضخمة:

تزايد حجم البيانات المستخدمة في النظام، مثل عدد الطلاب، المقررات، وقاعات الامتحانات، شكل تحديًا كبيرًا في ضمان الأداء السريع للنظام. تم اختبار الخوارزميات للتأكد من قدرتها على معالجة البيانات الضخمة بكفاءة ودون أي بطء أو أخطاء. النظام أظهر قدرة على تحسين الأداء بعد تعديل الخوارزميات لتصبح أكثر فعالية.

3. تعيين القاعات والمراقبين:

التعامل مع تعيين القاعات والمراقبين كان تحديًا تقنيًا ولوجستيًا. تطلب الأمر تصميم خوارزمية مرنة قادرة على حل التعارضات المتعلقة بالسعة الزمنية أو القيود اللوجستية. تم تطبيق خاصية السحب والإفلات لتسهيل عملية التعيين وضمان مرونة وسرعة التعامل مع البيانات في السيناريوهات المعقدة.

2. التحديات الإدارية

على الصعيد الإداري، واجه الفريق تحديات كبيرة نتيجة لعدم وجود تكامل بين أنظمة الجامعة المختلفة. لم يكن هناك واجهة برمجية (API) موحدة تتيح ربط الأنظمة، مما أثر على كفاءة تبادل البيانات.

هذا القيد أدى إلى تعقيد عملية جمع البيانات المطلوبة للنظام. لم يكن بالإمكان الحصول على البيانات بشكل مباشر من الأنظمة، مما اضطر الفريق إلى استخدام ملفات Excel تحتوي على البيانات الخام. بعد ذلك، كان لا بد من رفع هذه الملفات وتحليلها يدويًا لتنسيقها مع النظام.

هذه العملية تطلبت وقتًا إضافيًا وجهدًا كبيرًا، وزادت من تعقيد العمليات الإدارية. على الرغم من ذلك، ساعد استخدام ملفات Excel كحل بديل على سد الفجوة التقنية مؤقتًا، مما مكن الفريق من إتمام المشروع ضمن الإطار الزمني المحدد.

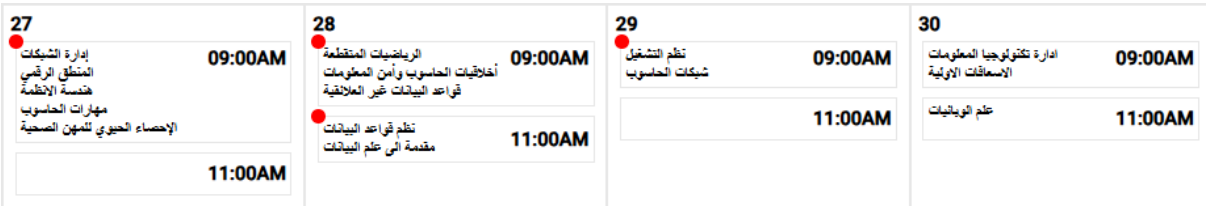
6.2 النتائج

1. كفاءة إعداد الجداول

ساهم النظام في تقليل الوقت اللازم لإنشاء الجداول بنسبة تزيد عن 50% مقارنة بالطريقة التقليدية، بفضل أتمتة العمليات الأساسية وتحسينها.

2. تحسين إدارة التعارضات

بفضل آلية الكشف التلقائي عن التعارضات وإعداد تقارير مفصلة لحلها، تم تقليل الأخطاء بشكل كبير، مما أدى إلى تحسين دقة الجداول.



الشكل42: جدول الامتحانات مع التعارضات

الشكل43: جدول الامتحانات بعد معالجة التعارضات

3. تجربة المستخدم

ساعدت الواجهات التفاعلية والبسيطة في تحسين تجربة المستخدمين، خاصة فيما يتعلق بتخصيص القاعات والمراقبين بسرعة وسهولة.

الشكل44: ميزة السحب والافلات

6.3 التوصيات

1. توسيع النظام ليشمل كافة الكليات

نظرًا للمرونة التي يتمتع بها النظام، يُوصى بتطبيقه على مستوى كافة كليات الجامعة، مما يُسهم في تحسين عمليات الجدولة بشكل شامل على مستوى الجامعة.

2. تطوير تطبيق موبايل

يُوصى بتطوير تطبيق مخصص للهواتف الذكية لعرض الجداول للمستخدمين بشكل مباشر، مع إرسال الإشعارات الفورية لتحديثاتهم اليومية، مما يُسهل الوصول إلى المعلومات.

3. تحسين التكامل مع أنظمة الجامعة

يُوصى بدمج النظام مع أنظمة الجامعة الداخلية، مثل نظام تسجيل الطلاب، لتقليل الاعتماد على العمليات اليدوية وضمان دقة البيانات وتحديثها بشكل مستمر.

4. إضافة تقارير تحليلية

- تقديم تقارير تنبؤية مثل نسب الضغط على القاعات وأوقات الذروة.
- تحليل البيانات لتوفير إحصائيات دقيقة تدعم الإدارة في تحسين توزيع الموارد.

5. تطوير واجهة متعددة اللغات

يُوصى بإتاحة واجهة النظام بلغات متعددة، مما يضمن سهولة الاستخدام لجميع المستخدمين، خاصة في البيئات التعليمية ذات الثقافات المتنوعة.

هذه النتائج والتوصيات تمثل خارطة طريق لتطوير النظام مستقبلًا بما يضمن تحقيق أهدافه الأكاديمية والإدارية بشكل أكثر شمولية وكفاءة.

6.4 الخاتمة

يُمثل نظام إدارة جداول الامتحانات خطوة جوهرية نحو تحقيق كفاءة أعلى وفعالية أكبر في تنظيم وإدارة الجداول الزمنية داخل المؤسسات التعليمية. من خلال أتمتة العمليات المعقدة وإدارة البيانات بدقة، يسهم النظام في تقليل التحديات المرتبطة بالجدولة اليدوية وتحسين العملية التعليمية بشكل شامل.

أبرز مميزات النظام تكمن في قدرته على تقليل الأخطاء الناتجة عن العمليات اليدوية، مما يضمن توزيعًا دقيقًا للامتحانات والقاعات والمراقبين. كما يقلل بشكل كبير من الوقت والجهد المطلوبين لإعداد الجداول من خلال أتمتة المهام الروتينية، مما يُتيح لفريق العمل التركيز على الأنشطة الأكثر أهمية.

يُوفر النظام مرونة عالية في التعامل مع التغييرات الطارئة، مثل تعديل مواعيد الامتحانات أو إعادة تخصيص القاعات، حيث تتيح الواجهات التفاعلية والبسيطة للمسؤولين تنفيذ هذه التعديلات بسهولة وكفاءة.

إضافةً إلى ذلك، يعمل النظام كمنصة مركزية تُعزز التعاون بين مختلف الكليات داخل المؤسسة التعليمية، مما يُسهم في تحسين التنظيم والتخطيط على مستوى الجامعة بشكل عام. كما يتيح إنشاء تقارير مفصلة وإحصاءات دقيقة تساعد الإدارة في اتخاذ قرارات مستنيرة وتعزز من جودة التخطيط المستقبلي.

في النهاية، يُعد نظام إدارة جداول الامتحانات أداة حيوية تعزز الكفاءة والتنظيم، وتُسهم في تقديم تجربة أكثر سهولة للمسؤولين والطلاب على حد سواء. فهو لا يقتصر على حل المشكلات اليومية فحسب، بل يُمثل استثمارًا مستدامًا لتحسين العمليات التعليمية ودعم أهداف المؤسسات الأكاديمية

7. المراجع

نظام Caphub - جامعة بوليتكنك فلسطين

نظام Caphub يمثل أحد الحلول البرمجية الرائدة التي طورتها جامعة بوليتكنك فلسطين لتلبية احتياجات الجدولة الأكاديمية. تم تصميم هذا النظام ليكون مرنًا وشاملًا، حيث يدمج بين التقنيات الحديثة وأفضل الممارسات الإدارية لضمان توفير تجربة فعالة للمستخدمين. يساعد النظام في تسهيل إدارة جداول الامتحانات، توزيع القاعات، وجدولة المراقبين، مما يعزز من كفاءة العملية التعليمية.

Microsoft Office

تم استخدام مجموعة أدوات Microsoft Office، مثل Excel، لتحضير وإدخال بيانات الطلاب والموارد الأخرى للنظام. تساهم التكاملات مع Office في تسهيل التعامل مع البيانات الكبيرة وضمان توافقها مع النظام. كما يُعد Excel أداة قوية لتحليل البيانات الأولية قبل رفعها إلى قاعدة البيانات.

UNITIME

نظام UNITIME هو أحد النظم المعروفة عالميًا في مجال إدارة الجداول الأكاديمية، ويُستخدم كمصدر إلهام في تصميم نظام Caphub. من خلال دراسة ميزات UNITIME، تم دمج العديد من التقنيات والأساليب المتقدمة في نظام Caphub، مثل تحسين توزيع الموارد والكشف التلقائي عن التعارضات. ومع ذلك، يتميز Caphub بمرونة أكبر تتيح الجمع بين الأتمتة والتدخل البشري لتلبية احتياجات المؤسسات الأكاديمية بشكل أكثر تخصيصًا.