****

**وزارة التعليم العالي والبحث العلمي**

**جامعة طرابلس**

**كلية تقنية المعلومات/ قسم هندسة البرمجيات**

**­­مشروع في مقرر جافا متقدمة ITSE322 بعنوان**

**أعد كجزء لهذا المقرر**

**اعداد**

**سـلـمـى حـمـدي الـصـغـيّـر 2211845034**

**صـفـاء مـحـمـد الـقـضـقـاض 2211845182**

**نـدى يـوسـف خـلـيـفـة 2211846481**

**هـدايـة عـبـدالـنـاصـر بـنّـيـس 2211846597**

**تحت إشراف الأستاذة**

**مي البعباع**

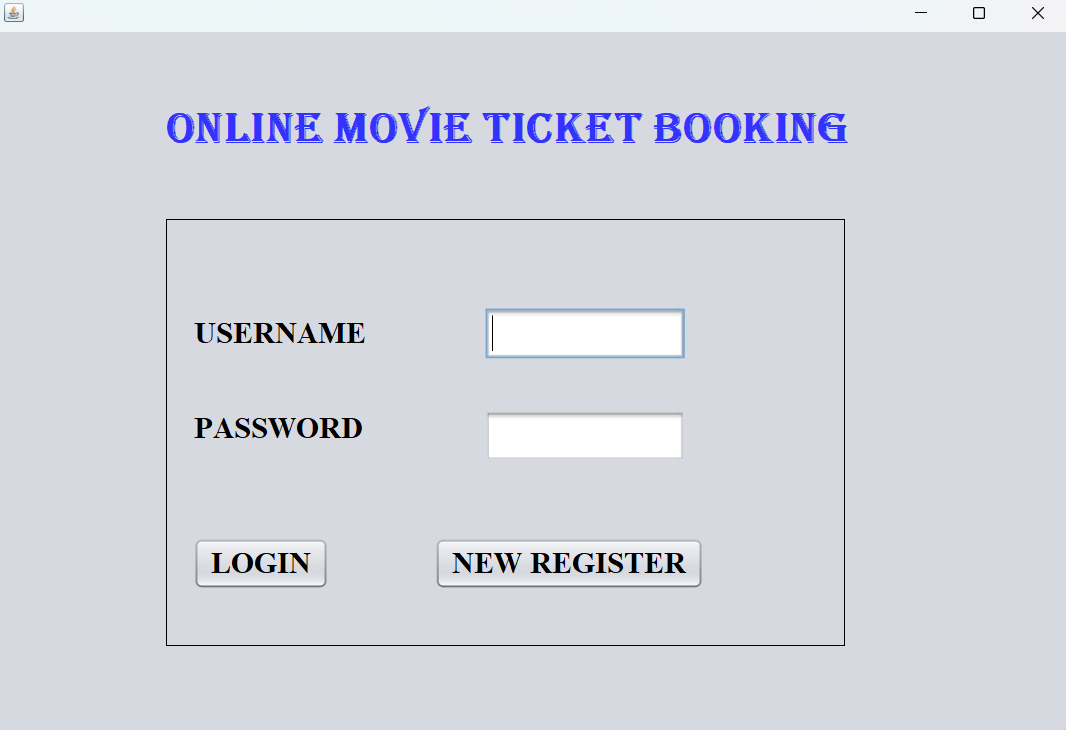
1. نظرة عامة على النظام :

نظام حجز تذاكر الأفلام عبر الإنترنت هو تطبيق يهدف إلى توفير وسيلة بديلة ومريحة للعملاء لشراء تذاكر السينما. يتيح النظام للمستخدمين تسجيل الدخول (أو التسجيل كجدد) ، ثم اختيار الفيلم ، والمسرح الذي يعرض الفيلم فيه ، وتوقيت العرض ، وعدد التذاكر المطلوبة. بعد تأكيد التفاصيل، يمكن للمستخدمين المتابعة لحجز التذاكر ، مع إمكانية الدفع عبر خيارات مختلفة مثل بطاقات الائتمان أو الخصم أو الخدمات المصرفية عبر الإنترنت أو المحافظ الإلكترونية. يعرض النظام تفاصيل الحجز ويقوم بتوليد إيصال يمكن طباعته لتقديمه في شباك التذاكر للحصول على التذاكر الفعلية. بالإضافة إلى ذلك، يوفر النظام وظيفة لإلغاء التذاكر المحجوزة ولان النظام مصمم ليكون آليًا، لا يحتاج الموظفون إلى التدخل يدويًا في الطلبات بمجرد تلقيها عبر النظام بعد تغذية البيانات في قاعدة البيانات.

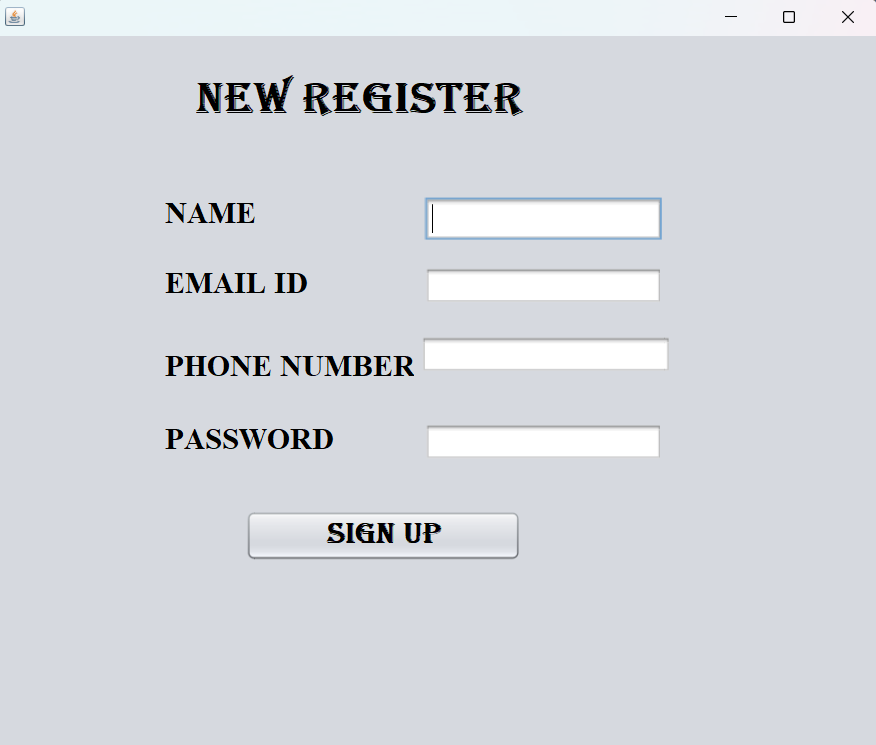
1. التقنيات والأدوات المستخدمة :

* **لغة البرمجة** : java
* **بيئة التطوير المتكاملة (IDE) :** NetBeans
* **بناء واجهة المستخدم الرسومية (GUI Framework)** **:** Java Swing
* **قاعدة البيانات :** MySQL
* **اتصال قاعدة البيانات :**  JDBC (Java Database Connectivity)
* **نظام التحكم بالإصدار (Version Control System):** GitHub لاستضافة مستودع المشروع

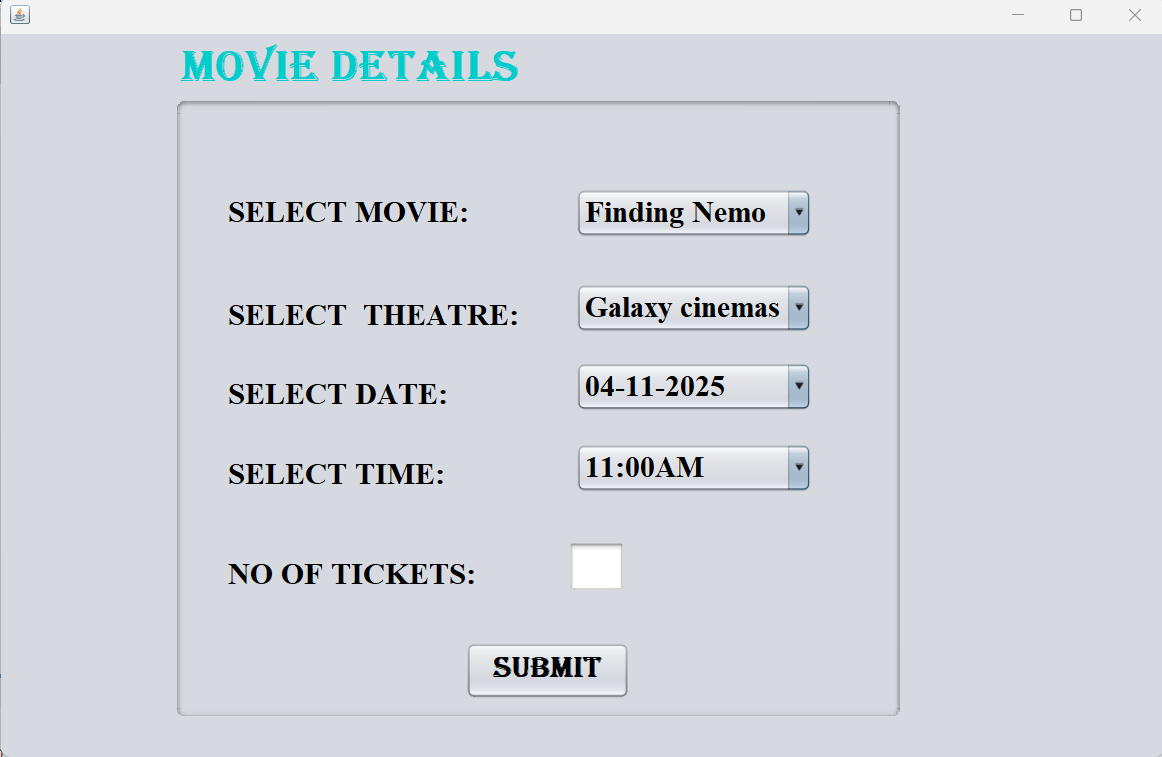
1. الميزات الرئيسية المنفذة :
   1. وحدة تسجيل دخول المستخدمين الحاليين.
   2. وحدة تسجيل مستخدمين جدد.
   3. اختيار الفيلم.
   4. اختيار المسرح.
   5. اختيار التاريخ.
   6. اختيار وقت العرض.
   7. تحديد عدد التذاكر المطلوب حجزها.
   8. عرض تفاصيل الحجز (عدد التذاكر، المسرح، الفيلم، التاريخ، وقت العرض، السعر).
   9. إمكانية حجز التذكرة بعد التأكيد.
   10. إمكانية إلغاء التذكرة.
   11. اختيار نوع الدفع (بطاقة ائتمان، بطاقة خصم، خدمات مصرفية عبر الإنترنت، محفظة ).
   12. إدخال تفاصيل البطاقة (رقم البطاقة، اسم حامل البطاقة، CVV، تاريخ انتهاء الصلاحية).
   13. توليد إيصال تفصيلي يحتوي على معلومات الحجز والسعر.
   14. رسالة شكر بعد إتمام الحجز.
2. لقطات شاشة (screenshots) من التطبيق :



**الشكل1**: واجهة تسجيل الدخول (User Login Interface)



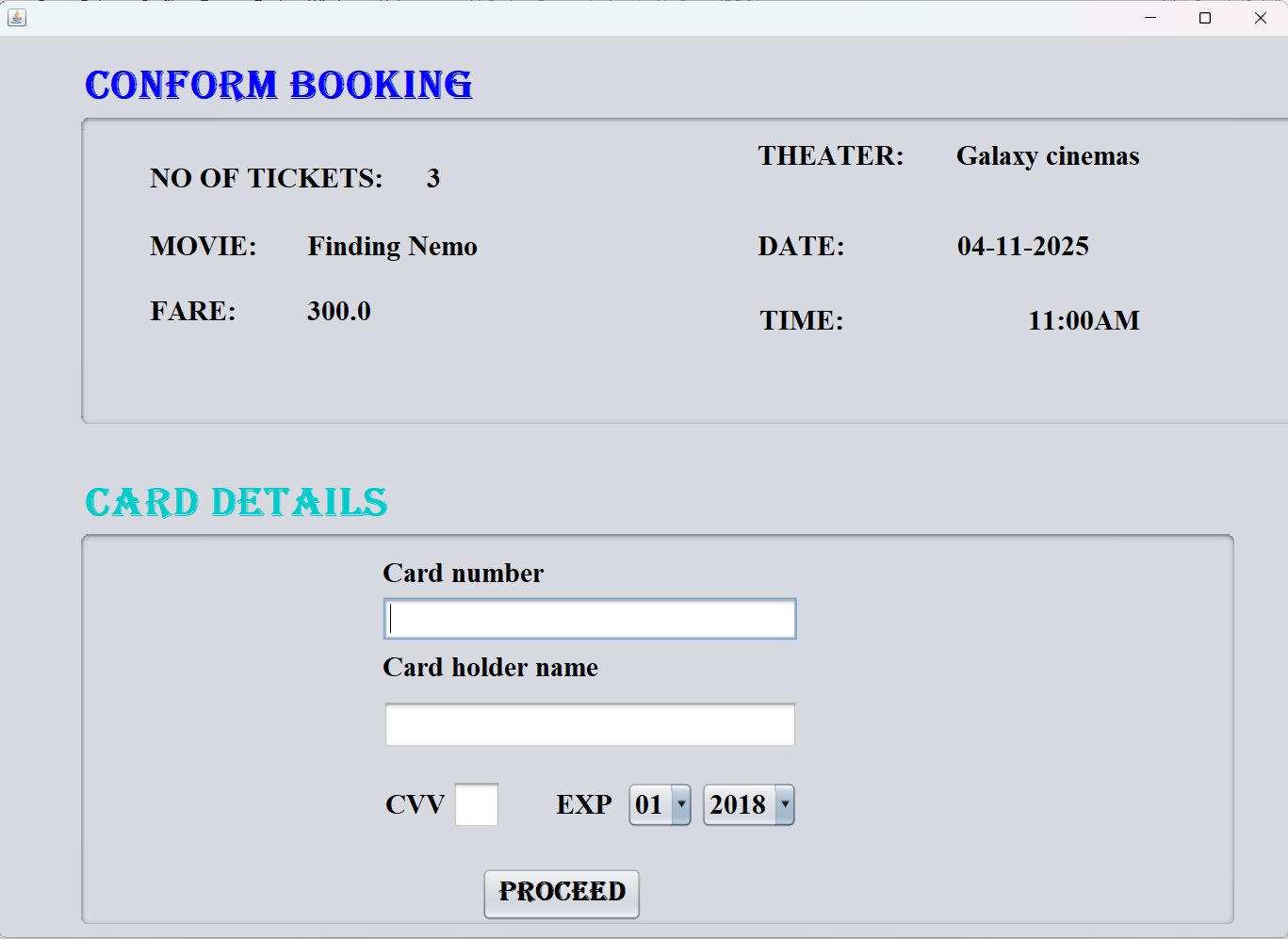
**الشكل2**: واجهة تسجيل مستخدم جديد (New User Registration Screen)



**الشكل3**: واجهة اختيار الفيلم وتفاصيل العرض (Movie and Show Details Selection Interface)



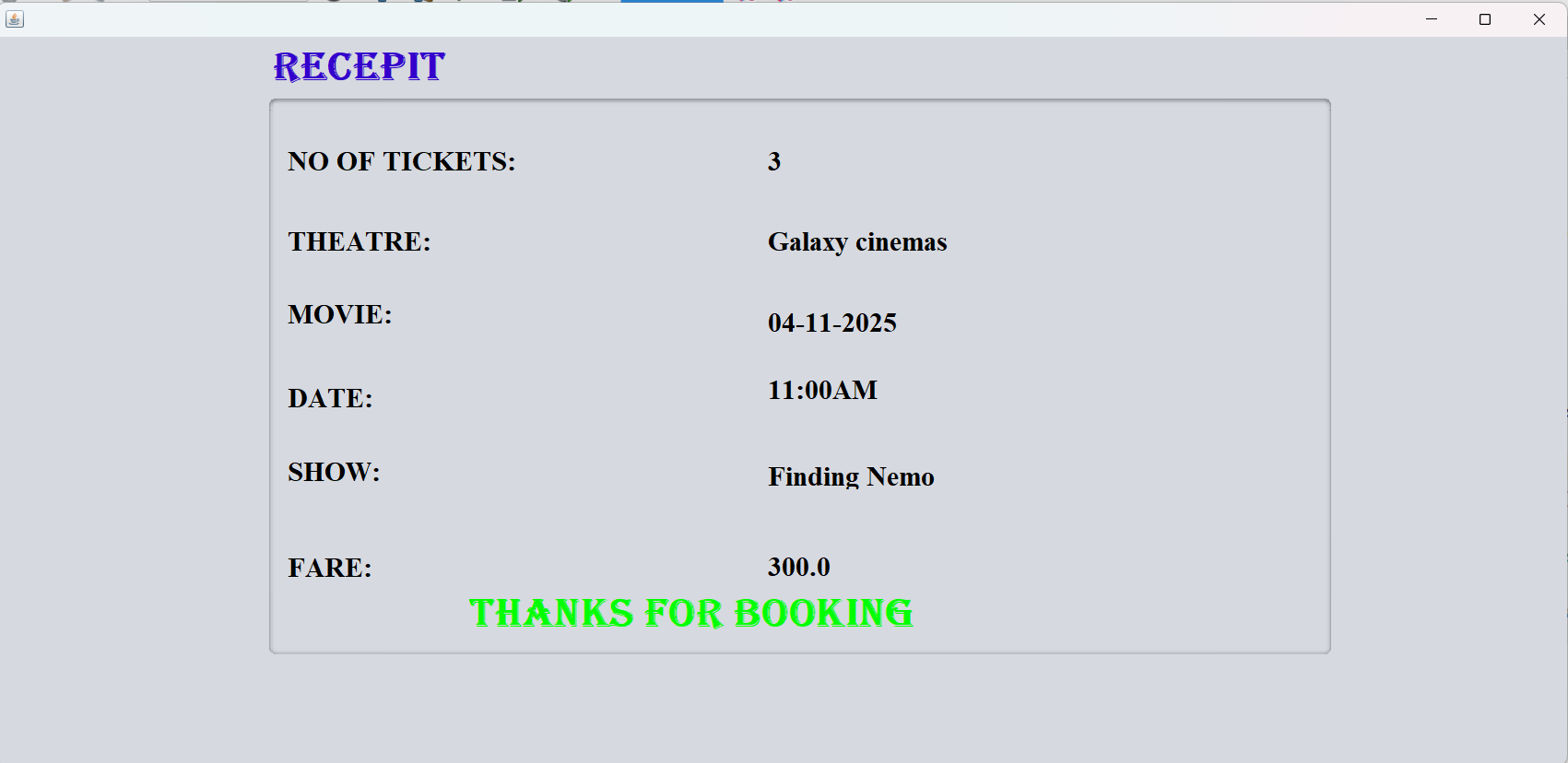
**الشكل4**: واجهة تفاصيل الحجز وتأكيده (Booking Details and Confirmation Interface)



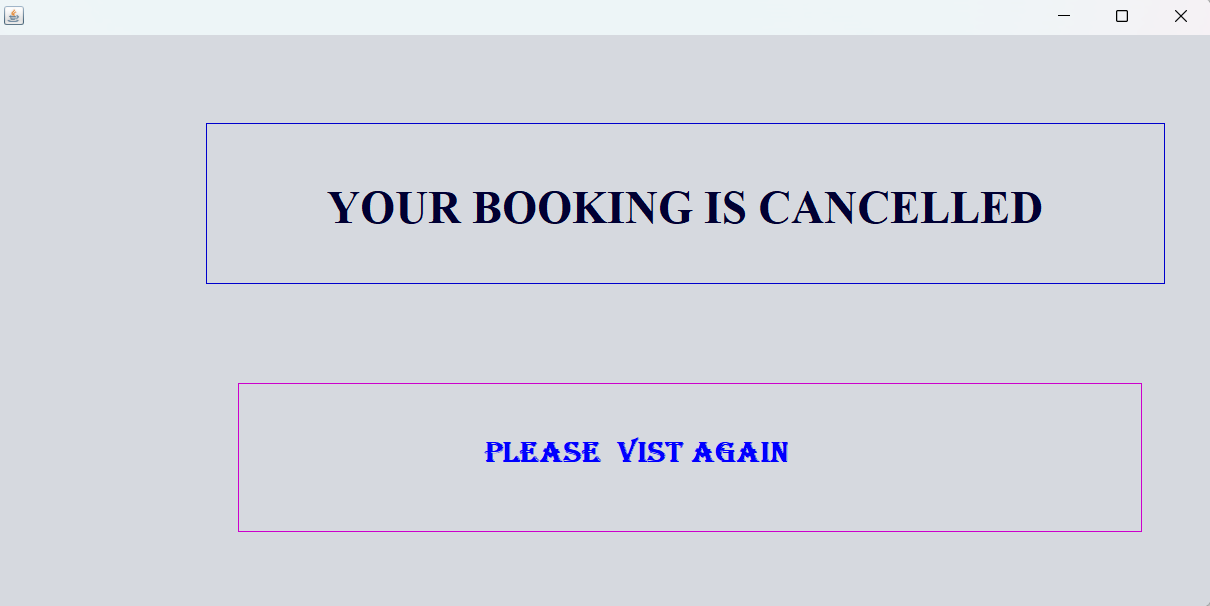
**الشكل5**: واجهة إدخال تفاصيل الدفع (Payment Details Entry Interface)



**الشكل6**: واجهة اختيار نوع الدفع (Payment Type Selection Interface)



**الشكل7**: واجهة الإيصال النهائي (Final Receipt Interface)



**الشكل8**: واجهة تأكيد الإلغاء (Booking Cancellation Confirmation Interface)

1. شرح لكيفية تنفيذ كل موضوع مطلوب :
2. **معالجة الاستثناءات (Exception Handling):**  
   استخدم المشروع تقنية معالجة الاستثناءات بفعالية لضمان عدم توقف التطبيق عند حدوث أي أخطاء. في كلاس movie، تم احتواء العمليات الحساسة داخل كتل try-catch-finally، وخصوصًا أثناء تنفيذ الحجز والتعامل مع قاعدة البيانات. في recepit، ظهرت هذه التقنية بشكل أكثر تقدمًا داخل دالة done() التابعة لـ SwingWorker، حيث يتم التعامل مع النتائج النهائية للعمليات الخلفية وعرض رسائل واضحة للمستخدم عند نجاح الحجز أو فشله، مع إعادة تمكين الأزرار بغض النظر عن النتيجة. كما استخدمت في دالة jButton2ActionPerformed() أثناء تنفيذ إلغاء الحجز، حيث يتم تنفيذ استعلامات حذف وتحديث تتطلب حماية كاملة عبر try-catch, ويتم تطبيق rollback() لتفادي تلف البيانات إذا حدث خطأ و في register، نجدها أثناء تنفيذ عملية إدخال المستخدم الجديد في قاعدة البيانات، لحماية العملية من الفشل غير المتوقع. كذلك، في كل من register و payment، تم استخدام catch داخل دالة main() للتحكم بالأخطاء الناتجة من تحميل واجهة المستخدم الرسومية بنظام Nimbus .
3. **الاتصال بقاعدة البيانات (JDBC):**  
   يعتمد المشروع بشكل أساسي على JDBC للاتصال بقاعدة بيانات MySQL، وقد تم تصميم كلاس خاص لذلك باسم JavaApplication4 يحتوي على دالة ConnecrDb() التي تنشئ الاتصال وتُستخدم من جميع الكلاسات مثل movie, recepit, و register. في كلاس movie, يُستخدم JDBC لاستعلام جداول movies\_inventory و bookings من أجل تعبئة عناصر الواجهة، وكذلك لتنفيذ عمليات الحجز. أما في كلاس recepit, فيتم تنفيذ سلسلة من العمليات المتقدمة مثل جلب بيانات الحجز، حذف السجل من جدول bookings, وإرجاع التذاكر إلى جدول movies\_inventory وكل ذلك داخل معاملة واحدة يتم التحكم بها من خلال setAutoCommit(false) و commit() لضمان الاتساق وأما register, فيقوم بتنفيذ استعلام INSERT لتسجيل بيانات المستخدم الجديد باستخدام Statement .
4. **واجهة المستخدم الرسومية ومعالجة الأحداث (GUI + Event Handling):**  
   تم تصميم واجهة المشروع بالكامل باستخدام مكتبة Swing في Java، مع تخصيص كل نافذة لتقوم بوظيفة معينة بطريقة منظمة وسلسة. كلاس login يحتوي على نافذة بسيطة لإدخال بيانات المستخدم، تشمل حقول إدخال وكلمة مرور وزر تسجيل الدخول. كلاس movie يُعد الأكثر تفاعلية، حيث يعرض قائمة بالأفلام والمواعيد باستخدام JComboBox, ويتعامل مع اختيار المستخدم للتوقيت والفيلم لعرض التفاصيل المناسبة. في payment, يتم إدخال بيانات البطاقة البنكية من خلال عناصر مثل JPasswordField, JTextField, و JComboBox, ويتم تنفيذ الحدث عند الضغط على زر "proceed" بنقل المستخدم إلى واجهة إيصال الحجز. recepit يعرض تأكيد الحجز ويحتوي على أزرار للتنقل أو الإلغاء، ويستخدم myStatusLabel لعرض حالة تنفيذ العملية ديناميكيًا. أما register, فيوفر واجهة تسجيل متكاملة تشمل الاسم، الإيميل، رقم الهاتف، وكلمة المرور، مع زر تسجيل لتنفيذ العملية.
5. **إطار عمل المجموعات (Collections Framework):**  
   في كلاس movie, تم استخدام بنية Vector<String> لتخزين البيانات المستخرجة من قاعدة البيانات مثل أسماء الأفلام، التوقيتات، والمسارح، ثم تعبئتها داخل عناصر JComboBox بشكل ديناميكي. هذا الاستخدام يُظهر كيف تم الاعتماد على إطار المجموعات لجعل الواجهة تفاعلية وقابلة للتحديث المباشر بناءً على بيانات قاعدة البيانات.
6. **تيارات الإدخال والإخراج (I/O Streams):**  
   ضمن كلاس register، تم استخدام تقنية Logger لتوثيق الأحداث والتفاعلات داخل البرنامج بشكل احترافي. حيث تم تهيئة ConsoleHandler و FileHandler لتسجيل الرسائل البرمجية من نوع INFO، وحفظها في ملف مخصص داخل المسار C:/temp/application\_test.log. هذه العملية تتيح تتبّع الأداء العام للتطبيق، والكشف المبكر عن أي أخطاء أو استثناءات قد تحدث أثناء تشغيل الواجهة أو تنفيذ الاستعلامات، مما يساهم في تحسين صيانة البرنامج وتطويره مستقبلًا.
7. **تعدد الخيوط (Multithreading)**   
   تم تطبيق تعدد الخيوط باحترافية داخل كلاس recepit باستخدام SwingWorker عندما يضغط المستخدم زر التأكيد للحجز، يتم تنفيذ العملية في الخلفية دون تجميد واجهة المستخدم، بينما يتم عرض رسالة "جاري المعالجة..." داخلmyStatusLabel بعد اكتمال المعالجة، يتم التفاعل مع المستخدم من خلال دالة done() التي تُظهر رسالة نجاح أو فشل، وتعيد عرض واجهة الإيصال أو تنقله إلى النافذة التالية و هذا يضمن تجربة مستخدم سلسة دون انتظار أو توقف واجهة البرنامج.
8. توزيع المهام بين أعضاء الفريق:

تم إنجاز هذا المشروع من قبل فريق عمل متكامل، حيث تم تقسيم الأدوار الفنية والبرمجية ،على النحو التالي:

* صفاء محمد القضقاض

قامت بتطوير كامل عملية تسجيل الدخول، من تصميم واجهة المستخدم إلى تنفيذ التحقق من صحة البيانات وربطها بقواعد البيانات، مع الاهتمام بإظهار رسائل الخطأ والتنبيهات .

* هداية عبد الناصر بنيس

تولّت مهمة تصميم وتنفيذ واجهة تسجيل مستخدم جديد، بما في ذلك إعداد حقول الإدخال ، بالإضافة إلى تنفيذ استعلامات الإدخال الخاصة بقاعدة البيانات وتوثيق العملية ضمن الكود البرمجي.

* سلمى حمدي الصغير

تولّت مسؤولية تطوير وظيفة حجز التذاكر، حيث قامت ببناء الواجهة الخاصة بعرض العروض والتوقيتات، وإعداد الاستعلامات الخاصة بالتحديث والإدراج في قاعدة البيانات بما يضمن دقة عملية الحجز.

* + ندى يوسف خليفة

نفّذت ميزة إلغاء الحجز من خلال تطوير واجهة مخصصة لذلك، بالإضافة إلى إعداد العمليات الخلفية التي تتضمن حذف البيانات المستهدفة وتحديث حالة المخزون، مع ضمان أمان المعاملة عبر التعامل الصحيح مع قاعدة البيانات.

وبالإضافة إلى المهام المذكورة أعلاه، تم تنفيذ بقية الواجهات والوظائف البرمجية الأخرى كعمل جماعي مشترك بين أعضاء الفريق، بما في ذلك التنسيق بين النوافذ، إعداد التصميم العام، وتكامل البيانات بين الأجزاء المختلفة لضمان تقديم مشروع متكامل .

1. التحديات والدروس المستفادة:

* من التحديات استخدام SwingWorker لأول مرة:

تم تطبيق SwingWorker لتجنب تجميد الواجهة أثناء تنفيذ الحجز، لكن كانت هناك صعوبة في استيعاب تدفق doInBackground ثم done()، خاصة في إعادة تمكين الأزرار وتحديث حالة الحجز.

* ومن التحديات ضبط المعاملة (Transactions) في الإلغاء:

تنفيذ عملية commit() و rollback() داخل إلغاء الحجز تطلب دقة كبيرة في ترتيب الاستعلامات والتأكد من أن كل خطوة تنفذ بشكل صحيح، وإلا قد تُلغى التذاكر دون أن تُعاد فعليًا إلى المخزون.

* ومن التحديات نقل البيانات بين الكلاسات:

تم تمرير البيانات بين النوافذ باستخدام Constructors، ولكن التعامل مع أنواع مختلفة (مثل String و double و int) بشكل منظم كان تحديًا في أول الأمر، خاصة في التنقل بين payment و recepit.

* ومن الدروس المستفادة هو ان التوثيق في وقت التنفيذ لا يقل أهمية عن الكود نفسه: حيث ان اعتماد Logger ساعد لاحقًا في معرفة نقاط الفشل بسرعة دون الحاجة لطباعة في الكونسول فقط، وأعطى المشروع شكلًا احترافيًا أكثر
* ابرز تحدي : تنفيذ المشروع كان تحت ضغط كبير بسبب ضيق الوقت، وتزامنه مع امتحانات ومشاريع أخرى في مواد مختلفة، مما جعل تنظيم الوقت والتفرغ الجزئي للعمل البرمجي تحديًا حقيقيًا.