الجمهورية الجزائرية الديموقراطية الشعبية

وزارة التعليم العالي و البحث العلمي

جامعة الشهيد حمه لخضر – الوادي

كلية العلوم الدقيقة

قسم الإعلام الآلي

مذكرة نهاية الدراسة
لنيل شهادة ليسانس أكادمية

في مجال : الرياضيات و الإعلام الآلي

تخصص : أنظمة الإعلام الآلي

من إعداد الطلبة :

عقبة ضياء الدين

لعيس إبراهيم

مديني علي مهدي

عنوان المذكرة :

تصميم و تطوير تطبيق موبايل لنظام تسيير مكتبة

تحت إشراف : الأستاذ. ندیوي محمد عبد الحميد

السنة الجامعية : 2024 / 2025

ملخص المذكرة

شهدت المكتبات، عبر العصور، تطورًا ملحوظًا في أشكالها ووظائفها، وخاصة مع التحول الرقمي المتسارع. في هذا السياق، تهدف هذه المذكرة إلى تحليل واقع مكتبة "محمد الطاهر التليلي" للمطالعة العمومية بقمار، وتقديم حل تقني متمثل في تطوير تطبيق لنظام مكتبة يسهل عمليات الإعارة، والفهرسة، وإدارة الأعضاء، مع تحسين تجربة المستخدم.

تمحورت الدراسة حول ثلاث مراحل أساسية: دراسة تمهيدية شملت تحليلًا للمكتبة ووظائفها، ثم تصميم نظام المعلومات باستخدام لغة**UML** ، وأخيرًا تطوير التطبيق عمليًا باستخدام بيئة **Android Studio** ، ولغة **Kotlin** ، وقواعد بيانات **MySQL** و **Firebase**

ساهم هذا العمل في تحديد المتطلبات التقنية والوظيفية للنظام، وتصميم واجهات استخدام تفاعلية، كما تم تقديم حلول عملية تساعد على تجاوز التحديات المرتبطة بالأنظمة التقليدية. والنتيجة نظام رقمي متكامل يعزز كفاءة إدارة المكتبة ويواكب التحولات الرقمية الحديثة.

الكلمات المفتاحية

مكتبة رقمية، أتمتة الإعارة، إدارة المكتبات، تطبيق مكتبي، UML، Kotlin، Android Studio، MySQL، XAMPP، فهرسة رقمية، تصميم البرمجيات، التحول الرقمي.

شكر و تقدير

الحمد لله الذي علم الإنسان ما لم يعلم، وسخرّ له سبل العلم والمعرفة، وألهمه التوفيق والسداد.
قال تعالى في كتابه العزيز:

"وفوق كل ذي علمٍ عليم" [سورة يوسف – الآية 76].

إن من تمام النبل ورد الجميل، أن نخص بالشكر من كان لعطائه وتوجيهه الأثر البالغ في إتمام هذا العمل.

وعليه نتقدم بجزيل الشكر لأستاذنا الفاضل : **نديوي محمد عبد الحميد** عرفانا

منا بجهوده القيمّة، وحرصه الكبير على مرافقتنا خلال جميع مراحل إعداد مشروع تخرجنا.

لقد كان لتفانيه في التأطير، وحرصه الدائم على دقةّ العمل وجودته، دور محوري في توجيهنا وتطوير مهاراتنا العلمية والعملية، فله منا كل الثناء والامتنان، ونسأل الله

أن يبارك له في علمه، وأن يجزيه خير الجزاء.

جدول المحتويات

ملخص المذكرة

الكلمات المفتاحية

شكر وتقدير

مقدمة و ملخص عام للمشروع 1

تمهيد..... 2

2- إشكالية البحث..... 2

3- خطة البحث..... 3

الفصل الأول: الدراسة التمهيدية..... 4

1- نظرة عامة عن المؤسسة..... 5

2- نظرة عامة على نظام تسيير المكتبة..... 5

1.2- التعريف العام بالمكتبة..... 5

2.2- المهام الأساسية لنظام تسيير المكتبة..... 6

3.2- المخطط الوظيفي لنظام تسيير المكتبة..... 7

4.2- الخدمات الأساسية لنظام تسيير المكتبة..... 8

5.2- دراسة الوثائق و السجلات لنظام تسيير المكتبة..... 8

6.2- الفاعلون والمهام في نظام تسيير المكتبة..... 10

3- العرض العام للمشروع..... 11

1.3- عرض وتحليل الإشكالية..... 11

2.3- عرض الحل المقترح و أهمية تطبيقه..... 12

3.3- الأهداف المسطرة للمشروع..... 13

4.3- خطة تنفيذ المشروع..... 14

7- خاتمة الفصل..... 15

الفصل الثاني: تصميم النظام وفق منهجية UML..... 16

تمهيد حول دور UML في نمذجة الأنظمة..... 17

2- التعريف بمنهجية UML..... 17

3- أداة نمذجة مخططات UML 17

4- تطبيق منهجية UML على نظام تسيير المكتبة..... 18

1.4- المخططات السلوكية..... 18

1.1.4- مخطط حالات الاستخدام (Use Case Diagram)..... 18

2.1.4- مخطط الأنشطة (Activity Diagram)..... 19

3.1.4- مخطط التسلسل الزمني للعمليات (Sequence Diagram).....21
4.2- المخططات البنيوية.....24
1.4.2- مخطط الفئات (Class Diagram).....24
5-عرض تصميم جداول قاعدة البيانات وفق MLD.....27
6- **خاتمة الفصل**.....28

الفصل الثالث: تطوير وتنفيذ تطبيق تسيير المكتبة.....**29**
مقدمة الفصل.....30
2- بيئة التطوير والأدوات البرمجية المعتمدة.....30
1.2- لغة البرمجة Kotlin.....30
1.1.2- تعريف بلغة kotlin.....30
2.1.2- سبب إعتمادها في تطوير التطبيق.....30
3.1.2- المميزات التي تقدمها في إطار المشروع.....31
4.1.2- مثال تطبيقي للغة Kotlin.....31
2.2- بيئة التطوير Android Studio.....32
1.2.2- تعريف بيئة Android Studio.....32
2.2.2- أهمية Android Studio في تطوير التطبيق.....32
3.2.2- صورة واجهة Android Studio أثناء تطوير التطبيق.....33
3.2- إطار العمل Jetpack Compose.....33
1.3.2- تعريف إطار العمل Jetpack Compose.....33
2.3.2- أهمية Jetpack Compose في تصميم واجهات المشروع.....34
4.3.2- مقارنة بين Jetpack Compose و XML.....34
5.3.2- أمثلة من الواجهات المطورة ب Jetpack Compose.....35
6.3.2- صورة واجهة مطورة باستخدام Jetpack Compose.....35
4.2- قاعدة البيانات MySQL.....36
1.4.2- تعريف MySQL.....36
2.4.2- سبب اختيار MySQL بدل Firebase.....36
3.4.2- صورة توضيحية لاستعمال MySQL.....36
5.2- الخادم المحلي XAMPP.....37
1.5.2- مقدمة عن XAMPP.....37
2.5.2- دور XAMPP في المشروع.....37
3.5.2- كيفية إستخدام XAMPP في المشروع.....38
4.5.2- أهمية XAMPP في المشروع.....38
5.5.2- صورة توضيحية لإستعمال XAMPP في المشروع.....38

6.2- لغة PHP.....39

1.6.2- مقدمة عن PHP 39
2.6.2- دور PHP Backend في التطبيق..... 39
3.6.2- الأسباب التقنية لإختيار PHP 40
4.6.2- مثال تطبيقي عن PHP في التطبيق..... 40
7.2- أداة Ngrok..... 41
1.7.2- مقدمة عن أداة **Ngrok** 41
2.7.2- الدور التقني لأداة **Ngrok** في المشروع..... 41
3.7.2- مزايا أداة **Ngrok** 41
4.7.2- نافذة أوامر **Ngrok** 42
3- عرض هيكل المشروع 42
1.3- أهم المجلدات في هيكل المشروع..... 43
2.3- صورة توضيحية للمجلدات في هيكل المشروع..... 44
4- تنفيذ الواجهات الأساسية للمشروع..... 45
5- ربط الواجهات بالوظائف البرمجية 56
6- التحديات والمشاكل أثناء التطوير 58
7- خاتمة الفصل..... **59**

خاتمة المذكرة..... **60**
المراجع **61**

فهرس الأشكال

1_ الشكل 1: وثيقة بطاقة عضوية مكتبية
1_ الشكل 2: وثيقة إستمارة طلب تسجيل عضوية مكتبية
1_ الشكل 3: واجهة من تطبيق Libib للتسير المكتبي

2_ الشكل 1: مخطط حالات الاستخدام (Use Case Diagram) لنظام تسيير المكتبة
2_ الشكل 2: مخطط النشاط (Activity Diagram) لعملية إعارة كتاب
2_ الشكل 3: مخطط النشاط (Activity Diagram) لعملية إضافة كتاب
2_ الشكل 4: مخطط التسلسل الزمني للعمليات (Sequence Diagram) لعملية البحث عن كتاب
2_ الشكل 5: مخطط التسلسل الزمني للعمليات (Sequence Diagram) لعملية إعارة كتاب
2_ الشكل 6: مخطط الفئات (Class Diagram) لنظام استعارة الكتب
2_ الشكل 7: مخطط الفئات (Class Diagram) لنظام الإضافة و البحث عن كتاب

3_ الشكل 1 : كود بلغة Kotlin لواجهة تسجيل الدخول
3_ الشكل 2 : صورة توضيحية لواجهة بيئة التطويرAndroid Studio
3_ الشكل 3 : صورة لواجهة عرض كتب مطورة باستخدام Jetpack Compose
3_ الشكل 4 : صورة بنية جدول طلبات الإعارة في قاعدة البيانات باستعمالMySQL
3_ الشكل 5 : صورة إضافة طلبات إعارة بواسطةMySQL
3_ الشكل 6 : صورة تشغيل Apache و MySQL في لوحة تحكمXAMPP
3_ الشكل 7 : صورة واجهة phpMyAdmin لجدول طلبات الإعارة في قاعدة البيانات
3_ الشكل 8 : صورة لكود ملفLogin.php
3_ الشكل 9 : صورة توضح نافذة أوامر Ngrok لربط الخادم المحلي بالإنترنت
3_ الشكل 10: صورة لشجرة هيكل مجلدات المشروع فيAndroid Studio
3_ الشكل 11: واجهة تسجيل الدخول في تطبيق تسيير المكتبة
3_ الشكل 12: واجهة إنشاء حساب في تطبيق تسيير المكتبة
3_ الشكل 13: واجهة عرض الكتب بالنسبة لعضو المكتبة
3_ الشكل 14: واجهة عرض الكتب بالنسبة لعضو إدارة المكتبة
3_ الشكل 15: واجهة طلبات الإعارة بالنسبة لعضو المكتبة
3_ الشكل 16: واجهة طلبات الإعارة بالنسبة لعضو إدارة المكتبة
3_ الشكل 17: كود واجهة تطبيق ترسل بيانات تسجيل مستخدم إلى الخادم
3_ الشكل 18: ملف PHP لمعالجة تسجيل مستخدم وتخزينه في قاعدة البياناتMySQL

فهرس الجداول

1_ الجدول 1 : مقارنة بين المكتبة الحديثة و المكتبة التقليدية
3_ الجدول 1 : مقارنة بين xml و jetpack compose

مقدمة و ملخص عام للمشروع

تمهيد

شهدت المكتبات على مر العصور تطورا ملحوظا في أشكالها ووظائفها، إذ انتقلت من مجرد أماكن لتخزين الكتب والمخطوطات إلى مؤسسات ثقافية وتربوية متكاملة تسهم في نشر المعرفة وتيسير الوصول إلى المعلومات. ومع التحولّات التكنولوجية المتسارعة، أصبح من الضروري رقمنة العمليات والخدمات المكتبية لتواكب متطلبات العصر الرقمي.

في هذا السياق، شهدت نظم تسيير المكتبات تطورا كبيرا سواء على المستوى العالمي أو المحلي. فعلى المستوى العالمي، ظهرت نظم متكاملة مثل Koha و Aleph بداية من تسعينيات القرن الماضي، وهي أنظمة مفتوحة المصدر ساعدت في أتمتة الإعارة، الفهرسة، وخدمات البحث والاسترجاع. أما في الجزائر، فقد بدأت بعض المكتبات الجامعية والعمومية منذ أوائل الألفية الثالثة في اعتماد نظم رقمية بديلة لتسيير الرصيد الوثائقي وتحسين الخدمات المقدمة للمستفيدين، رغم التحديات المرتبطة بالبنية التحتية الرقمية ونقص التكوين المتخصص.

وانطلاقاً من هذا الواقع، جاءت فكرة مشروعنا المتمثلة في تطوير تطبيق موبايل لتسيير مكتبة المطالعة العمومية محمد الطاهر تليلي بقمار، بهدف تحسين كفاءة التسيير، تسهيل عمليات الإعارة، وتوفير واجهة رقمية تفاعلية تواكب تطلعات المستفيدين.

تظهر هذه الدراسة كيف أسهم التحليل الأولي في تحديد المتطلبات، والأهداف، والوظائف المتوقعة، مما ساعد على التحقق من الحلول التقنية الممكنة لتنفيذ المشروع. وفي النهاية، يستعرض البحث مختلف المراحل المرتبطة بتطوير التطبيق واعتماده كنظام تسيير مكتبي.

2. إشكالية البحث :

يرتكز هذا البحث بشكل عام حول الإشكالية التالية:

كيف يمكن لأتمتة نظام مكتبة تقليدي أن تساهم في تحسين كفاءة إدارة المكتبة، وتسهيل الوصول إلى الموارد، وتعزيز تجربة المستخدم؟

3. خطة البحث :

يهدف هذا البحث إلى تطوير تطبيق موبايل لتسيير مكتبة، يهدف إلى تبسيط وتحسين المهام الإدارية والعملية داخل مكتبة المطالعة العمومية "محمد الطاهر تليلي" بقمار، بما يخدم مصلحة كل من الإدارة والمشتركين.
وينقسم البحث إلى ثلاثة فصول رئيسية:

- **الفصل الأول: الإطار العام للدراسة والمنهجية المعتمدة .**

يتناول هذا الفصل تقديماً شاملاً لموضوع الدراسة وأهميته، مع تحليل البنية التنظيمية والوظيفية للمكتبة محل الدراسة. كما يتضمن شرحاً مفصلا للمنهجية البحثية المتبعة، والأدوات والأساليب التي تم اعتمادها في جمع البيانات وتحليلها.

- **الفصل الثاني: تحليل وتصميم النظام باستخدام لغة UML.**

يركز هذا الفصل على دراسة المتطلبات وتحليلها بدقة، متبوعا بعملية تصميم النظام بالاعتماد على لغة النمذجة الموحدة **(UML)** يتم خلاله تقديم مختلف الرسومات والمخططات التوضيحية للعمليات والمهام التي تتم في النظام، مثل:

- مخططات حالات الاستخدام **(Use Case Diagram)**
- مخططات سير الأنشطة **(Activity Diagram)**
- مخططات الأصناف أو الكائنات البرمجية **(Class Diagram)**
- مخططات التسلسل الزمني للعمليات بين الكائنات البرمجية **(Sequence Diagram)**

وذلك بهدف تجسيد التصور الكامل لكيفية عمل النظام داخلياً وخارجيا.

- **الفصل الثالث: تطوير النظام وتطبيقه البرمجي .**

يخصص هذا الفصل لاستعراض مراحل تطوير التطبيق بدءا من اختيار لغة البرمجة والتقنيات والأطر البرمجية الأنسب، مرورا بشرح معمق لخطوات الإنجاز البرمجي، وصولا إلى تقديم وصف دقيق وشامل للوظائف والخدمات التي يوفرها التطبيق المنجز.

الفصل الأول : الدراسة التمهيدية

1. نظرة عامة عن المؤسسة :

تعد مكتبة المطالعة العمومية **"محمد الطاهر تليلي" بقمار** من أبرز المؤسسات الثقافية في بلدية **قمار** بولاية **الوادي، الجزائر**. منذ تأسيسها سنة **2005** وإفتتاحها الرسمي سنة **2013**، و قد لعبت المكتبة دورا محوريا في نشر الثقافة وترسيخ عادة القراءة لدى مختلف الفئات العمرية حيث تقدم خدماتها للباحثين، والطلبة، والأطفال، ومحبي المطالعة.
و من خلال جمعنا للمعلومات حول المكتبة وجدنا أن المكتبة تتكون من طابقين:

الطابق الأرضي: يضم أماكن تخزين الكتب والإدارة والمكاتب الإدارية ومكتب التوثيق وخدمة الأنشطة.

الطابق الأول: يحتوي على قاعات للقراءة، فضاء مخصص للأطفال، قاعات بحث مجهزة للدراسات الأكاديمية، وقاعة إنترنت لدعم الوصول إلى الموارد الرقمية.

توفر المكتبة خدمات متنوعة للمشتركين عبر بطاقة اشتراك تخولهم استعارة الكتب لفترات محددة، والوصول إلى قواعد بيانات رقمية، وحجز الكتب مسبقاً عبر النظام الداخلي للمكتبة. كما توفر فضاءات مخصصة للدراسة والبحث الأكاديمي، مدعومة بنظام تنبيه آلي بخصوص توفر الكتب أو مواعيد إرجاعها.

بالإضافة إلى خدمات الإعارة والقراءة، تعد المكتبة مركزا ثقافيا متكاملا ينظم لقاءات أدبية وورشات قراءة ودورات تدريبية في مختلف المجالات، مما يعزز دورها كمركز علمي وثقافي في المنطقة.

2. لمحة عامة عن نظام تسيير المكتبة :

1.2. التعريف العام بالمكتبة :

المكتبة الرقمية هي مؤسسة ثقافية وعلمية تهدف إلى جمع وتنظيم وحفظ مصادر المعرفة بمختلف أنواعها، مثل الكتب، والمخطوطات، والمجلات، والأطروحات، والمواد الرقمية الحديثة. وتتمثل مهمتها الأساسية في توفير بيئة مناسبة للبحث والتعليم عبر إتاحة هذه الموارد للقراء من خلال الاطلاع المباشر أو خدمات الإعارة.
تتجاوز المكتبات الرقمية الحديثة دورها التقليدي، حيث تعتمد على التقنيات الرقمية المتطورة لتمكين المستخدمين من البحث عن الموارد، وحجز الكتب، وإدارة اشتراكاتهم عبر نظام إلكتروني. كما تسعى إلى تحسين تجربة المستخدم من خلال سرعة الخدمات و تحسين جودتها.

2.2. المهام الأساسية لنظام تسيير المكتبة :

1. إدارة الرصيد الوثائقي والمحتوى الرقمي :

يتناول هذا المحور عمليات إدخال الكتب والمصادر الجديدة إلى النظام المكتبي، مع تصنيفها وتنظيمها وفقا للمعايير المهنية المعتمدة في إدارة المعلومات. كما يشمل تحديث قواعد البيانات بشكل دوري لضمان تنظيم الموارد المعرفية، وتيسير عمليات البحث والاسترجاع، مع تعزيز كفاءة الوصول إلى المحتوى بطريقة منهجية ومنظمة.

2. إدارة اشتراكات الأعضاء ونظام العضوية الرقمية :

يتضمن هذا الجانب تطوير منظومة متكاملة لتسجيل الأعضاء إلكترونياً، إصدار بطاقات عضوية ذكية، إدارة بيانات الأعضاء، مراقبة صلاحية الاشتراكات، بالإضافة إلى تفعيل نظام إشعارات آلي لتنبيه المستخدمين بمواعيد التجديد أو إعلامهم بالتحديثات المتعلقة بعضويتهم.

3. إدارة عمليات الإعارة والإرجاع وتتبع حركة المواد المكتبية :

يعنى هذا المجال برقمنة عملية الإعارة والإرجاع من خلال نظام إلكتروني متطور يتيح تسجيل كل المعاملات بدقة، مع إرسال تنبيهات تلقائية للمستخدمين بقرب موعد الإرجاع، واحتساب الغرامات بشكل آلي في حال التأخير، مما يسهم في تحسين الانضباط وتحقيق أفضل استغلال للرصيد المكتبي.

4. أتمتة البحث والاسترجاع وفق المعايير الببليوغرافية :

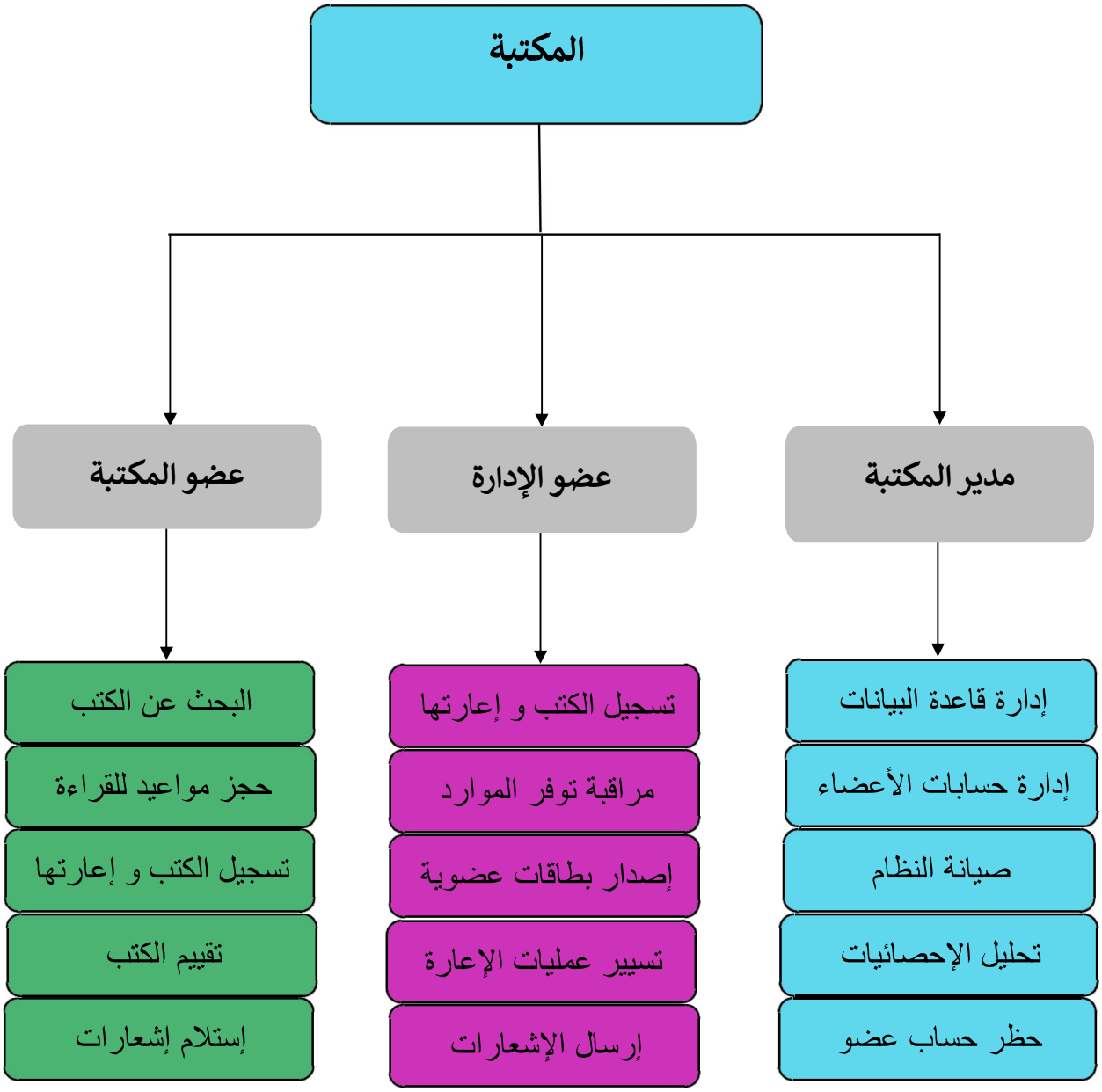
يركز هذا المحور على تطوير محرك بحث ذكي ومتكامل يتيح للمستخدمين إمكانية استكشاف الموارد حسب معايير متعددة تشمل العنوان، المؤلف، التصنيف الموضوعي، أو الكلمات المفتاحية، مع دعم نظام فهرسة متقدم لضمان سرعة الاسترجاع ودقته وفق أحدث المعايير الببليوغرافية

.

5. تحليل بيانات الاستخدام لدعم تحسين الأداء المكتبي :

يهدف هذا الجانب إلى جمع وتحليل بيانات استخدام المكتبة عبر أدوات إحصائية متطورة، مثل تحديد عدد الكتب المستعارة، الكتب الموجودة والمتوفرة، وتفضيلات المستخدمين، وعدد القراء. وهذا يساعد الإدارة في اتخاذ قرارات مبنية على معطيات فعلية لتحسين الخدمات المقدمة، وتوجيه خطط التطوير المستقبلية بفعالية.

3.2. المخطط الوظيفي لنظام تسيير مكتبة :



4.2. الخدمات الأساسية لنظام تسيير المكتبة :

1. إدارة الإعارة والإرجاع :

تتيح هذه الخدمة للمستخدمين تقديم طلبات استعارة المواد المكتبية إلكترونياً بطريقة ميسرّة، مع إمكانية متابعة مدد الإعارة بشكل دقيق من خلال النظام. كما تقوم الخدمة بإرسال إشعارات تلقائية لتذكير المستخدمين بمواعيد الإرجاع المستحقة، وإعلامهم بالإجراءات اللازمة في حال التأخير، بما يضمن سير عمليات الإعارة بسلاسة وكفاءة.

2. تسجيل الأعضاء وإدارة الحسابات الشخصية :

تمكن هذه الخدمة المشتركين من إنشاء حسابات عضوية رقمية بطريقة آمنة وسريعة، مع توفير خاصية حجز أماكن القراءة داخل المكتبة إلكترونياً، ومتابعة سجلات الاستعارة السابقة لكل مستخدم. كما تساهم هذه الخدمة في تحسين تجربة المستخدم، وتعزيز قدرة الإدارة على تتبع وإدارة بيانات الأعضاء بشكل منظم وفعال.

3. الفهرسة الرقمية والبحث المتقدم :

تعمل هذه الخدمة على تطوير قدرات البحث داخل النظام المكتبي من خلال تفعيل أدوات تصفية متقدمة تمكن المستخدمين من تحديد نتائج البحث وفق معايير متعددة مثل العنوان، اسم المؤلف، التصنيف الموضوعي، أو الكلمات المفتاحية. كما تضمن الفهرسة الذكية تنظيماً دقيقاً للمصادر، مما يسهل استرجاع المعلومات بطريقة سريعة ودقيقة.

4. الجرد الإلكتروني وإدارة المجموعات الوثائقية :

تركز هذه الخدمة على إجراء عمليات جرد دورية للرصيد الوثائقي باستخدام أنظمة إلكترونية متخصصة، مع متابعة دقيقة لحالة المواد المكتبية من حيث التوفر أو الفقدان. كما يتم إعداد تقارير تحليلية مفصلة حول وضعية المجموعات، لدعم القرارات الإدارية المتعلقة بالتحديث، الصيانة، واقتناء موارد جديدة.

5.2. دراسة الوثائق والسجلات لنظام تسيير المكتبة :

1. بطاقة العضوية المكتبية :

تعُتبر بطاقة العضوية وثيقة رسمية تصُدر لكل مستخدم مسجل في نظام المكتبة، وتحتوي على معلوماته الشخصية الأساسية مثل الاسم الكامل، رقم العضوية الفريد، وتاريخ إصدار وانتهاء صلاحية البطاقة. تمكنّ البطاقة حاملها من الاستفادة من مختلف خدمات المكتبة، بما في ذلك الإعارة وحجز الموارد.

2. إستمارة طلب التسجيل في العضوية :

تُستخدم هذه الاستمارة كآلية رسمية لتسجيل الأعضاء الجدد، حيث يتم من خلالها جمع وتوثيق المعلومات الأساسية اللازمة مثل البيانات الشخصية ووسائل الاتصال، مما يساهم في إدارة العضويات بشكل أسهل.

3. إستمارة طلب إعارة المواد المكتبية :

يتم اعتماد هذه الاستمارة خلال عملية استعارة الكتب أو المواد الأخرى، حيث توثق جميع تفاصيل العملية بما في ذلك بيانات المستعير، وصف المادة المعارة، وتحديد تواريخ الإعارة والإرجاع. تساعد هذه الوثيقة في متابعة حركة الرصيد المكتبي وضمان استرداده ضمن الآجال المحددة.

4. إشعارات التأخير في إعادة المواد المكتبية :

تصُدر هذه الإشعارات تلقائياً أو يدوياً لفائدة الأعضاء الذين تجاوزوا المدة المحددة لإرجاع المواد المستعارة. وتتضمن الإشعارات تذكيراً بوجوب الإرجاع، مع التنبيه إلى إمكانية فرض غرامات مالية أو تعليق مؤقت للعضوية إذا استمر التأخير، بما يهدف إلى الحفاظ على جودة خدمات الإعارة.

الشكل 1 : وثيقة بطاقة عضوية مكتبية

الشكل 2: وثيقة إستمارة طلب تسجيل عضوية مكتبية

6.2. الفاعلون والمهام في نظام تسيير المكتبة :

1. مدير المكتبة :

يتولى مدير المكتبة الإشراف الكامل على جميع المهام الإدارية المرتبطة بتسيير المؤسسة، حيث تشمل مسؤولياته إدارة إدخال الكتب الجديدة، مراقبة وتنظيم عمليات الجرد الدوري، الإشراف على أنشطة الإعارة والإرجاع، إصدار التقارير الدورية حول أداء المكتبة، وضمان الالتزام بالسياسات الداخلية والمعايير التنظيمية المعتمدة، بما يحقق جودة الخدمات واستمراريتها.

2 . المشترك (عضو المكتبة) :

يعُتبر المشترك مستخدماً معتمداً يستفيد من مختلف خدمات المكتبة، بما في ذلك البحث في الفهارس الرقمية، استعارة المواد المكتبية وإرجاعها، وتجديد فترات الإعارة إلكترونياً من خلال حسابه الشخصي. ويعُد الالتزام بشروط الاستخدام وسياسات المكتبة من المتطلبات الأساسية للاستفادة المستدامة من هذه الخدمات.

3. الطاقم الإداري :

يعنى الطاقم الإداري بدعم سير العمليات اليومية داخل المكتبة، من خلال استقبال وتسجيل الأعضاء الجدد، إصدار بطاقات العضوية، مساعدة المستفيدين في عمليات البحث عن المواد المكتبية، وتسيير مختلف الإجراءات الإدارية، بما في ذلك معالجة المخالفات المتعلقة بالتأخير أو فقدان المواد. ويسهم الطاقم بشكل فعّال في ضمان تقديم خدمات ذات جودة عالية وتحقيق رضا المستفيدين.

3. العرض العام للمشروع :

1.3. عرض و تحليل الإشكالية :

رغم التطورات التقنية، لا تزال العديد من المكتبات تعتمد على أنظمة تقليدية ، مثل التسجيل اليدوي للأعضاء ، تدوين عمليات الإعارة يدويا أو عبر جداول إكسل ، وتصنيف الكتب بطريقة يدوية .
هذا الوضع يؤدي إلى مشكلات تنظيمية متعددة ، مثل ضياع السجلات و الوثائق ، صعوبة تتبع الإعارات ، وبطء في استرجاع الكتب . كما أن غياب نظام تحليلي يجعل من الصعب اتخاذ قرارات إدارية مبنية على معطيات دقيقة و في ما يلي مقارنة توضح بشكل أعمق الإشكالية المدروسة :

الجانب المقارن	النظام التقليدي	النظام الإلكتروني المقترح
إدارة الأعضاء	تسجيل ورقي أو Excel	تسجيل إلكتروني بقاعدة بيانات موحدة
تتبع الإعارات	يدوي و بطيء	سريع و آلي مع إشعارات تلقائية
تصنيف الكتب	يدوي و غير منظم	مصنف وفق معايير ببلوغرافية رقمية (تصنيف ديوي العشري)
إسترجاع المعلومات	صعب و يتطلب وقتا طويلا	بحث سريع عن طريق كلمات مفتاحية
حماية الوثائق	معرضة للضياع و ضعيفة	مؤمنة ضمن نظام رقمي
دعم القرار الإداري	غياب للبيانات الدقيقة	وجود إحصائيات و تقارير فورية

الجدول 1 : مقارنة بين المكتبة الحديثة و المكتبة التقليدية

2.3 . عرض الحل و أهمية تطبيقه :

يمثل التحول نحو نظام رقمي متكامل خطوة محورية في تطوير أداء المكتبة وتعزيز جودة خدماتها.

يسهم هذا الحل في أتمتة مختلف العمليات التقليدية، مما يقلل من نسبة الأخطاء البشرية، ويزيد من سرعة معالجة طلبات البحث والإعارة، بالإضافة إلى تحسين إدارة الرصيد الوثائقي بشكل أكثر دقة وفعالية.

كما يوفر النظام الرقمي أدوات تحليل متقدمة لإعداد تقارير تفصيلية تدعم عملية اتخاذ القرار الإداري، وتسهم في التخطيط الاستراتيجي لتطوير خدمات المكتبة بما يلبي تطلعات المستفيدين ويواكب التحولات التقنية الحديثة.

وكأمثلة على أنظمة وتطبيقات رقمية طبقت هذه الحلول لدينا:

تطبيق Libib :

هو تطبيق لإدارة المكتبات يستخدم من طرف الأفراد، المدارس، والمكتبات الصغيرة. يتميز بواجهة بسيطة وسهلة الاستعمال، يعمل على الويب وكذلك على الهواتف Android وiOS وقد ساهم في تحسين نظام تسيير المكتبات من خلال:

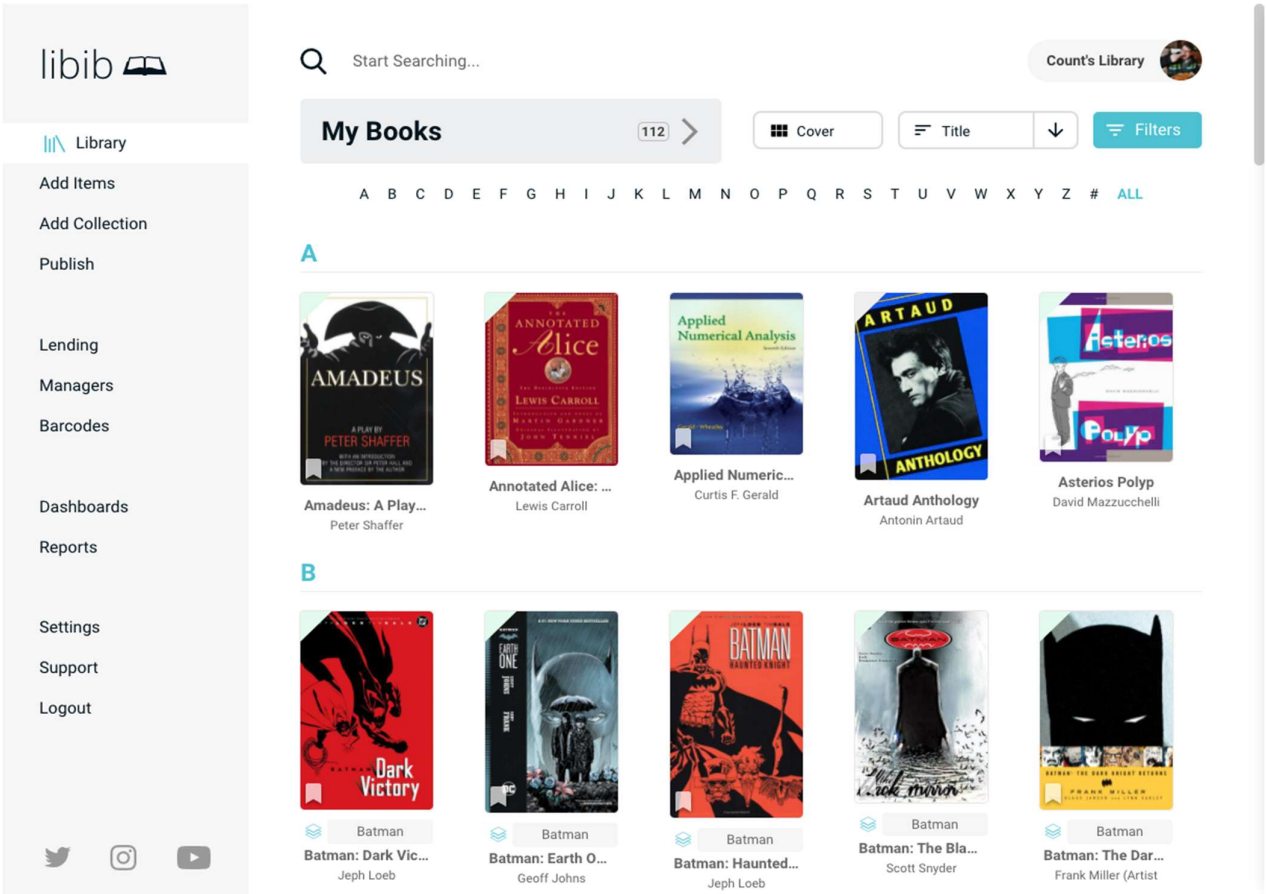
1. **توفير إمكانية إضافة الكتب بسهولة عبر الهاتف باستخدام الباركود.**
2. **تسهيل عملية إعارة واسترجاع الكتب عبر واجهة بسيطة ومباشرة.**
3. **تمكين المشرفين من تنظيم الكتب حسب التصنيفات ومتابعة الإعارات دون أوراق.**

تطبيق BookBuddy:

هو تطبيق موجه أكثر لهواة القراءة أو المكتبات الصغيرة، ويعمل على أجهزة iOS

يسمح للمستخدمين بإنشاء قاعدة بيانات شخصية لمجموعاتهم من الكتب، ويسُتخدم أيضاً في بعض المكتبات لتسيير الإعارات يدوياً بطريقة منظمة و يتميز ب :

1. **تقديم نظام لعرض حالة كل كتاب (متاح، معار، مفقود).**
2. **تحسين عملية البحث والوصول إلى الكتاب عبر فلترة وتصنيفات دقيقة.**
3. **المساعدة في تتبع سجل الإعارات لكل مستخدم مما يسهل سير العملية بوضوح.**



الشكل 3 : واجهة من تطبيق Libib للتسيير المكتبي

3.3. الأهداف المسطرة للمشروع :

- **أتمتة عمليات تسيير المكتبة :**

يسهم الحل المقترح في استبدال الأساليب التقليدية بأنظمة إلكترونية متكاملة تعمل على إدارة عضويات المستخدمين ، تسهيل عمليات البحث عن المواد المكتبية ، وتسيير إجراءات الإعارة والإرجاع بطرق أكثر كفاءة ، مما يعزز من مستوى التنظيم الداخلي للمكتبة .

- **تحسين تجربة المستخدم :**

من خلال توفير واجهة استخدام رقمية حديثة ، يتيح الحل تعزيز سهولة التفاعل مع خدمات المكتبة عبر دعم خاصية البحث المتقدم ، وتمكين إرسال الإشعارات التلقائية للمستخدمين ، بما يضمن تواصلا فعالا ويثري تجربة الاستفادة من الموارد المكتبية .

- **تقليل الأخطاء التشغيلية :**

يساهم النظام الرقمي في تقليص نسبة الأخطاء الناتجة عن المعالجة اليدوية للبيانات ،
عبر إعتماد التوثيق الإلكتروني لكل العمليات الإدارية ، مما يضمن دقة المعلومات وسلامة سير مختلف الخدمات المقدمة
.

- **تحليل أداء المكتبة :**

يوفر الحل آليات فعآلة لإعداد تقارير دورية وتحليلات إحصائية دقيقة ، تدعم صانعي القرار داخل المكتبة في متابعة الأداء العام ، تحديد نقاط القوة والضعف ، وتوجيه خطط التطوير المستقبلية على أسس موضوعية .

4.3. خطة تنفيذ المشروع :

- **المرحلة الأولى : الدراسة التمهيدية .**

تتضمن هذه المرحلة تحليل الوضعية الحالية للمكتبة ، بهدف تشخيص أوجه القصور و نقاط الضعف في أساليب التسيير المكتبي التقليدية ، مع اقتراح الحلول الرقمية الممكنة لمعالجتها ، بما يضمن الانتقال السلس نحو نظام إلكتروني متكامل .

- **المرحلة الثانية : تحليل النظام وتحديد المتطلبات .**

في هذه المرحلة ، يتم بناء تصور دقيق للنظام من خلال دراسة وتحليل الاحتياجات التقنية والوظيفية . ويعتمد على لغة النمذجة الموحدة UML لإنشاء مجموعة من المخططات التمثيلية مثل :

مخطط حالات الإستخدام (Uses Cases Diagram) .

مخطط الأصناف أو الكائنات البرمجية (Class Diagram) .

مخطط التسلسل الزمني للعمليات (Sequence Diagram) .

مخطط سير النشاطات (Activity Diagram) .

و كل هذا بهدف وضع تصور شامل لكيفية تفاعل النظام مع مستخدميه وبياناته .

- **المرحلة الثالثة : تطوير التطبيق و تنفيذه .**

تركز هذه المرحلة على ترجمة التصاميم التحليلية إلى تطبيق عملي باستخدام بيئة التطوير **Android Studio** ولغة البرمجة **Kotlin** مع إطار العمل **Jetpack Compose** لإنشاء واجهة مستخدم حديثة **(Modern UI)** ، ديناميكية ، ومتجاوبة **(Responsive)** مع مختلف أحجام الشاشات .

كما يتم إنشاء قاعدة بيانات باستخدام **MySQL** مدعومة بخادم محلي **XAMPP** ، مع ربط التطبيق بقاعدة البيانات عبر سكريبتات **PHP** مصممة خصيصا لضمان التكامل الوظيفي **(Functional Integration)** بين التطبيق و قاعدة البيانات .

و لتسهيل إختبار التطبيق على أجهزة متعددة وشبكات مختلفة سنستخدم أداة **Ngrok** لإنشاء نفق آمن **(Tunnel Connection)** يتيح الوصول إلى الخادم المحلي من خلال رابط عام **(Public URL)**، مما يسمح بمحاكاة بيئة العمل الفعلية دون الحاجة إلى استضافة خارجية.

تُختتم هذه المرحلة بإجراء اختبارات شاملة للتحقق من صحة الوظائف ، ضمان استقرار الأداء ، واكتشاف أي أخطاء أو نقاط خلل محتملة قبل الإعتماد النهائي للتطبيق .

خاتمة الفصل

تعد هذه الدراسة التمهيدية حجر الأساس للمشروع ، حيث وفرت تحليلا شاملا للوضع الراهن لنظام تسيير المكتبة المستهدفة ، وحددت التحديات المرتبطة بالأساليب التقليدية التي تتبعها ، وأبرزت الحاجة إلى حلول رقمية متطورة . كما رسمت أهداف المشروع وحددت الإطار المنهجي الذي سيقود المراحل القادمة من التطوير ، مما يضمن إنتقالا منهجيا ومدروسا نحو إنشاء نظام رقمي متكامل لإدارة المكتبات بكفاءة وفعالية

الفصل الثاني : تصميم النظام وفق منهجية UML

1. تمهيد حول دور UML في نمذجة النظام :

يعتبر التصميم الممنهج للنظم أحد الركائز الأساسية في نجاح أي مشروع برمجي، حيث يسهم في فهم دقيق لمتطلبات النظام وتحديد بنيته ووظائفه قبل الشروع في عملية التطوير. ومن بين أبرز الأدوات المعتمدة لتحقيق هذا الهدف نجد لغة النمذجة الموحدة **UML** ، التي تعد وسيلة فعالة لتمثيل مكونات النظام وعلاقاته بطريقة بصرية ومنهجية .

في إطار مشروعنا المتمثل في تطوير تطبيق موبايل لتسيير مكتبة **"محمد الطاهر تليلي" بقمار**، إعتمدنا على **UML** لتصميم النظام بشكل يسمح بفهم شامل لجميع العمليات الرئيسية، من تسجيل المستخدمين واستعارة الكتب، إلى تسيير قاعدة البيانات والمهام الإدارية. يساعد استخدام **UML** على تبسيط التواصل بين أعضاء الفريق وتوثيق مكونات النظام بطريقة معيارية، مما يمُهد لتطوير تطبيق مستقر وقابل للتوسع مستقبلا .

2. التعريف بمنهجية UML :

لغة النمذجة الموحدة **UML (Unified Modeling Language)** هي لغة معيارية تسُتخدم على نطاق واسع في مجال هندسة البرمجيات لنمذجة وتحليل وتصميم الأنظمة المعلوماتية. ظهرت **UML** في تسعينيات القرن الماضي كتوحيد لمجموعة من المنهجيات السابقة **(OMT, Booch, OOSE)** وذلك بهدف توفير وسيلة مرئية، موحدة وقابلة للفهم تساعد محللي ومطوري الأنظمة على تصور بنية وسلوك الأنظمة المعقدة بطريقة دقيقة ومنظمة.

تعتمد **UML** على مجموعة من المخططات **(Diagrams)** التي تصنف إلى نوعين رئيسيين :
مخططات بنيوية **(Structural Diagrams)** مثل مخطط الفئات **(Class Diagram)** و مخططات سلوكية **(Behavioral Diagram)** مثل مخطط حالات الإستخدام **(Use Case Diagram)** حيث يمكن إستخدام هذه المخططات على طول دورة حياة تطوير النظام بدءا من تحليل المتطلبات مرورا بتصميم النظام و إنتهاءا بتنفيذ التطبيق .

في إطار هذا المشروع، المتمثل في تطوير تطبيق موبايل لتسيير مكتبة المطالعة العمومية **"محمد الطاهر تليلي" بقمار**، تم الاعتماد على منهجية **UML** لتوفير تصور شامل ودقيق لمختلف جوانب النظام، سواء من حيث الوظائف التي يقدمها للمستخدمين مثل **(إستعارة الكتب، البحث عن العناوين، إدارة الحسابات)**، أو من حيث البنية الداخلية للتطبيق والكيانات المكونة له. تسمح هذه المنهجية بضمان وضوح الرؤية لجميع المعنيين بالمشروع **(محللين، مطورين، مشرفين)**، كما تساهم في تسهيل الصيانة المستقبلية وإمكانية تطوير النظام لاحقاً.

بالتالي، تشكل **UML** أداة فعالة لتجسيد الأفكار وتحويل المتطلبات النصية إلى نماذج بصرية قابلة للتحليل والمراجعة، مما يقلل من الأخطاء المحتملة ويزيد من كفاءة عملية تطوير البرمجيات.

3. أداة نمذجة مخططات UML :

لإستعراض المخططات بطريقة دقيقة ومنظمة، تم اعتماد أداة **PlantUML Editor** عبر الإنترنت، وهي أداة مفتوحة المصدر تستخدم على نطاق واسع في الأوساط الأكاديمية والمهنية لإنشاء مخططات **UML** باستخدام لغة وصف نصي بسيطة. تتميز هذه الأداة بواجهة سهلة الاستخدام ومتاحة عبر المتصفح، مما يتيح إمكانية الوصول إليها دون الحاجة إلى تثبيت برامج إضافية. كما توفر مرونة عالية في تعديل المخططات، وتصديرها كصور أو ملفات قابلة للدمج داخل الوثائق التقنية والعلمية.

تدعم **PlantUML Editor** عبر الإنترنت جميع أنواع المخططات البنيوية والسلوكية، مما يجعلها أداة مثالية لتجسيد بنية وسلوك النظام الذي يتم تطويره في هذا المشروع. باستخدام هذه الأداة، تمكنا من تحسين وضوح التصاميم وتسريع عملية التواصل بين مراحل التحليل والتطوير، مما ساهم في تنظيم العمل وتوضيح التصورات المطلوبة بسهولة ودقة.

4. تطبيق منهجية UML على نظام تسيير المكتبة :

بعد تقديم المفاهيم النظرية المتعلقة بمنهجية **UML**، سنقوم بتطبيق هذه المنهجية من خلال نمذجة نظام تسيير مكتبة المطالعة العمومية. حيث نهدف إلى ترجمة متطلبات النظام إلى مخططات **UML** تسهل فهم النظام، وتوضح بنيته وسلوكاته المختلفة .

سنقوم بتصنيف هذه المخططات إلى قسمين رئيسيين :

1.4. المخططات السلوكية (Behavioral Diagrams) :

تركز المخططات السلوكية على وصف ديناميكية النظام، أي كيف يتفاعل المستخدمون مع النظام وكيف تتغير حالاته بناءً على هذه التفاعلات.

تهدف هذه المخططات إلى توضيح سلوك النظام عبر الزمن، وكيفية تنفيذ العمليات، وتسلسل الخطوات، وتدفق البيانات أو الأوامر داخله.

في تطبيق تسيير المكتبة، يمكن استخدام المخططات السلوكية لشرح تفاعل المستخدم مع النظام أثناء تنفيذ عملية إستعارة كتاب، تسجيل الدخول، أو كيفية إرسال إشعار للمستخدم عند قبول طلبه.

في هذا القسم، سنعرض مجموعة من المخططات السلوكية و تطبيقها في النظام المكتبي و تتمثل في :

1.1.4. مخطط حالات الإستخدام (Use Case Diagram) :

- **التعريف بالمخطط :** مخطط حالات الاستخدام **(Use Case Diagram)** هو أحد المخططات السلوكية في **UML**، يستخدم لتحديد الوظائف الأساسية التي يوفرها النظام للمستخدمين **(Actors)**، من خلال وصف التفاعلات التي يمكن أن تحدث بين النظام والمستخدمين الخارجيين. يعد هذا المخطط من الأدوات الأساسية لفهم المتطلبات الوظيفية للنظام بطريقة مبسطة وواضحة.

- **أهمية المخطط في النظام :** في تطبيق تسيير المكتبة، يساهم مخطط حالات الاستخدام في تحديد وتوضيح جميع العمليات التي يمكن للفاعلين في النظام القيام بها، مثل "تسجيل الدخول"، "البحث عن كتاب"، "طلب استعارة كتاب". هذا التحديد المسبق يساعد في تنظيم منطق العمل داخل التطبيق وتحديد الأولويات أثناء التطوير.

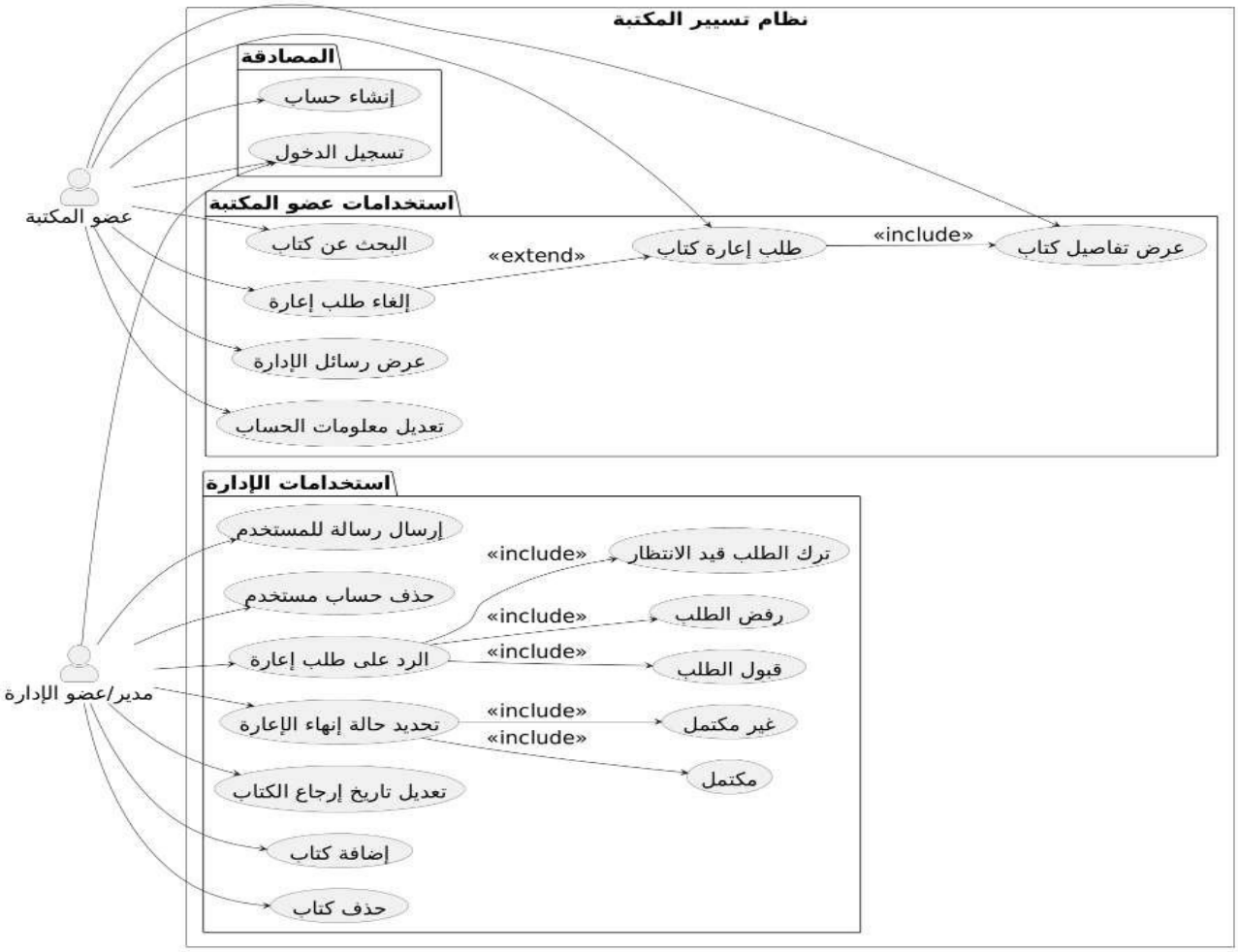
- **مثال تطبيقي للمخطط في النظام :** نفترض أن لدينا فاعلين رئيسيين:

  - **عضو المكتبة :** المستخدم العادي الذي يستفيد من خدمات استعارة الكتب والاطلاع عليها.

  - **عضو الإدارة/المدير :** الجهة المسيرة للنظام، ويقوم بإدارة العمليات الإدارية مثل معالجة الطلبات، تنظيم البيانات، وتحديث محتوى المكتبة.

يعكس هذا السيناريو التفاعلات الأساسية بين المستخدمين والنظام، دون الخوض في تفاصيل كل وظيفة، حيث سيتم توضيح هذه العمليات لاحقاً من خلال الرسم التخطيطي.

- **تطبيق المثال في المخطط :**



الشكل 4 : مخطط حالات الإستخدام (Use Case Diagram) لنظام تسيير المكتبة

2.1.4. مخطط الأنشطة و العمليات (Activity Diagram) :

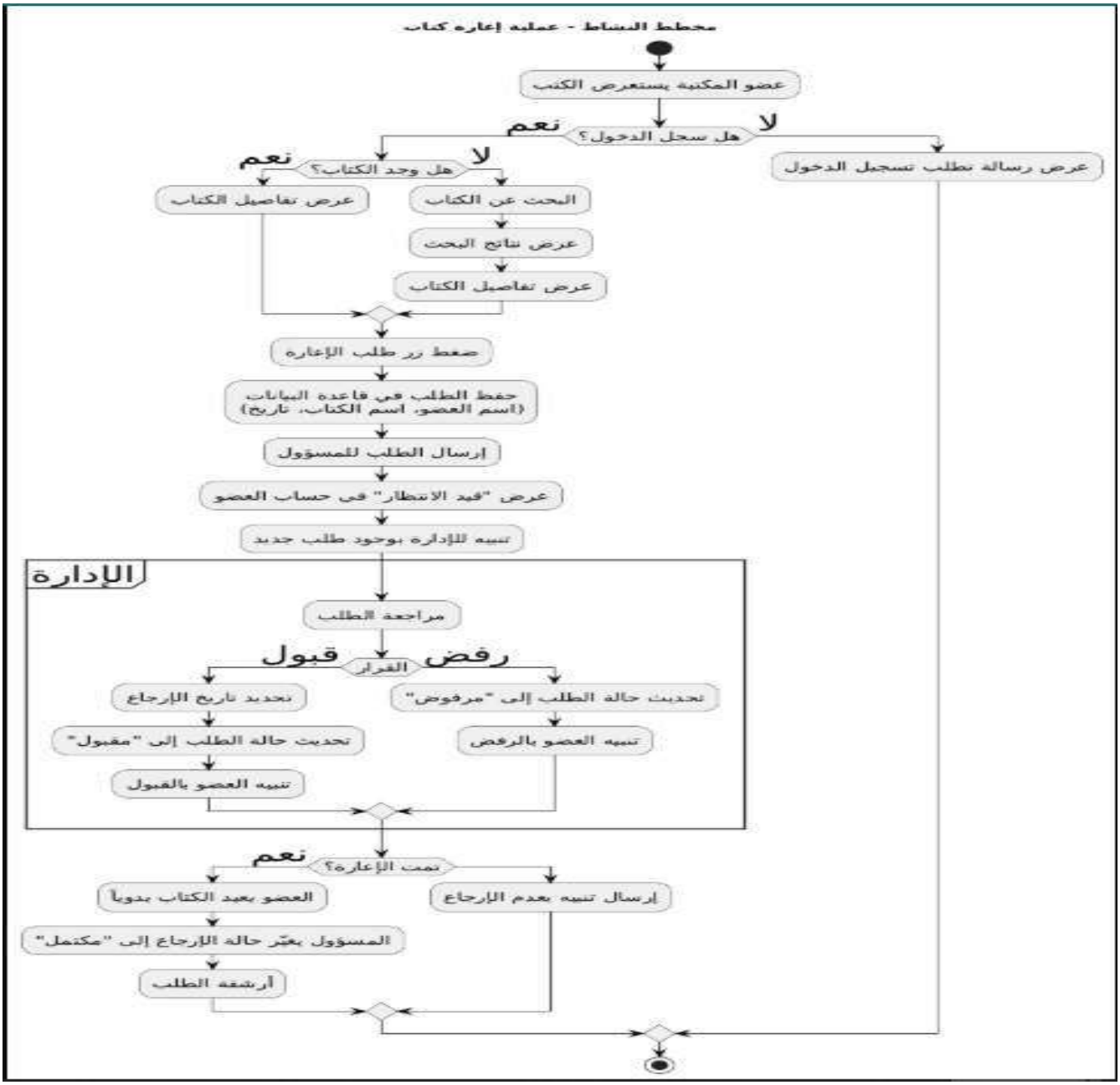
- **التعريف بالمخطط :** مخطط الأنشطة **(Activity Diagram)** هو أحد المخططات السلوكية في منهجية **UML**، يستخدم لتمثيل تدفق الأنشطة داخل النظام بشكل متسلسل أو متوازي. يوضح هذا النوع من المخططات كيفية إنتقال التحكم بين الأنشطة المختلفة، ويستخدم غالباً لنمذجة الإجراءات **(Processes)** أو الوظائف **(Functions)** داخل النظام. يعتمد هذا المخطط على رموز بسيطة كالعقُد **(Nodes)** والسهام **(Edges)** لتوضيح بداية النشاط، نهايته، والتفرعات أو التزامن بين الأنشطة.

- **أهمية المخطط في النظام :** في إطار تطوير تطبيق لتسيير مكتبة عمومية، يعد مخطط النشاط أداة فعالة لفهم وتنظيم خطوات تنفيذ العمليات المعقدة داخل النظام مثل عملية إستعارة الكتب، أو معالجة الطلبات من طرف مدير المكتبة. يظهر هذا المخطط التسلسل المنطقي الذي تمر به كل عملية، ويساعد على تحديد النقاط الحرجة في التدفق والقرارات التي تؤثر على سلوك النظام. بذلك، يسهم في تحسين جودة التصميم وتجنب الغموض في فهم الإجراءات الداخلية للتطبيق.

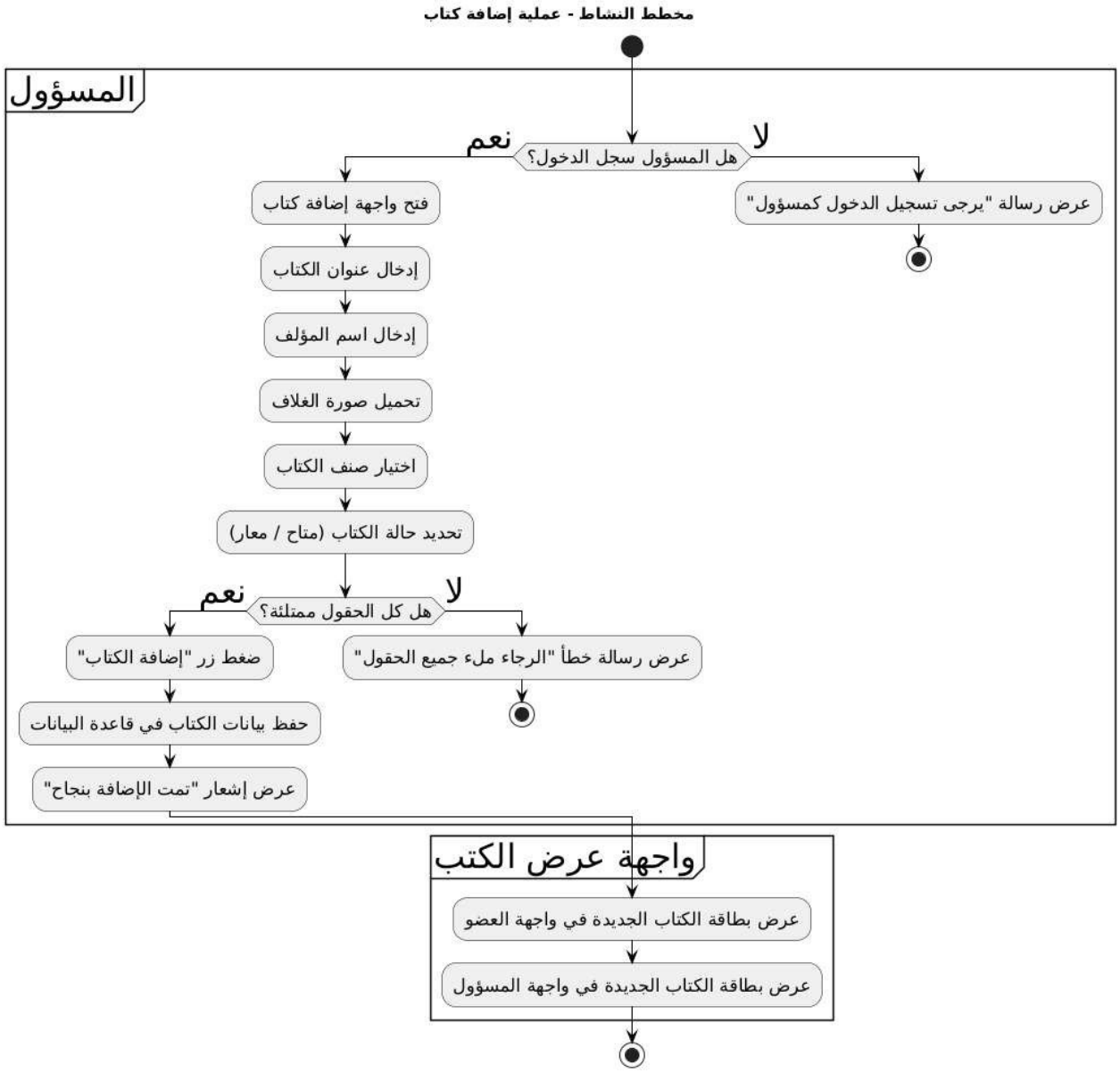
- **مثال تطبيقي للمخطط في النظام :** كمثال عام يمكن إعتماده هو عملية استعارة كتاب من طرف عضو المكتبة، حيث يوضح المخطط الخطوات التي يمر بها هذا الطلب بداية من قيام العضو بتقديم الطلب، مرورا بمعالجة المدير لهذا الطلب، ووصولاً إلى إتمام أو رفض العملية بناءً على توفر الكتاب والشروط الأخرى و كذلك إكمال عملية الإعارة عند إرجاع الكتاب من طرف عضو المكتبة.

كذلك في تطبيق تسيير المكتبة، يعد "إضافة كتاب جديد" من المهام الأساسية التي يقوم بها مدير المكتبة أو عضو الإدارة. تتم هذه العملية عند توفر كتاب جديد يراد إدراجه ضمن فهرس المكتبة حتى يصبح متاحا للأعضاء للاطلاع أو الإعارة. تتضمن العملية عادةً سلسلة من الأنشطة المنظمة، إدخال معلومات الكتاب، التحقق من صحتها، ثم حفظها في قاعدة البيانات و عرضها سواء بالنسبة لعضو المكتبة أو العضو الإداري.

- **تطبيق المثال في المخطط :**



الشكل 5 : مخطط النشاط (Activity Diagram) لعملية إعارة كتاب



الشكل 6 : مخطط النشاط (Activity Diagram) لعملية إضافة كتاب

3.1.4. مخطط التسلسل الزمني للعمليات (Sequence Diagram) :

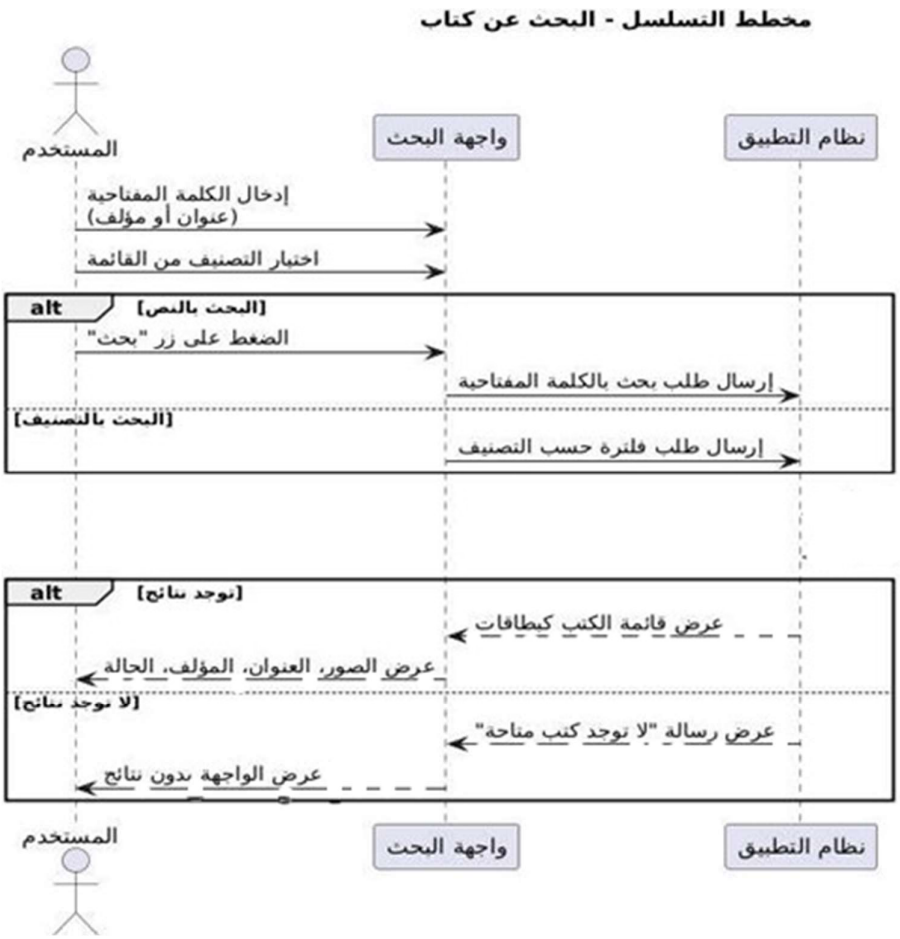
- **التعريف بالمخطط :** مخطط التسلسل **(Sequence Diagram)** هو أحد المخططات السلوكية في منهجية **UML**، يستخدم لتمثيل التفاعل بين الكائنات **(Objects)** في النظام وفق تسلسل زمني معين. يوضح هذا المخطط كيف تتبادل الكائنات الرسائل فيما بينها لإنجاز وظيفة معينة، مع التركيز على ترتيب الأحداث وتدفق الرسائل خلال الزمن. يتكون عادة من مجموعة من الكائنات مرتبة أفقيا، ومحور زمني عمودي يمثل تتابع الرسائل من الأعلى إلى الأسفل.

- **أهمية المخطط في النظام :** في سياق تطوير تطبيق مخصص لتسيير مكتبة عمومية، يساعد مخطط التسلسل على فهم تفاصيل التفاعل بين مكونات النظام عند تنفيذ وظائف محددة مثل تسجيل الدخول، استعارة كتاب، أو التحقق من توفر عنوان معين. يسهم هذا المخطط في الكشف عن العلاقات الديناميكية بين الكائنات، ما يتيح تحسين بنية البرمجيات وضمان ترابط المكونات بالشكل الصحيح وفق السيناريوهات الواقعية التي يتعامل معها التطبيق.

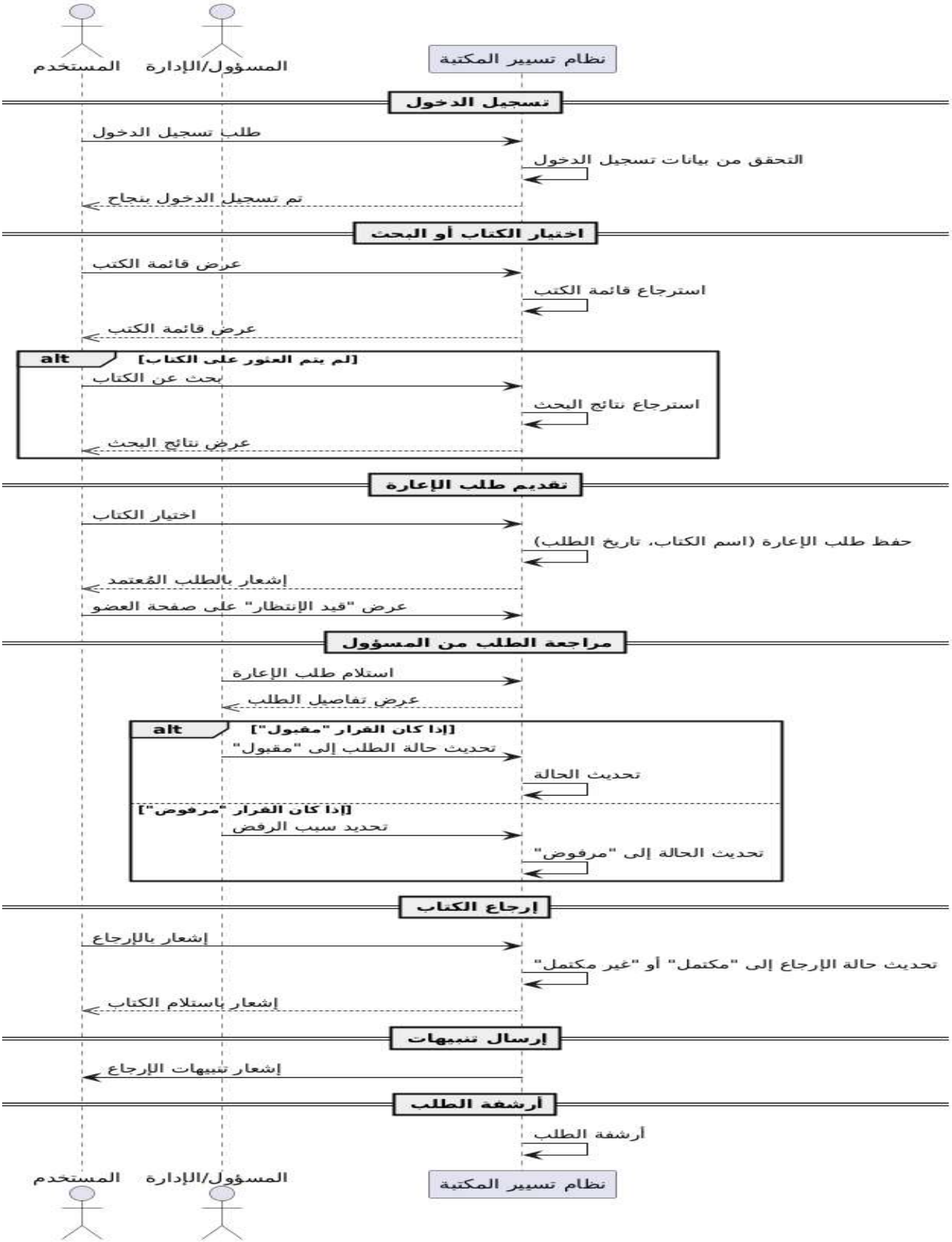
- **مثال تطبيقي للمخطط في النظام :** كمثال عام، يمكن تمثيل سيناريو معالجة طلب استعارة كتاب، بحيث يظهر المخطط كيفية تفاعل الكائنات الرئيسية في النظام **(**عضو المكتبة**،** النظام**،** قاعدة البيانات**،** مدير المكتبة**)** بداية من إدخال العضو لطلبه**،** مرورا بمعالجة الطلب**،** وانتهاء بإعلامه بنتيجة العملية**.**

تعتبر ميزة "البحث عن كتاب" من الوظائف الأساسية التي يستخدمها عضو المكتبة عند رغبته في العثور على كتاب معين داخل التطبيق**،** سواء للاطلاع على معلوماته أو التقدم بطلب استعارة**.** تتم هذه العملية بطريقة تفاعلية بين المستخدم والنظام**،** حيث يقوم العضو بإدخال كلمة مفتاحية **(**مثل العنوان أو المؤلف**)،** ليقوم النظام باسترجاع النتائج المطابقة من قاعدة البيانات**.**

- **تطبيق المثال في المخطط :**



الشكل 7 : مخطط التسلسل الزمني للعمليات (Sequence Diagram) لعملية البحث عن كتاب



الشكل 8 : مخطط التسلسل الزمني للعمليات (Sequence Diagram) لعملية إعارة كتاب

2.4. المخططات البنيوية (Structural Diagrams) :

تهتم المخططات البنيوية في **UML** بتمثيل الهيكل الداخلي للنظام والعلاقات الثابتة بين مكوناته، مثل الكائنات **(Objects)**، الأصناف **(Classes)**، والمكونات المختلفة التي يتكون منها النظام.
تستخدم هذه المخططات لتصميم قاعدة الكود ومكوناته بشكل واضح ودقيق، مما يسهل على المطورين بناء نظام متماسك وقابل للتطوير.
في سياق مشروعنا الخاص بتسيير مكتبة المطالعة، تساعدنا هذه المخططات على تحديد بنية النظام من حيث الأصناف (مثل كتاب، طالب، أمين المكتبة...)، وعلاقتها ببعضها البعض، وهي ضرورية في المرحلة الأولى من البرمجة.
سنقوم في هذا القسم بعرض أهم المخططات البنيوية المستخدمة في تصميم النظام، وهي :

1.2.4. مخطط الفئات (Class Diagram) :

- **التعريف بالمخطط :** يعد مخطط الفئات أحد أهم المخططات البنيوية في منهجية **UML**، ويستخدم لتمثيل الهيكل الداخلي للنظام من خلال عرض الكائنات الأساسية **(Classes)** وخصائصها **(Attributes)** ووظائفها **(Methods)**، إضافة إلى العلاقات التي تربطها ببعضها مثل التوريث، التجميع، والاعتماد. يعد هذا المخطط أساسا في عملية التصميم الشيئي **(Object-Oriented Design)**، حيث يحدد الهيكل العام للكيانات التي يتكون منها النظام.

- **أهمية المخطط في النظام :** في مشروع تطوير تطبيق لتسيير مكتبة عمومية، يلعب مخطط الصنف دورا مركزيا في نمذجة الكيانات الأساسية مثل: الكتب، الأعضاء، طلبات الإعارة، المشرفين، والعمليات المرتبطة بكل كيان. يساعد هذا المخطط في فهم البنية العامة للبيانات التي يديرها النظام، كما يسهل على المطورين لاحقا تحويل هذه النماذج إلى كود برمجي ضمن هيكل منظم ومتماسك.

- **مثال تطبيقي للمخطط في النظام :**

بشكل عام، يمكن أن يتضمن المخطط أصنافاً بالنسبة لعلملية إعارة الكتب مثل :

- **Book** (كتاب) ويحتوي على خصائص مثل id, titre, author, statut.

- **User** (عضو مكتبة أو مسؤول إداري) ويمتلك خصائص مثل id, name, email,.

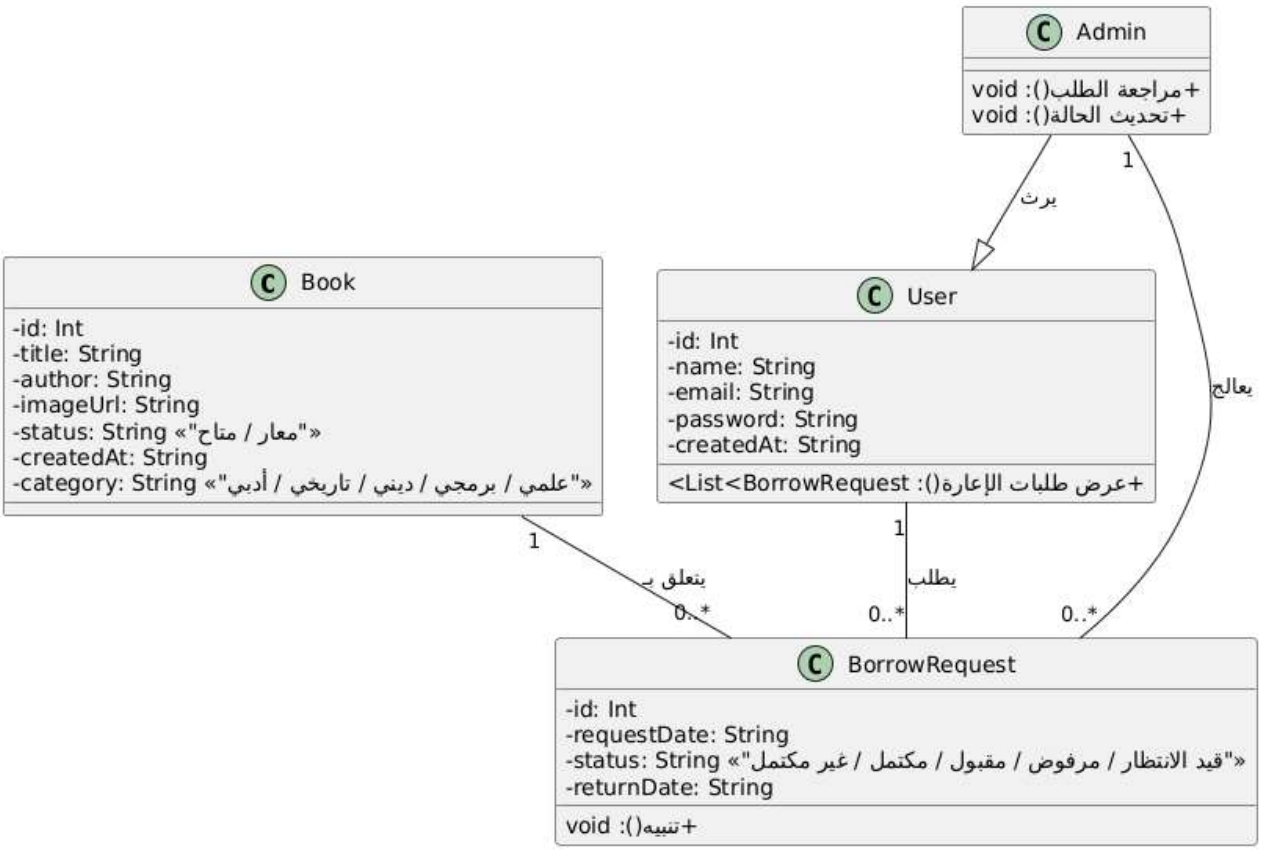
- **BorrowRequest** (طلب استعارة) تمثل العلاقة بين عضو المكتبة والكتاب و مسؤول المكتبة.

ويتم الربط بين هذه الأصناف بواسطة علاقات تمثل كيفية تفاعلها، مثل:
"عضو يقدمّ عدة طلبات استعارة"، أو **"كتاب يمكن أن يكون موضوعاً لعدة طلبات"**.

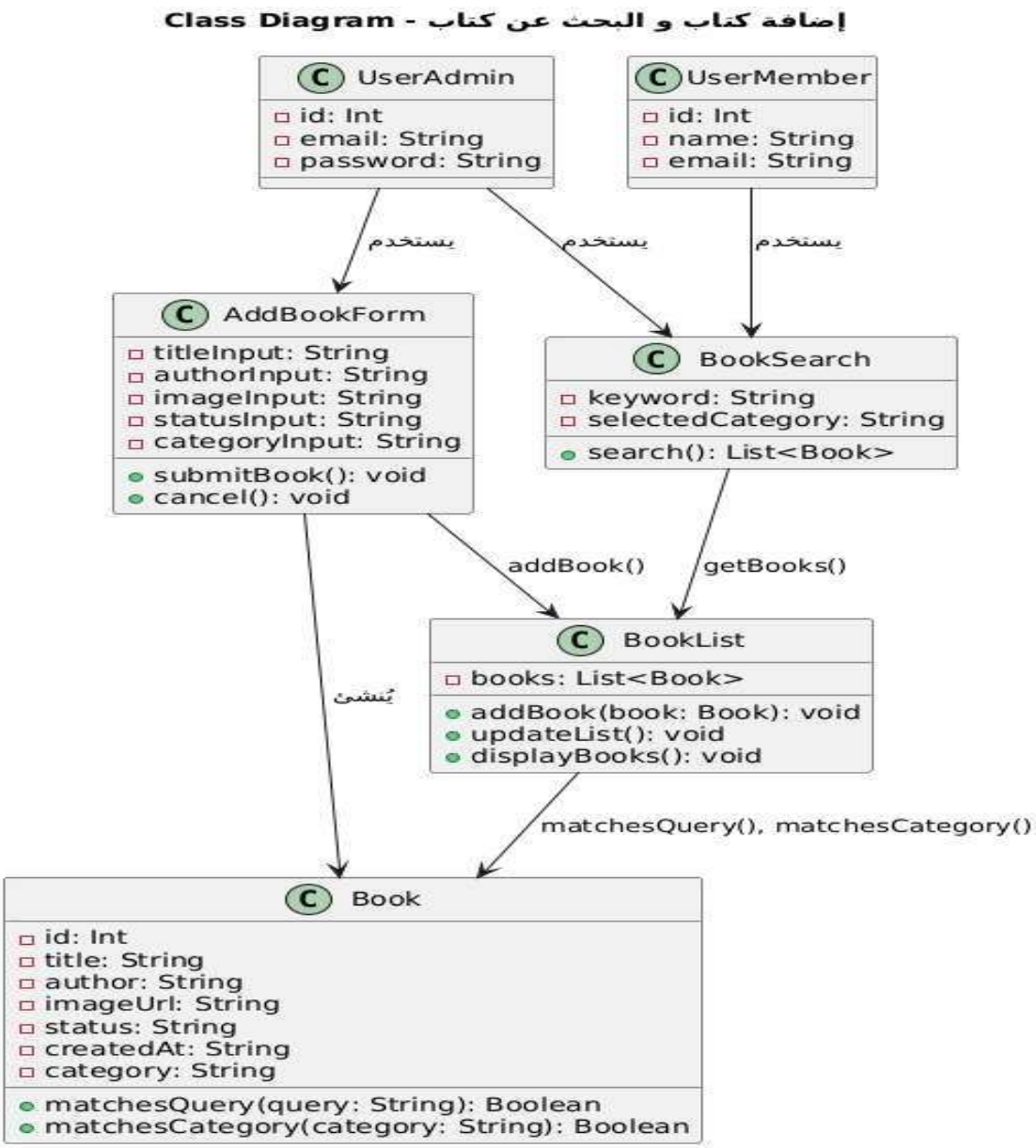
كذلك يتيح النظام لمستخدميه (وخاصة مدير المكتبة أو عضو الإدارة) إمكانية إضافة كتب جديدة إلى قاعدة البيانات، وذلك عبر إدخال معلومات مثل العنوان، المؤلف، التصنيف، والكمية المتاحة. من جهة أخرى، يتيح نفس النظام للأعضاء البحث عن الكتب عبر إدخال كلمات مفتاحية، لتظهر النتائج وفق مطابقة تلك البيانات.

لتمثيل هاتين العمليتين في مخطط الكلاسات، نحتاج إلى إبراز البنية العامة للنظام، وعلاقاته بين الكائنات الأساسية في المخطط.

- **تطبيق المثال في المخطط :**



الشكل 9 : مخطط الفئات (Class Diagram) لنظام إستعارة الكتب



الشكل 10 : مخطط الفئات (Class Diagram) لنظام الإضافة و البحث عن كتاب

5. عرض تصميم جداول قاعدة البيانات وفق MLD :

في هذا الجزء، نقدم النموذج المنطقي للبيانات **(MLD)** الذي يوضح بنية الجداول و تصميمها داخل قاعدة البيانات والعلاقات التي تربطها ببعضها البعض. حيث تم تحديد المفاتيح الأساسية بواسطة **خط تحتي**، بينما تم تمييز المفاتيح الخارجية بواسطة الرمز #، وذلك وفقًا لمنهجية **Merise** المعتمدة في تحليل وتصميم قواعد البيانات.

📄جدول المستخدمين(USERS) :

USERS(id , name, email, password, phone, birth_date, gender, profile_image, created_at, verification_token, is_verified, token_expiry)

📚جدول الكتب(BOOKS) :

BOOKS(id , title, author, image_url, status, created_at, category)

📄جدول طلبات الاستعارة(BORROW_REQUESTS) :

BORROW_REQUESTS(id, user_name, request_date, due_date, status, pdf_file_path, #user_id, #book_id)

✉️جدول رسائل المشرف(ADMIN_MESSAGES) :

ADMIN_MESSAGES(id, message, sent_at, #user_id, #borrow_request_id)

🔔جدول إشعارات المستخدم(USER_NOTIFICATIONS) :

USER_NOTIFICATIONS(id, is_read, created_at, #user_id, message)

خاتمة الفصل

يعد اعتماد منهجية **UML** في تصميم نظام تسيير المكتبة خطوة أساسية تمهد لمرحلة تطوير التطبيق، حيث شكلت المخططات المنجزة في هذا الفصل قاعدة مرجعية واضحة لبناء مكونات النظام رقميا. فكل مخطط ساهم في توضيح جانب معين من سلوك التطبيق أو بنيته، مما يسمح بتحويل المتطلبات النظرية إلى شيفرة برمجية منظمة ومترابطة.

هذه التصاميم ستوجه بشكل مباشر عملية تطوير التطبيق عبر **Jetpack Compose** ، إذ يمكن الاعتماد على مخطط الفئات لتحديد الكيانات وقواعد البيانات، ومخططات الحالات والنشاطات لضبط واجهات المستخدم ومسارات التفاعل، بينما تضمن مخططات التتابع والهيكلة وضوح العلاقات الداخلية وتنظيم الكود بشكل قابل للصيانة والتوسعة.

و إختصارا، فإن هذه المرحلة التصميمية لا تختزل في كونها توثيقا نظريا، بل تمثل أساسا عمليا لبناء تطبيق فعال، متكامل الوظائف، ومطابق لاحتياجات المكتبة .

الفصل الثالث : تطوير و تنفيذ تطبيق تسيير المكتبة

مقدمة الفصل

تعد مرحلة تطوير التطبيق وتنفيذه من أبرز المراحل في دورة حياة إنجاز الأنظمة البرمجية، حيث يتم خلالها تحويل التصاميم والنماذج النظرية التي تم إعدادها في المرحلة السابقة باستخدام منهجية **UML** إلى مكونات عملية وواجهات تفاعلية قابلة للاستخدام من طرف المستفيدين. في هذا الفصل، سيتم التركيز على مختلف الجوانب التقنية التي تدخل في عملية بناء تطبيق تسيير مكتبة المطالعة العمومية **"محمد الطاهر تليلي" بقمار**، وذلك عبر إستعراض الأدوات البرمجية المعتمدة، وشرح مراحل تطوير الواجهات، وربط التطبيق بقاعدة البيانات، إلى جانب تحليل أبرز التحديات التي واجهت الفريق أثناء الإنجاز.

لقد تم اعتماد مجموعة من التقنيات والأدوات البرمجية الحديثة والمناسبة لطبيعة المشروع، من بينها لغة **Kotlin** وبيئة التطوير **Android Studio** لتصميم تطبيق أندرويد بواجهات عصرية مبنية باستخدام **Jetpack Compose**، إلى جانب **MySQL** كقاعدة بيانات علائقية يتم التعامل معها عبر ملفات **PHP** على خادم محلي باستخدام **XAMPP**، مع الاعتماد على مكتبة **Retrofit** مدعومة بـ **Gson** لتمرير البيانات من وإلى الخادم وتحليلها بصيغة **JSON**.

يهدف هذا الفصل إلى تقديم وصف دقيق لكيفية بناء التطبيق، عبر تحليل كل أداة برمجية إستعملت و دورها التقني في تحقيق وظائف التطبيق، مع إرفاق صور و شروحات تطبيقية لتوضيح خطوات التنفيذ.و سيتم عرض الهيكل العام للمشروع، الواجهات التي تم تطويرها،كيفية إدارة البيانات،لتشكيل أفضل منهجية متبعة قبل إعتماد التطبيق نهائيا.

2. بيئة التطوير و الأدوات البرمجية المعتمدة :

1.2. لغة البرمجة Kotlin :

1.1.2. تعريف بلغة Kotlin :

تعد لغة **Kotlin** لغة برمجة حديثة موجّهة أساسا لتطوير تطبيقات الأندرويد، طورتها شركة **JetBrains**، وتم الإعلان عنها لأول مرة سنة **2011**، قبل أن تعتمدها شركة **Google** كلغة رسمية لتطوير تطبيقات أندرويد سنة **2017**. تتميز **Kotlin** بكونها لغة ساكنة الأنماط **(*statically typed*)** وتشتغل بشكل كامل على منصة **JVM Java (Virtual Machine)**، مما يجعلها متوافقة مع مكتبات وأطر **Java** التقليدية، وقابلة للتكامل بسهولة معها.

2.1.2. سبب إعتمادها في تطوير التطبيق :

تم اختيار لغة **Kotlin** لتطوير هذا التطبيق لعدة اعتبارات تقنية، من أبرزها:

- الدعم الرسمي من **Google**، مما يضمن استمرار التحديثات والدعم المجتمعي الواسع.
- السهولة والبساطة في الكتابة مقارنة بلغة **Java**، ما يقلل من عدد الأسطر البرمجية ويزيد من وضوح الكود.
- قابلية التعلّم السريعة، خاصة للمبتدئين والمطورين القادمين من خلفيات **Java**.
- التكامل الكامل مع **Android Studio** و**Jetpack Compose**، مما يسهل التصميم والتطوير في بيئة موحدة.

3.1.2. المميزات التي تقدمها في إطار المشروع :

في سياق تطوير تطبيق تسيير المكتبة، ساهمت لغة **Kotlin** في تحقيق عدة مزايا:

- كتابة كود آمن ومتين، بفضل آليات **Kotlin** في التعامل مع القيم الفارغة **(null safety)**، ما ساهم في تقليل أخطاء التشغيل المتعلقة بالمتغيرات غير المهيأة.

- التكامل مع **Compose UI**، حيث وفرّت **Kotlin** مرونة كبيرة في بناء واجهات المستخدم عبر مكونات **Compose** الحديثة.

- إنتاجية أعلى، إذ سمحت الصياغة المختصرة للغة بإنجاز المهام البرمجية بسرعة أكبر مع الحفاظ على مقروئية الكود.

- سهولة ربط الواجهات بالبيانات، من خلال دعمها الممتاز للبرمجة الكائنية والدوال من الدرجة الأولى ***(first-Class Functions)***.

4.1.2. مثال تطبيقي للغة Kotlin :

الصورة التالية تمثل مثالا واقعيا من كود التطبيق يظهر كيفية استخدام **Kotlin** في تصميم واجهة تسجيل الدخول عبر **Jetpack Compose** :

```kotlin
@Composable
fun LoginScreen(viewModel: LoginViewModel = viewModel()) {
    val email by viewModel.email.collectAsState()
    val password by viewModel.password.collectAsState()

    Column(
        modifier = Modifier
            .fillMaxSize()
            .padding(16.dp),
        verticalArrangement = Arrangement.Center
    ) {
        TextField(
            value = email,
            onValueChange = { viewModel.onEmailChanged(it) },
            label = { Text("Email") }
        )
        Spacer(modifier = Modifier.height(16.dp))
        TextField(
            value = password,
            onValueChange = { viewModel.onPasswordChanged(it) },
            label = { Text("Password") },
            visualTransformation = PasswordVisualTransformation()
        )
        Spacer(modifier = Modifier.height(24.dp))
        Button(onClick = { viewModel.login() }) {
            Text("Login")
        }
    }
}
```

الشكل 11 : كود بلغة Kotlin لواجهة تسجيل الدخول

2.2. بيئة التطوير Android Studio :

1.2.2. تعريف بيئة Android Studio :

يعد **Android Studio** البيئة الرسمية لتطوير تطبيقات نظام التشغيل أندرويد، وهو محرر متكامل **(IDE – Integrated Development Environment)** تم تطويره من قبل شركة **Google** بالاعتماد على منصة **IntelliJ IDEA**.

يوفر هذا المحرر مجموعة شاملة من الأدوات اللازمة لتصميم، تطوير، واختبار التطبيقات، مما يجعله الخيار الأمثل للمطورين في بناء تطبيقات متكاملة وعالية الجودة.

2.2.2. أهمية Android Studio في تطوير التطبيق :

يلعب **Android Studio** دورا محوريا في تسهيل عملية تطوير تطبيقات الأندرويد بفضل ما يوفره من:

- محرر واجهات رسومي **(Layout Editor)** يمكن المطور من تصميم واجهات المستخدم **(UI)** باستخدام السحب والإفلات **(Drag & Drop)** أو من خلال كتابة **XML**.

- دعم قوي لتقنيات **Jetpack Compose**، والتي تسمح ببناء واجهات تفاعلية ديناميكية باستخدام لغة **Kotlin** بطريقة مباشرة وسلسة.

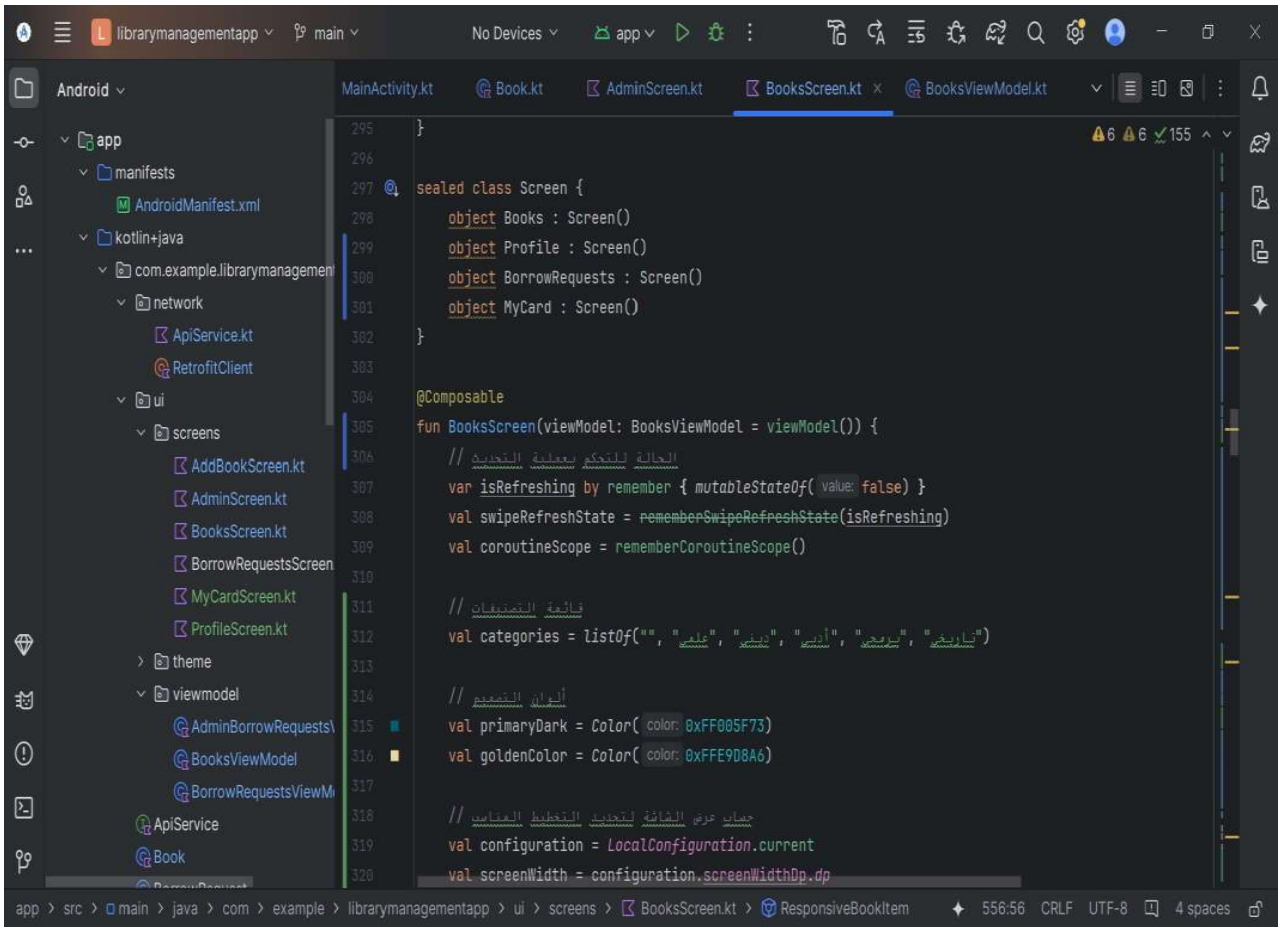
- تكامل تام مع ملفات الكود البرمجي **(Kotlin)**، مما يضمن تزامنا آنيا بين الواجهة الرسومية والمنطق البرمجي للتطبيق.

- أدوات المعاينة **(Preview Tools)** التي تمكن من عرض شكل الواجهة مباشرة دون الحاجة لتشغيل المحاكي أو الجهاز الحقيقي، ما يسرع من وتيرة التطوير.

- نظام إدارة الحزم **Gradle**، الذي يتيح تنظيم الإعتمادات والمكتبات الخارجية وضبط الإعدادات العامة للمشروع.

في إطار مشروع تسيير المكتبة، مكن **Android Studio** من تنظيم ملفات المشروع بشكل هرمي واضح، وتطوير الواجهات باستعمال **Jetpack Compose** بشكل مرن، إلى جانب تسهيل عملية الاختبار والتجريب على مختلف أحجام الشاشات وأنظمة التشغيل.

3.2.2. صورة لواجهة Android Studio أثناء تطوير التطبيق :



الشكل 12 : صورة توضيحية لواجهة بيئة التطوير Android Studio

3.2. إطار العمل Jetpack Compose :

1.3.2. تعريف إطار العمل Jetpack Compose :

تعد **Jetpack Compose** مكتبة حديثة ومفتوحة المصدر من تطوير شركة **Google**، تستخدم لبناء واجهات المستخدم **(UI)** في تطبيقات الأندرويد باستخدام لغة **Kotlin** فقط، دون الحاجة إلى استخدام ملفات **XML** كما كان شائعا في الأساليب التقليدية. تمثل **Compose** ثورة في تطوير الواجهات، حيث تعتمد على البرمجة التصريحية **(Declarative Programming)**، ما يسمح بكتابة واجهات تفاعلية وديناميكية بكود أقل وأكثر مرونة.

2.3.2. أهمية Jetpack Compose في تصميم واجهات المشروع :

تبرز أهمية **Jetpack Compose** من خلال الخصائص التالية:

- التكامل الكامل مع لغة **Kotlin**، مما يجعل عملية تطوير الواجهة منسجمة مع منطق التطبيق البرمجي.
- إمكانية إنشاء واجهات مرنة وتفاعلية بسهولة دون الحاجة إلى فصل الواجهة عن الكود.
- تقليل عدد الأسطر البرمجية وتفادي التعقيدات المرتبطة بإدارة ملفات **XML**.
- دعم معاينة مباشرة للواجهة **(Live Preview)** داخل **Android Studio**، مما يسرع من اختبار التصميم وتعديله.
- سهولة إعادة الاستخدام **(Reusability)** للمكونات عبر بناء عناصر واجهة مخصصة**(Composable Function)**

في مشروع تسيير المكتبة، استخدمت مكتبة **Jetpack Compose** لتصميم جميع واجهات التطبيق، مثل شاشة تسجيل الدخول، الصفحة الرئيسية، تفاصيل الكتب، وقائمة الإعارات، ما أضفى لمسة عصرية وسهلّ إدارة الحالة **(State Management)** داخل كل واجهة.

3.3.2. مقارنة بين Jetpack Compose و XML :

الجانب المقارن	Jetpack Compose	XML
لغة التطوير	Kotlin فقط	XML + KOTLIN
البرمجة	تصريحية (Declarative)	أمرية (Imperative)
سهولة القراءة	تنظيم و سهولة أكثر	تنظيم و سهولة أقل
عدد الملفات	قليلة حيث نجد واجهة كود موحد	كثيرة (ملفات XML منفصلة)
التفاعل مع البيانات	مباشر و سلس	يتطلب ربط يدوي (ViewBinding)

الجدول 2 : مقارنة بين JetpackCompose و XML

هذه المقارنة توضح كيف أن **JetpackCompose** تبسط عملية تطوير الواجهة، مما يجعلها الأنسب لتطبيقات حديثة مثل هذا المشروع.

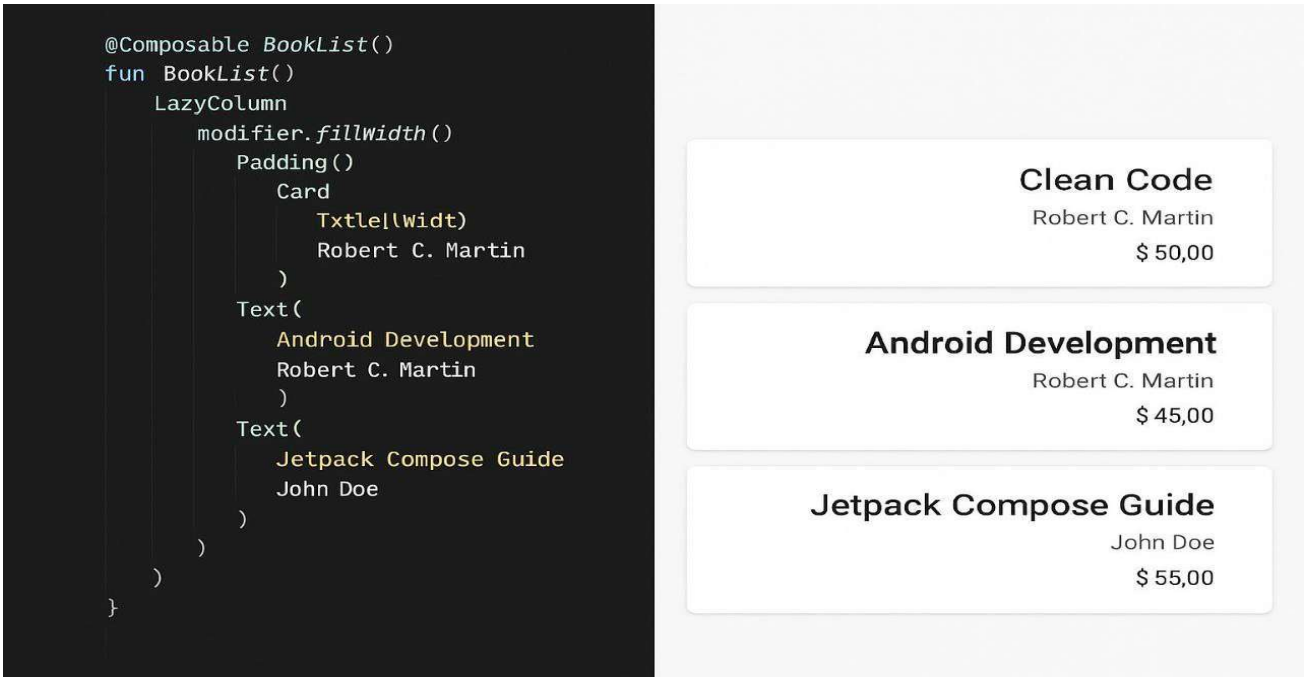
4.3.2. أمثلة من واجهات التطبيق المطورة ب Jetpack Compose :

في هذا المشروع، تم بناء عدة شاشات باستخدام **JetpackCompose**، مثل:

- واجهة تسجيل الدخول **(LoginScreen)**
- واجهة عرض الكتب المتاحة **(BooksListScreen)**
- واجهة تفاصيل الكتاب وطلب الإعارة **(BookDetailScreen)**
- واجهة الإدارة **(AdminPanelScreen)**

يتم في كل واجهة إستخدام مكونات مثل **Image ,LazyColumn ,Column ,Button ,Text**، مع تنظيمها داخل دوال **@Composable** لتسهيل التعديل والتوسع مستقبلا.

5.2.2. صورة لواجهة مطورة بإستخدام Jetpack Compose :



شكل 13 : صورة لواجهة عرض كتب مطورة بإستخدام Jetpack Compose

4.2. قاعدة البيانات MySQL :

تعد **MySQL** واحدة من أشهر نظم إدارة قواعد البيانات العلائقية **(Relational Database Management Systems – RDBMS)**، وهي مفتوحة المصدر وتتميز بالقوة، الاستقرار، وسهولة الاستخدام، ما يجعلها خيارا مثاليا لمختلف التطبيقات البرمجية، خاصة تلك التي تعتمد على تخزين ومعالجة البيانات بطريقة منظمة وقابلة للتوسع.

1.4.2. تعريف قاعدة البيانات MySQL :

MySQL هي نظام لإدارة قواعد البيانات يستخدم لغة **SQL** إختصار **(Structured Query Language)** من أجل تنظيم البيانات، تنفيذ الاستعلامات، وضمان التكامل المنطقي والعلائقي بين الجداول. تدعم **MySQL** العديد من المزايا، كالأمان، القيود، الفهارس، والاستعلامات المركبة، ما يجعلها مناسبة لتطبيقات إدارة المعلومات، مثل نظام تسيير مكتبة.

2.4.2. أسباب إختيار MySQL بدلا من FireBase :

رغم انتشار **Firebase** كحل سحابي لتخزين البيانات في تطبيقات الأندرويد، تم الاعتماد على **MySQL** للأسباب التالية:

- سهولة الاستضافة المحلية باستخدام أدوات مثل **XAMPP**، ما يسمح باختبار النظام دون الحاجة إلى اتصال دائم بالإنترنت.
- إمكانية التحكم الكامل في بنية الجداول والاستعلامات وقواعد البيانات.
- المرونة العالية في التعامل مع البيانات العلائقية، وهي ضرورية في حالة المكتبة، مثل الربط بين جدول الكتب وجدول المستخدمين وجدول الإعارات.
- مجانية التكوين المحلي وعدم الحاجة لحسابات أو اشتراكات إضافية.

3.4.2. صورة توضيحية لإستعمال قاعدة البيانات MySQL :

```sql
CREATE TABLE `borrow_requests` (
  `id` int(11) NOT NULL,
  `user_id` int(11) NOT NULL,
  `book_id` int(11) NOT NULL,
  `user_name` varchar(100) NOT NULL,
  `request_date` timestamp NOT NULL DEFAULT current_timestamp(),
  `due_date` timestamp NULL DEFAULT NULL,
  `status` enum('طلب مرفوض','طلب مقبول','قيد الانتظار') NOT NULL DEFAULT 'قيد الانتظار',
  `pdf_file_path` varchar(255) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

الشكل 14 : صورة بنية جدول طلبات الإعارة في قاعدة البيانات بإستعمال MySQL

```sql
INSERT INTO `borrow_requests` (`id`, `user_id`, `book_id`, `user_name`, `request_date`,
`due_date`, `status`, `pdf_file_path`) VALUES
(92, 161, 1, 'okba', '2025-05-08 20:42:31', '2025-05-23 05:42:31', 'طلب مقبول', NULL),
(94, 161, 22, 'okba', '2025-05-08 20:43:17', '2025-05-23 05:43:17', 'قيد الانتظار', NULL),
(95, 161, 35, 'okba', '2025-05-08 20:43:22', '2025-05-23 05:43:22', 'قيد الانتظار', NULL),
(97, 156, 1, 'قيد الانتظار' ,'21:01:19 23-05-2025' ,'12:01:19 09-05-2025' ,'محمد إسلام', NULL),
(98, 162, 22, 'zakaria', '2025-05-09 12:11:51', '2025-05-23 21:11:51', 'قيد الانتظار', NULL),
(99, 162, 23, 'zakaria', '2025-05-09 12:11:55', '2025-05-23 21:11:55', 'قيد الانتظار', NULL);
```

الشكل 15 : صورة إضافة طلبات إعارة بواسطة MySQL

5.2. الخادم المحلي XAMPP :

1.5.2. مقدمة عن XAMPP :

يعد **XAMPP** من بين الحلول البرمجية المفتوحة المصدر الأكثر شيوعا لإنشاء بيئة تطوير محلية، حيث يوفر حزمة متكاملة تضم مجموعة من الخدمات الأساسية لتطوير تطبيقات الويب. يتكونّ **XAMPP** أساساً من:

- **Apache**: خادم ويب **(Web Server)** يستخدم لتشغيل تطبيقات **PHP** محليا.
- **MySQL/MariaDB**: نظام إدارة قواعد بيانات علائقية يستخدم لتخزين البيانات واسترجاعها.
- **PHP**: لغة برمجة من جهة الخادم تستخدم لإنشاء سكريبتات لمعالجة البيانات والربط مع قاعدة البيانات.
- **phpMyAdmin**: واجهة رسومية لإدارة قواعد بيانات **MySQL** بطريقة سهلة وسلسة.

2.5.2. دور XAMPP في المشروع :

في إطار هذا التطبيق، تم اعتماد **XAMPP** لإنشاء خادم محلي يربط تطبيق الأندرويد بقاعدة بيانات **MySQL** عبر ملفات **PHP**. فعوض استخدام خدمات سحابية مثل **Firebase**، تم تفضيل **XAMPP** لعدة اعتبارات أهمها:

- العمل دون الحاجة إلى اتصال دائم بالإنترنت أثناء مرحلة التطوير.
- تحكم كامل في إعدادات الخادم وقواعد البيانات.
- إمكانية اختبار وربط الوظائف بين الواجهة الأمامية **(تطبيق الهاتف)** والخلفية **(الخادم المحلي)** بطريقة مرنة.

3.5.2. كيفية إستخدام XAMPP في المشروع :

تشغيل الخادم: يتم فتح لوحة تحكم **XAMPP** وتفعيل خدمتي **Apache** و **MySQL**.

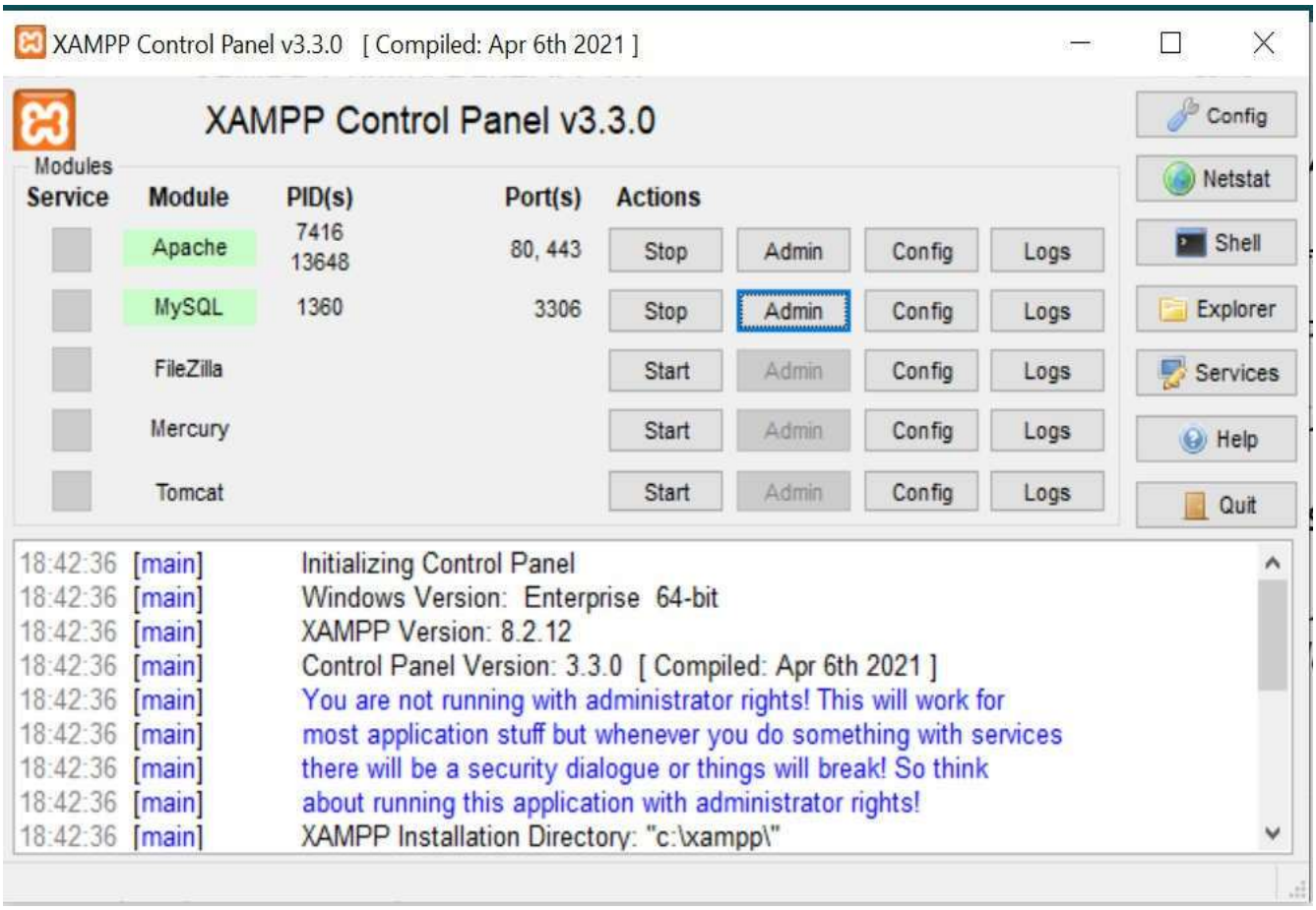
إدارة قاعدة البيانات: عبر **phpMyAdmin**، تم إنشاء قاعدة بيانات **"library_db"** وإعداد الجداول الخاصة بالكتب، المستخدمين، والإعارات.

ربط مع التطبيق: تم تطوير ملفات **PHP** داخل مجلد **htdocs**، تقوم هذه الملفات باستقبال الطلبات من تطبيق الهاتف **(عبر Retrofit)**، ومعالجتها باستخدام استعلامات **SQL**، ثم إرسال النتائج بصيغة **JSON**.

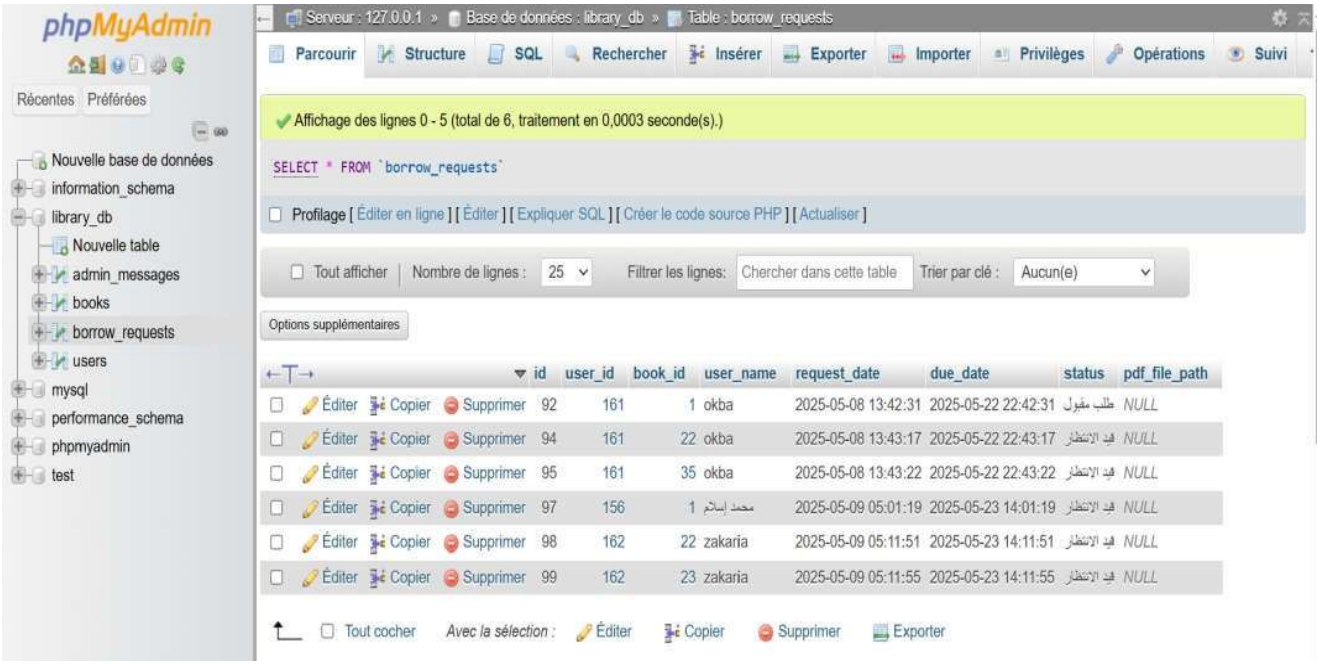
4.5.2. أهمية XAMPP في التطبيق :

- لعب دورا محوريا كحلقة وصل بين تطبيق الهاتف وبيانات المستخدمين والكتب.
- ساعد على اختبار سيناريوهات متعددة قبل نشر النظام.
- مكن الفريق من العمل بمرونة دون الحاجة إلى خادم خارجي مدفوع.

5.5.2. صورة توضيحية لإستعمال XAMPP في المشروع :



الشكل 16 : صورة تشغيل Apache و MySQL في لوحة تحكم XAMPP



الشكل 17 : صورة واجهة phpMyAdmin لجدول طلبات الإعارة في قاعدة البيانات

6.2. لغة PHP :

1.6.2. مقدمة عن PHP :

تعد لغة **PHP** (اختصارا لـ ***Hypertext Preprocessor***) من أبرز لغات البرمجة المستخدمة في تطوير تطبيقات الويب من جهة الخادم (**Server Side**). وهي لغة مفتوحة المصدر، تمتاز بسهولة التعلم، السرعة في التنفيذ، والقدرة على التكامل السلس مع قواعد البيانات، خاصة **MySQL**.

2.6.2. دور PHP Backend في التطبيق :

في إطار مشروع تطوير تطبيق تسيير المكتبة، استخدمت لغة **PHP** لتطوير واجهات برمجية (**APIs**) تعمل كوسيط بين تطبيق الأندرويد وقاعدة البيانات المخزنة محليا على خادم **XAMPP**. هذه الواجهات البرمجية تعالج الطلبات القادمة من التطبيق (مثل تسجيل الدخول، جلب الكتب، إرسال طلب إعارة، إلخ)، وتنفيذ الاستعلامات على قاعدة البيانات، ثم ترسل الرد بصيغة **JSON**.

وقد تم اختيار **PHP** نظراً لسهولة استخدامها مع **MySQL**، ولقدرتها على إنشاء واجهات برمجية (APIs) مرنة باستخدام بروتوكول **HTTP**، مما يسمح للتطبيق بالتواصل مع الخادم محلياً أو عن بُعد.

كما تم اعتماد **تنسيق JSON** في تبادل البيانات بين التطبيق والخادم، لما يتميز به من خفة وسهولة في المعالجة داخل بيئة **Android**.

تمت كتابة ملفات PHP بشكل يحافظ على فصل منطق التطبيق (Business Logic) عن الواجهة، مما يسهّل لاحقاً عملية الصيانة أو التوسعة مستقبلاً، مثل إضافة صلاحيات، أو تتبع سجلات الاستخدام.

3.6.2. الأسباب التقنية لإختيار PHP في التطبيق :

- التكامل الكامل مع **MySQL**، وسهولة تنفيذ الاستعلامات.
- الدعم الواسع من مجتمع المطورين والمصادر التعليمية.
- إمكانية استضافة الملفات محليا باستخدام **XAMPP** بدون تعقيدات.
- سرعة التطوير والمرونة في إدارة البيانات.

4.6.2. مثال تطبيقي عن PHP في التطبيق :

تم إنشاء ملف **login.php** داخل مجلد **htdocs** في **XAMPP**. هذا الملف يستقبل بيانات المستخدم (البريد وكلمة السر) من التطبيق عبر بروتوكول **POST**، ثم يتحقق من صحتها في قاعدة البيانات ويعيد نتيجة على شكل **JSON**.

```php
<?php
$host = "localhost";
$user = "root";
$password = "";
$dbname = "library_db";
$conn = new mysqli($host, $user, $password, $dbname);

if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    $email = $_POST['email'];
    $password = $_POST['password'];

    $query = "SELECT * FROM users WHERE email='$email' AND password='$password'";
    $result = $conn->query($query);

    if ($result->num_rows > 0) {
        echo json_encode(["success" => true, "message" => "Login successful"]);
    } else {
        echo json_encode(["success" => false, "message" => "Invalid credentials"]);
    }
}
?>
```

الشكل 18 : صورة لكود ملف Login.php

7.2. أداة Ngrok :

1.7.2. مقدمة عن أداة Ngrok :

تعد **Ngrok** من الأدوات المساعدة الهامة في تطوير واختبار التطبيقات التي تتطلب اتصالاً بخوادم محلية من أجهزة خارجية، مثل تطبيقات الهاتف المحمول. وهي أداة مفتوحة المصدر تسُتخدم لإنشاء نفق آمن بين الجهاز المحلي وشبكة الإنترنت، مما يسمح للمطورين بكشف تطبيقاتهم أو واجهاتهم البرمجية **(APIs)** المستضافة على بيئة محلية مثل **XAMPP** إلى الإنترنت مؤقتا، دون الحاجة إلى نشر الخادم فعليا.

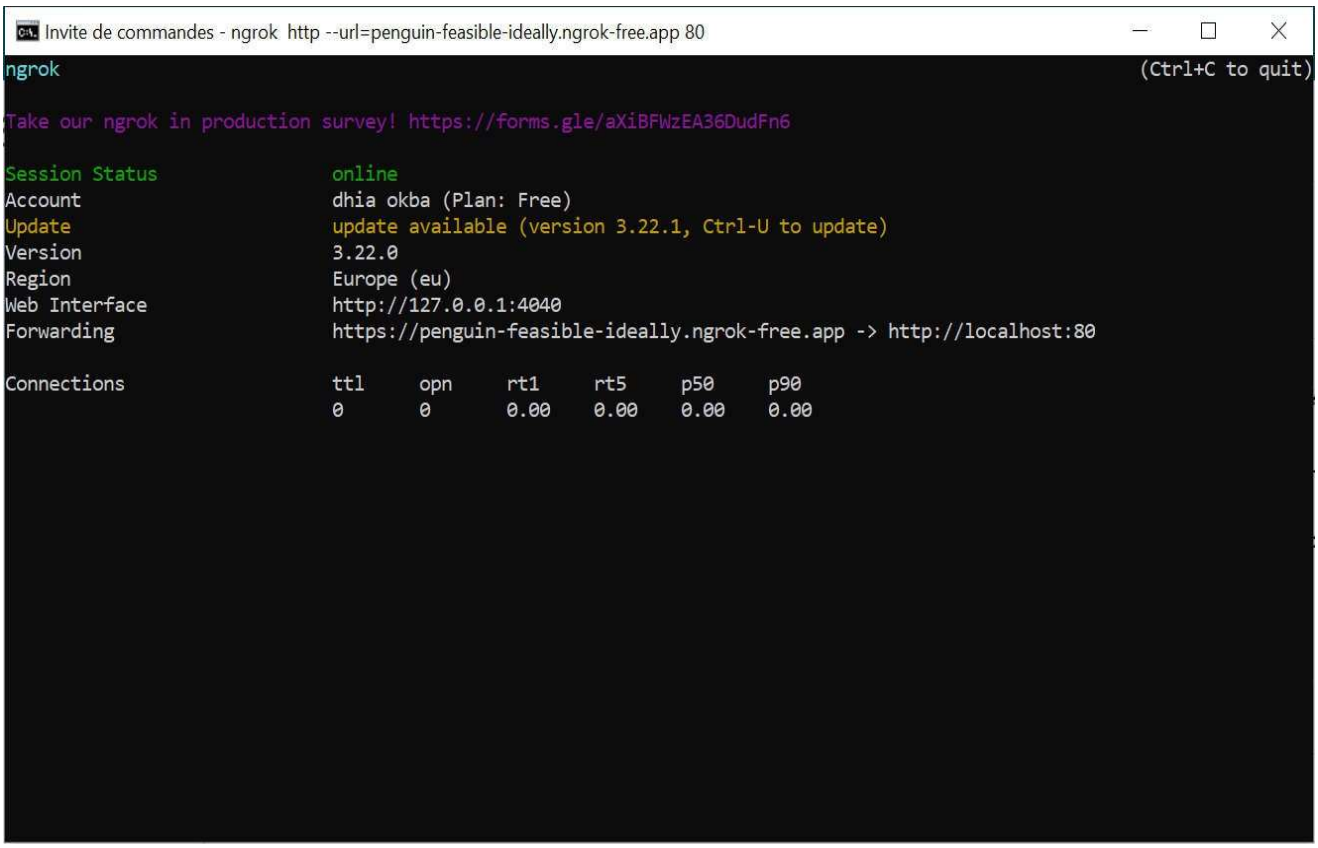
2.7.2. الدور التقني لأداة Ngrok في المشروع :

في سياق تطوير تطبيق تسيير المكتبة، استعملت أداة **Ngrok** لتمكين تطبيق الأندرويد من الوصول إلى ملفات **PHP** المستضافة على **XAMPP** في جهاز المطور، حيث لا يمكن للتطبيق، بطبيعته، الوصول إلى العناوين المحلية (مثل **localhost**) مباشرة من الهاتف أو من جهاز افتراضي (**Emulator**). لذلك، تقوم **Ngrok** بإنشاء عنوان **URL** عام وآمن (عادةً يبدأ بـ **https://**) يسُتخدم داخل التطبيق لتمرير البيانات إلى الواجهات البرمجية المكتوبة بلغة **PHP**.

3.7.2. مزايا أداة Ngrok :

- **سهولة الربط:** لا يتطلب إعدادات معقدة، فقط أمر بسيط في الطرفية.
- **الأمان:** يتم إنشاء اتصال عبر **HTTPS** بشكل آلي.
- **المرونة في الاختبار:** يمكنّ من اختبار التطبيق كما لو كان متصلا بسيرفر حقيقي على الإنترنت.
- **توفير الوقت:** يجنب المطور إعداد خادم خارجي خلال مرحلة التطوير.

4.7.2. نافذة أوامر Ngrok :



الشكل 19 : صورة توضح نافذة أوامر Ngrok لربط الخادم المحلي بالإنترنت

3. عرض هيكل المشروع (Project Structure) :

يعد هيكل المشروع **(Project Structure)** من العناصر الأساسية في تنظيم شفرة المصدر الخاصة بالتطبيق، حيث يساهم في تسهيل عملية التطوير، تحسين قابلية الصيانة، وتوزيع المسؤوليات بين مكونات التطبيق. يعتمد تنظيم المشروع بشكل سليم على مبدأ الفصل بين المسؤوليات **(Separation of Concerns)**، مما يجعل كل مكون مسؤولا عن وظيفة محددة.

في تطبيق تسيير المكتبة، تم تنظيم هيكل المشروع داخل بيئة **Android Studio** وفقا لتقسيم معماري مبسط، يراعي وضوح التخصصات لكل مجلد، بما في ذلك واجهات المستخدم، البيانات، الخدمات الشبكية، والمصادر.

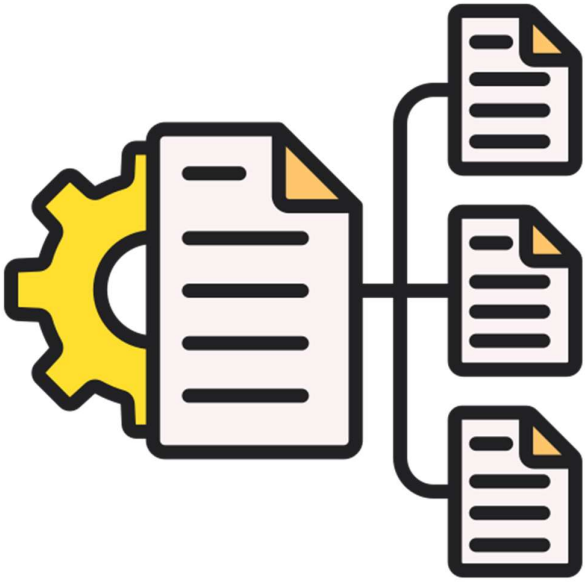
1.3. أهم المجلدات في هيكل المشروع :

- **/ui**: يحتوي على جميع ملفات الواجهات الرسومية المبنية بـ **Jetpack Compose**، مثل شاشات عرض الكتب، تسجيل الدخول، والصفحة الرئيسية.

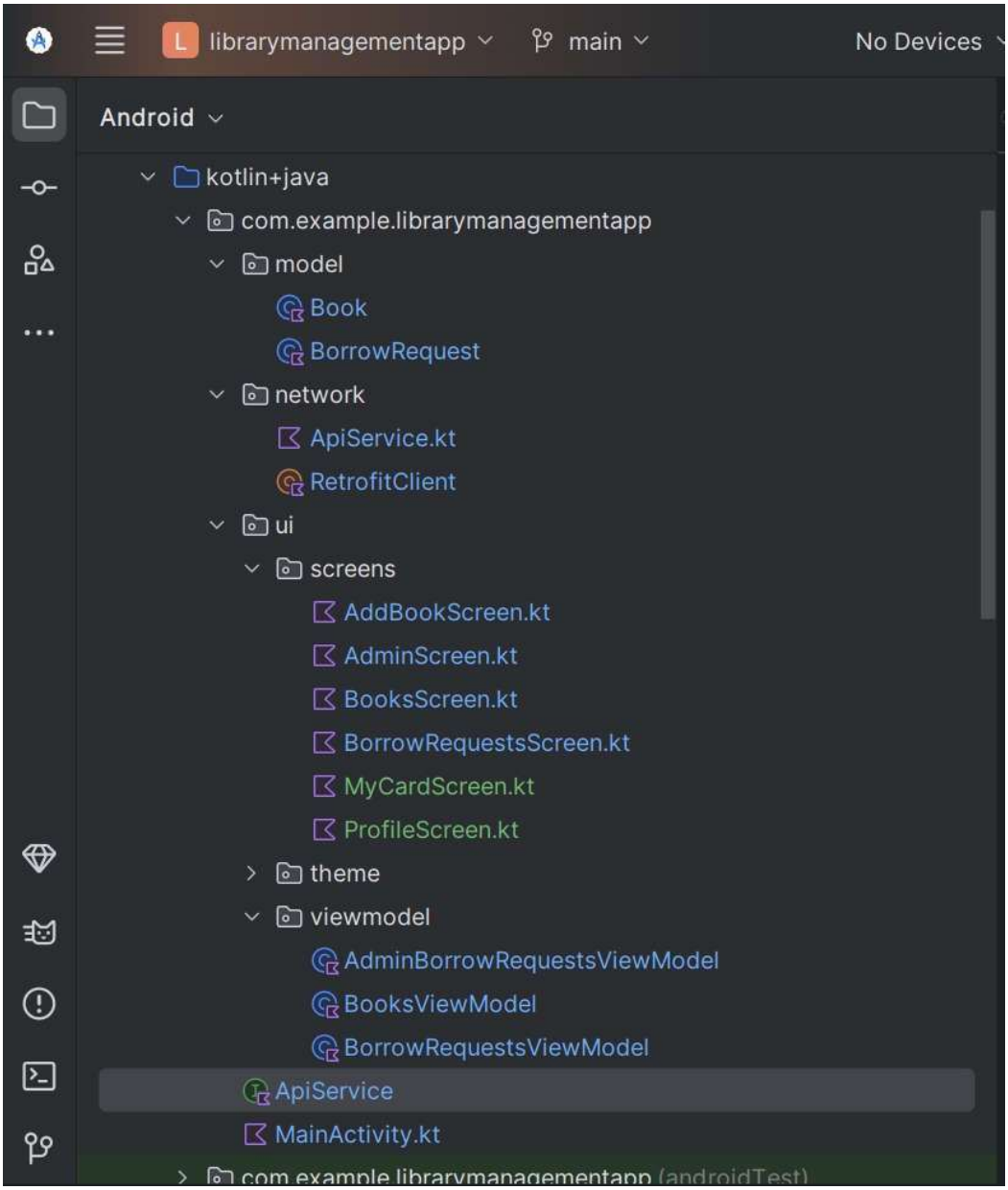
- **/model**: يتضمن أصناف البيانات **(Data Classes)** التي تمثل الكيانات الأساسية في النظام، مثل **Book.kt, BorrowRequest.kt...** وهي بمثابة نماذج للبيانات المتبادلة بين واجهات المستخدم وقاعدة البيانات.

- **/network**: يحتوي على الأصناف المسؤولة عن الاتصال بالخادم عبر بروتوكول **HTTP**، باستخدام **Retrofit**.

- **/viewmodel**: يحتوي على الأصناف من نوع **ViewModel**، وهي طبقة منطقية وسيطة بين واجهة المستخدم وطبقة البيانات. يتم فيها تخزين الحالة الحالية للواجهة **(state)**، وتنفيذ المعالجة الأولية للبيانات المستلمة من قاعدة البيانات قبل إرسالها إلى واجهة المستخدم. على سبيل المثال، عند طلب قائمة الكتب من الخادم، يتم استدعاءها من **ViewModel**، وتحضيرها للعرض داخل الشاشة.



2.3. صورة توضيحية للمجلدات في هيكل المشروع :



الشكل 20 : صورة لشجرة هيكل مجلدات المشروع في Android Studio

4. تنفيذ الواجهات الأساسية للمشروع :

تمثل واجهات المستخدم **(User Interfaces)** الجزء الظاهر من التطبيق والذي يتفاعل معه المستخدم النهائي. ويعد تصميم الواجهات عملية جوهرية لضمان تجربة مستخدم مريحة وسلسة، حيث تم الاعتماد في هذا المشروع على مكتبة **Jetpack Compose** لتطوير واجهات تفاعلية تعتمد على مبدأ التصميم التفاعلي **(Declarative UI)**.

1.4. واجهة تسجيل الدخول :

تعتبر واجهة تسجيل الدخول من أهم الواجهات في تطبيق تسيير المكتبة، إذ تشكل البوابة الرئيسية التي تمكن المستخدم من الوصول إلى محتوى التطبيق وخدماته. تم تصميم هذه الواجهة وفقاً لمعايير تجربة المستخدم، مع التركيز على البساطة، الأمن، وسهولة التفاعل.

1.1.4. المكونات البرمجية المستعملة :

- **TextField**: لاستقبال مدخلات المستخدم مثل البريد الإلكتروني وكلمة المرور، مع تخصيص الخصائص مثل **placeholder, label**, وإخفاء كلمة المرور باستخدام **visualTransformation**.

- **Button**: زر تنفيذ العملية، مرفق بنص توضيحي (مثل "تسجيل الدخول")، ومربوط بدالة تنفيذية في **ViewModel** للتحقق من صحة المعلومات والاتصال بالخادم.

- **Text** و **Spacer**: تم استخدام هذه العناصر لتحسين تنظيم وتنسيق محتوى الشاشة.

- **Snackbar** أو **Toast** *(عند التنفيذ)*: لإظهار رسائل فورية للمستخدم في حالة نجاح أو فشل عملية تسجيل الدخول.

2.1.4. الجوانب التقنية :

- التحقق من صحة البيانات **(Validation)**: قبل إرسال المعلومات إلى الخادم، يتم التحقق من صحة تنسيق البريد الإلكتروني وطول كلمة المرور داخل **ViewModel**.

- الاتصال بالخادم **(Backend Integration)**: بعد الضغط على الزر، يتم إرسال البيانات إلى ملف **PHP** موجود على الخادم المحلي **(XAMPP)**، باستخدام بروتوكول **HTTP**. يستقبل الملف البيانات ويتحقق من صحة الحساب باستخدام قاعدة بيانات **MySQL**.

- الإدارة بالحالة **(State Management)**: يتم استخدام **ViewModel** لتخزين الحالة الحالية للمدخلات والردود من الخادم، مما يسمح بتحديث الواجهة تلقائيا عند تغير البيانات.

3.1.4. تنفيذ الواجهة :

شكل 21 : واجهة تسجيل الدخول في تطبيق تسيير المكتبة

2.4. واجهة إنشاء حساب :

تعد واجهة إنشاء الحساب من الواجهات الأساسية في تطبيق تسيير المكتبة، حيث تتيح للمستخدمين الجدد التسجيل والوصول إلى خدمات التطبيق بطريقة آمنة ومنظمة. تم تصميم هذه الواجهة اعتمادا على مبدأ التصميم التفاعلي الذي توفره مكتبة **Jetpack Compose**، بما يتوافق مع معايير تجربة المستخدم وأمن المعلومات.

1.2.4. المكونات البرمجية المستعملة :

- **TextField**: لاستقبال معلومات المستخدم، مثل الاسم الكامل، البريد الإلكتروني، اسم المستخدم، وكلمة المرور. تم تخصيص هذه الحقول بعناصر مرئية مناسبة لتوضيح الغرض منها، مع استعمال **label** و**placeholder**.

- **PasswordField**: حقل خاص بكلمة المرور، مزود بخاصية الإخفاء (**visualTransformation**) للحفاظ على خصوصية المستخدم.

- **Button**: زر تنفيذ عملية التسجيل، مرتبط بدالة معالجة في **ViewModel** تقوم بإرسال المعلومات إلى الخادم للتحقق من صحتها وتخزينها.

- **Snackbar** أو **Text**: لعرض رسائل تنبيهية بعد محاولة التسجيل، توضح للمستخدم حالة العملية (نجاح أو خطأ).

2.2.4. الجوانب التقنية :

- التحقق من البيانات **(Input Validation)**: قبل الإرسال، يتم التحقق من تطابق كلمات المرور، وصحة تنسيق البريد الإلكتروني، وعدم ترك الحقول فارغة.

- الربط بالخادم **(Backend Integration)**: ترسل البيانات عبر بروتوكول **HTTP** إلى ملف **PHP** على خادم **XAMPP**. يقوم هذا الأخير بالتحقق من وجود الحساب مسبقا، ثم إدراج المستخدم الجديد في قاعدة البيانات **MySQL** إذا كانت البيانات صالحة.

- إدارة الحالة **(State Handling)**: تم استخدام **ViewModel** لإدارة حالة المدخلات وردود الخادم، مما يسمح بتفاعل فوري بين واجهة المستخدم ومعالجة البيانات.

3.2.4. تنفيذ الواجهة :

الشكل 22 : واجهة إنشاء حساب في تطبيق تسيير المكتبة

3.4. واجهة عرض الكتب :

تعتبر واجهة عرض الكتب من الواجهات المحورية في تطبيق تسيير مكتبة المطالعة العمومية، حيث تتيح للمستخدمين تصفح قائمة الكتب المتوفرة بسهولة وبتنظيم بصري جذآب. تم تطوير هذه الواجهة باستخدام مكتبة **Jetpack Compose** التي تسمح ببناء عناصر واجهة تفاعلية وديناميكية، وتستمد بياناتها مباشرة من قاعدة البيانات عبر واجهات برمجية (**APIs**) مكتوبة بلغة **PHP**.

1.3.4. المكونات البرمجية المستعملة :

- **LazyColumn / LazyVerticalGrid**: لعرض قائمة الكتب بشكل قابل للتمرير، سواء في شكل قائمة عمودية أو شبكة **Grid**، حسب التصميم المطلوب.

- **Card**: تستخدم لاحتواء كل عنصر كتاب، وتعرض معلومات مثل صورة الغلاف، عنوان الكتاب، اسم المؤلف، وسنة النشر، مع تنسيق مرئي متناسق.

- **Image**: تستخدم لعرض صورة غلاف الكتاب، مع تحميلها من الخادم باستخدام مكتبة Coil لتحميل الصور من الإنترنت بطريقة سلسة.

- **Text**: لعرض معلومات الكتاب بتنسيق أنيق ومقروء.

- **ViewModel** : لاستدعاء دالة جلب البيانات من الخادم وتحويلها إلى قائمة كتب جاهزة للعرض في الواجهة.

2.3.4. الجوانب التقنية :

- **جلب البيانات من الخادم:** يتم التواصل مع ملف **PHP** عبر بروتوكول **HTTP** باستخدام مكتبة **Retrofit**، حيث يرسل الطلب ويستقبل الرد بصيغة **JSON**، ثم يتم تحويله إلى كائنات **Book** باستخدام محول **Gson**.

- **التفاعل مع البيانات الحية (LiveData / State):** يتم عرض البيانات بشكل لحظي عبر خاصية **mutableStateListOf( )** أو **StateFlow**، مما يضمن استجابة الواجهة عند تغير البيانات.

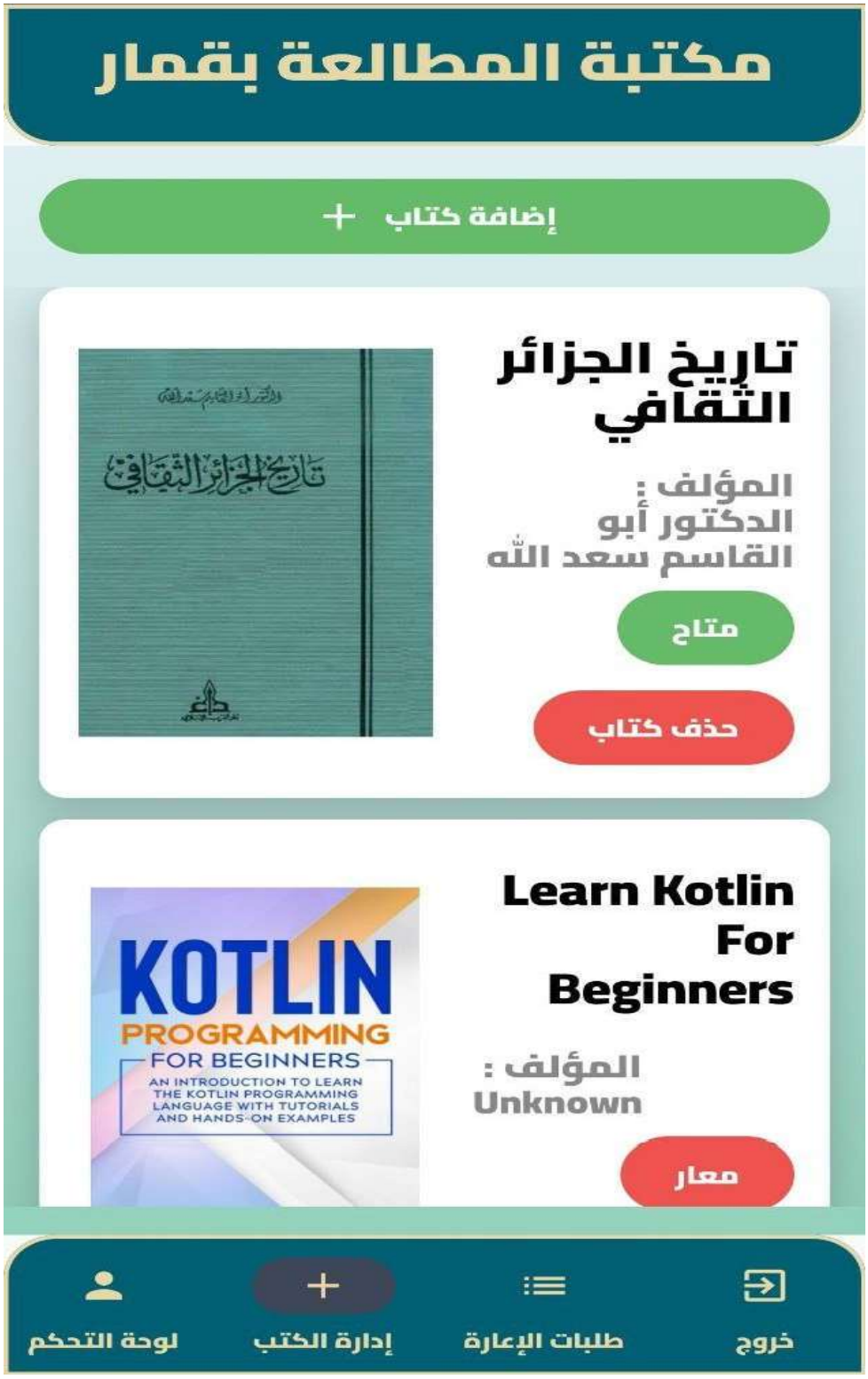
- **تحميل الصور ديناميكيا:** يتم استخدام **Coil** لتحميل الصور من روابط يتم توفيرها من الخادم، مع دعم الصور البديلة في حال تعذر التحميل.

3.2.4. تنفيذ الواجهة :

- واجهة عرض الكتب بالنسبة لعضو المكتبة :

الشكل 23 : واجهة عرض الكتب بالنسبة لعضو المكتبة

- واجهة عرض الكتب بالنسبة لعضو إدارة المكتبة :



الشكل 24 : واجهة عرض الكتب بالنسبة لعضو إدارة المكتبة

4.4. واجهات طلبات الإعارة :

1.4.4. واجهة طلبات الإعارة بالنسبة لعضو المكتبة :

تعد هذه الواجهة جزءا محوريا من تجربة المستخدم في التطبيق، حيث تتيح له متابعة جميع طلبات الإعارة التي قام بها سابقا. توفر الواجهة عرضا مفصلا لحالة كل طلب
(قيد المعالجة، مقبول، مرفوض)، إلى جانب معلومات أساسية حول الكتاب وتاريخ الإعارة.

_ المكونات البرمجية المستعملة :

- **LazyColumn** لعرض الطلبات في قائمة قابلة للتمرير.
- عناصر **Card** تحتوي على:
  - ❑ عنوان الكتاب.
  - ❑ تاريخ الطلب.
  - ❑ حالة الطلب (مرفقة بلون مميز أو أيقونة).
- آلية جلب البيانات من الخادم عبر استدعاء **HTTP** باستخدام **Retrofit** أو طريقة أخرى مناسبة.

_ الأهداف الوظيفية :

- تمكين المستخدم من الاطلاع على كافة طلباته السابقة.
- عرض حالة الطلب بشكل واضح.
- تسهيل التفاعل والمتابعة دون الحاجة للحضور الفعلي إلى المكتبة.

2.4.4. تنفيذ الواجهة :

الشكل 25 : واجهة طلبات الإعارة بالنسبة لعضو المكتبة

3.4.4. واجهة طلبات الإعارة بالنسبة لعضو إدارة المكتبة :

من جهة الإدارة، توفر هذه الواجهة أداة عملية لمراقبة الطلبات الواردة من أعضاء المكتبة والرد عليها بالقبول أو الرفض. تمثل هذه الواجهة الجسر بين النظام اليدوي التقليدي والإدارة الرقمية الحديثة لمكتبات الإعارة.

_ المكونات البرمجية المستعملة :

- **LazyColumn** لعرض قائمة الطلبات .

- عناصر **Card** تحتوي على:
  - ❑ إسم العضو .
  - ❑ عنوان الكتاب.
  - ❑ تاريخ الطلب.
  - ❑ زر تغيير الحالة (مرفقة بلون مميز للحالة).

- إرسال القرار إلى الخادم بإستخدام طلبات HTTP نحو ملف PHP مخصص لمعالجة الردود .

_ الأهداف الوظيفية :

- تمكين المسؤول من مراجعة كل طلبات الإعارة المستلمة.
- توفير واجهة تفاعلية لاتخاذ قرارات القبول أو الرفض.
- تحديث حالة الطلب مباشرة في قاعدة البيانات وإشعار المستخدم بذلك.

4.4.4. تنفيذ الواجهة :

الشكل 26 : واجهة طلبات الإعارة بالنسبة لعضو إدارة المكتبة

5. ربط الواجهات بالوظائف البرمجية :

تمثل عملية ربط الواجهات الرسومية للتطبيق بالوظائف البرمجية الخلفية **(Back-End)** عنصرا أساسيا في نجاح أي نظام رقمي، حيث يتم من خلالها تمكين التطبيق من التفاعل مع البيانات المخزنة على الخادم ومعالجتها في الوقت الفعلي. في مشروع تسيير مكتبة ، تم اعتماد لغة **PHP** كوسيط للتعامل مع قاعدة بيانات **MySQL**، ما يسمح بإنشاء واجهات برمجية **(APIs)** يستدعيها التطبيق عند تنفيذ عمليات معينة كالتسجيل، تسجيل الدخول، طلب إعارة كتاب، وغيرها.

1.5. آلية الربط بين الواجهة و MySQL عبر PHP :

عند تفاعل المستخدم مع واجهة معينة (مثلا واجهة إنشاء حساب جديد)، يتم إرسال البيانات المدخلة **(كالاسم، البريد الإلكتروني، كلمة المرور...)** من الواجهة الأمامية **(Frontend)** نحو خادم محلي بواسطة **HTTP POST** نحو ملف **PHP** محدد. يقوم هذا الأخير بمعالجة البيانات وإجراء العمليات المطلوبة على قاعدة البيانات، ثم يرسل استجابة **(Response)** إلى التطبيق على شكل **JSON** لتحديد نجاح أو فشل العملية.

2.5. مثال تطبيقي لربط الواجهة بقاعدة البيانات :

- **من واجهة التطبيق (Kotlin / Compose) :**

```kotlin
val user = User(name, email, password)
RetrofitClient.instance.registerUser(user).enqueue(object : Callback<ServerResponse> {
    override fun onResponse(call: Call<ServerResponse>, response:
Response<ServerResponse>) {
        if (response.isSuccessful && response.body()?.success == true) {
            Toast.makeText(context, "تم التسجيل بنجاح", Toast.LENGTH_SHORT).show()
        } else {
            Toast.makeText(context, "فشل في التسجيل", Toast.LENGTH_SHORT).show()
        }
    }

    override fun onFailure(call: Call<ServerResponse>, t: Throwable) {
        Toast.makeText(context, "خطأ: ${t.message}", Toast.LENGTH_SHORT).show()
    }
})

```

الشكل 27 : كود واجهة تطبيق ترسل بيانات تسجيل مستخدم إلى الخادم

- كود ملف Register.php :

```php
<?php
header("Content-Type: application/json");
include("db.php"); // الاتصال بقاعدة البيانات

$data = json_decode(file_get_contents("php://input"));

$name = $data->name;
$email = $data->email;
$password = $data->password;

$response = array();

if (!empty($name) && !empty($email) && !empty($password)) {
    $query = "INSERT INTO users (name, email, password) VALUES (?, ?, ?)";
    $stmt = $conn->prepare($query);
    $stmt->bind_param("sss", $name, $email, $password);

    if ($stmt->execute()) {
        $response["success"] = true;
        $response["message"] = "تم التسجيل بنجاح";
    } else {
        $response["success"] = false;
        $response["message"] = "حدث خطأ أثناء التسجيل";
    }
} else {
    $response["success"] = false;
    $response["message"] = "يرجى ملء جميع الحقول";
}

echo json_encode($response);
?>
```

الشكل 28 : ملف PHP لمعالجة تسجيل مستخدم وتخزينه في قاعدة البيانات MySQL.

6. المشاكل و التحديات أثناء تطوير التطبيق :

خلال عملية تطوير تطبيق تسيير مكتبة المطالعة العمومية **"محمد الطاهر تليلي" بقمار**، وجدت مجموعة من التحديات التقنية والوظيفية التي إستلزمت حلولا متنوعة وتكيفا مستمرا مع البيئة التطويرية. وفيما يلي أبرز هذه التحديات مع توضيح كيفية التعامل معها:

- **صعوبات ربط التطبيق بقاعدة البيانات MySQL عبر PHP:**

من أكثر التحديات التي تم مواجهتها هي إشكالية ربط تطبيق الأندرويد بقاعدة بيانات **MySQL** المستضافة على خادم محلي باستعمال **XAMPP**، حيث لم يكن بالإمكان الوصول إلى ملفات **PHP** من التطبيق عند العمل على المحاكي داخل **Android Studio** بسبب عدم توافق عنوان **localhost**.

للتغلب على هذا العائق، تم أولا استخدام العنوان الخاص بالمحاكي **(10.0.2.2)**، غير أن هذا الحل ظلّ محدوداً بالتجريب المحلي. ولهذا، تم اعتماد أداة **ngrok** لإنشاء نفق **(Tunnel)** يربط الخادم المحلي بالإنترنت عبر عنوان **HTTPS** خارجي مؤقت، مما مكنّ التطبيق من الوصول إلى ملفات **PHP** كأنها مستضافة على خادم فعلي، وساهم في اختبار الوظائف عن بعد بشكل فعآل وآمن .

- **مشاكل معالجة الاستجابات (Responses) من الخادم:**

واجه الفريق أخطاء في تفسير البيانات المستلمة من ملفات **PHP**، خصوصا في ما يتعلق بصيغة **JSON** المطلوبة. تم استخدام مكتبة **Gson** مع **Retrofit** لتسوية تنسيق البيانات المرسلة والمستلمة، مع مراجعة دقيقة لبنية الكائنات المستعملة في جهة الخادم والعميل.

- **أخطاء منطقية في واجهات Compose:**

نظراً لطبيعة **Jetpack Compose** التي تعتمد على البرمجة التفاعلية **(Declarative UI)**، ظهرت بعض السلوكيات غير المتوقعة عند تغير حالة الواجهات **(مثل إعادة تحميل مكونات الشاشة بعد تعديل المتغيرات)**. تم حل هذه المشكلات عبر تطبيق مبادئ إدارة الحالة **(Management State)** بشكل دقيق باستخدام متغيرات من نوع **mutableStateOf** وتوظيف **ViewModel** للفصل بين الواجهة والبيانات.

- **تحديات تتعلق بالأمان عند نقل البيانات:**

في مرحلة أولى من التطوير، كانت البيانات تنقل بدون تشفير بين التطبيق والخادم. ولتدارك هذه النقطة، تم اقتراح استعمال بروتوكول **HTTPS** في النسخ المستقبلية من التطبيق عند نقله إلى خادم إنتاج حقيقي، وذلك ضمانا لسلامة المعلومات الشخصية للمستخدمين.

خاتمة الفصل

يعد هذا الفصل تتويجا للمرحلة التطبيقية من مشروع التخرج، حيث تم فيه الانتقال من النماذج النظرية التي تم إعدادها باستخدام منهجية **UML** إلى مرحلة التنفيذ البرمجي الفعلي لتطبيق يهدف إلى تحسين آليات تسيير مكتبة **"محمد الطاهر تليلي" بقمار**. تم التطرق بالتفصيل إلى الأدوات والتقنيات المعتمدة، مثل بيئة **Android Studio** ولغة **Kotlin** ومكتبة **Jetpack Compose** لتصميم الواجهات، بالإضافة إلى **MySQL** لإدارة قاعدة البيانات وربطها بالتطبيق من خلال **XAMPP** و **PHP**، مع دعم النفاذ الخارجي باستعمال أداة **ngrok**.

كما تم عرض الهيكلية العامة للمشروع، والواجهات الأساسية الموجهة للمستخدمين والإدارة، مع شرح التفاعل البرمجي بين هذه الواجهات وقاعدة البيانات. وأدرجت أيضا التحديات التي واجهت عملية التطوير وكيف تم تجاوزها باستخدام منهجيات برمجية عملية، مما يبرز قدرة الفريق على حل المشكلات والابتكار ضمن بيئة تطوير واقعية.

خاتمة المذكرة

مع اختتام هذا العمل، نكون قد وصلنا إلى نهاية مشروع تخرج يهدف إلى تقديم حل رقمي فعال لمشكلة واقعية تعاني منها العديد من المكتبات التقليدية، والمتمثلة في غياب أداة تسيير رقمية عصرية تساعد في تنظيم عمليات الإعارة والاطلاع على الكتب.

من خلال تحليل الوضع الحالي لمكتبة "محمد الطاهر تليلي" بقمار، وتصميم نظام رقمي بناءً على منهجية **UML**، ثم تطوير تطبيق فعلي باستخدام أدوات وتقنيات حديثة، تمكنا من تجسيد مشروع متكامل يعكس قدراته في التحليل، التصميم، والبرمجة، ويبرز أهمية الرقمنة في تحسين الخدمات العمومية.

هذا المشروع لا يمثل فقط ثمرة لسنوات من التكوين الأكاديمي، بل يشكل أيضاً نقطة انطلاق لتطويرات مستقبلية قد تشمل تعزيز الأمان، دعم تعدد اللغات، دمج نظام إشعارات وتنبيهات، وإمكانية العمل دون اتصال بالإنترنت.

في الأخير، نأمل أن يكون هذا العمل قد قدم إضافة ملموسة في ميدان التطبيقات التعليمية والخدمية، وأن يشكّل مصدر إلهام لطلبة ومطوري المستقبل نحو تبني الحلول الرقمية لخدمة المجتمع المحلي و تطويره.

المراجع

1. Android Developers: https://developer.android.com
موقع رسمي لتعلم كيفية تطوير تطبيقات أندرويد باستخدام Kotlin و Jetpack Compose.
2. W3Schools: https://www.w3schools.com
موقع تعليمي شهير لتعلم لغة PHP وربطها مع قاعدة البيانات MySQL بطريقة مبسطة.
3. Libib: https://www.libib.com
تطبيق مكتبي رقمي استُخدم كمثال تطبيقي في المشروع، مخصص للمكتبات الصغيرة.
4. BookBuddy: https://www.bookbuddyapp.com
تطبيق بسيط لإدارة الكتب، تم الاستئناس به في المشروع كمثال عملي.
5. PlantUML: https://plantuml.com
أداة مجانية لإنشاء المخططات UML التي استُعملت في تحليل وتصميم النظام.
6. مقابلة مع موظفي مكتبة "محمد الطاهر تليلي" بقمار – 2025
مصدر ميداني مباشر لتحليل نظام التسيير الحالي المعتمد في المكتبة.
7. Koha Integrated Library System: https://koha-community.org
نظام مكتبات رقمي مفتوح المصدر، يُستخدم في عدة مكتبات جامعية حول العالم.
8. Kotlin Official Documentation: https://kotlinlang.org/docs/getting-started.html
وثائق رسمية لتعلم أساسيات لغة Kotlin لتطوير تطبيقات أندرويد.
9. Learn Kotlin in Arabic: https://learnwithfares.com/kotlin-intro/
شرح مبسط باللغة العربية حول تعلم لغة Kotlin موجه للمبتدئين.
10. Introduction to XAMPP: https://www.geeksforgeeks.org/how-to-install-and-use-xampp-on-windows
مقال يشرح طريقة تثبيت واستخدام XAMPP لإنشاء خادم محلي وربطه بقاعدة MySQL.
11. What is Library Digitization: https://www.electronic-resources.org/what-is-library-digitization
مقال توعوي حول أهمية رقمنة المكتبات في العصر الحديث.
12. Library Management Systems Overview: https://www.lisbdnetwork.com/library-management-system/
شرح بسيط حول مفهوم أنظمة تسيير المكتبات والفرق بينها وبين الطرق التقليدية.