# الفصل الأول

التعريف بالمشروع

يتضمن هذا الفصل التعريف بالمشروع ومتطلباته.

## مقدمة

تعتبر صيدلية مركز البحوث العلمية من المراكز الخدمية الرئيسية في المؤسسة، تقتصر مهمتها الأساسية على صرف الأدوية للموظفين وأفراد عائلاتهم المباشرين، معتمدة بذلك على نظام تغطية مالية داخلية. في ظل التطور السريع في مجال هندسة البرمجيات وظهور أطر عمل حديثة توفر أداءً عالياً وتجربة مستخدم أفضل، أصبح النظام الحالي المستخدم في الصيدلية قديماً ومحدود القدرات، مع توقف الدعم الفني له منذ فترة طويلة. من هذا المنطلق، جاء المشروع كحل بديل لإعادة بناء وتطوير نظام الصيدلية بشكل كامل. يهدف المشروع إلى تقديم نظام ويب متكامل وعصري، وذلك بهدف أتمتة العمليات اليومية، وتحسين دقة إدارة المخزون، وتقديم واجهات سهلة الاستخدام تعزز من كفاءة العمل.

## الهدف من المشروع

إن الهدف الأساسي من المشروع هو إنشاء نظام برمجي متكامل وموثوق لأتمتة العمليات اليومية في صيدلية مركز البحوث، مما يؤدي إلى تحسين الكفاءة والدقة وتقليل الأخطاء اليدوية. يسعى النظام لتحقيق ذلك من خلال مركزية إدارة الأدوية عبر توفير قاعدة بيانات شاملة يمكن إدارتها بسهولة. كما يركز على أتمتة إدارة المخزون، حيث يتم تتبع الكميات بشكل دقيق ولحظي وإدارة دفعات الأدوية بناءً على تاريخ انتهاء الصلاحية لضمان تطبيق مبدأ (FIFO) بالإضافة إلى ذلك، يهدف النظام إلى تحسين عملية صرف الوصفات الطبية عن طريق تبسيطها وتسريعها، مع ضمان التطبيق الدقيق لقواعد التغطية المالية الداخلية. ولدعم اتخاذ القرار، سيوفر النظام القدرة على إنشاء تقارير تحليلية إدارية ومالية، وأخيراً، سيحقق التكامل مع أنظمة المنظمة الأخرى من خلال توفير واجهات برمجية (APIs) آمنة.

## المتطلبات الوظيفية

* + 1. يجب أن يسمح النظام للمسؤولين بإنشاء حسابات مستخدمين.
    2. يجب أن يسمح النظام للمسؤولين بتعديل معلومات حسابات المستخدمين.
    3. يجب أن يفرض النظام المصادقة عبر اسم المستخدم وكلمة المرور
    4. يجب أن يطبق النظام آليات للتحكم بصلاحيات الوصول بناءً على أدوار المستخدمين.
    5. يجب أن يسمح النظام للصيادلة بإضافة سجلات أدوية جديدة تشمل الاسم، الشركة المصنعة، السعر، التصنيف، العيار، الوصف، التركيب، الشكل الدوائي، الحجم (عدد الكبسولات، حجم عبوة الشراب..)، تاريخ التصنيع وتاريخ انتهاء الصلاحية
    6. يجب أن يسمح النظام بالتعديل على خصائص الأدوية الموجودة (مثل الاسم، الشركة المصنعة، السعر، التصنيف، الجرعة، الوصف).
    7. يجب أن يسمح النظام بإضافة معلومة عن المستوى الأدنى المسموح لكل دواء.
    8. يجب أن يسمح النظام بحذف سجلات الأدوية إن لم يتم تنفيذ عمليات عليها.
    9. يجب أن يسمح النظام بتسجيل شحنات الأدوية الواردة بما في ذلك معلومات عن الأدوية في الشحنة وكمياتها والمورد.
    10. يجب أن يقوم النظام بتحديث كميات المخزون تلقائيًا بناءً على الشحنات الجديدة.
    11. يجب أن يقوم النظام بتحديث المخزون عندما يتم صرف الأدوية.
    12. يجب أن يمنع النظام صرف الأدوية إذا كانت غير متوفرة في المخزون.
    13. يجب أن يجب أن يقوم النظام بتوليد تقارير مالية تلخص المبيعات ضمن مدة زمنية محدّدة.
    14. يقوم النظام بتوليد تقارير إدارية للمخزون في الصيدلية.

## المتطلبات غير الوظيفية

1. يجب أن يكون النظام آمناً، حيث يسمح فقط للمستخدمين المسجّلين باستخدامه.
2. يجب أن يكون التطبيق متاحاً بشكل دائم
3. يجب أن يكون زمن الاستجابة سريعاً، لا يتجاوز ال 10 ثوان.
4. يجب على النظام أن يكون مصمماً بطريقة قابلة للتحديث والتطوير المستمر، بالإضافة إلى سهولة الصيانة في حال ظهور خطأ أو خلل في الأداء
5. يجب أن يوفر النظام واجهات مرنة، سهلة الفهم الاستخدام وجيدة المظهر.

# الفصل الثاني

الدراسة النظرية

يوضّح هذا الفصل مجموعة من المفاهيم النظرية المستخدمة ضمن العمل المقدم.

# الفصل الثالث

الدراسة المرجعية

يعرض هذا الفصل الأبحاث والبيانات المرتبطة بالعمل المقدم.

# الفصل الرابع

الدراسة التحليلية

يعرض هذا الفصل عمليّة تحليل النظام ودراسة متطلباته.

# الفصل السادس

تصميم النظام

يعرض هذا الفصل القرارات التصميمية التي بني من خلالها النظام.

## فصل الواجهة الأمامية عن الخلفية (Decoupled Frontend/Backend)

تم اعتماد بنية برمجية حديثة ومنظمة لبناء النظام، حيث يتكون من تطبيقين منفصلين يعملان معاً بشكل متكامل: تطبيق خلفي (Backend) مسؤول عن منطق العمل وإدارة البيانات، وتطبيق أمامي (Frontend) مسؤول عن عرض واجهات المستخدم. تقدم هذه البنية مجموعة من الفوائد الأساسية التي تضمن جودة واستدامة المشروع

* 1. **فصل الاهتمامات (Separation of Concerns):** إن الفصل الواضح بين الواجهة الخلفية والواجهة الأمامية، بالإضافة إلى الفصل الداخلي لمكونات الواجهة الخلفية، يجعل كل جزء من النظام مسؤولاً عن مهمة محددة. هذا يقلل من التعقيد ويجعل تطوير وفهم كل جزء أكثر بساطة.
  2. **قابلية الصيانة والتوسع (Maintainability & Scalability):** تسمح البنية المتبعة بإصدار نسخة جديدة من الواجهة الأمامية أو الخلفية بشكل مستقل، سواء لإصلاح خطأ أو إضافة ميزة، دون الحاجة لإعادة بناء النظام بأكمله. كما يمكن توسيع كل جزء على حدة لمواجهة أي زيادة في الطلب مستقبلاً
  3. **التنوع التقني:** يمكّننا هذا الفصل من اختيار التقنية الأنسب لكل جزء. تم استخدام ASP.NET Core للواجهة الخلفية لفعاليتها في بناء APIs آمنة وعالية الأداء، بينما تم استخدام React مع TypeScript للواجهة الأمامية لمرونتها في بناء واجهات مستخدم تفاعلية وحديثة.
  4. **سهولة الاختبار:** يمكن اختبار منطق العمل وقواعد البيانات في الواجهة الخلفية بشكل مستقل تماماً عن واجهات المستخدم، مما يسرّع من دورة التطوير ويضمن جودة الكود.

## استراتيجية الرماز أولاً (Code-First)

كأحد القرارات التصميمية في الواجهة الخلفية، تم اعتماد منهجية الرماز أولاً(Code-First) لتطوير قاعدة بيانات النظام، وذلك بالاعتماد على الكامل على إطار العمل (Entity Framework Core) . تضع هذه المنهجية الرماز البرمجي في مقدمة عملية التصميم ، حيث يتم تعريف نماذج و هياكل البيانات على شكل صفوف، هذه الصفوف تمثل كيانات العمل مثل (Medication ) و (Patient) .

بعد تعريف هذه النماذج، يتم استخدام أداة (Migrations) المدمجة في (Entity Framework) التي تقوم بتحليل الصفوف وتوليد الشيفرة اللازمة (SQL Scripts) لأنشاء مخطط قاعدة البيانات (Schema) بالكامل أو تعديله ليتطابق مع أي تعديلات تطرأ على الكود. تم اختيار هذه الطريقة لما توفره من مزايا جوهرية؛ فهي تسرّع من وتيرة التطوير بشكل ملحوظ، وتضمن التوافق الدائم بين الكود وقاعدة البيانات، مما يمنع حدوث أخطاء ناتجة عن عدم تطابق الهياكل. كما أنها تسهّل إدارة التغييرات على بنية قاعدة البيانات مع تطور متطلبات المشروع، حيث يتم حفظ كل تغيير في ملف (Migrations)يمكن تتبعه والعودة إليه ضمن نظام التحكم بالإصدارات (Version Control)

## بنية قاعدة البيانات (Database Design)

تم إنشاء مخطط قاعدة البيانات العلائقية في SQL Server بناءً على الكيانات البرمجية المعرفة ضمن طبقة المجال (Domain). فيما يلي عرض لأهم الجداول والغرض منها:

1. **AspNetUsers(المستخدمون):** يعبر هذا الجدول عن مستخدمي النظام من مديرين وصيادلة. نهتم فيه بتخزين معلومات تسجيل الدخول بشكل آمن، بالإضافة إلى الأسماء الشخصية لتمييزهم.
2. **AspNetRoles(الأدوار):** يعبر عن الأدوار المتاحة في النظام. نهتم بتخزين اسم الدور (مثل "Admin"أو "Pharmacist") لربطه بالصلاحيات المحددة.
3. **InsuredPerson (الأشخاص المؤمن عليهم):** يعبر عن السجل الأساسي لكل شخص يستفيد من خدمات الصيدلية. نهتم بتخزين معلوماته الشخصية، وحالته في النظام، وتحديد نوعه كموظف أو فرد عائلة لتطبيق قواعد التغطية المالية.
4. **Employee(الموظفون):** جدول يمثل دور الموظف، وهو يرث من جدول InsuredPerson ويضيف معلومات وظيفية خاصة به، كما يحتوي على مفتاح أجنبي لجدول الأقسام بهدف معرفة القسم الذي يعمل به الموظف.
5. **FamilyMember (أفراد العائلة):** جدول يرث من InsuredPerson ويخزن معلومات حول أقارب الموظف، حيث يحتوي على مفتاح أجنبي لجدول الموظفين لتحديد الموظف المسؤول عنه، بالإضافة إلى تحديد نوع صلة القرابة.
6. **Medication(الأدوية):** يعبر عن كتالوج الأدوية. نهتم بتخزين المعلومات التعريفية للدواء كالاسم والجرعة والباركود، بالإضافة إلى ربطه بالشركة المصنعة وتصنيفه وشكله الصيدلاني.
7. **InventoryItem (عناصر المخزون):** يعبر عن وجود دواء معين في المخزون. نهتم بربطه بالدواء وتخزين سعر بيعه.
8. **InventoryItemDetail (تفاصيل المخزون):** يمثل دفعة (batch) محددة من دواء في المخزون. نهتم بتخزين الكمية المتوفرة وتاريخ انتهاء الصلاحية لهذه الدفعة.
9. **Diagnosis(التشخيصات):** يعبر عن التشخيصات الطبية التي يمكن ربطها بالوصفات. نهتم بتخزين وصف التشخيص.
10. **Prescription (الوصفات الطبية):** يعبر عن عملية صرف متكاملة. نهتم بتخزين معلومات المريض والطبيب والصيدلي الذي قام بالصرف، بالإضافة إلى تاريخ وقيمة الصرف الإجمالية.
11. **Sale(المبيعات):** يمثل الجانب المالي لكل وصفة. نهتم بتخزين القيمة الإجمالية، وقيمة التغطية التي قدمتها المنظمة، والمبلغ المستلم من المريض.
12. **PrescriptionItem (عناصر الوصفة الطبية):** هو جدول كسر علاقة many-to-many بين الوصفات والأدوية. نهتم فيه بتخزين الكمية المصروفة وسعر الوحدة لكل دواء في الوصفة.
13. **Department(الأقسام):** يعبر هذا الجدول عن الأقسام المختلفة داخل المنظمة. نهتم بتخزين اسم القسم لربط الموظفين به.
14. **Doctor(الأطباء):** يعبر عن الأطباء الذين يكتبون الوصفات. نهتم بتخزين أسمائهم وتخصصاتهم.
15. **Manufacturer (الشركات المصنعة):** يعبر عن الشركات المنتجة للأدوية. نهتم بتخزين اسم الشركة.
16. **) MedicationClassتصنيفات الأدوية:(**يعبر عن التصنيفات العلاجية. نهتم بتخزين اسم التصنيف ووصفه.
17. **) MedicationFormأشكال الأدوية:(**يعبر عن الشكل الصيدلاني للدواء. نهتم بتخزين اسم الشكل ووحدة القياس.
18. **ActiveIngredient (المكونات الفعالة):** يعبر عن المواد الكيميائية الفعالة في الأدوية. نهتم بتخزين اسم المكون.
19. **MedicationActiveIngredient:** هو جدول كسر علاقة many-to-many بين الأدوية والمكونات الفعالة. نهتم فيه بتحديد كمية كل مكون في الدواء.
20. **Supplier (الموردون):** يعبر عن الشركات التي تورد الأدوية للصيدلية. نهتم بتخزين اسم المورد.
21. **Order (طلبات الشراء):** يعبر عن طلبات الشراء من الموردين. نهتم بتخزين معلومات المورد وتاريخ الطلب.
22. **OrderItem (عناصر طلب الشراء):** هو جدول كسر علاقة many-to-many بين طلبات الشراء والأدوية. نهتم فيه بتخزين الكمية المطلوبة وسعر الوحدة وتاريخ انتهاء الصلاحية لكل دواء في الطلب.
23. **InventoryCheck (جرد المخزون):** يعبر عن عمليات الجرد الدورية. نهتم بتخزين تاريخ الجرد والملاحظات المتعلقة به.
24. **InventoryCheckItem (عناصر جرد المخزون):** هو جدول كسر علاقة بين عمليات الجرد والأدوية. نهتم فيه بتخزين الكمية المتوقعة في النظام والكمية التي تم عدها فعلياً لكل دواء.

.

