

الجمهورية العربية السورية

المعهد العالي للعلوم التطبيقية والتكنولوجيا

اختصاص هندسة البرمجيات والذكاء الصناعي

العام الدراسي 2022-2023

مشروع السنة الرابعة

اختبارات الصيدلة

تقديم

جلّنار بشار علي

إشراف

م. محمود الياس

31/8/2023

الشكر

أتقدم بجزيل الشكر لكل من ساعدني في هذا المشروع، وأخص بالذكر المهندس محمود الياس الذي أشرف على جميع مراحل المشروع وأغنى بملاحظاته طريقة تنظيم العمل.

الإهداء

إلى قدوتي في الحياة ومن له الفضل الأكبر بما أنا عليه اليوم..... والدي بشار

إلى جميلتي وملهمتي والداعم الأكبر لي في حياتي..... والدتي أنديرا

إلى رفيق دربي منذ الصغر، وسندي في هذه الحياة..... أخي محمد نور

الملخص

شهدت السنوات الأخيرة انتشاراً واسعاً للهواتف الذكية، فلم تُعد ميزات الهاتف المحمول إجراء مكالمات وإرسال واستقبال رسائل نصية، بل أصبح يحوي ميزات عديدة تسهّل من صعوبات الحياة على الرغم من حجمه الصغير.

يهدف هذا المشروع إلى الاستفادة من الميزات التي تقدّمها الهواتف الذكية لتخفيف معاناة طلاب الصيدلة في فترة الملائمة، فنقوم في هذا المشروع بتصميم تطبيق موبايل يعمل على أنظمة تشغيل Android أو iOS، يتيح تخزين بعض المعلومات عن الأدوية التي من المفترض على طالب الصيدلة تعلّمها في فترة الملائمة، وليتمكّن من استعراض هذه المعلومات والتدرّب عليها. يتيح التطبيق أيضاً للمستخدم إمكانية إجراء اختبارات للتأكد من إتمامه عملية التدريب.

Abstract

Recent years, there has been a widespread of smartphones. The features of mobile phones have extended beyond making calls and sending/receiving text messages. But rather, they now contain many features that facilitate the difficulties of life despite its small size.

This project aims to take advantage of the features provided by smartphones to relief the suffering of pharmacy students in the training period, so in this project we design a mobile application that works on Android or iOS operating systems that allows storing some information about the drugs that the pharmacy student is supposed to learn during the training period, so that he can review this information to practice it. The application also allows the user to take tests to ensure that he has completed the training process.

جدول الاختصارات

الاختصار	الاسم الكامل	المعنى
API	Application Programming Interface	واجهة برمجة التطبيقات
ARM	Advanced RISC Machine	مجموعة من التعليمات تعتمد على RISC
PDA	Personal Digital Assistant	المساعد الرقمي الشخصي
RDBMS	Relational Database Management System	نظام إدارة قواعد المعطيات العلاقاتية
RIM	Research In Motion	شركة برمجيات وعتاد كندية
RISC	Reduced Instruction Set Computer	حوسبة بمجموعة تعليمات مُصَغَّرة (موجزة)
SDK	Software Development Kit	مجموعة تطوير البرمجيات

جدول المصطلحات

المصطلح	المعنى / الدلالة
Dalvik VM	آلة Dalvik الافتراضية
Objective-C	لغة برمجة غرضية التوجه

الفهرس

1	الإطار العام للمشروع.....	1
2	1.1. هدف المشروع	2
2	2.1. متطلبات المشروع.....	2
2	1.2.1. المتطلبات الوظيفية لتطبيق المستخدم:	2
3	2.2.1. المتطلبات الوظيفية لموقع الويب الخاص بالمدير:	3
4	3.2.1. المتطلبات غير الوظيفية للنظام:	4
5	2. الدراسة المرجعية.....	5
6	1.2. الهواتف المحمولة	6
6	2.2. الهواتف الذكية	6
7	3.2. أنظمة تشغيل الهواتف الذكية.....	7
7	1.3.2.Symbian.....	7
8	2.3.2.BlackBerry.....	8
9	3.3.2.Windows Phone.....	9
9	4.3.2.Android.....	9
11	5.3.2.IOS.....	11
14	3. الدراسة التحليلية.....	14
15	1.3. مخطط حالات الاستخدام.....	15
15	1.1.3. موقع الويب.....	15

252.1.3. تطبيق الهاتف الذكي
332.3. مخططات التعاون
331.2.3. موقع الويب
372.2.3. تطبيق الهاتف الذكي
394. تصميم النظام
401.4. بنية قاعدة المعطيات Database
432.4. البنية المعمليّة للنظام البرمجي المعتمد Software Architecture
431.2.4. Repository Pattern
442.2.4. MVC Pattern
465. التنفيذ والاختبارات
471.5. التنفيذ
471.1.5. بيئة العمل
492.1.5. أدوات التنفيذ
493.1.5. مكونات النظام التفصيلية
562.5. تصميم واجهات التطبيق
626. الخاتمة والآفاق المستقبلية
637. المراجع

فهرس الأشكال

- الشكل 1: بنية نظام Android. 11
- الشكل 2: بنية iPhone. 13
- الشكل 3: مخطط حالات الاستخدام للجزء الخاص بالأدوية في موقع الويب. 16
- الشكل 4: مخطط حالات الاستخدام للجزء الاختبارات في موقع الويب. 23
- الشكل 5: مخطط حالات الاستخدام للجزء الخاص بالأدوية في تطبيق الهاتف الذكي. 26
- الشكل 6: مخطط حالات الاستخدام لتطبيق الهاتف الذكي للجزء الخاص بالاختبارات. 31
- الشكل 7: مخطط التعاون لحالة استخدام إضافة شركة أدوية. 34
- الشكل 8: مخطط التعاون لحالة استخدام تعديل شركة أدوية. 35
- الشكل 9: مخطط التعاون لحالة استخدام حذف شركة أدوية. 36
- الشكل 10: مخطط التعاون لحالة استخدام استعراض شركات الأدوية. 37
- الشكل 11: مخطط التعاون لحالة استخدام إجراء اختبار. 38
- الشكل 14: تصميم قاعدة المعطيات للجزء الخاص بالأدوية. 40
- الشكل 15: تصميم قاعدة المعطيات للجزء الخاص بالاختبارات. 41
- الشكل 16: رسم توضيحي يعبر عن مبدأ عمل Repository Pattern. 44
- الشكل 17 : شكل توضيحي لمبدأ عمل النمط MVC. 45
- الشكل 19: الواجهة الرئيسية لتطبيق الموبايل. 56
- الشكل 20: واجهة الأدوية في تطبيق الموبايل. 57
- الشكل 21: واجهة الأشكال الدوائية لتطبيق الموبايل. 58.....
- الشكل 22: واجهة الشكل الدوائي Tablet لتطبيق الموبايل. 58
- الشكل 23: واجهة الفئات المرضية لتطبيق الموبايل. 59.....
- الشكل 24: واجهة أحد الفئات المرضية لتطبيق الموبايل. 59
- الشكل 25: واجهة شركات الأدوية لتطبيق الموبايل. 60.....
- الشكل 26: واجهة إحدى شركات الأدوية لتطبيق الموبايل. 60

- الشكل 27: واجهة تسجيل الدخول لتطبيق الموبايل 61
- الشكل 28: واجهة إنشاء حساب لتطبيق الموبايل 61

فهرس الجداول

الجدول 1: السيناريو الرئيسي الناجح لحالة استخدام تسجيل الدخول للمدير.....	17
الجدول 2: المسار البديل 1 لحالة استخدام تسجيل الدخول للمدير.....	18
الجدول 3: السيناريو الرئيسي الناجح لحالة استخدام إضافة دواء.....	19
الجدول 4: السيناريو البديل 1 لحالة استخدام إضافة دواء الخاصة بالمدير.....	20
الجدول 5: السيناريو الرئيسي الناجح لحالة استخدام التعديل على دواء الخاصة بالمدير.....	21
الجدول 6: السيناريو الرئيسي الناجح لحالة استخدام حذف دواء الخاصة بالمدير.....	22
الجدول 8: السيناريو الرئيسي الناجح لحالة استخدام استعراض نتائج اختبارات أحد المستخدمين الخاصة بالمدير.....	24
الجدول 9: السيناريو الرئيسي الناجح لحالة استخدام استعراض المستخدمين الخاصة بالمدير.....	25
الجدول 10: المسار البديل 2 لحالة استخدام تسجيل الدخول للمستخدم.....	27
الجدول 11: المسار البديل 3 لحالة استخدام تسجيل الدخول للمستخدم.....	28
الجدول 12: المسار البديل 4 لحالة استخدام تسجيل الدخول للمستخدم.....	28
الجدول 13: السيناريو الرئيسي الناجح لحالة استخدام استعراض الأدوية.....	29
الجدول 14: السيناريو الرئيسي الناجح لحالة استخدام فترة الأدوية.....	30
الجدول 15: السيناريو الرئيسي الناجح لحالة استخدام إجراء اختبار.....	32

المقدمة

لاقت الهواتف الذكية رواجاً واسعاً في السنوات الأخيرة، فلم نعد نبحت عن هاتف نتمكّن من خلاله من إجراء مكالمات صوتية أو إرسال رسائل نصية، بل أصبحنا نبحت عن هاتف ذو ميزات أكثر مثل دقة الكاميرا، مساحة التخزين، نظام التشغيل، إمكانية الوصول للإنترنت. ومن هنا جاءت أفكار المطوّرين لزيادة التطبيقات التي تزيد من رغبة الناس بالتعامل مع هذه الهواتف، وعند انتشار تطبيق معيّن بين الناس يبحث المطوّر عن تحسينات يمكن إضافتها لهذا التطبيق من أجل زيادة ترغيب الناس في العمل مع هذا التطبيق دون غيره. لیتمكّن المطوّر من تحقيق هذا الهدف، يبحث عن الصعوبات التي تواجه الناس في حياتهم ويحاول العمل على تطبيق موبايل يخفّف قليلاً من هذه الصعوبات.

في الآونة الأخيرة ازدادت أعداد طلاب الصيدلة في سوريا، وأدّت هذه الزيادة إلى معاناة كبيرة لطلاب الصيدلة في البحث عن صيدلية تحوي شاعراً ليتدرّب بها في فترة الملائمة، بالإضافة لذلك الكثير من هؤلاء الطلاب يقطنون في أماكن ريفية لا تحوي صيدليات كثيرة فيضطر طالب الصيدلة إلى الذهاب لمسافات كبيرة كل يوم من أجل التدريب على المعلومات المتعلقة بالأدوية التي يجب أن يتعلّمها في فترة الملائمة. من هنا جاءت فكرة المشروع، والذي يهدف إلى تخفيف هذه المعاناة عن طريق توفير المعلومات المطلوبة عن الأدوية في تطبيق موبايل يمكن تنصيبه على هاتف ذكي يعمل بنظام Android أو iOS، لیتمكّن طلاب الصيدلة من استعراض هذه المعلومات وإجراء اختبارات تثبت أنّهم خضعوا لفترة التدريب المطلوبة.

الفصل الأول

الإطار العام للمشروع

نهدف في هذا الفصل إلى توضيح الإطار العام للمشروع من خلال تحديد هدف المشروع والمتطلبات الوظيفة والغير الوظيفية للمشروع.

1.1. هدف المشروع

ازدادت أعداد طلاب كلية الصيدلة في سوريا بشكل كبير في السنوات الأخيرة، وبالرغم من عدد الصيدليات الكبير إلا أنها لا تمتلك طاقة لاستيعاب كل هؤلاء الطلاب في فترة الملازمة. بالإضافة لذلك، إنّ فترة انتشار COVID-19 أدّت لتعطّل الجميع عن وظائفهم بما فيهم طلاب الصيدلة الذين كانوا يلازمون في صيدليات، حيث تمّ إعلامهم بالبقاء في منازلهم حتى نهاية الحجر، كما أنّ ازدياد صعوبات النقل في الفترة الأخيرة أدّى إلى حاجة ماسّة لهؤلاء الطلاب لإجراء الملازمة عن بعد. ومن هنا كان تطبيق الملازمة هو الحلّ الأمثل لتجاوز هذه الصعوبات، فيهدف التطبيق إلى تجميع كافة المعلومات التي يتعلّمها طالب الصيدلة في فترة الملازمة في صيدلية لتصبح متوقّرة على هواتف الطلاب دون الحاجة للبحث عن صيدلية تستوعب شاعراً يلازم فيها، ودون عناء المواصلات. وللتأكد من أنّ الطالب أكمل التدريب على التطبيق، يوفّر هذا التطبيق اختبارات للطلاب أيضاً يستطيع القيام بها متى ما أراد ويعطيه درجته المستحقّة بعد انتهاء الاختبار.

2.1. متطلبات المشروع

نوضح فيما يلي المتطلبات الوظيفية لتطبيق ملازمة صيدلية الخاص بالمستخدم، وموقع الويب الخاص بالمدير، ثم سنتحدث عن المتطلبات غير الوظيفية التي تشمل كلاهما معاً.

1.2.1. المتطلبات الوظيفية لتطبيق المستخدم:

1. يسمح التطبيق للمستخدم بالدخول والاستفادة من ميزات التطبيق عن طريق إدخال البريد الإلكتروني وكلمة المرور.
 - 1.1. يتيح التطبيق ميزة إنشاء حساب في حال عدم وجود حساب سابق.
 - 2.1. يمكن للمستخدم تعديل معلومات ملفه الشخصي (كلمة المرور، ...).
2. يتيح التطبيق للمستخدم استعراض جميع شركات الأدوية (أسماء الشركات، شعارها، والأدوية التي تصنعها مع جميع المعلومات الخاصة بهذه الأدوية).
3. يمكن للمستخدم من خلال هذا التطبيق استعراض جميع المواد الفعالة الموجودة في مختلف الأدوية.
4. يتيح التطبيق للمستخدم استعراض جميع الأشكال الدوائية (حب، شراب، حقن، مرهم موضعي، ...).
5. يمكن للمستخدم من خلال هذا التطبيق استعراض جميع الفئات المرضية (أمراض الكبد، أمراض المعدة، ...)، ويمكنه أيضاً استعراض جميع الأدوية الخاصة بكل فئة ومعلومات عن هذه الأدوية.

6. يتيح التطبيق للمستخدم استعراض المعلومات الكاملة لجميع الأدوية (اسم الدواء في كل من اللغتين العربية والإنكليزية، توصيف الدواء، اسم الشركة المصنّعة، الفئة المرضية التي ينتمي لها الدواء، جميع الأشكال الدوائية الموجودة لهذا الدواء، جميع المواد الفعالة في هذا الدواء، الآثار الجانبية له، ...).
7. يوفر التطبيق للمستخدم اختبارات تحوي أسئلة (اختيار من متعدد)، وتحوي هذه الأسئلة معلومات عن الأدوية للتأكد من أن الطالب قام بمرحلة التدريب.
- 1.7. قد تحوي هذه الاختبارات صورة وصفة طبية، وتتكوّن الخيارات من أسماء أدوية، بحيث تكون إحدى هذه الخيارات مكوّنة من أسماء الأدوية الموجودة في الصورة.
8. يتيح التطبيق للمستخدم الإجابة عن هذه الأسئلة.
- 1.8. يعرض التطبيق نتيجة الاختبار الذي قام به المستخدم.

2.2.1. المتطلبات الوظيفية لموقع الويب الخاص بالمدير:

1. يسمح النظام للمدير بالدخول للموقع عن طريق إدخال البريد الإلكتروني وكلمة المرور.
2. يمكن للمدير استعراض شركات الأدوية، وإضافة شركة جديدة، أو التعديل على شركة موجودة، أو حذفها.
3. يمكن للمدير استعراض الفئات المرضية، وإضافة فئة مرضية جديدة، أو التعديل على فئة مرضية موجودة سابقاً، أو حذفها.
4. يمكن للمدير استعراض أشكال الأدوية المختلفة، وإضافة شكل دوائي جديد، والتعديل على أحد الأشكال الموجودة، أو حذفها.
5. يمكن للمدير استعراض المواد الفعّالة الممكنة، وإضافة مادة فعّالة جديدة، والتعديل على مادة فعّالة موجودة مسبقاً، أو حذفها.
6. يمكن للمدير استعراض جميع الأدوية الموجودة، وإدخال دواء جديد مع جميع معلوماته (اسم الدواء في اللغتين العربية والإنكليزية، توصيفه، آثاره الجانبية، الشركة المصنّعة، الفئة المرضية التي يعالجها، المواد الفعّالة الموجودة داخله، الأشكال الدوائية له، ...).
7. يمكن للمدير استعراض الأسئلة الموجودة عن الأدوية (نص السؤال، علامة السؤال في حال كانت الإجابة صحيحة، علامة السؤال في حال كانت الإجابة خاطئة، العلامة في حال لا يوجد إجابة)، كما يمكنه إضافة أسئلة جديدة من نمط اختبار من متعدد، وإضافة الخيارات لهذه الأسئلة مع تحديد الخيار الصحيح من بين هذه الخيارات)، يمكنه أيضاً التعديل على هذه الأسئلة، أو حذف أسئلة موجودة.

- 1.7. يمكن للمدير إضافة سؤال يحوي صورة طبية مع خيارات بأسماء الأدوية المحتمل وجودها في هذه الوصفة، ويمكنه التعديل على هذا السؤال أو الصورة أو حذفهما.
8. يتيح النظام للمدير استعراض الخيارات الخاصة بكل سؤال وإمكانية إضافة خيارات لسؤال معين أو التعديل على خيار موجود أو حذفها.
9. يتيح النظام للمدير استعراض علامات المستخدمين.

3.2.1. المتطلبات غير الوظيفية للنظام:

1. متطلبات الأداء

- 1.1. الموثوقية: ينجز التطبيق جميع المهام المطلوبة دون خطأ.
- 2.1. التوفر: يجب أن يكون التطبيق متاحاً بشكل دائم.
- 3.1. الاستجابة: يجب أن يكون زمن الاستجابة سريعاً، لا يتجاوز 10 ثانية.

2. متطلبات الأمان

- 1.2. يجب أن يسمح النظام فقط للمستخدمين المسجلين بالدخول للتطبيق واستخدام ميزات التطبيق المصرح عنها لهم فقط.
- 2.2. يجب أن يكون هناك حماية لخصوصية المستخدمين.
- 3.2. لا يمكن للمستخدم أن يقوم بإجراءات تؤدي لمشاكل في التطبيق.

3. الواجهات

- 1.3. يجب أن يحوي النظام واجهات سهلة التعامل لكل من المستخدم والمدير.
- 2.3. يجب أن يلائم التطبيق جميع شاشات الهواتف الذكية بمختلف مقاساتها.

الفصل الثاني

الدراسة المرجعية

نعرض في هذا الفصل دراسة نظرية عن الهواتف المحمولة والذكية بأنواعها، وأنظمة التشغيل المستخدمة فيها.

1.2. الهواتف المحمولة

الهاتف المحمول هو جهاز اتصال لا سلكي يعمل عن طريق شبكات الاتصال اللاسلكية مثل شبكات الجيل الثاني (2G)، وشبكات الجيل الثالث (3G)، وشبكات الجيل الرابع (4G)، وغيرها، التي توفر الاتصال اللاسلكي بأبراج الاتصال المنتشرة على مساحات جغرافية واسعة، ليسمح للناس بالتواصل عبر المكالمات الصوتية أو الرسائل النصية.

شغلت الهواتف المحمولة جانباً أساسياً في تقدّم التكنولوجيا حيث ظهرت الهواتف الذكية التي تحاكي الحواسيب الشخصية من حيث عدّة ميزات.

2.2. الهواتف الذكية

حققت الهواتف الذكية نجاحاً كبيراً في جذب الزبائن، حيث انتشرت بسرعة كبيرة ويعود الفضل في هذا النجاح إلى الميزات العديدة الموجودة في الهواتف الذكية، حيث أنّها تقدّم خدمات كثيرة يقدّمها الحاسوب الشخصي على الرغم من صغر حجمها، فهي تتكوّن من شاشة قابلة للمس Touchscreen، لوحة مفاتيح مضمّنة داخلياً، كاميرا أمامية وكاميرا خلفية، ومن خلال هذا العتاد فخدماتها لم تُعد تقتصر على إجراء مكالمات صوتية وإرسال واستقبال رسائل نصية، وإنّما أصبحت تتيح للمستخدم إمكانية الوصول للإنترنت عن طريق Wi-Fi باستخدام المتصفح Browser، وإجراء مكالمات فيديو وتشغيل الوسائط من خلال مشغّل الوسائط Media Player بالإضافة لميزة تحديد الموقع GPS.

أدّى الانتشار الواسع الذي حقّقه الهواتف الذكية إلى تسارع كبير في تطوير التطبيقات والأنظمة التي تعمل على هذه الهواتف، كما تعدّدت أنواعها لتشمل عدّة أنظمة تشغيل أشهرها: Android و iOS، وهذا ما ألزم مطوّري التطبيقات تطوير تطبيقاتهم لكلّ نظام تشغيل على حدة من أجل تلبية احتياجات المستخدمين.

يمكن تنصيب هذه التطبيقات مسبقاً على الجهاز، أو بالإمكان تحميلها من قبل المستخدم عن طريق متجر التطبيقات. وبفضل هذه التطبيقات تصبح هذه الهواتف كأنّها حواسيب شخصية تحوي معالجات متعدّدة النواة، ووحدات تخزين كبيرة الحجم، إضافة إلى نظام التشغيل [1].

3.2. أنظمة تشغيل الهواتف الذكية

أنظمة التشغيل هي البنية الأساسية في أي جهاز لأنها هي التي تسمح للجهاز بتشغيل خدماته. يوجد العديد من أنظمة التشغيل الخاصة بالهواتف الذكية، ولكن أنظمة التشغيل الرئيسية الموجودة في الهواتف الذكية الأحدث هي ما يلي:

- Symbian التابع لشركة Nokia.
- BlackBerry التابع لشركة RIM.
- Windows Phone التابع لشركة Microsoft.
- Android التابع لشركة Google.
- iOS التابع لشركة Apple.

يمكن تنصيب أنظمة التشغيل الموجودة أعلاه على مختلف نماذج الهواتف، وكل جهاز يستقبل تحديثات لنظام التشغيل الخاص به طوال دورة حياته [2].

سيتم توضيح كل نظام تشغيل بشكل مبسّط:

1.3.2 Symbian

هو نوع من أنواع نظم التشغيل الخاصّة بالهواتف الذكية، تمّ إطلاقه في 24 حزيران عام 1998 كشراكة بين Ericsson، Nokia، Motorola، وPsion. وكان ذلك بهدف بناء جهاز يجمع بين أجهزة المساعد الرقمي الشخصي PDAs، والهواتف المحمولة. وفي 2 كانون الأول عام 2008 استحوذت شركة Nokia على هذا النظام بالكامل، وعندها أصبح جميع موظفي Symbian هم موظفين في شركة Nokia. ولكن تراجع استخدام هذا النظام لعدة أسباب أهمها:

- مظهر واجهات المستخدم القديمة، والتي أصبحت مملة والتحديثات عليها قليلة جداً [3].
- ضعف التطبيق، حيث أنّه على الرغم من وجود أكثر من 10,000 تطبيق أصلي للهواتف المحمول يعمل على Symbian، لكن استغرق نظام التشغيل هذا 7 سنوات للوصول إلى هذه العلامة على عكس أنظمة التشغيل الأخرى حيث أنّ شركات تطوير تطبيق iPhone تطلق الكثير من التطبيقات الجديدة كل ثلاثة أشهر [3].
- السرعة التشغيلية العامة لشركة Symbian منخفضة إلى حدّ ما، وكانت هناك شكاوى حول تعليق الأجهزة دون سبب واضح. كما أنّ حالات ضياع المكالمات على الهواتف التي تعمل بهذا النظام لم تكن مُستبعدة [3].

- أهم جزء في تصفّح الويب عبر الهاتف المحمول هو السرعة (على الأقل بالنسبة لمعظم المستخدمين)، وفشل Symbian فشلاً ذريعاً في هذا الجانب. لذلك، أصبح من الضروري للأشخاص الذين يمتلكون هواتف تعمل بهذا النظام تنزيل تطبيق متصفّح تابع لجهة خارجية، حتى يتمكنوا من الوصول إلى الويب. ومع ذلك، لم تكن سرعة الإنترنت قريبة من أجهزة iOS أو Android المتطورة [3].
- النجاح الكبير الذي حققه كل من نظام Android ونظام iOS، حيث أصبح معالج جهاز Android أفضل بكثير من معالج جهاز Nokia، كما أصبح النظام غير قابل للتحديث على عكس نظام Android و iOS [3].

BlackBerry 2.3.2

تم تطوير جهاز BlackBerry والنظام الداعم له بواسطة شركة (Research In Motion (RIM، وهي شركة برمجيات وعتاد كندية سُمّيت بعدها BlackBerry. يؤمّن نظام مراسلة لاسلكية متكامل، ممّا يوفّر الوصول إلى البريد الإلكتروني عبر الشبكات اللاسلكية الخلوية في جميع أنحاء العالم. أجهزة BlackBerry متعددة الاستخدامات، ويمكن استخدامها لمجموعة من الوظائف بما في ذلك الاتصالات الهاتفية والرسائل النصية القصيرة والبريد الإلكتروني وتصفح الويب من بين أشياء أخرى.

بينما يُستخدم BlackBerry نظام تشغيل خاص به، فإنّ إطار عمل تطبيق الطرف الثالث يعتمد بالكامل على Java. وبالتالي تتم كتابة التطبيقات المستهدفة لأجهزة BlackBerry بلغة Java ثم يتم تجميعها في ملفات (.cod). خاصة ويتم التحقق من صحة كود Java من جهة الحاسوب قبل أن يتم تحويله إلى ملف (.cod). ليقوم بعدها النظام بتنفيذه. العامل الرئيسي في نجاح هذا النظام هو منهجيته في الأمن، حيث أنّه افتراضياً، تتمتع التطبيقات غير الموقّعة بوصول محدود للغاية إلى الوظائف التي يُقدّمها النظام. كما يجب توقيع الطلبات من قبل RIM من أجل تنفيذ الإجراءات التي تُعتبر حسّاسة مثل تعداد مدير المعلومات الشخصية أو قراءة رسائل البريد الإلكتروني. إضافةً لذلك فإنّ التطبيقات الموقّعة أيضاً قد تطلب سمّاحة من المستخدم للقيام بأفعال حسّاسة مثل إجراء مكالمات هاتفية. على الرغم من هذه الميزات إلا أنّه في عام 2007 قامت شركة Apple بتطوير iPhone ولاقت رواجاً أكثر من BlackBerry لعدّة أسباب أهمها تصميم الواجهات، وأيضاً عندما أطلقت Google نظام Android تراجع نظام BlackBerry، وعلى الرغم من محاولة شركة RIM تطوير واجهات BlackBerry إلا أنّه عند انتهائها كان كل من Google و Apple قد أطلقوا متجر تطبيقات لكلٍ منهما وهذا أدّى لانتشار Android و iOS أكثر من BlackBerry [4].

Windows Phone 3.3.2

نظام تشغيل مفتوح المصدر تم تطويره من قبل شركة Microsoft عام 2008، وتكون واجهات الهواتف التي تعمل بهذا النظام شبيهة بواجهات نظام تشغيل الحاسب الشخصي Windows. يوفر هذا النظام تزامناً بين أرقام الهواتف والرسائل والمهام والمنبّهات بين الهاتف والحاسب. يعتمد تطوير التطبيقات الخاصة به على لغة ++C، أو C# [5].

Android 4.3.2

نظام التشغيل Android هو نظام تشغيل مفتوح المصدر قائم على Linux أطلقته شركة Google، ويتكون من عدة طبقات سيتم عرضها من الطبقة السفلى إلى الطبقة العليا:

- **نواة نظام لينوكس Linux Kernel:**

هذه الطبقة تدعم خدمات النظام الأساسية مثل الأمان، وإدارة الذاكرة، وإدارة العمليات، ومكدس الشبكة (Network stack)، وبرنامج التشغيل (المقود Driver). تعمل هذه النواة كطبقة مجردة بين عتاد الجهاز وبرمجياته الأخرى [2].

- **Runtime:**

تتضمن المكتبات الأساسية Core libraries، وآلة Dalvik الافتراضية، وبالتالي نظام Android على عكس باقي الأنظمة التي تدعم تطبيقات Java لا يدعم آلة Java الافتراضية. يتم تشغيل كل تطبيق Android بالإجرائية الخاصة به مع النسخة الخاصة به من آلة Dalvik الافتراضية. حيث يقوم النظام بترجمة التطبيقات المكتوبة بلغة Java إلى كود (dex). Dalvik executable، وهذا النمط من الملفات يعتبر مثاليًا للمساحات الصغيرة من الذاكرة. تقوم الآلة الافتراضية Dalvik بتنفيذ الكود وتعتمد على طبقة نواة Linux في الوظائف الأساسية مثل النسيبة Threading، وإدارة الذاكرة في المستوى الأدنى [2].

• المكتبات Libraries:

نظام Android يحوي مكتبات C/C++ تُستخدم من قبل مكونات النظام، يمكن للمطورين الاستفادة منها. إضافةً إلى ذلك فإن النظام يحوي مكتبات وسائط تعتمد على إطار عمل الوسائط المتعددة OpenCORE الخاص بشركة PacketVideo، الذي يدعم تشغيل وتسجيل الوسائط [2].

• إطار عمل التطبيق Application Framework:

جميع تطبيقات Android تُكتب بلغة Java، ويتم تزويدها بمجموعة تطبيقات أساسية تتضمن برنامج الرسائل النصية SMS، email client، التقويم، الخرائط، المتصفح، وجهات الاتصال وغيرها. يمتلك مطورو التطبيقات إمكانية الوصول الكامل لواجهات إطار عمل مُشتركة مُستخدمة من قبل التطبيقات الأساسية. يتم بناء التطبيقات بحيث يكون من السهل إعادة استخدام مكوناتها، حيث أن أي تطبيق يمكنه أن ينشر إمكانياته للتطبيقات الأخرى وأي تطبيق آخر يمكنه الاستفادة من هذه الإمكانيات. يتيح موقرو المحتوى للتطبيقات بأن تصل لبيانات التطبيقات الأخرى، ويؤمن مدير الموارد الوصول إلى الموارد غير البرمجية مثل السلاسل المترجمة والرسومات وملفات التخطيط. يمكن مدير الإشعارات جميع التطبيقات من عرض التنبيه المخصص في شريط الحالة. يدير مدير النشاط دورة حياة التطبيقات ويوفر حزمة عودة عامة للتنقل [2].

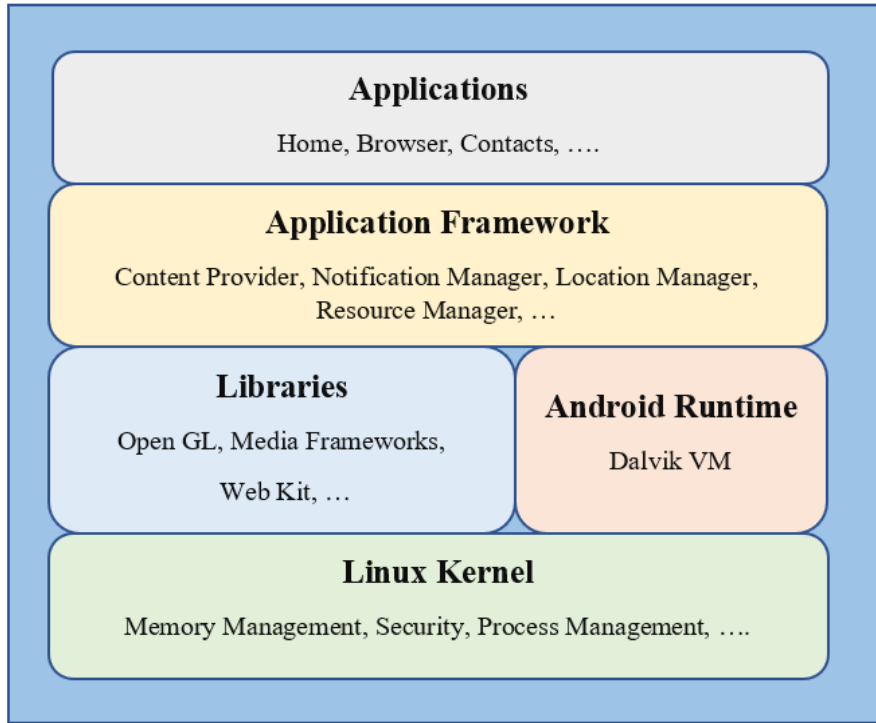
• التطبيقات Applications:

هي أهم ميزة في نظام التشغيل Android وهي غالباً تُطور بلغة Java باستخدام مجموعة تطوير برامج Android (Android SDK)، ويوجد أدوات تطوير أخرى منها مجموعة أدوات التطوير الأصلية للتطبيقات Native Development Kit أو الإضافات في C، C++.

من جانب آخر، بما أن Android هو نظام مفتوح المصدر فإنه يتيح للمطورين إمكانية إنشاء تطبيقات مبتكرة، حيث يمكنهم الاستفادة من عتاد الجهاز، الوصول للموقع، تشغيل خدمات الخلفية، إضافة إشعارات، ضبط منبهات وغيرها.

هذه الميزات التي يوفرها نظام Android أدت إلى ترغيب المطورين بتطوير تطبيقات Android عوضاً عن غيرها. حيث أعلنت شركة Google في عام 2011 أنه تمّ تنصيب حوالي 100,000 تطبيق Android. يتم استخدام نظام التشغيل Android على الهواتف الذكية والحواسيب المحمولة، والأجهزة اللوحية Tablets، بما في ذلك Dell Streak، Samsung Galaxy Tab، والتلفزيون والأجهزة الأخرى [2].

يبيّن الشكل (1) بنية نظام Android.



الشكل 1: بنية نظام Android [6].

IOS 5.3.2

هو نظام تشغيل خاص بشركة Apple، تمّ تطويره بشكل أساسي من أجل أجهزة iPhone، ولكن الآن نظام التشغيل iOS يدعم أجهزة Apple الأخرى مثل iPad، itouch و Apple TV. على الرغم من أنّه نظام مشتقّ من نظام التشغيل Mac OS X، إلا أنّه يمتلك تقنيّات متاحة على الأجهزة التي تعمل به فقط مثل: واجهة متعددة اللمس، قياس التسارع. الميزة الأساسية لهذا النظام هي العدد الهائل من التطبيقات، حيث أنه يملك حوالي 300,000 تطبيق موجودة في متجر التطبيقات الخاص بشركة Apple، والتي يبلغ عدد مرّات تنزيلها أكثر من 10 مليارات مرة، ويمكن أن يُنسب ذلك إلى مجموعة أدوات تطوير برامج (iOS SDK)، والتي تحتوي على التعليمات البرمجية والمعلومات والأدوات التي يحتاجها الأشخاص لتطوير واختبار وتشغيل وتصحيح الأخطاء وضبط التطبيقات لنظام iOS [2].

بنية iPhone:

سيتم توضيح الطبقات الموجودة في جهاز iPhone.

• العتاد Hardware:

يشير العتاد في أجهزة iPhone إلى الأجزاء المادية الملموسة بدارات iPhone، يقع المعالج الفعلي ضمن هذه الطبقة، ولكن يتم تضمين مجموعة التعليمات وجداول عنوان الذاكرة داخل طبقة المعالج [6].

• البرمجيات الراسخة Firmware:

تشير البرمجيات الراسخة إلى الرّماز الخاص بالشريحة الموجود إمّا مع الذاكرة داخل/حول الجهاز الطرفي، أو داخل محرّك الأقراص للأجهزة الطرفية [6].

• المعالج Processor:

هذه الطبقة تعتمد على مجموعة تعليمات ARM، وتتميز معالجات ARM بكفاءة استهلاك الطاقة، مما يسمح بعمر بطارية أطول للأجهزة المحمولة. كما توفر معمارية ARM مرونة وتعددية في التطبيقات، مما يتيح للمطوّرين تصميم أنظمة مخصّصة ومنخفضة التكلفة. بالإضافة إلى ذلك، توفر معمارية ARM دعماً قوياً لتقنيات الأمان والتشفير. وتحتوي هذه الطبقة جدول المقاطعات والذي يتم تحديده بواسطة نظام التشغيل عند الإقلاع [6].

• نظام التشغيل OS:

هذه الطبقة هي نواة برامج التشغيل والخدمات التي يتكوّن منها نظام التشغيل، وهي صلة وصل بين واجهات المستخدم والعتاد [6].

• Objective-C Runtimes:

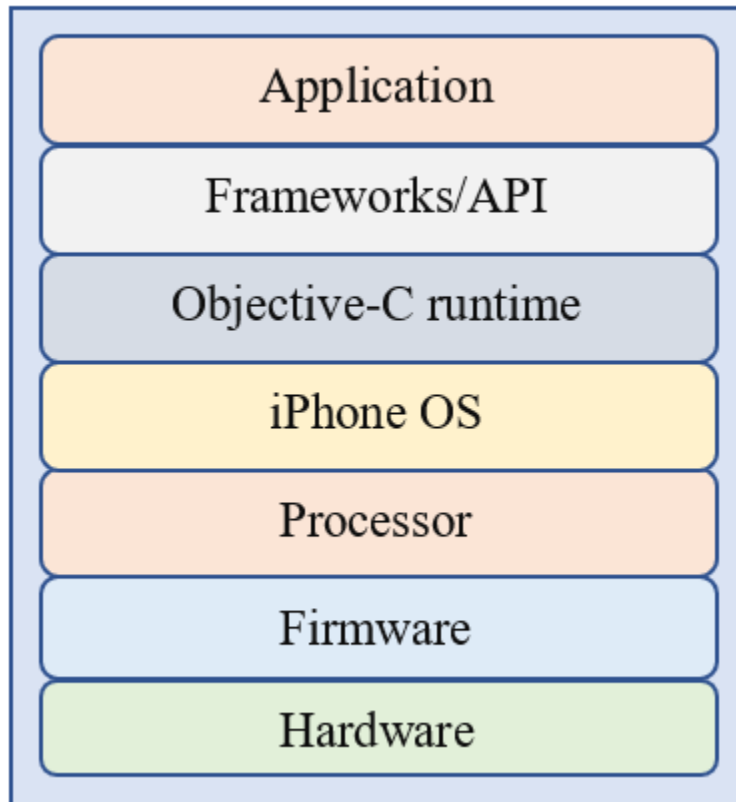
إن لغة Objective-C هي الأساس لشركة Apple، تم تطويرها في ثمانينيات القرن العشرين، وتم استخدامها في أنظمة التشغيل المبكرة، والآن يتم استخدامها في أنظمة التشغيل OS X و iOS. تعتمد هذه اللغة على لغة C وتضيف إليها مبادئ غرضية التوجه. وبالتالي فإنّ هذه الطبقة تتكوّن من مكتبات وقت تشغيل Objective-C المرتبطة ديناميكياً، بالإضافة لمكتبات C الأساسية [6].

- أطر العمل / الواجهات Frameworks/API:

تحتوي هذه الطبقة على استدعاءات API وهي عبارة عن ترويسات موزعة مع iPhone SDK، مع حدوث بعض الارتباط الديناميكي في وقت التشغيل، وتوجد هذه الطبقة أعلى طبقة Objective-C، حيث تتم كتابة العديد منها في لغة Objective-C [6].

- التطبيقات Applications:

التطبيق المخزن في iPhone، يجب شراؤه من مخزن التطبيقات، تم تجميع هذا التطبيق من التعليمات البرمجية الأصلية بواسطة مترجم iPhone الموزع من Apple، وتم ربطه بوقت تشغيل Objective-C ومكتبة C بواسطة الرابط. يعمل التطبيق أيضاً بالكامل داخل بيئة المستخدم التي تم إعدادها بواسطة نظام تشغيل iPhone [6]. يوضح الشكل (2) بنية نظام iPhone.



الشكل 2: بنية iPhone [6].

الفصل الثالث

الدراسة التحليلية

في هذا الفصل سيتم تحليل المشروع من خلال تحليل المتطلبات، وستشمل عملية التحليل كل من مخطط حالات الاستخدام والسرد النصي لكلٍ منها.

1.3. مخطط حالات الاستخدام

النظام البرمجي الذي نسعى لتحقيقه مؤلف من تطبيق الهاتف الذكي للمستخدم وموقع ويب للمدير، كما أنه مؤلف من قسمين: قسم خاص بالأدوية وقسم للاختبارات، وبالتالي سنقوم بتوصيف حالات الاستخدام للنظام على الشكل التالي:

- مخطط حالات الاستخدام للمدير (خاص بالأدوية).
- مخطط حالات الاستخدام للمدير (خاص بالاختبارات).
- مخطط حالات الاستخدام للمستخدم (خاص بالأدوية).
- مخطط حالات الاستخدام للمستخدم (خاص بالاختبارات).

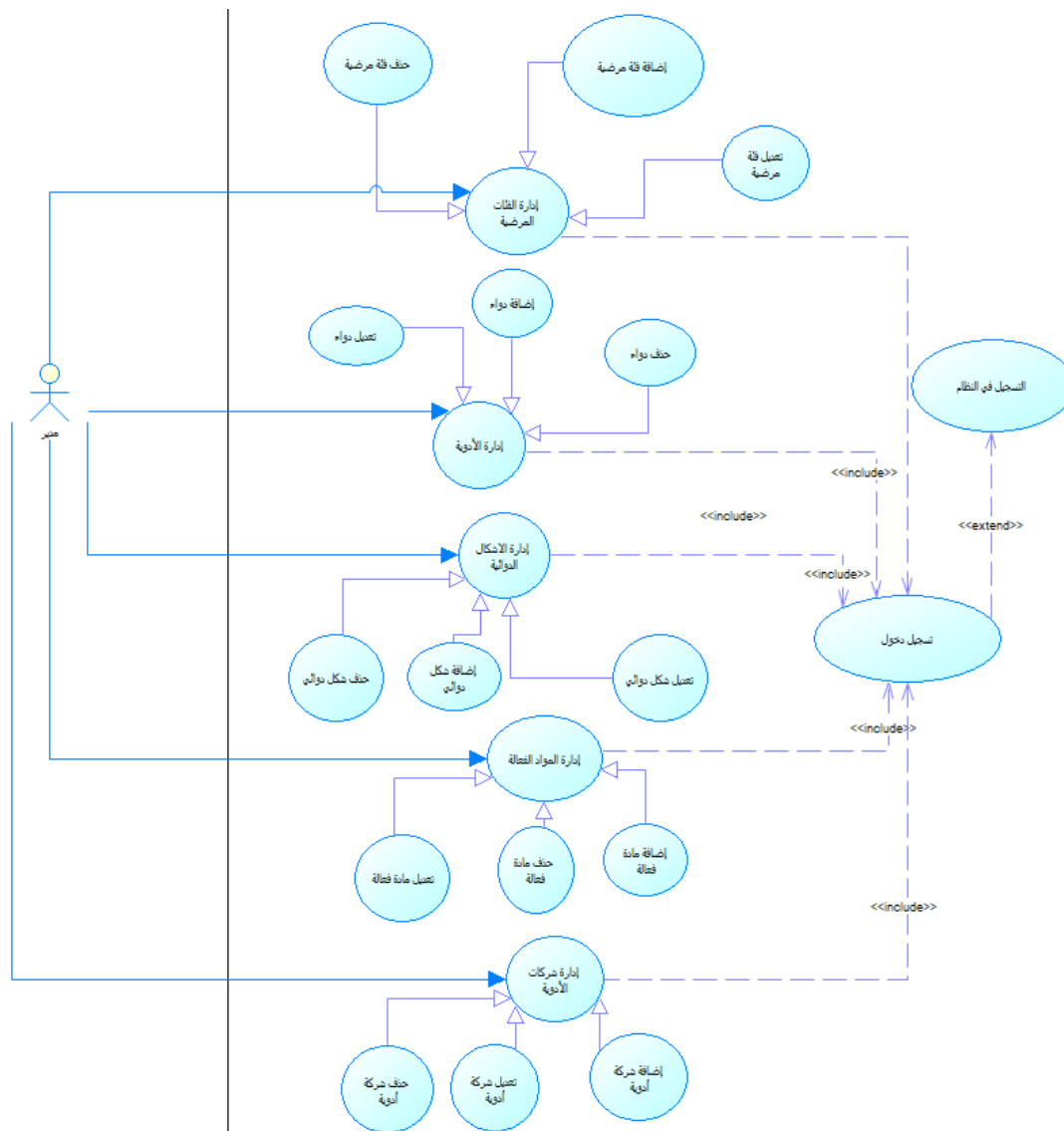
1.1.3. موقع الويب

يتفاعل المدير فقط مع هذا النظام، الذي يمتلك الصلاحيات التالية:

- إدارة المستخدمين.
- إدارة محتوى النظام والمتمثلة بالشكل التالي:
 - إدارة شركات الأدوية.
 - إدارة الأدوية.
 - إدارة المواد الفعالة.
 - إدارة الأشكال الدوائية.
 - إدارة الفئات المرضية.
 - إدارة الاختبارات.

وتتمثل إدارة المحتوى بإمكانية استعراض المحتوى بالكامل، الإضافة أو التعديل عليه، والحذف منه.

- مخطط حالات الاستخدام لموقع الويب للجزء الخاص بالأدوية فقط:
- لتحسين مظهر المخطط، لم يتم ذكر حالة الاستخدام (الاستعراض) لكل شكل من أشكال الإدارة، وإنما تم وضعها مع مخطط حالات الاستخدام للمستخدم لأنّ المستخدم يمكنه استعراض جميع محتويات النظام، يبيّن الشكل (3) مخطط حالات الاستخدام للجزء المتعلّق في الأدوية.



الشكل 3: مخطط حالات الاستخدام للجزء الخاص بالأدوية في موقع الويب.

➤ السرد النصي لحالات الاستخدام:

1. حالة استخدام تسجيل الدخول:

- الشروط اللاحقة: تمت عملية تسجيل الدخول للمستخدم.
- الأخطاء: لا يوجد.
- اسم حالة الاستخدام: تسجيل الدخول.
- الفاعلون الأساسيون: مدير النظام.
- الفاعلون الثانويون: لا يوجد.
- الملخص: يقوم مدير النظام بطلب تسجيل الدخول لموقع الويب.
- الشروط المسبقة: المدير مُسجّل في النظام.
- السيناريو الرئيسي الناجح:

الفاعل	النظام
1. تبدأ حالة الاستخدام عندما يطلب المدير تسجيل الدخول.	
	2. يطلب النظام من المدير إدخال البريد الإلكتروني وكلمة المرور.
3. يُدخِل المدير البريد الإلكتروني وكلمة المرور.	
	4. يتحقق النظام من وجود حساب لهذا البريد الإلكتروني ومن صحة كلمة المرور

الجدول 1: السيناريو الرئيسي الناجح لحالة استخدام تسجيل الدخول للمدير.

■ المسارات البديلة:

المسار البديل 1: كلمة المرور غير صحيحة.

يبدأ هذا المسار بعد الخطوة 4 من السيناريو الرئيسي الناجح.

الفاعل	النظام
	5. يُظهر النظام رسالة تدلّ على عدم صحة كلمة المرور ويطلب من المدير إعادة إدخالها.
6. يدخل المدير كلمة المرور مرة أخرى.	
	7. يتحقق النظام من صحة كلمة المرور.

الجدول 2: المسار البديل 1 لحالة استخدام تسجيل الدخول للمدير.

2. حالة استخدام إضافة دواء:

- اسم حالة الاستخدام: إضافة دواء.
- الفاعلون الأساسيون: مدير النظام.
- الفاعلون الثانويون: لا يوجد.
- الملخص: يقوم مدير النظام بطلب إضافة دواء جديد.
- الشروط المسبقة: المدير قام بتسجيل الدخول في النظام.

■ السيناريو الرئيسي الناجح:

الفاعل	النظام
1. تبدأ حالة الاستخدام عندما يطلب المدير من النظام عملية إضافة دواء.	
	2. يطلب النظام من المدير إدخال معلومات الدواء عن طريق إظهار استمارة تمكّن المدير من إدخال معلومات الدواء من خلالها.
3. يُدخِل المدير معلومات الدواء عن طريق ملء ال استمارة.	
	4. يتحقق النظام من أنّ المعلومات التي أدخلها المدير تتوافق مع الأنماط المحددة لكلّ منها.
	5. يقوم النظام بإضافة الدواء.
	6. يقوم النظام بحفظ التغييرات.

الجدول 3: السيناريو الرئيسي الناجح لحالة استخدام إضافة دواء.

■ المسارات البديلة:

المسار البديل 1: المعلومات التي أدخلها المدير لا تتوافق مع الأنماط المحددة داخل النظام.
يبدأ هذا المسار بعد الخطوة 4 من السيناريو الرئيسي الناجح.

الفاعل	النظام
	5. يُظهر النظام رسالة تدلّ على أنّ المعلومات التي أدخلها المدير غير متوافقة مع الأنماط المحددة لكلّ منها. ويطلب منه إعادة ملء الاستمارة.
	6. يقوم المدير بإعادة إدخال معلومات الدواء عن طريق ملء الاستمارة.
	7. يتحقق النظام من أنّ المعلومات التي أدخلها المدير تتوافق مع الأنماط المحددة لكلّ منها.
	8. يقوم النظام بإضافة الدواء.
	9. يقوم النظام بحفظ التغييرات.

الجدول 4: السيناريو البديل 1 لحالة استخدام إضافة دواء الخاصة بالمدير.

■ الشروط اللاحقة: أُضيف دواء جديد للنظام.

■ الأخطاء: لا يوجد.

3. حالة استخدام تعديل دواء:

■ اسم حالة الاستخدام: تسجيل الدخول.

■ الفاعلون الأساسيون: مدير النظام.

■ الفاعلون الثانويون: لا يوجد.

■ الملخص: يقوم مدير النظام بطلب عملية تعديل دواء.

■ الشروط المسبقة: المدير قام بتسجيل الدخول للنظام.

■ السيناريو الرئيسي الناجح:

الفاعل	النظام
1. تبدأ حالة الاستخدام عندما يطلب المدير من النظام عملية تعديل دواء.	
	2. يطلب النظام من المدير إدخال المعلومات الجديدة للدواء المراد تعديله عن طريق إظهار استمارة تحوي معلومات الدواء الحالية ليقوم المدير بالتعديل عليها.
3. يقوم المدير بالتعديل على المعلومات الحالية للدواء.	
	4. يتحقق النظام من أنّ المعلومات التي أدخلها المدير تتوافق مع الأنماط المحددة لكلّ منها.
	5. يقوم النظام بإضافة الدواء.
	6. يقوم النظام بحفظ التغييرات.

الجدول 5: السيناريو الرئيسي الناجح لحالة استخدام التعديل على دواء الخاصة بالمدير.

■ المسارات البديلة:

المسار البديل 1: المعلومات التي أدخلها المدير لا تتوافق مع الأنماط المحددة داخل النظام.

يبدأ هذا المسار بعد الخطوة 4 من السيناريو الرئيسي الناجح وهو تماماً مثل المسار البديل 1 لحالة استخدام إضافة دواء.

■ الشروط اللاحقة: تمّ تعديل دواء موجود في النظام.

■ الأخطاء: لا يوجد.

4. حالة استخدام حذف دواء:

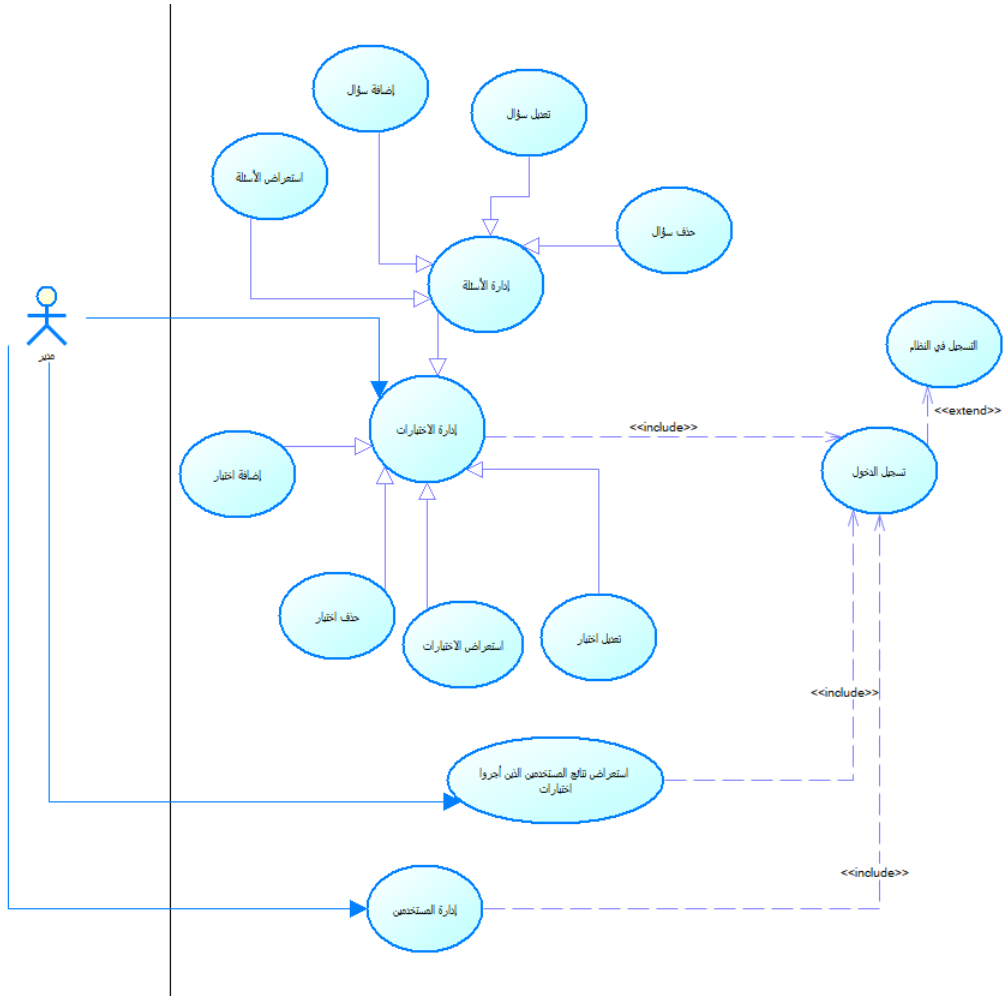
- اسم حالة الاستخدام: حذف دواء.
- الفاعلون الأساسيون: مدير النظام.
- الفاعلون الثانويون: لا يوجد.
- الملخص: يقوم مدير النظام بطلب عملية حذف دواء.
- الشروط المسبقة: المدير قام بتسجيل الدخول للنظام.
- السيناريو الرئيسي الناجح:

الفاعل	النظام
1. تبدأ حالة الاستخدام عندما يطلب المدير عملية حذف دواء.	
	2. يقوم النظام بحذف الدواء.
	3. يقوم النظام بحفظ التغييرات.

الجدول 6: السيناريو الرئيسي الناجح لحالة استخدام حذف دواء الخاصة بالمدير.

- المسارات البديلة: لا يوجد.
 - الشروط اللاحقة: تمّ تعديل دواء موجود في النظام.
 - الأخطاء: إذا كان الدواء مرتبطاً بأشكال دوائية أو مواد فعالة، لا يمكن حذفه إلا إذا تم حذف الأشكال الدوائية والمواد الفعالة الخاصة به.
- ملاحظة: بالنسبة لحالات الاستخدام: الإضافة، التعديل، والحذف لكلٍ من شركات الأدوية، الفئات المرضية، المواد الفعالة، والأشكال الدوائية فهي تماماً مثل حالات الاستخدام: الإضافة، التعديل، والحذف للدواء. لذلك لن يتم ذكرها مرة أخرى.

➤ مخطط حالات الاستخدام لموقع الويب للجزء الخاص بالاختبارات:



الشكل 4: مخطط حالات الاستخدام للجزء الاختبارات في موقع الويب.

➤ السرد النصي لحالات الاستخدام:

1. حالة استخدام استعراض نتائج اختبارات المستخدمين:

- اسم حالة الاستخدام: استعراض نتائج اختبارات المستخدمين.
- الفاعلون الأساسيون: مدير النظام.
- الفاعلون الثانويون: لا يوجد.

- الملخص: يقوم مدير النظام بطلب استعراض نتائج أحد المستخدمين الذين أجروا اختبارات.
- الشروط المسبقة: المدير قام بتسجيل الدخول إلى النظام، والمستخدم قام بإجراء اختبار.
- السيناريو الرئيسي الناجح:

الفاعل	النظام
1. تبدأ حالة الاستخدام عندما يقوم المدير بطلب عملية استعراض نتائج اختبارات أحد المستخدمين.	
	2. يقوم النظام بعرض نتائج ذلك المستخدم.

الجدول 7: السيناريو الرئيسي الناجح لحالة استخدام استعراض نتائج اختبارات أحد المستخدمين الخاصة بالمدير.

- المسارات البديلة: في حال لم يكن المستخدم قد قام بإجراء اختبار ويبدأ هذا المسار بعد المرحلة 1 من السيناريو الرئيسي الناجح، فيُظهر النظام رسالة تدلّ على أنّ المستخدم لم يقم بإجراء اختبار بعد.
- الشروط اللاحقة: عُرضت نتائج المستخدم.
- الأخطاء: لا يوجد.

2. حالة استخدام استعراض المستخدمين:

- اسم حالة الاستخدام: استعراض المستخدمين.
- الفاعلون الأساسيون: مدير النظام.
- الفاعلون الثانويون: لا يوجد.
- الملخص: يقوم مدير النظام بطلب استعراض المستخدمين المسجلين في النظام.
- الشروط المسبقة: المدير قام بتسجيل الدخول إلى النظام.

■ السيناريو الرئيسي الناجح:

الفاعل	النظام
1. تبدأ حالة الاستخدام عندما يقوم المدير بطلب عملية استعراض المستخدمين.	
	2. يقوم النظام بعرض معلومات المستخدمين المسجلين في النظام.

الجدول 8: السيناريو الرئيسي الناجح لحالة استخدام استعراض المستخدمين الخاصة بالمدير.

■ المسارات البديلة: في حال لم يكن هناك مستخدمين في النظام، يبدأ هذا المسار بعد المرحلة 1 في السيناريو

الرئيسي الناجح، فيُظهر النظام رسالة تدلّ على عدم وجود مستخدمين مسجلين في النظام بعد.

■ الشروط اللاحقة: عُرضت معلومات المستخدمين.

■ الأخطاء: لا يوجد.

ملاحظة: بالنسبة لحالات الاستخدام: الإضافة، التعديل، والحذف لكلٍ من الاختبار، والأسئلة فهي تماماً مثل

حالات الاستخدام: الإضافة، التعديل، والحذف للدواء. لذلك لن يتم ذكرها مرة أخرى.

2.1.3. تطبيق الهاتف الذكي

يتفاعل مع هذا النظام المستخدم الذي يمتلك الصلاحيات التالية:

● استعراض محتوى النظام والمتمثلة بالشكل التالي:

○ استعراض شركات الأدوية وفلترتها.

○ استعراض الأدوية وفلترتها.

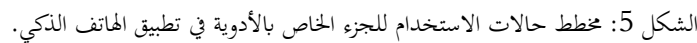
○ استعراض المواد الفعالة وفلترتها.

○ استعراض الأشكال الدوائية وفلترتها.

○ استعراض الفئات المرضية وفلترتها.

○ إجراء اختبارات.

كما تم التوضيح سابقاً أنه تم وضع المدير في مخطط حالات الاستخدام هذا لتحسين المنظر لأنّ المدير أيضاً يمكنه استعراض محتوى النظام، يبيّن الشكل (5) مخطط حالات الاستخدام لتطبيق الهاتف الذكي للجزء المتعلّق بالأدوية.



1. حالة استخدام تسجيل الدخول للمستخدم: هي نفسها حالة استخدام تسجيل الدخول للمدير ولكن مع وجود

■ اسم حالة الاستخدام: تسجيل الدخول.

- الفاعلون الأساسيون: مدير النظام.
- الفاعلون الثانويون: لا يوجد.
- الملخص: يقوم مدير النظام بطلب تسجيل الدخول لموقع الويب.
- الشروط المسبقة: المستخدم غير مسجل في النظام.
- المسارات البديلة:

المسار البديل 1: كلمة المرور غير صحيحة وهو مثل المسار البديل لحالة استخدام تسجيل الدخول للمدير الموجودة في الجدول (2).

المسار البديل 2: المستخدم غير مسجل في النظام.

يبدأ هذا المسار بعد الخطوة 4 في السيناريو الأساسي الناجح الموجود في الجدول (1).

الفاعل	النظام
	5. يظهر النظام رسالة تدلّ على أنّ المستخدم غير مسجل في النظام.
6. يطلب المستخدم إنشاء حساب.	
	7. يطلب النظام من المستخدم إدخال البريد الإلكتروني وتعيين كلمة مرور وإعادة إدخال كلمة المرور للتأكد من توافق الكلمتين.
8. يدخل المستخدم البريد الإلكتروني ويدخل كلمة المرور مرتين.	
	9. يتحقق النظام من أنّ كلمة المرور موافقة لنمط معين مُحدّد في النظام.
	10. يتحقق النظام من توافق كلمتي المرور.

الجدول 9: المسار البديل 2 لحالة استخدام تسجيل الدخول للمستخدم.

المسار البديل 3: كلمة المرور غير موافقة للنمط المحدّد في النظام.

يبدأ هذا المسار بعد الخطوة 9 في المسار البديل 2.

الفاعل	النظام
	10. يُظهر النظام رسالة تتضمن النمط الذي يجب أن تكون وفقه كلمة المرور ويطلب من المستخدم إعادة إدخالها.
11. يدخل المستخدم كلمة مرور جديدة موافقة للنمط.	
	12. يتحقق النظام من توافق كلمتي المرور.

الجدول 10: المسار البديل 3 لحالة استخدام تسجيل الدخول للمستخدم.

المسار البديل 4: كلمتي المرور غير متوافقتين.

يبدأ هذا المسار بعد الخطوة 10 في المسار البديل 2.

الفاعل	النظام
	11. يُظهر النظام رسالة تدلّ على عدم توافق كلمتي المرور ويطلب من المستخدم إعادة إدخالها.
12. يُدخل المستخدم كلمة المرور مرة أخرى.	
	13. يتحقق النظام من أنّ كلمة المرور موافقة لنمط معين مُحدّد في النظام.
	14. يتحقق النظام من توافق كلمتي المرور.
	15. يظهر النظام رسالة بأنّه تمت عملية إنشاء حساب جديد بنجاح.

الجدول 11: المسار البديل 4 لحالة استخدام تسجيل الدخول للمستخدم.

- الشروط اللاحقة: تمت عملية تسجيل الدخول للمستخدم.
- الأخطاء: لا يوجد.

2. حالة استخدام استعراض الأدوية:

- اسم حالة الاستخدام: استعراض الأدوية.
- الفاعلون الأساسيون: المستخدم بالنسبة للتطبيق ومدير النظام بالنسبة للويب.
- الفاعلون الثانويون: لا يوجد.
- الملخص: يقوم المستخدم أو المدير بطلب استعراض الأدوية الموجودة في النظام.
- الشروط المسبقة: المدير أو المستخدم قاموا بتسجيل الدخول إلى النظام.
- السيناريو الرئيسي الناجح:

الفاعل	النظام
1. تبدأ حالة الاستخدام عندما يقوم المدير أو المستخدم بطلب عملية استعراض الأدوية.	
	2. يقوم النظام بعرض جميع الأدوية والمعلومات الخاصة بها.

الجدول 12: السيناريو الرئيسي الناجح لحالة استخدام استعراض الأدوية.

- المسارات البديلة: في حال لم يكن هناك أدوية في النظام، يبدأ هذا المسار بعد المرحلة 1 في السيناريو الرئيسي الناجح، فيُظهر النظام رسالة تدلّ على عدم وجود أدوية في النظام بعد.
- الشروط اللاحقة: عُرضت معلومات الأدوية.
- الأخطاء: لا يوجد.

3. حالة استخدام فترة الأدوية:

- اسم حالة الاستخدام: فترة الأدوية.
- الفاعلون الأساسيون: المستخدم بالنسبة للتطبيق ومدير النظام بالنسبة للويب.
- الفاعلون الثانويون: لا يوجد.
- الملخص: يقوم المستخدم أو المدير بطلب فترة الأدوية الموجودة في النظام وذلك حسب اسم الدواء أو الشركة المصنّعة أو الفئة المرضية.
- الشروط المسبقة: المدير أو المستخدم قاموا بتسجيل الدخول إلى النظام.

■ السيناريو الرئيسي الناجح:

الفاعل	النظام
1. تبدأ حالة الاستخدام عندما يقوم المدير أو المستخدم بطلب عملية فلترة الأدوية.	
	2. يقوم النظام بعرض جميع الأدوية التي تحقق شرط الفلترة والمعلومات الخاصة بتلك الأدوية.

الجدول 13: السيناريو الرئيسي الناجح لحالة استخدام فلترة الأدوية.

■ المسارات البديلة: في حال لم يكن هناك أدوية في النظام تحقق شرط الفلترة، يبدأ هذا المسار بعد المرحلة 1

في السيناريو الرئيسي الناجح، فيُظهر النظام رسالة تدلّ على عدم وجود أدوية تحقق شرط الفلترة في النظام.

■ الشروط اللاحقة: عُرضت معلومات الأدوية التي تم البحث عنها.

■ الأخطاء: لا يوجد.

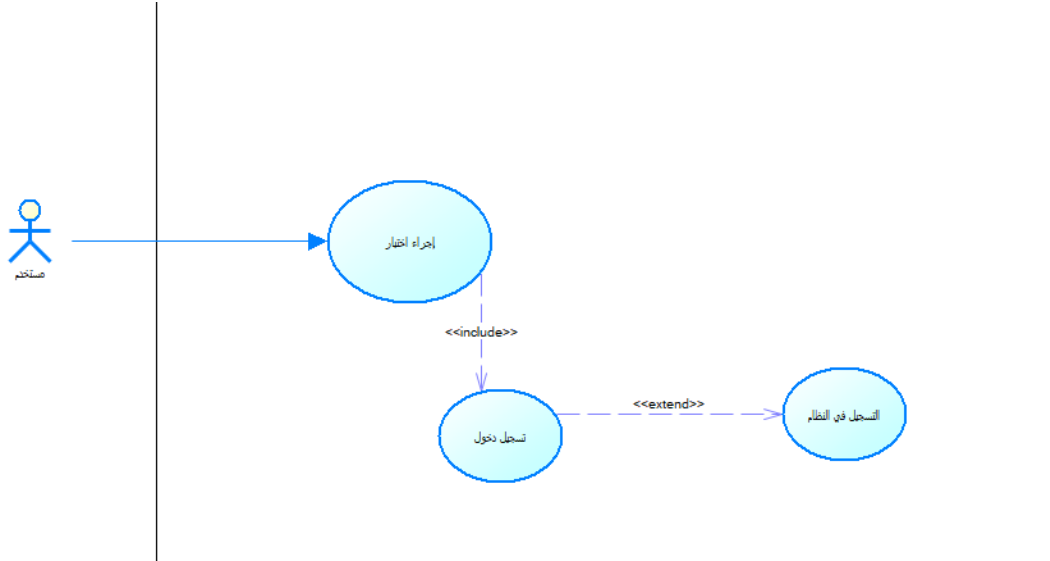
ملاحظة: بالنسبة لحالات الاستخدام: الاستعراض، والفلترة لكلٍّ من شركات الأدوية، الفئات المرضية، الأشكال

الدوائية، والمواد الفعالة، فهي تماماً مثل حالات الاستخدام: الاستعراض، والفلترة للدواء. لذلك لن يتم ذكرها

مرة أخرى.

● مخطط حالات الاستخدام لتطبيق الهاتف الذكي للجزء الخاص بالاختبارات:

يبيّن الشكل (6) مخطط حالات الاستخدام لجزء الاختبارات الخاص بتطبيق المستخدم.



الشكل 6: مخطط حالات الاستخدام لتطبيق الهاتف الذكي للجزء الخاص بالاختبارات.

1. حالة استخدام إجراء اختبار:

- اسم حالة الاستخدام: إجراء اختبار.
- الفاعلون الأساسيون: المستخدم.
- الفاعلون الثانويون: لا يوجد.
- الملخص: يقوم المستخدم بطلب إجراء اختبار، ومن ثم يختار اختبار من الاختبارات الموجودة في النظام.
- الشروط المسبقة: المستخدم قام بتسجيل الدخول إلى النظام، يوجد اختبارات في النظام.

■ السيناريو الرئيسي الناجح:

الفاعل	النظام
1. تبدأ حالة الاستخدام عندما يقوم المستخدم بطلب إجراء اختبار.	
	2. يقوم النظام بعرض جميع الاختبارات الموجودة.
3. يختار المستخدم اختبار ليقوم به.	
	4. يقوم النظام بعرض سؤال من أسئلة هذا الاختبار وتكون طريقة اختيار السؤال بشكل عشوائي.
	5. يضع النظام العلامة 0 كنتيجة أولية للاختبار.
6. يقوم المستخدم بالإجابة على السؤال.	
	7. يُظهر النظام رسالة توضّح فيما إذا كانت الإجابة صحيحة أم لا.
	8. يقوم النظام بتعديل علامة المستخدم حسب الإجابة التي اختارها.
9. يقوم المستخدم بالانتقال للسؤال التالي.	
	10. يقوم النظام بعرض سؤال آخر من الاختبار أيضاً بشكل عشوائي وهكذا يتم تكرار الخطوات 4، 5، 6، 7 حتى نهاية الأسئلة.
	11. يُظهر النظام نتيجة الاختبار النهائية.

الجدول 14: السيناريو الرئيسي الناجح لحالة استخدام إجراء اختبار.

- **المسارات البديلة:** في حال لم يكن هناك اختبارات في النظام بعد، يبدأ هذا المسار بعد المرحلة 1 في السيناريو الرئيسي الناجح، فيُظهر النظام رسالة تدلّ على عدم وجود اختبارات.
- **الشروط اللاحقة:** تمّ إجراء اختبار من قبل المستخدم وسُجّلت نتيجة الاختبار في النظام.
- **الأخطاء:** لا يوجد.

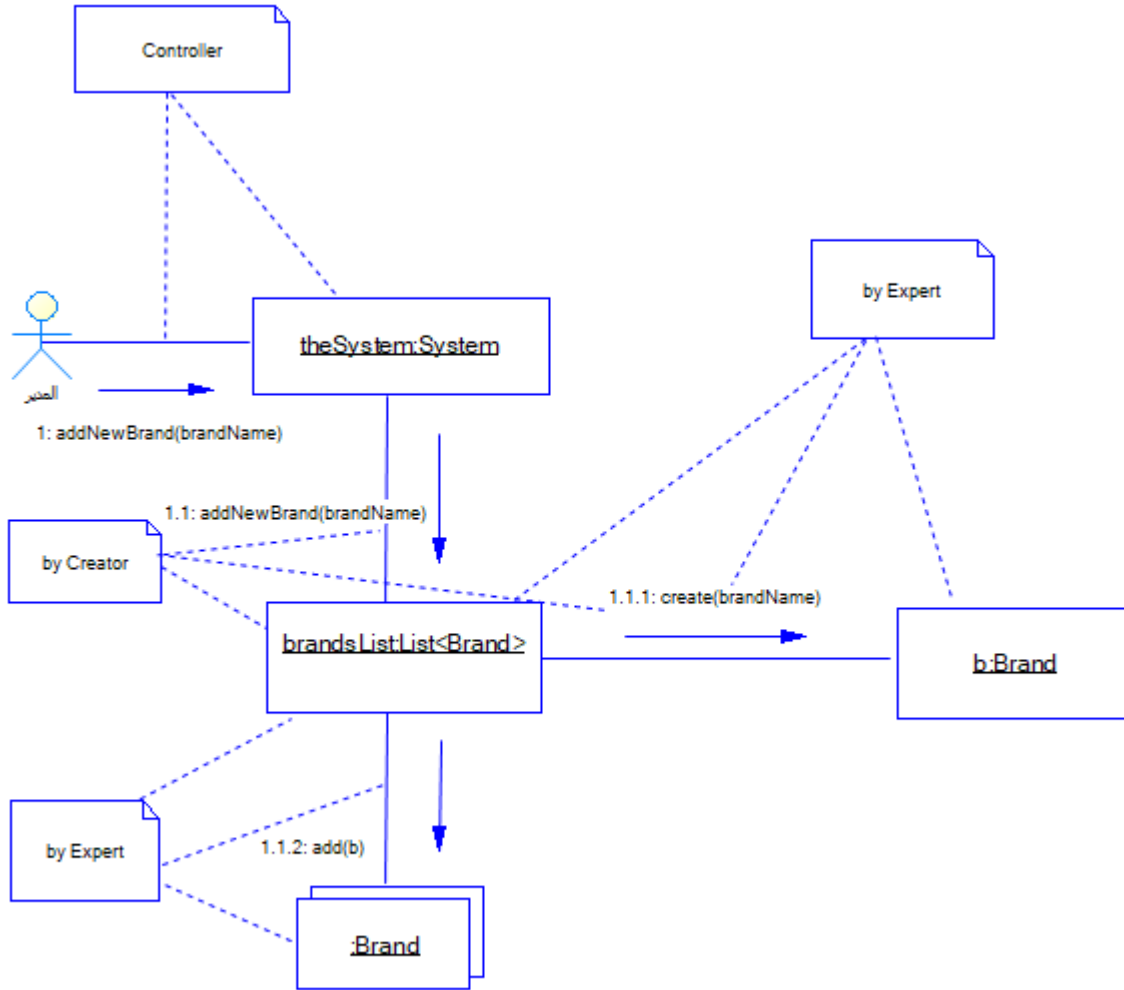
2.3. مخططات التعاون

1.2.3. موقع الويب

سيتمّ رسم مخططات التعاون لحالات الاستخدام المعيّنة عن إدارة شركات الأدوية فقط، لأنّ إدارة باقي المحتوى نفس الأمر.

- حالة استخدام إضافة شركة أدوية:

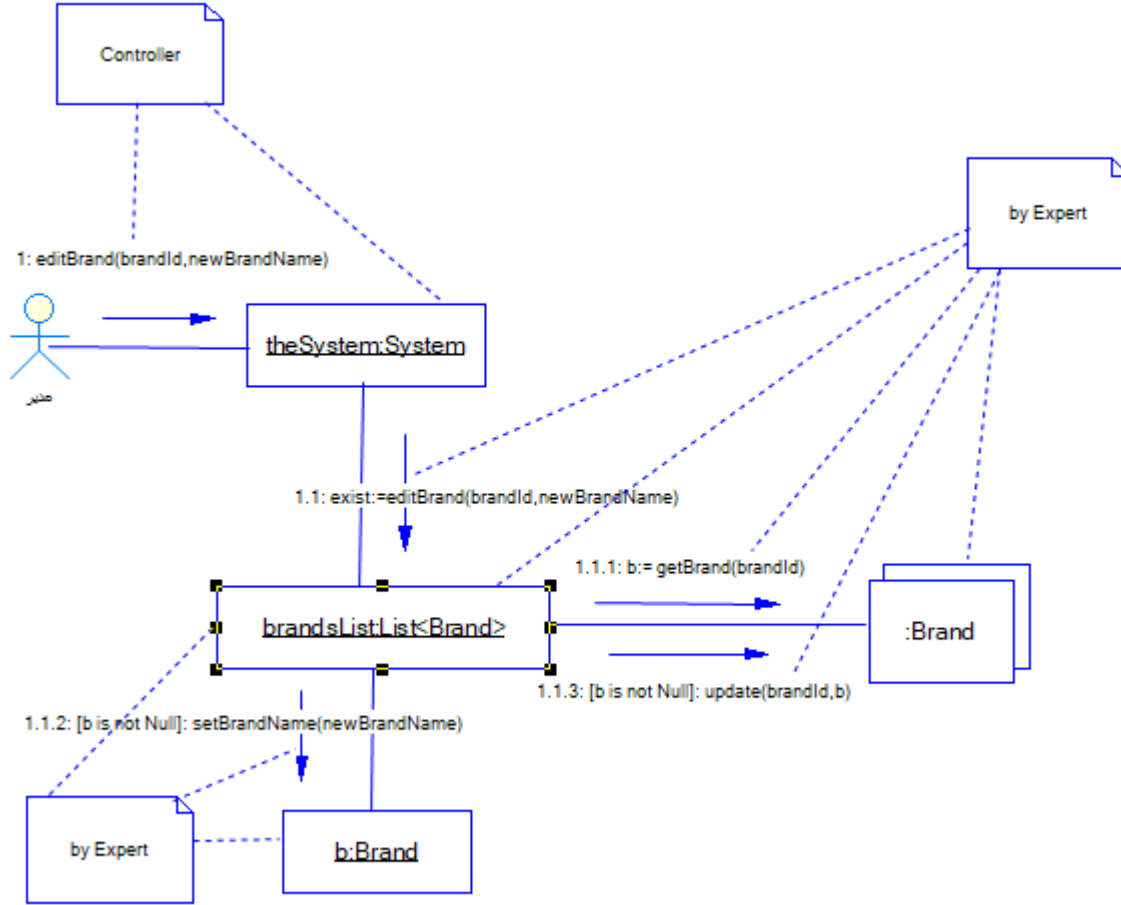
يبيّن الشكل (7) مخطط التعاون لحالة استخدام إضافة شركة أدوية.



الشكل 7: مخطط التعاون لحالة استخدام إضافة شركة أدوية.

- حالة استخدام تعديل شركة أدوية:

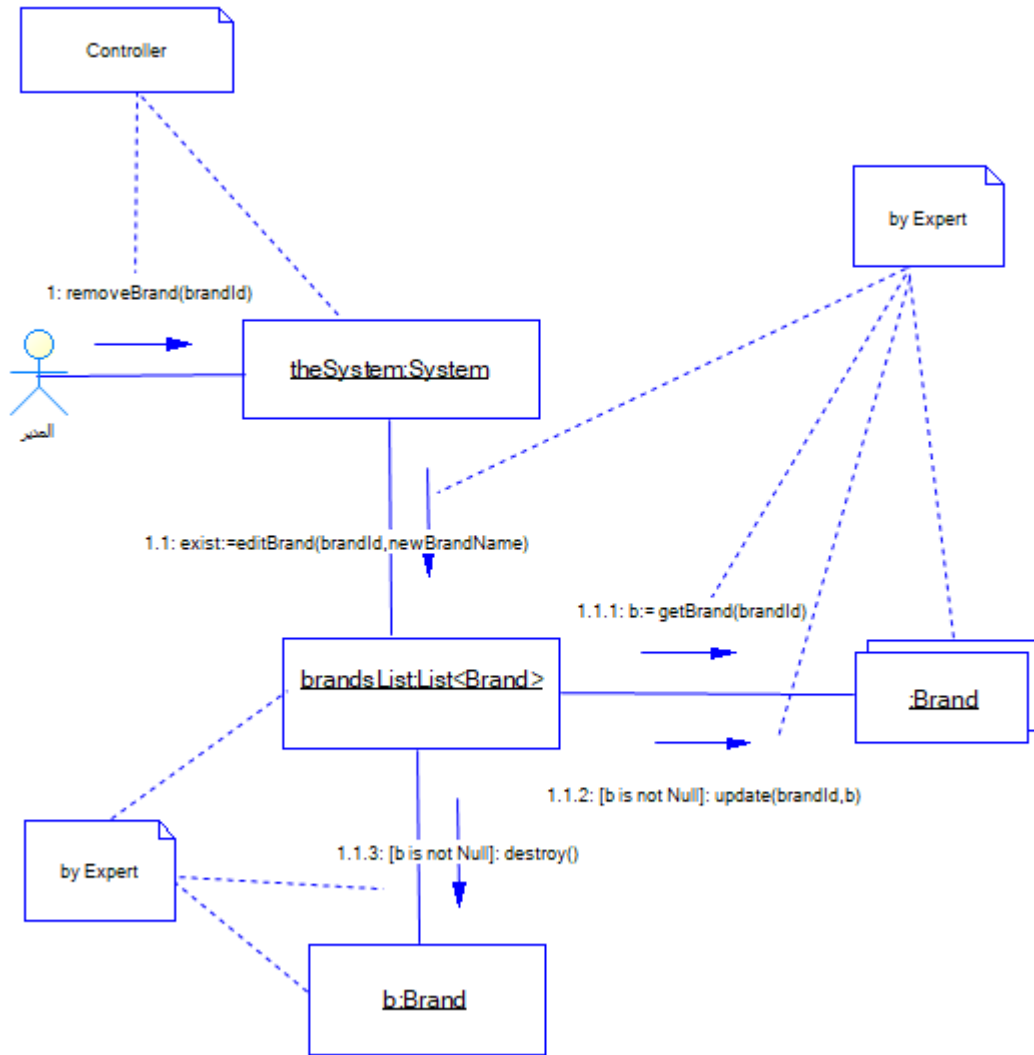
يبيّن الشكل (8) مخطط التعاون لحالة استخدام تعديل شركة أدوية.



الشكل 8: مخطط التعاون لحالة استخدام تعديل شركة أدوية.

- حالة استخدام حذف شركة أدوية:

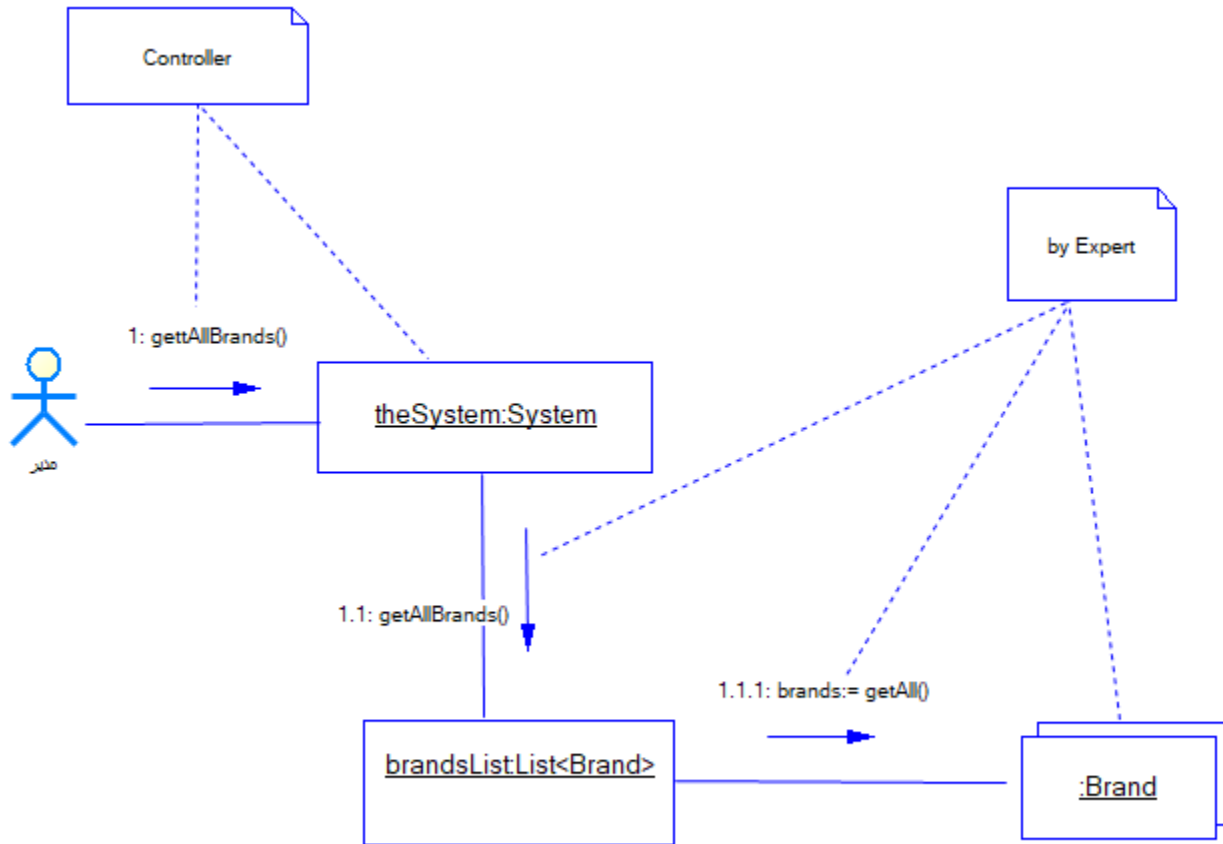
يبيّن الشكل (9) مخطط التعاون لحالة استخدام حذف شركة أدوية.



الشكل 9: مخطط التعاون لحالة استخدام حذف شركة أدوية.

- حالة استخدام استعراض شركات الأدوية:

يبيّن الشكل (10) مخطط التعاون لحالة استخدام استعراض شركات الأدوية.



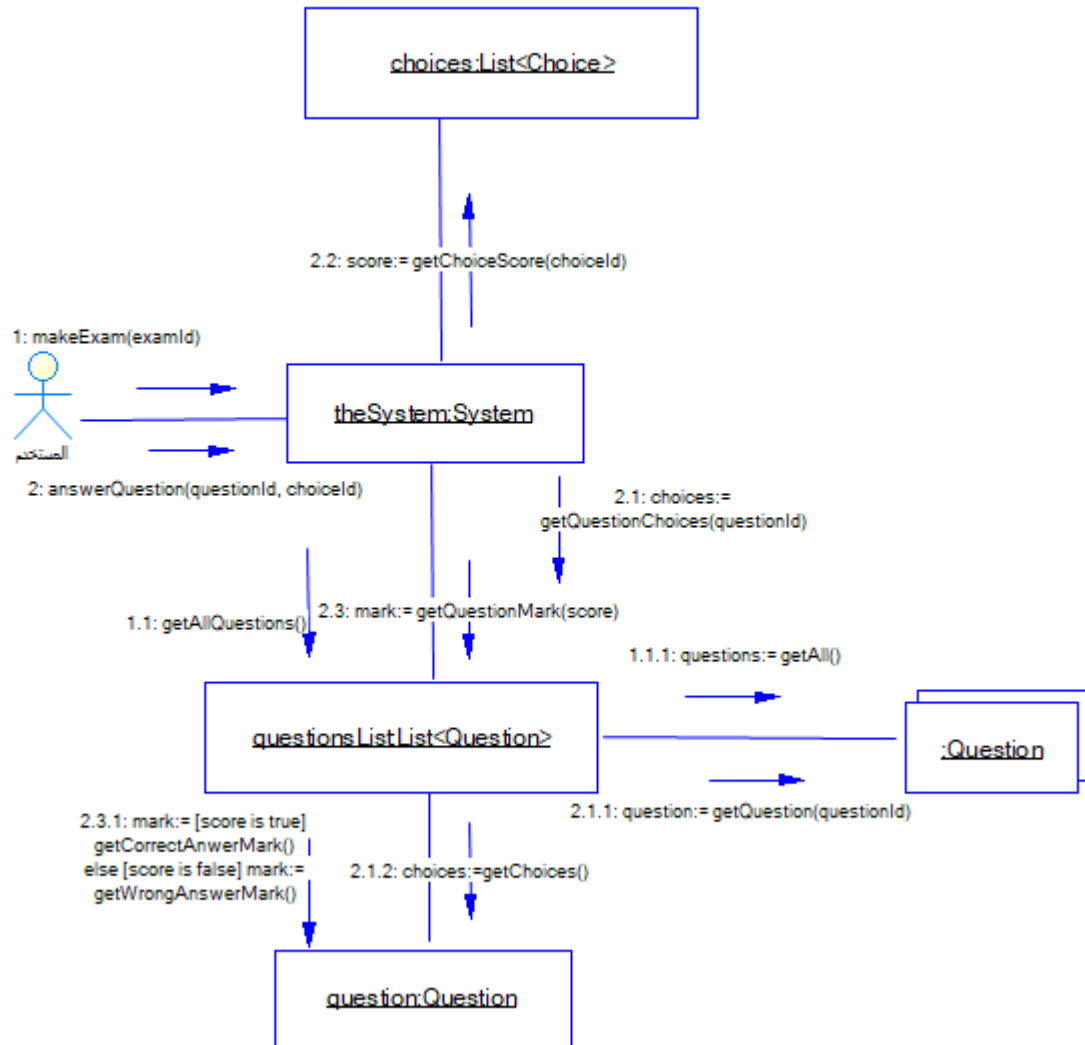
الشكل 10: مخطط التعاون لحالة استخدام استعراض شركات الأدوية.

2.2.3. تطبيق الهاتف الذكي

سيتم توضيح مخطط التعاون لحالة استخدام إجراء اختبار فقط، لأنه تمّ عرض مخطط التعاون لحالة استخدام استعراض محتوى ما.

- حالة استخدام إجراء اختبار:

يبيّن الشكل (11) مخطط التعاون لحالة استخدام إجراء اختبار.



الشكل 11: مخطط التعاون لحالة استخدام إجراء اختبار.

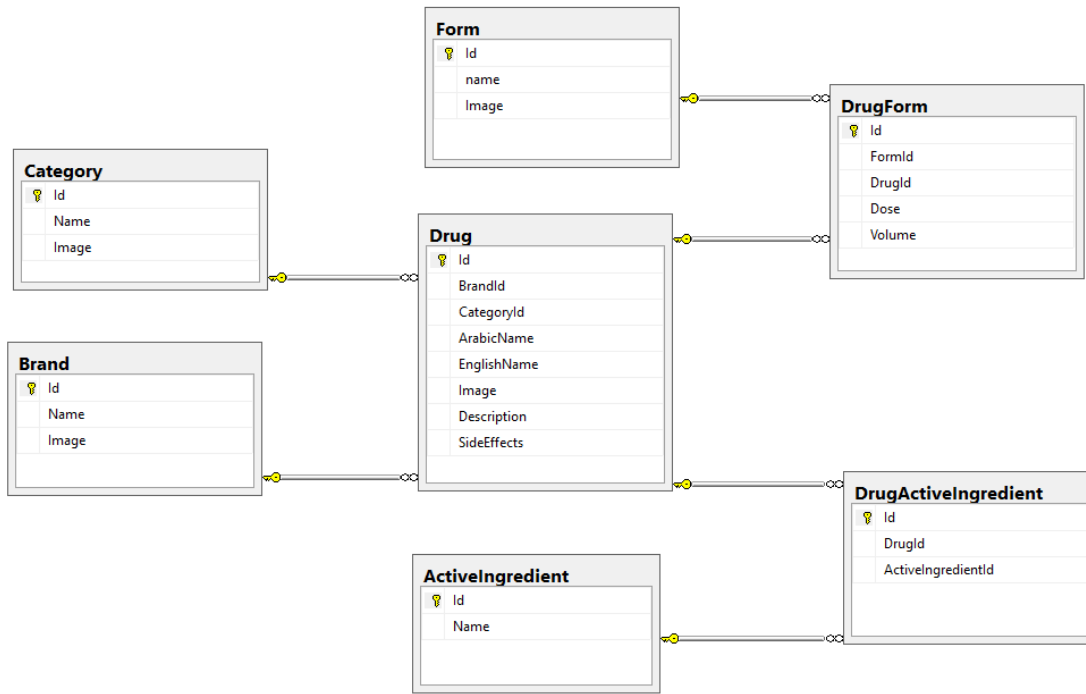
الفصل الرابع

تصميم النظام

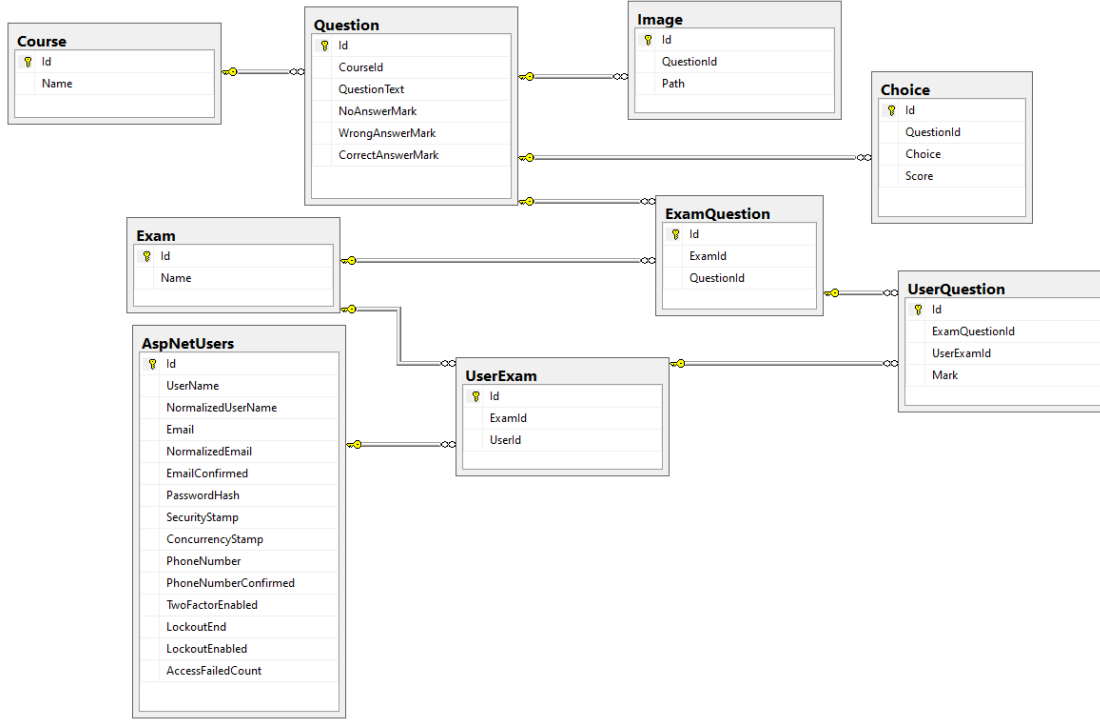
في هذا الفصل سيتم تصميم النظام عن طريق توضيح قاعدة المعطيات وتوضيح البنية المعمارية للنظام والأنماط التصميمية المستخدمة.

1.4. بنية قاعدة المعطيات Database

تتألف قاعدة المعطيات من قسمين رئيسيين من الجداول المترابطة، أولها قسم توصيف الأدوية وفتحاتها وأشكالها الدوائية والشركات المصنعة للأدوية، وثانيها قسم الاختبارات، يوضح الشكلين (14) و (15) مخطط قاعدة المعطيات لكل من الجزء الخاص بالأدوية والجزء الخاص بالاختبارات.



الشكل 12: تصميم قاعدة المعطيات للجزء الخاص بالأدوية.



الشكل 13: تصميم قاعدة المعطيات للجزء الخاص بالاختبارات.

حيث تحوي قاعدة المعطيات الجداول التالية:

1. **Category**: وهذا الجدول يعبر عن الفئة المرضية التي ينتمي لها دواء ما، ونحتّم بتخزين اسم الفئة وصورة تعبّر عنها.
2. **Brand**: يعبر عن الشركة المصنّعة للدواء ونحتّم بتخزين اسم الشركة وصورة تعبّر عن شعار الشركة.
3. **Form**: يعبر هذا الجدول عن الشكل الدوائي، ونحتّم بتخزين اسم الشكل وصورة تعبّر عنه.
4. **ActiveIngredient**: يعبر عن المادة الفعالة الموجودة في الدواء، ونحتّم بتخزين اسم المادة الفعالة فقط.
5. **Drug**: هو الجدول الخاص بالدواء، ونحتّم بتخزين اسم الدواء في كلّ من اللغتين العربية والإنكليزية، توصيف الدواء، الآثار الجانبية، وصورة للدواء. يحوي هذا الجدول مفتاحين خارجيين على جدولي **Category**، و **Brand** يشيران إلى كلّ من الفئة المرضية التي يتبع لها الدواء، والشركة المصنّعة له.
6. **DrugForm**: وهو الجدول الذي نحتّم فيه بتخزين معلومات الدواء المرتبط بشكل معين لأنّ الشكل الدوائي يرتبط بالكثير من الأدوية، والدواء له أشكال متعددة، أي أنّ العلاقة بين الجدولين **Drug** و **Form** هي many to

- many وبالتالي نحتاج لكسر هذه العلاقة، لذلك نجد مفتاح خارجي يشير إلى الدواء، ومفتاح خارجي يشير إلى الشكل الدوائي، ونهتمّ في هذا الجدول بتخزين مقدار الجرعة من هذا الدواء والحجم.
7. **DrugActiveIngredient**: في هذا الجدول نخزن معلومات الدواء المرتبط بمادة فعالة معينة، لأنّ المادة الفعالة قد تكون موجودة في عدّة أدوية، والدواء قد يحوي أكثر من مادة فعّالة، وبالتالي العلاقة بين الجدولين Drug وActiveIngredient هي علاقة many to many لذلك نحتاج لجدول كسر علاقة.
8. **Course**: وهو يُعبّر عن الاختصاصات الموجودة في التطبيق من أجل التوسّع لاحقاً، حيث يمكن إضافة اختصاصات أخرى غير الصيدلة، ونهتمّ بتخزين اسم ال Course.
9. **Exam**: يُعبّر عن الاختبارات التي يجب أن يجتازها المستخدم ونهتمّ بتخزين عنوان الاختبار والتاريخ.
10. **Question**: و هو يعبّر عن الأسئلة الموجودة في الاختبارات، وهو يحوي مفتاح خارجي على الجدول Course يشير إلى الاختصاص الذي ينتمي له السؤال. نهتم في هذا الجدول بتخزين نصّ السؤال، والعلامة في حال الإجابة كانت صحيحة، والعلامة في حال لا يوجد إجابة، والعلامة في حال كانت الإجابة خاطئة.
11. **Image**: يُعبّر هذا الجدول عن صور الوصفات الطبية التي من الممكن أن تكون موجودة في الاختبار على شكل سؤال لذلك نجد مفتاح خارجي على الجدول Question يشير إلى السؤال الذي يحوي الصورة. ونهتم بتخزين اسم الصورة فقط.
12. **ExamQuestion**: نهتمّ في هذا الجدول بتخزين معلومات الفحص المرتبط بسؤال معيّن وذلك لأنّ الفحص قد يحوي عدة أسئلة والسؤال ممكن أن يوجد في أكثر من فحص أي أنّ العلاقة بين الجدولين Exam، وQuestion هي علاقة many to many وبالتالي نحتاج لجدول كسر علاقة بينهما لذلك نجد مفتاح خارجي على الجدول Exam ومفتاح خارجي على الجدول Question.
13. **Choice**: يُعبّر هذا الجدول عن الخيارات المتعلقة بسؤال معيّن، فنجد مفتاح خارجي على الجدول Question يشير إلى السؤال الذي يمتلك هذه الخيارات، ونهتمّ في هذا الجدول بتخزين نصّ الخيار، وتحديد فيما إذا كان هذا الخيار صحيح أم لا وهذا ما تُشير إليه الوصفة Score وهي من النمط Boolean.
14. **AspNetUsers**: نهتمّ في هذا الجدول بتخزين معلومات عن مستخدمين تطبيق الهاتف وتتضمن هذه المعلومات اسم المستخدم وكلمة المرور وغيرها من معلومات الحساب.
15. **AspNetRoles**: يعبّر هذا الجدول عن الأدوار التي يمكن منحها للمستخدمين.

16. **AspNetUserRoles**: يعرّف هذا الجدول عن مجموعة الأدوار الممنوحة لمستخدم معين، حيث أنّ الدور ممكن أن يُمنَح لعدّة مستخدمين، والمستخدم ممكن أن يملك أكثر من دور وبالتالي العلاقة بين الجدولين **AspNetUsers**، و**AspNetRoles** هي علاقة many to many وبالتالي تحتاج لجدول كسر علاقة لذلك نجد أنه في هذا الجدول يوجد مفتاح خارجي على **AspNetUsers**، ومفتاح خارجي على **AspNetRoles**.

17. **UserQuestion**: نهتم في هذا الجدول بتخزين معلومات الأسئلة التي أجاب عنها المستخدم.

2.4. البنية المعماريّة للنظام البرمجي المُعتمد Software Architecture

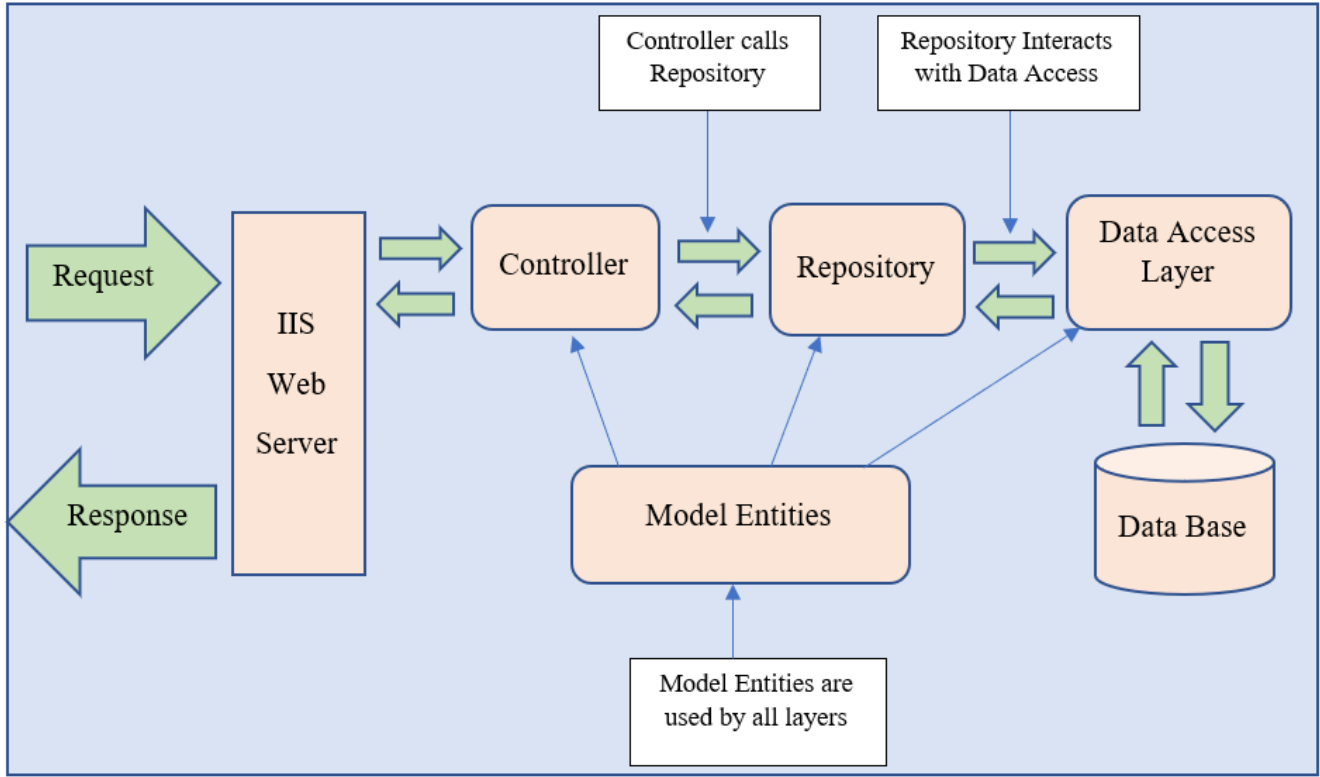
من المهم قبل البدء بتصميم برمجية معيّنة معرفة البنية المعماريّة التي سوف نبني وفقها النظام، وعند وضع هذه البنية يجب أن نأخذ بعين الاعتبار المشاكل التصميمية التي من الممكن أن تواجهها. تُعتبر الأنماط التصميمية حلاً لهذه المشاكل. ومتى ما استخدمنا نمطاً تصميمياً معيّناً أثناء بدء التنفيذ نجد أننا نستخدم أنماطاً تصميمية أخرى تلقائياً، ممّا يساهم في الوقاية من بعض المشاكل التصميمية.

1.2.4 Repository Pattern

هو نمط تصميمي يُستخدم بشكل كبير في (.Net) واعتمدنا بشكل أساسي في تقسيم طبقات قسم ال Back-End على هذا النمط، يعتمد على فكرة فصل الجزء من الرّماز الذي يتعامل مع قاعدة المعطيات عن الجزء الذي يتعامل معه المستخدم (Controllers). لتحقيق ذلك يعتمد هذا النمط على فصل النظام إلى طبقات بحيث يكون هناك طبقة واحدة فقط تتعامل مع قاعدة المعطيات تسمى Data Access Layer وهي تحوي Entity Framework، وبحيث يتم فصل هذه الطبقة عن الطبقة التي تحوي ال Controllers باستخدام طبقة أخرى تحوي ال Business Logic. وهذا يساهم في جعل الرّماز منظم أكثر حيث يحقق مبدأ Separation of Concerns، كما يحدّ من تكرار الاستعلامات فمثلاً في حال وجود أكثر من Controller كلّ منها يحوي Action يطلب نفس الاستعلام، يوفّر هذا النمط التصميمي إمكانية وضع ال logic في مكان وحيد ويمكن استخدامه في كل Controller للحدّ من تكرار الاستعلام نفسه.

من الميزات المهمّة الأخرى التي يقدّمها هذا النمط هي أنه يفصل اعتمادية طبقة ال Controller عن الطبقة التي تتعامل مع قاعدة المعطيات باستخدام طبقة بينهما وبالتالي في حال تمّ مستقبلاً تغيير قاعدة المعطيات من بيئة SQL إلى بيئة أخرى مثل MySQL، أو Oracle، لن نواجه مشاكل تصميمية ولن نحتاج لإعادة التصميم من جديد أو تعديل جميع ال Actions الموجودة في ال Controllers كما هو الحال في حال عدم استخدام هذا النمط وجعل جميع ال Actions تتعامل مباشرة مع ال ApplicationDbContext، وإنما فقط نعدّل على طبقة ال Data Access Layer.

يوضح الشكل (16) كيفية عمل النمط التصميمي Repository.



الشكل 14: رسم توضيحي يعبر عن مبدأ عمل Repository Pattern

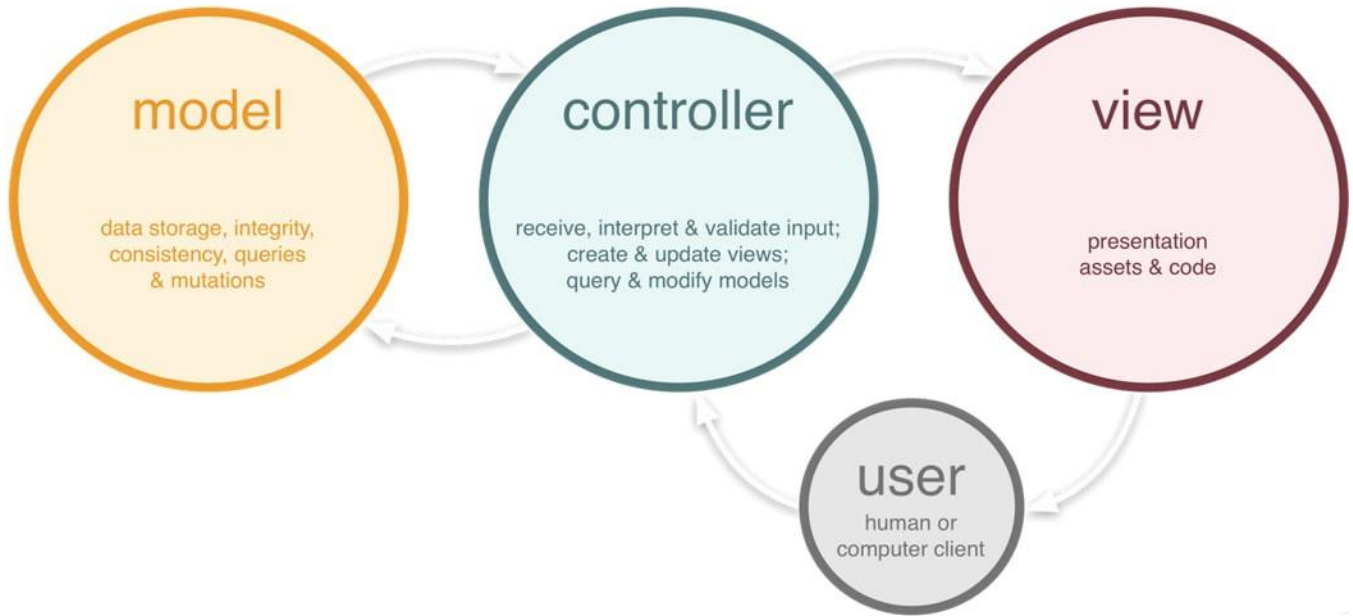
2.2.4 MVC Pattern

يقوم هذا النمط بفصل طبقة العرض عن الطبقة التي تحوي ال Business Logic وكذلك عن طبقة العرض.

ويتألف هذا النمط من 3 أقسام:

- **Model**: يقدم هذا القسم توضيح عن بنية قاعدة المعطيات فيحوي جداول قاعدة المعطيات ولكن بعد أن تم تحويلها إلى صفوف بواسطة Object Relational Mapping.
- **View**: هذا القسم المسؤول عن إظهار الواجهات للمستخدم، وقد يكون الإظهار على شكل صفحات HTML لطلبات الويب، أو على شكل أغراض JSON لطلبات API.

- **Controller**: يقوم بأخذ الطلبات القادمة من ال **View** ليقوم بمعالجتها عن طريق ال **Business Logic** ويقدمها لل **Model** الذي بدوره يقوم بالتعامل مع قاعدة المعطيات، ليعيد النتيجة لل **Controller** الذي بدوره يعيد النتيجة لل **View** ليتم إظهار الواجهات للمستخدم.



الشكل 15 : شكل توضيحي لمبدأ عمل النمط MVC.

بعد أن قمنا بتقسيم النظام إلى 3 طبقات:

- طبقة تتعامل مع قاعدة المعطيات.
- طبقة وسيطة تحوي ال **Repositories**.
- طبقة **Web API** تحوي **Controllers**.

أضفنا طبقة أخرى تعتمد على طبقة ال **Web API** من أجل تصميم واجهات تطبيق الويب الخاص بالمدير. وهذه الطبقة تعتمد النمط **MVC** حيث أنها تحوي **Views** تحوي تصميم الواجهات التي تعمل اعتماداً على ال **Controllers** التي تقوم بدورها بإرسال طلبات **HTTP** إلى طبقة ال **Web APIs**.

الفصل الخامس

التنفيذ والاختبارات

في هذا الفصل سيتمّ توضيح بيئة العمل المستخدمة في تنجيز النظام، وسيتم عرض أمثلة من التطبيق، وفي النهاية سيتم عرض الخلاصة والآفاق المستقبلية.

1.5. التنفيذ

يمكن تقسيم تنفيذ المشروع إلى عدة أجزاء:

1. الجزء الخاص بطرف المخدم Back-End

- تنجز خدمة ويب Web API لتخديم طلبات المستخدمين عن طريق تطبيق الهاتف الذكي، بحيث يتخاطب التطبيق معها لتقوم بدورها بالتخاطب مع قاعدة المعطيات بهدف تخديم تلك الطلبات.
- تنجز منطق عمل تطبيق الهاتف Business logic الذي يتخاطب مع خدمة ويب Web API المذكورة لتأمين خدمات المستخدم.

2. الجزء الخاص بطرف الزبون Front-End

- تصميم واجهات تطبيق الهاتف الذكي التي يتعامل معها المستخدم Views.
- تصميم واجهات تطبيق ويب المدير التي تمكنه من إدارة المحتوى وإدارة المستخدمين.

1.1.5. بيئة العمل

تم بناء قسم ال Back-End باستخدام ASP.Net Core 6.0، حيث تمّ تنجز خدمة ويب باستخدام Web API وتمّ تنجز تطبيق الويب باستخدام MVC كإطار عمل Framework بلغة C#، وتمّ بناء قاعدة المعطيات باستخدام Microsoft SQL Server 2014.

➤ ASP.Net Core MVC

هو إطار عمل لتطوير تطبيقات الويب تم تطويره بواسطة Microsoft كجزء من ASP.NET Core، وهو مصمم لإنشاء تطبيقات ويب قابلة للتطوير وعالية الأداء، يحوي مميزات ASP .Net، مثل التحقق من عضوية المستخدم وأدواره، بالإضافة لإمكانية تعديل هذه الأدوار. يتبع النمط التصميمي MVC الذي يعتمد على فصل المكونات، مما يتيح اختبار كل جزء من هذه المكونات بشكل منفصل، بالإضافة لإمكانية استخدام محرك العرض Razor الذي يسمح للمطورين بكتابة رماز باستخدام C# مع HTML.

➤ ASP .Net Web API

هو إطار عمل تمّ تطويره بواسطة Microsoft كجزء من ASP .Net. يسمح للمطوّرين إنشاء وعرض واجهات برمجة تطبيقات الويب التي يمكن أن تتعامل معها تطبيقات الزبون المختلفة، مثل تطبيقات الويب وتطبيقات الأجهزة المحمولة وتطبيقات سطح المكتب وغيرها. تمّ تصميم واجهة برمجة تطبيقات الويب لتسهيل إنشاء واجهات برمجة تطبيقات RESTful، مما يتيح الاتصال الفعال بين الزبون والمخدّم عبر بروتوكول HTTP مما يجعل من السهل الاستفادة من طرق HTTP مثل GET، POST، PUT، وDELETE للتفاعل مع المعطيات. وهي تدعم مبادئ بنية RESTful، والتي تؤكد على استخدام عناوين URL وطرق HTTP للتعامل مع المعطيات. تستخدم واجهة برمجة تطبيقات الويب ASP.NET آلية توجيه لتعيين عناوين URL إلى إجراءات تحكّم محدّدة تماماً مثل ASP.NET MVC، يمكن لواجهة برمجة تطبيقات الويب ربط بيانات JSON أو XML، تتيح هذه المرونة للزبائن تلقي المعطيات بالتنسيق الذي يناسبهم أكثر، كما يمكن أن ترجع إجراءات المتحكمات نتائج إجراءات محدّدة (مثل Ok، BadRequest، وNotFound وغيرها) التي تتوافق مع رموز استجابة HTTP المختلفة. هذا ييسّر عملية صياغة الاستجابات المناسبة لطلبات الزبائن.

➤ ASP Identity Membership System

هو نظام إدارة عضوية المستخدمين تمّ إنشاؤه من قبل Microsoft كجزء من ASP .Net، وهو مصمّم للتعامل مع التحقق من المصادقة، والصلاحيات ومهام أخرى متعلّقة بالعضوية في تطبيقات الويب، حيث يوفر ميزة إمكانية إدارة حسابات المستخدمين، الأدوار، والصلاحيات، ويتيح للمطوّر بناء نظام واحد لجميع أنواع تسجيل الدخول سواء كان تسجيل الدخول عن طريق البريد الإلكتروني وكلمة المرور أو إن كان عن طريق Google أو طرف ثالث مثل Facebook وغيرها.

➤ Flutter

هي مجموعة أدوات تطوير برامج واجهة المستخدم مفتوحة المصدر تمّ إنشاؤها بواسطة Google. يتم استخدامه لبناء تطبيقات الموبايل، تطبيقات الويب، وتطبيقات سطح المكتب من قاعدة معطيات واحدة. يسمح Flutter للمطوّرين بإنشاء تطبيقات جذابة بصرياً، سريعة، وعالية الأداء يمكن تشغيلها على منصات متعددة باستخدام لغة برمجة واحدة، حيث يمكن من خلال Flutter تصميم رماز واحد يعمل على منصات متعددة، بما في ذلك iOS، Android، الويب، و سطح المكتب وهذا يقلّل بشكل كبير من وقت التطوير والجهد مقارنةً بإنشاء تطبيقات منفصلة

لكل نظام أساسي. يستخدم Flutter لغة برمجة Dart، التي طورها Google. تشتهر Dart بتركيبها الحديث، تحسينات الأداء، وميزات أخرى مثل إعادة التحميل السريع، والتي تتيح للمطورين رؤية التغييرات في التعليمات البرمجية الخاصة بهم على الفور تقريباً دون إعادة تشغيل التطبيق بأكمله. يوفر Flutter إطار عمل مرّن لواجهة المستخدم يسمح للمطور بإنشاء تصميمات مخصصة وجذابة بصرياً. وهو يدعم الرسوم المتحركة، والانتقالات، والمؤثرات المرئية مما يجعل واجهة المستخدم أكثر جاذبية.

➤ Microsoft SQL Server 2014

هو نظام لإدارة نظام قواعد المعطيات RDBMS أطلقته شركة Microsoft كمخدّم لقواعد المعطيات.

2.1.5. أدوات التنفيذ

تم استخدام الأدوات التالية:

- Microsoft Visual Studio 2022: من أجل تطبيق الويب الخاص بالمدير والجزء الخاص بطرف المخدم .Back End
- Android Studio 2022.2.1: من أجل تطبيق الهاتف الذكي الخاص بالمستخدم.
- Microsoft SQL Server 2014: من أجل إدارة قاعدة المعطيات.

3.1.5. مكونات النظام التفصيلية

سنتحدث في هذه الفقرة عن البنية البرمجية للنظام حيث سيتم تفصيل البنية لثلاثة أقسام (بنية مخدم ويب، تطبيق الويب، تطبيق الهاتف المحمول) مع توضيح لبعض الصفوف التي قمنا باستخدامها.

1.3.1.5. بنية مخدم الويب

يمكن تقسيم بنية المخدم إلى الطبقات التالية:

1. طبقة DBContext.
2. طبقة BuisnessLogic.
3. طبقة Web API.
4. طبقة Web App.

ستحدّث عن كلّ منها بالتفصيل.

1. طبقة DBContext

هي الطبقة المخصّصة للتعامل مع قاعدة المعطيات التي تمّ تصميمها باستخدام Microsoft SQL Server 2014. وتمّ تغليفها بـ PharmacyDB الذي يتكوّن من Models وهي صفوف تعبّر عن جداول قاعدة المعطيات تمّ إنشاؤها تلقائياً باستخدام تعليمة Scaffold، ليتمّ استخدام هذه الصفوف من قبل بقية الطبقات بدلاً من التعامل مباشرةً مع جداول قاعدة المعطيات. وتمّ استخدام النمط التصميمي Repository Pattern في هذه الطبقة من أجل سهولة التعديل لاحقاً حيث ينحصر مكان التعديل على جزء واحد فقط من الرّماز، ومن أجل التعامل مع مختلف أنظمة إدارة قواعد المعطيات وليس فقط SQL حيث في حال تمّ استبدال نظام إدارة قواعد المعطيات يكفي تعديل طريقة استخدام بعض التوابع أو إضافة توابع جديدة، بالإضافة لذلك فهذا النمط يخفّف من تكرار مجموعة من التوابع التي تُستخدم عند التعامل مع كافة الجداول في قاعدة المعطيات بدلاً من تكرارها سيتم وضعها في جزء واحد فقط، سنذكر جزء من هذه التوابع والتي احتجناها خلال هذا المشروع:

- **GetAll**: خرج هذا التابع هو لائحة من نمط الـ Object الخاص بكل جدول.
- **GetById**: دخل هذا التابع هو عدد صحيح int يمثّل الـ id، وخرجه هو غرض من أحد الجداول الذي توافّق قيمة id له قيمة id المدخلة.
- **Create**: دخل هذا التابع Object من نمط الصف الخاص بأحد الجداول المراد الإضافة عليها ويقوم التابع بإضافة هذا الـ Object على شكل سطر إلى الجدول.
- **Update**: دخل هذا التابع Object من نمط الصف الخاص بالجدول المراد التعديل عليه ويقوم التابع بتعديل السطر الموافق لهذا الـ Object في الجدول.
- **Delete**: دخل هذا التابع Object من نمط الصف الخاص بالجدول المراد حذف السطر الموافق لهذا الـ Object منه ويقوم التابع بحذف ذلك السطر.

ولتنجيز هذا النمط تمّ إنشاء مجلد Interfaces يحوي Interface لكلّ جدول من جداول قاعدة المعطيات IRepository يخصّص للتوابع الخاصّة بهذا الجدول وجعلنا كل منها ابن لـ IGenericRepository تحوي هذه التوابع المذكورة، كما يحوي المجلد على IUnitOfWork من نمط Interface وهي تقوم بتغليف جميع الـ

Repositories لتتمكن من خلالها طبقة ال Business Logic من التخابط مع طبقة DBContext وبالإضافة لذلك فهي تحوي تابعين هما :

- **SaveChanges**: تابع يعيد قيمة من نط bool تدلّ على نجاح أو فشل عملية إجراء التعديلات على الجدول وحفظ التغييرات، وهو يُستخدم بعد كل عملية إضافة أو تعديل أو حذف.
- **Dispose**: تابع يُستخدم لإلغاء حفظ التغيرات التي حصلت على جدول معيّن في حال ظهر خطأ قبل انتهاء عملية ما.

تحتوي هذه الطبقة أيضاً على مجلد Models يغلف جداول قاعدة المعطيات بصفوف Classes يتم التعامل معها في حال الرغبة بإجراء تعديلات على الجداول.

2. طبقة Business Logic

تفصل هذه الطبقة بين طبقة العرض وطبقة ال DBContext تحوي مجموعة من الصفوف مُقسّمة ضمن مجلدات حسب وظيفة كل منها، وهذه المجلدات هي:

- **مجلد Repositories**: يحوي صف خاص بكل جدول Repository من جداول قاعدة المعطيات وهي أبناء لصف GenericRepository يحوي تنجيز لتوابع IGenericRepository. كما يحوي صف UnitOfWork الذي يحوي تنجيز لتوابع IUnitOfWork.
- **مجلد Requests**: يحوي صفوف متعلّقة بطلبات HTTP. تسهّل عملية استدعاء Action في مشروع Web API من Action في مشروع PharmacyAdminWebApp حيث أنّ بعض طلبات ال HTTP والتي تكون Post أو Put ترفض في أن يُمرّر فيها متحولين، وإنما يجب تمرير غرض Json في ال Body الخاص بها لذلك احتجنا هذه الصفوف لتساعدنا في بعض ال Actions، فيكون محتوى كل صفّ منها الواصفات الخاصة بصفوف ال Models التي نحتاج تمريرها فقط وليس جميع الواصفات.
- **مجلد ViewModels**: يحتوي مجموعة من الصفوف تقابل الصفوف الموجودة في Models تُستخدم لتخزين واسترجاع الأغراض من طبقة العرض، وهذا يفيد في حال كنا لا نحتاج في طبقة العرض جميع واصفات الصف الموجود في Models.

3. طبقة Web API

هذه الطبقة عبارة عن REST Web Services، مهمتها الإجابة على الطلبات المرسلة من تطبيق الموبايل الموجود على الهاتف الذكي وبالتالي لا تحوي Views لذلك تم إنشاء مشروع WebApi هو PharmacyWeb يحوي مجلد Controllers الذي بدوره يحوي على متحكمات Controllers الخاصة ببعض الجداول الموجودة في قاعدة المعطيات مهمتها استخدام الصفوف الموجودة في Models لتلبية طلبات المستخدم عن طريق إرسال إجابة على شكل غرض Json وهذه المتحكمات هي:

- **DrugsController**: مسؤول عن تلبية الاستعلامات التي تخص الأدوية.
- **CategoriesController**: مسؤول عن تلبية الاستعلامات التي تخص الفئات المرضية.
- **BrandsController**: مسؤول عن الاستعلامات التي تخص شركات الأدوية.
- **FormsController**: مسؤول عن تلبية الاستعلامات التي تخص الأشكال الدوائية.
- **ActiveIngredientsController**: مسؤول عن تلبية الاستعلامات التي تخص المواد الفعالة.
- **ExamsController**: مسؤول عن تلبية الاستعلامات التي تخص الاختبارات.
- **QuestionsController**: مسؤول عن تلبية الاستعلامات التي تخص الأسئلة.
- **ImagesController**: مسؤول عن تلبية الاستعلامات التي تخص صور الوصفات الطبية التي من المحتمل وجودها في الأسئلة.
- **ChoicesController**: مسؤول عن تلبية الاستعلامات التي تخص الخيارات الخاصة بكل سؤال.

4. طبقة WebApp

هي عبارة عن طبقة تطبيق ويب تم إنشاؤه باستخدام MVC واسمه PharmacyAdminWebApp. مهمة هذه الطبقة استلام طلبات مدير النظام وإرسالها إلى طبقة WebApp لتقوم بدورها بمعالجة الطلبات بالاستعانة بطبقة Business Logic التي تتخاطب مع طبقة DbContext من أجل تلبية هذه الطلبات وتخدمها. تحوي هذه الطبقة العديد من المجلدات:

- مجلدات تحوي ملفات CSS وJavaScript.
- مجلد Controllers: يحوي نفس ال Controllers الموجودة في طبقة Web API وتقوم هذه المتحكمات بإرسال طلبات HTTP إلى المتحكمات الموجودة في طبقة WebAPI لتليتها.

- مجلد Views: الذي يحوي الواجهات الخاصة بكل متحكّم، وهي مكوّنة بشكل رئيسي من الملفات التالية:

- Index
- Create
- Details
- Modal وهذه الواجهة هي من أجل التعديل.
- مجلد Models: يحوي على صفوف ViewModels.

2.3.1.5. بنية تطبيق الموبايل

تم تنجيز هذه البنية باستخدام بيئة Android Studio باستخدام إطار عمل Flutter Framework، بلغة Dart لتصميم واجهة التطبيق. سيتم توضيح بعض المفاهيم المستخدمة في Flutter:

- جميع العناصر في Flutter هي عبارة عن widget نذكر منها: النص Text، العمود Column، السطر Row، الصورة Image، الزر Button، وكل Widget لها مجموعة معينة من الخصائص.
- يوجد نوعين من العناصر:

○ العناصر الثابتة Stateless Widgets:

هي العناصر التي حالتها غير قابلة للتغيير. أي أنّه بمجرد إنشاء عنصر Stateless Widget، لا يمكن تغيير خصائصه. تستخدم هذه العناصر بشكل عام لعرض عناصر واجهة المستخدم الثابتة. على سبيل المثال، نص، أو صورة.

○ العناصر المرنة Stateful Widgets:

يمكن أن يحتفظ هذا النوع من العناصر بحالة قابلة للتغيير مع مرور الوقت. يتم استخدامها لعناصر واجهة المستخدم التي تحتاج إلى تحديث استجابةً لتفاعلات المستخدم أو تحديث البيانات أو الأحداث الأخرى.

يحتوي مشروع ال Flutter على عدة مجلدات منها:

- مجلد Android: يحتوي ملفات التطبيق لنظام Android.
- مجلد iOS: يحتوي ملفات التطبيق لنظام iOS.

- مجلد lib: يحتوي على الملفات الخاصة بالرمز البرمجي.
- مجلد Pubspec.yaml: يحتوي على المكتبات المستخدمة ضمن التطبيق حيث نضع اسم المكتبة ورقم النسخة بجانبها، كما أنه يحوي مسارات إلى مجلد assets الذي يحوي الصور والملفات المستخدمة في التطبيق.

بالنسبة لمحتوى المجلد lib:

- مجلد Services: وهو يحوي Models خاصة بالواجهات التي سيتم عرضها كل Model عبارة عن class يحوي واصفات خاصة به سيتم عرضها ضمن الواجهة المخصصة له.
- مجلد Screens: وهو يحوي واجهات التطبيق، كل ملف فيها يحوي التصميم الخاص بواجهة معينة وهذه الملفات هي:
 - home.dart: وهي الصفحة الرئيسية في التطبيق ومن خلالها يمكن للمستخدم الذهاب لأي صفحة يريد.
 - activeIngredients.dart: هي الصفحة الخاصة بالمواد الفعالة.
 - activeIngredientDrugs.dart: تعرض هذه الواجهة الأدوية التي تحوي مادة فعالة معينة اختارها المستخدم من صفحة المواد الفعالة.
 - brands.dart: هي الصفحة الخاصة بشركات الأدوية.
 - brandDrugs.dart: تعرض هذه الصفحة الأدوية المصنعة من قبل شركة معينة اختارها المستخدم من خلال صفحة شركات الأدوية.
 - categories.dart: هي الصفحة التي تعرض الفئات المرضية.
 - categoryDrugs.dart: تعرض هذه الصفحة الأدوية التي تعالج فئة أمراض معينة تم اختيارها من قبل المستخدم من خلال صفحة الفئات المرضية.
 - drugsMainScreen: تعرض هذه الصفحة جميع الأدوية الموجودة.
 - forms.dart: تحوي هذه الصفحة جميع الأشكال الدوائية الموجودة.
 - formDrugs.dart: تعرض هذه الصفحة جميع الأدوية التي لها شكل معين اختاره المستخدم من خلال صفحة الأشكال الدوائية.

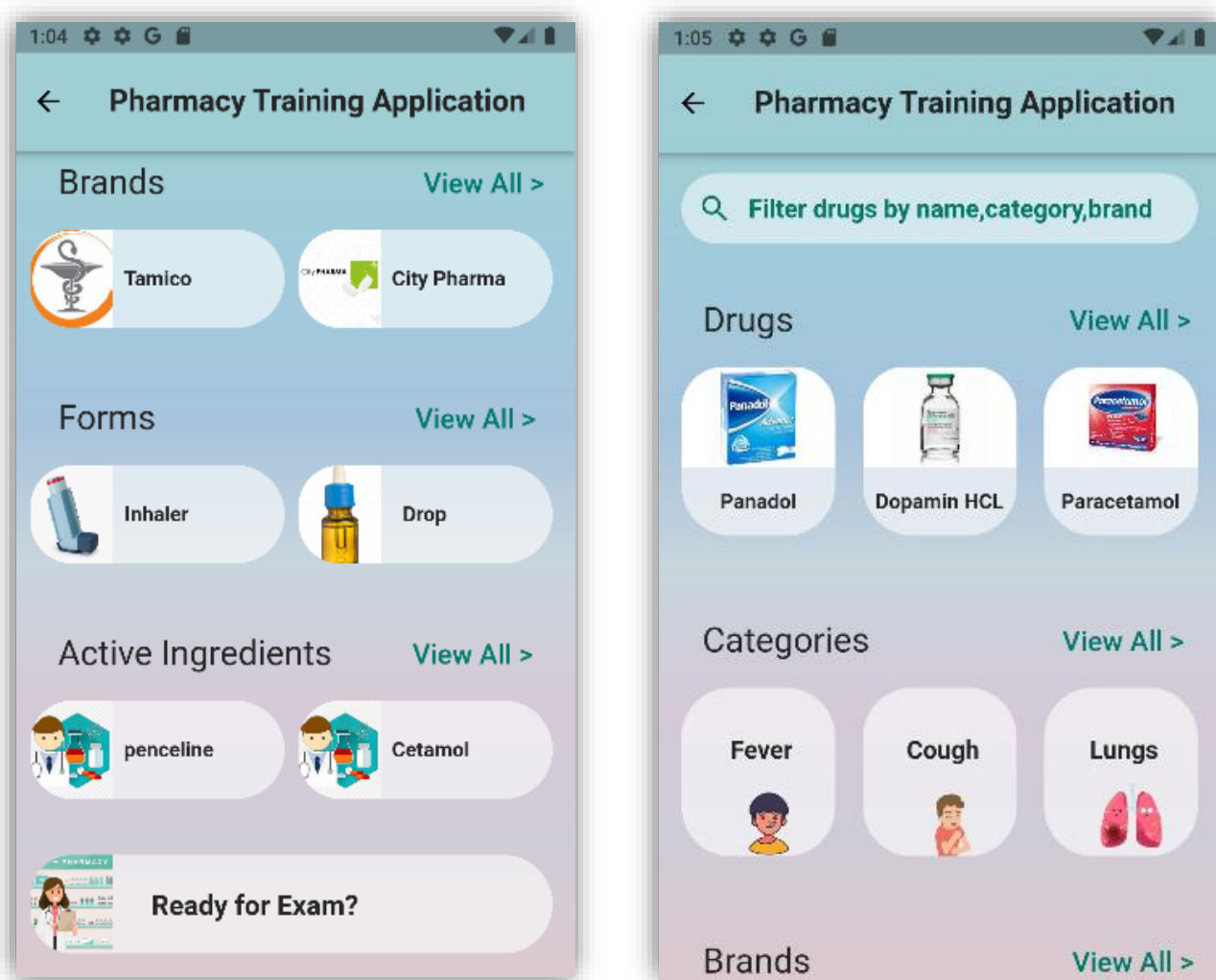
- `drug.dart`: هي الصفحة التي تعرض معلومات دواء معيّن اختاره المستخدم ليرى معلوماته ويمكن الانتقال لهذه الصفحة من أي صفحة تعرض الأدوية.
- `exams.dart`: تعرض هذه الصفحة جميع الاختبارات الموجودة.
- `examQuestion.dart`: عند اختيار اختبار ما من قبل المستخدم يتم عرض هذه الصفحة التي تحوي الأسئلة المتعلقة بهذا الاختبار.
- `loginPage.dart`: صفحة تسجيل الدخول.
- `registerPage.dart`: صفحة إنشاء حساب.

2.5. تصميم واجهات التطبيق

سنعرض فيما يلي بعض واجهات التطبيق.

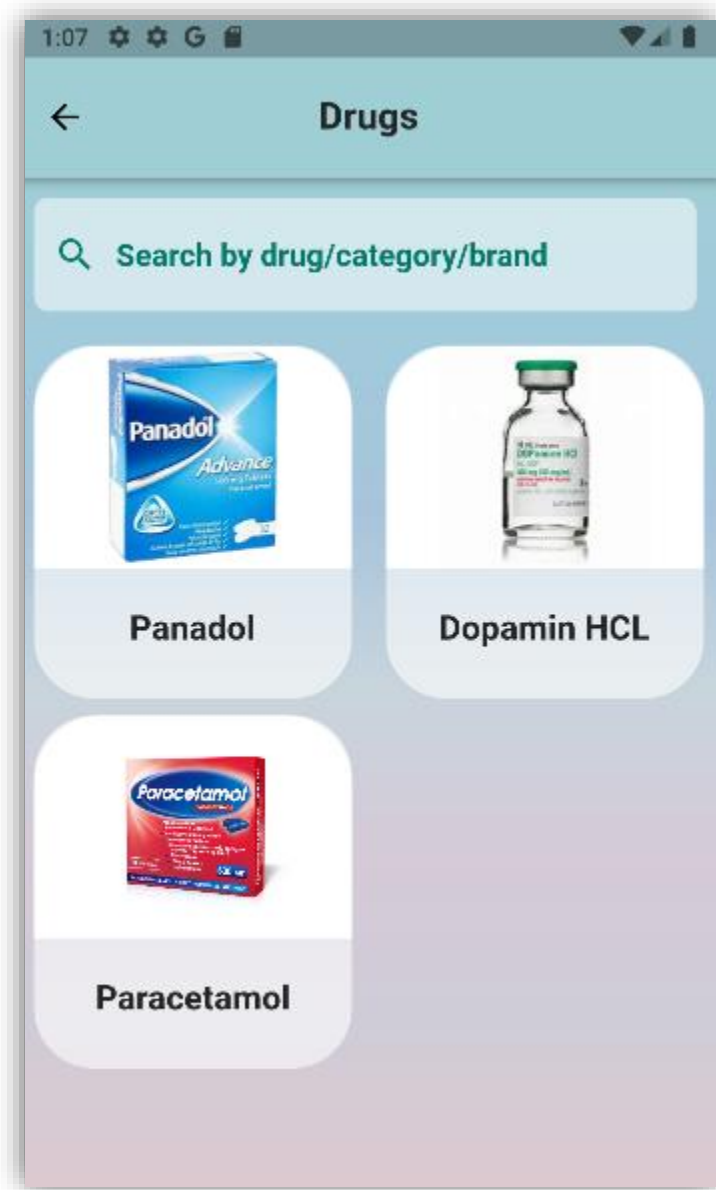
➤ الواجهة الرئيسية

تحتوي هذه الواجهة جميع متطلبات التطبيق وهي الأدوية، الفئات المرضية، شركات الأدوية، الأشكال الدوائية، المواد الفعالة، وزرّ للانتقال إلى صفحة الاختبارات، ويمكن من خلالها فلترة الأدوية حسب اسم الدواء واسم الشركة والفئة المرضية، فيما يلي تصميم للواجهة الرئيسية:



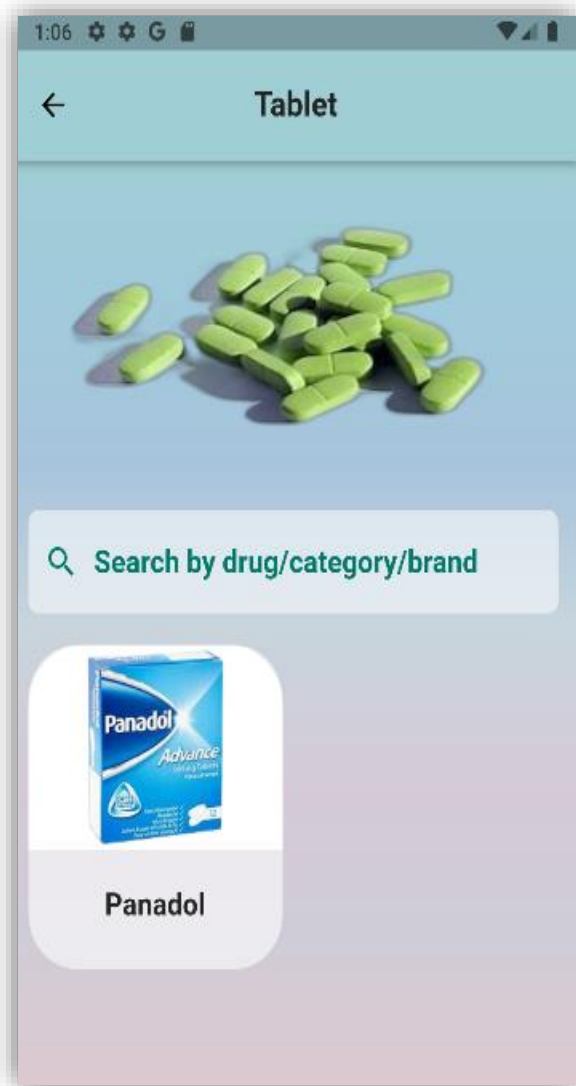
الشكل 16: الواجهة الرئيسية لتطبيق الموبايل.

➤ **واجهة الأدوية:** وهي تعرض كل الأدوية الموجودة ويمكن فلترتها حسب اسم الدواء، اسم الشركة المصنّعة أو الفئات المرضية، وعند اختيار أحد الأدوية تظهر واجهة تحوي معلومات تخص هذا الدواء، ويمثل الشكل التالي واجهة الأدوية:

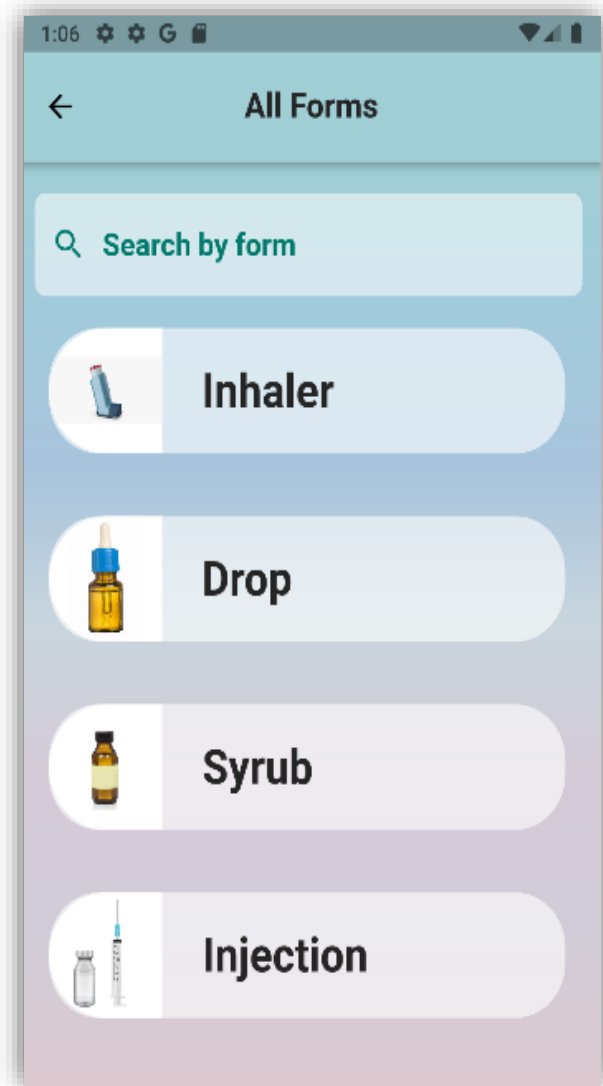


الشكل 17: واجهة الأدوية في تطبيق الموبايل.

➤ **واجهة الأشكال الدوائية:** وهي تعرض كل الأشكال الدوائية ويمكن عند اختيار أحدها عرض جميع الأدوية التي لها هذا الشكل الدوائي، كما يمكن فلترة الأشكال الدوائية بحسب اسمها، وتوضّح الأشكال التالية واجهة الأشكال الدوائية، وواجهة أحد هذه الأشكال:

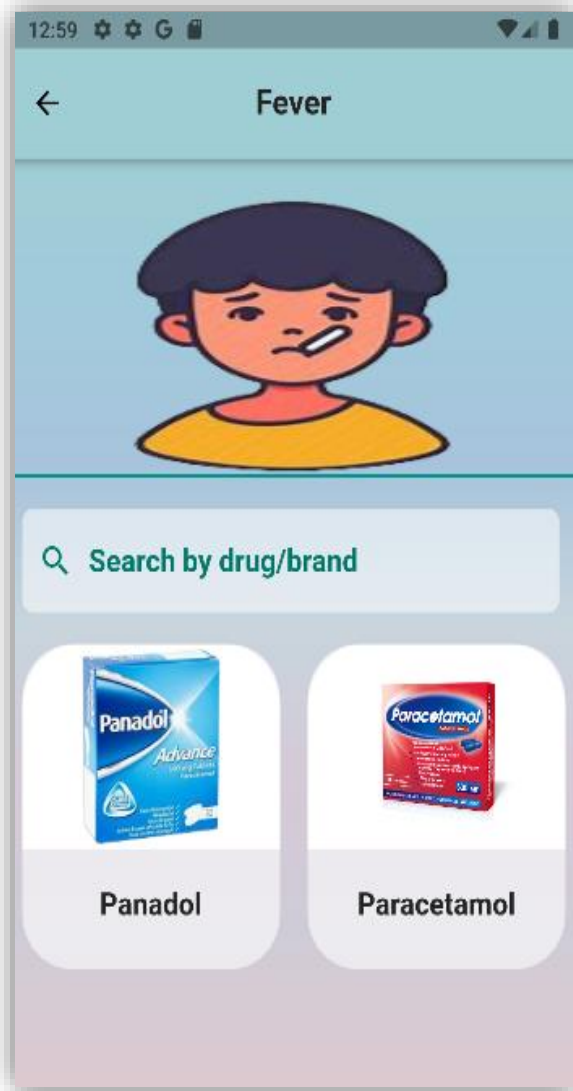


الشكل 19: واجهة الشكل الدوائي Tablet لتطبيق الموبايل.

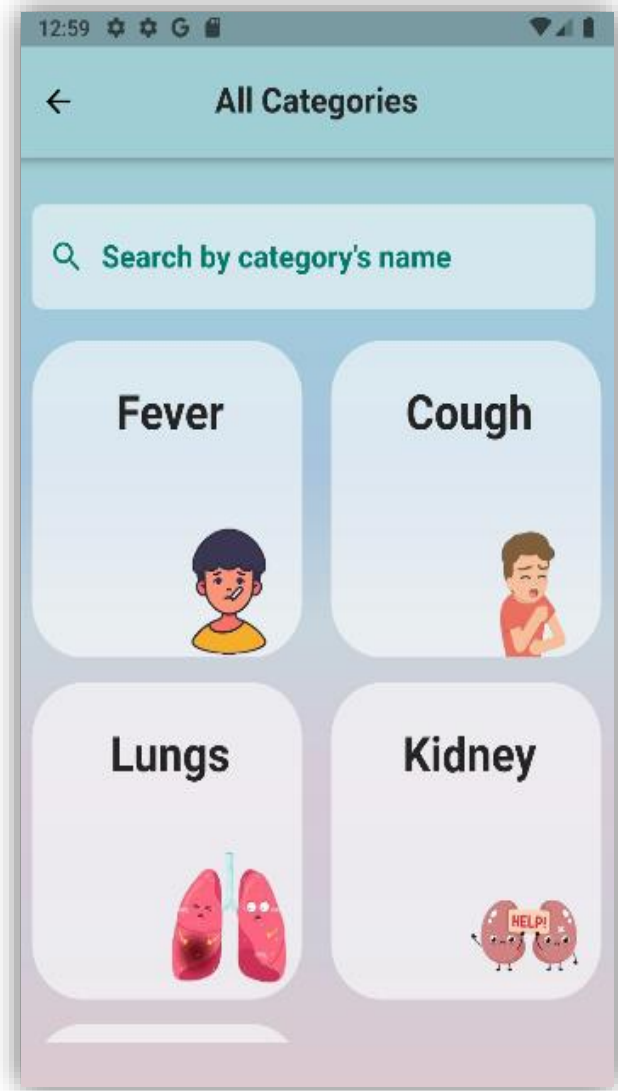


الشكل 18: واجهة الأشكال الدوائية لتطبيق الموبايل.

➤ واجهة الفئات المرضية: تعرض هذه الواجهة جميع الفئات المرضية ويمكن فلترتها حسب الاسم وعند اختيار أحدها تظهر الأدوية المتعلقة بهذه الفئة، وتمثل الأشكال التالية واجهة الفئات المرضية، وواجهة الأدوية المتعلقة بإحداها:

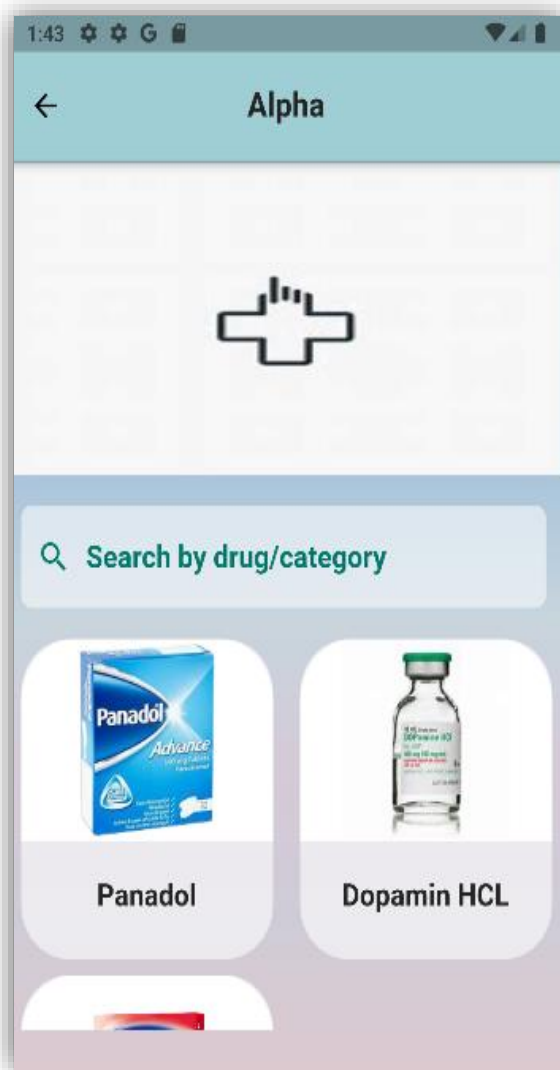


الشكل 21: واجهة أحد الفئات المرضية لتطبيق الموبايل.

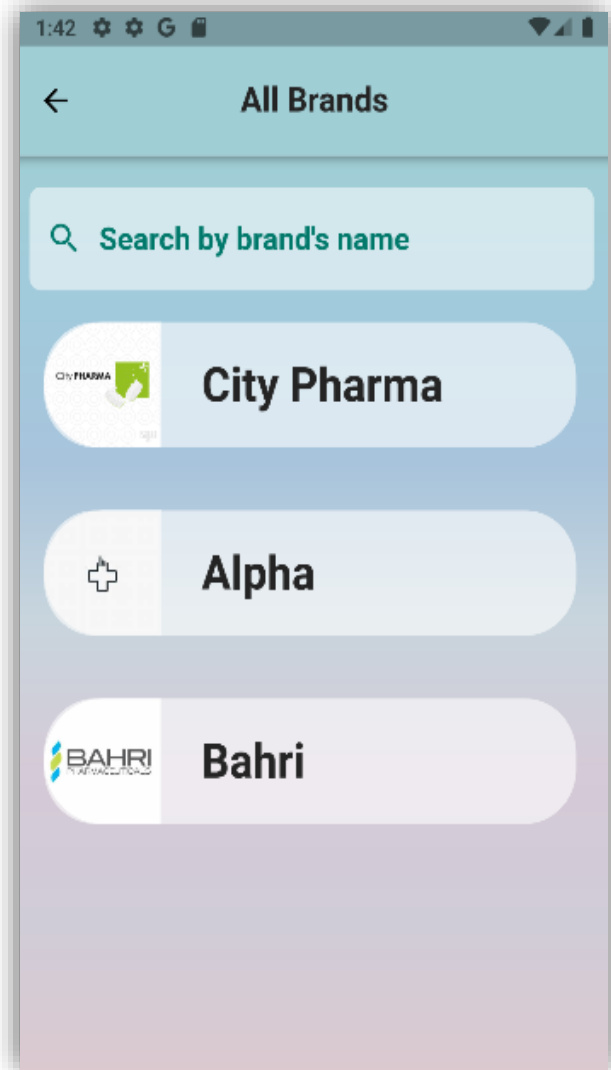


الشكل 20: واجهة الفئات المرضية لتطبيق الموبايل.

➤ **واجهة شركات الأدوية:** تعرض هذه الواجهة شركات الأدوية ويمكن فلترتها حسب اسم الشركة، وعند اختيار أحدها تظهر الأدوية المصنّعة من قبل شركة الأدوية هذه، وتمثّل الأشكال التالية واجهة شركات الأدوية وواجهة الأدوية المصنّعة من إحداها:



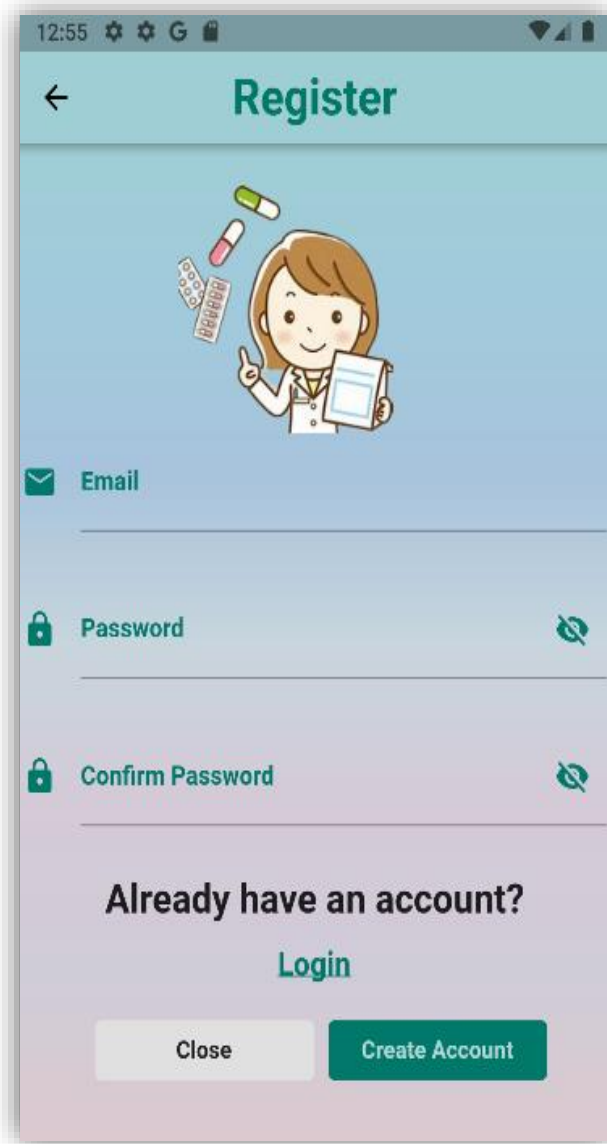
الشكل 23: واجهة إحدى شركات الأدوية لتطبيق الموبايل.



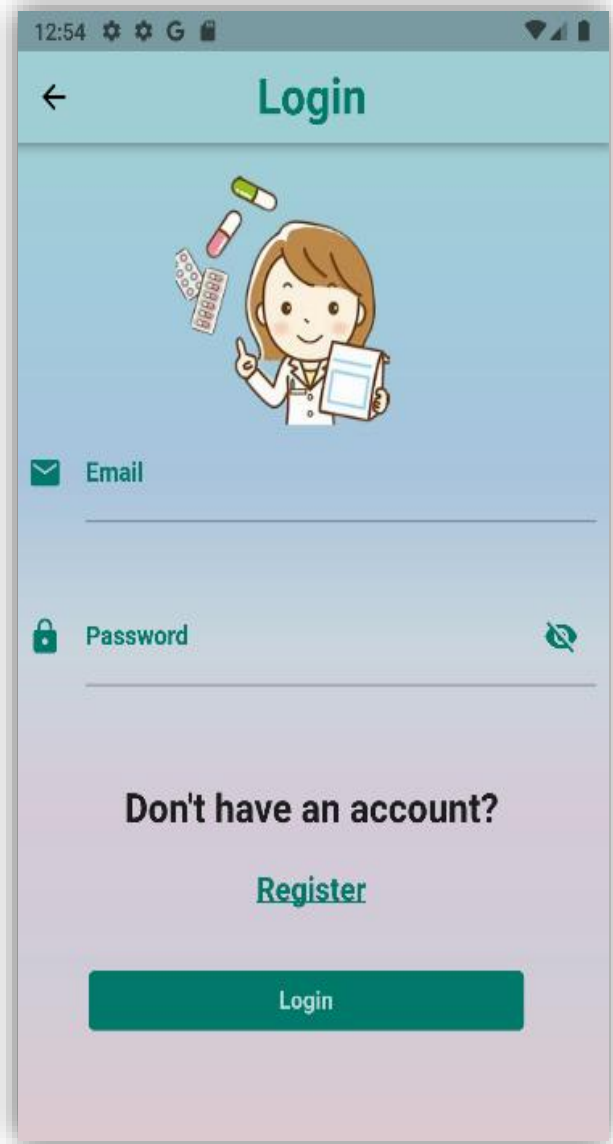
الشكل 22: واجهة شركات الأدوية لتطبيق الموبايل.

➤ واجهتي تسجيل الدخول وإنشاء حساب:

من خلال هاتين الواجهتين يمكن للمستخدم تسجيل الدخول أو إنشاء حساب في حال عدم وجود حساب سابق، وعندما تتم عملية تسجيل الدخول بنجاح تظهر الواجهة الرئيسية، يمثل الشكلين التاليين واجهتي تسجيل الدخول وإنشاء حساب:



الشكل 25: واجهة إنشاء حساب لتطبيق الموبايل.



الشكل 24: واجهة تسجيل الدخول لتطبيق الموبايل.

الخاتمة والآفاق المستقبلية

قمنا في هذا المشروع بتنفيذ تطبيق موبايل ملازمة الصيدلية وهو تطبيق يعمل على الهواتف الذكية التي تحوي نظام Android أو iOS، يقدم هذا التطبيق ميزة لطلاب الصيدلة حيث يوفر لهم معلومات عن الأدوية، ويقوم بإجراء اختبارات لهم للتأكد من إكمالهم عملية التدريب بنجاح. فيتيح لهم إجراء فترة ملازمة صيدلية عبر هذا التطبيق دون عناء الذهاب والبحث عن صيدلية تحوي شواغر للتدريب.

يوفر التطبيق معلومات كثيرة عن الأدوية، الشركات المصنعة لها، الفئات المرضية المتعلقة بها، الأشكال الدوائية لكلٍ منها، والمواد الفعالة الداخلة في تركيبها، مما يتيح مستقبلاً تطوير النظام ليصبح هناك منهجية معيّنة في التدريب بحيث يتم تقسيم المعلومات على أيام أو أسابيع محدّدة بحيث يساعد الطالب على تنظيم عملية التدريب.

أيضاً يمكن مستقبلاً إضافة اختصاصات جديدة غير الصيدلة (لغات أجنبية مثلاً، ...)، وهذا متاح لأنّه في تصميم قاعدة المعطيات يوجد جدول يعرّف عن الاختصاصات.

المراجع

- [1] Ritu Sharma, Soumya Verma, Vikash Kumar, Dr. Raghunath Verma, Android Application Development, International Journal for Research in Applied Science & Engineering Technology (IJRASET), ISSN: 2321-9653, Volume 9 Issue Jun 6, 2021- Available at www.ijraset.com.
- [2] Suhas Holla, Mahima M Katti, Android based Mobile Application Development and its Security, International Journal of Computer Trends and Technology- volume3Issue3- 2012.
- [3] Saurav Barua, The fall of Symbian, Rhine-Waal University of Applied Sciences, April,16,2017.
- [4] Maximiliano Firtman, Programming the Mobile Web, O'Reilly Media, Inc,1005 Gravenstein Highway North, Sebastopol, CA95472, United States of America, March 2013.
- [5] Anuja H. Vaidya, Sapan Naik, Comprehensive Study and Technical Overview of Application Development in iOS, Android and Window Phone 8, International Journal of Computer Applications (0975 – 8887) Volume 64– No.19, February 2013.
- [6] A ijaz Ahmad Sheikh, N isar Malik, P rince Tehseen Ganai, Khursheed Ahmad, Smartphone: Android Vs iOS, The Standard International Journals (The SIJ), ISSN: 2321 – 2381, Vol. 1, No. 4, September-October 2013