تعرَّفْتُ سابقًا دورة حياة تطوير البرمجيات، التي تتضمَّن خطوات تطوير المشروع، وتساعد على تنظيم عملية تطوير البرمجيات وإدارتها بصورة فاعلة، فهل توجد نماذج يُعتمَد عليها في تطوير البرمجيات؟

نشاط تمهيدي

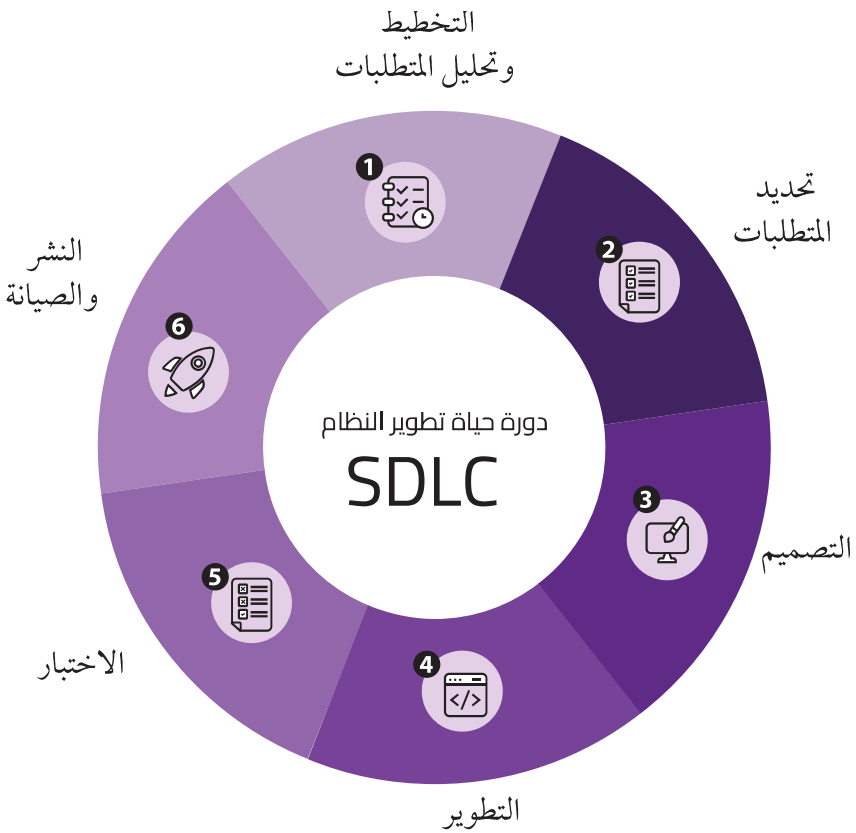
أُفكِّر في المراحل الرئيسة لدورة حياة تطوير النظام (SDLC)، ثمَّ أختار إحدى هذه المراحل للحديث عنها أمام زملائي/ زميلاتي في الصف، وأذكر أمثلة عملية عليها من واقع الحياة.

مفهوم دورة حياة تطوير النظام (SDLC):

تُعرَّف دورة حياة تطوير النظام بأنَّها عملية منهجية مُنظَّمة تَصِف كيف يُمكِن تطوير برامج مُعيَّنة وصيانتها وتحسينها خطوة خطوة بهدف إنتاج برامج فائقة الجودة، وقادرة على الوفاء باحتياجات المُستخدِمين ومُتطلَّباتهم.

مراحل دورة حياة تطوير النظام:

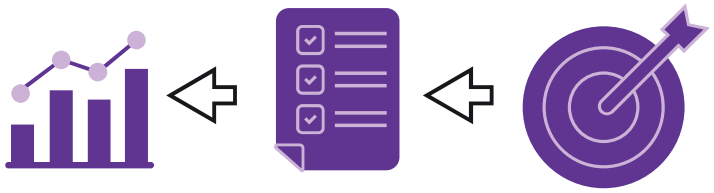
تَمُرُّ عملية تطوير النظام بمراحل مُنظَّمة ومُرتَّبة يتعيَّن على مهندس البرمجيات أو المُطوِّر للأنظمة الحاسوبية أنْ يتَّبعها أثناء هذه العملية لضمان الحصول على مُنتَج مُتميِّز يَفي باحتياجات المُستخدِمين ومُتطلَّباتهم. أنظر الشكل (1-1) الذي يُبيِّن مراحل دورة حياة تطوير النظام.



الشكل (1-1): دورة حياة تطوير النظام.

المرحلة الأولى: التخطيط وتحليل المُتطلَّبات
(Planning and Requirement Analysis)

تتمثَّل هذه المرحلة في تحليل المشروع والتخطيط له عن طريق دراسة الجوانب الآتية: نطاق المشروع، والمشكلة التي يُراد حَلُّها، والعناصر الواجب توافرها من موارد بشرية وأدوات مُتعدِّدة، وجدوى المشروع الاقتصادية، والمخاطر التي قد تعترض عملية تنفيذ المشروع وسُبُل التعامل معها. كذلك يجب الالتقاء بالعملاء لجمع البيانات الخاصة بالمشروع وتعرُّف مواصفاته، ثمَّ إعداد خارطة طريق للتطوير اعتمادًا على ما جُمِع من بيانات، وتضمين خارطة الطريق الجدول الزمني المُحدَّد لتنفيذ المشروع. يلي ذلك تحليل المُتطلَّبات، والإحاطة بما ينبغي للنظام أنْ يُحقِّقه عن طريق التفاعل مع الجهات المَعْنِيَّة من عملاء ومُستخدِمين؛ إمّا بإجراء مقابلات، وإمّا باستخدام استبانات، وإمّا بعمل دراسات وبحوث للوقوف على احتياجات السوق. أنظر الشكل (1–2) الذي يُبيِّن خطوات هذه المرحلة.



الشكل (1–2): خطوات مرحلة التخطيط وتحليل المُتطلَّبات في دورة حياة تطوير النظام.

مثال:

أُطوِّر مشروعًا خاصًّا بإدارة المهام اليومية، وأعمل لذلك على جمع المُتطلَّبات الآتية وتحليلها:

1. تعيين الفئة المُستهدَفة، وتحديد الميزانية المُتوقَّعة والأدوات التي يُراد استخدامها، ووضع جدول زمني مُحدَّد لتنفيذ المشروع.
2. إضافة المهام المطلوبة.
3. تعديل بعض المهام، وحذف ما يَلزم منها.
4. تصنيف المهام بحسب الأولوية.
5. دراسة أنظمة شبيهة لتعرُّف مَواطِن القوَّة ومَواطِن الضعف، وكيف يُمكِن الاستفادة منها.

نشاط فردي

يبحث طبيب أسنان عن برنامج لتنظيم المواعيد الخاصة بعيادته. أُفكِّر في التخطيط لهذا البرنامج عن طريق تحديد المُتطلَّبات الآتية: الفئة المُستهدَفة، والأهداف، والموارد. بعد ذلك أستعمل ورق قلّاب (Flipchart) لرسم مُخطَّط، ثمَّ أُناقِشه مع زملائي/ زميلاتي في الصف.

هل اختلف مُخطَّطي عن مُخطَّطات الزملاء؟ أُبرِّر إجابتي.

المرحلة الثانية: تحديد المُتطلَّبات (Defining Requirements)

تُعَدُّ وثيقة المواصفات الخاصة بمُتطلَّبات البرنامج (Software Requirement Specification: SRS) مُخرَجات هذه المرحلة؛ إذ يُوثَّق فيها كل ما يُحدَّد من مُتطلَّبات النظام الحاسوبي (البرنامج) المُستهدَف بالاتِّفاق مع العملاء ومُحلِّلي السوق. كذلك تشمل هذه المرحلة تحديد مهام النظام، والمُتطلَّبات التكنولوجية الخاصة به.

مثال:

يشتمل مشروع إدارة المهام اليومية على ما يأتي:

أ. **وثيقة مُتطلَّبات البرنامج التي تُحدِّد مهام المُستخدِم**

- إضافة المهام اليومية، وتحديد مواعيد تنفيذها النهائية.
- تعديل المهام.
- حذف المهام.
- تحديد حالة المهمة (مُكتمِلة/ غير مُكتمِلة).
- تصنيف المهام بحسب الأولوية.
- إرسال البرنامج إشعارات إلى المُستخدِم لتذكيره بالأولويات.

ب. **تحديد المُتطلَّبات التكنولوجية** : مثل الأجهزة والأدوات الرقمية والبرمجيات.

نشاط فردي

1- أُحدِّد مهام البرنامج الخاص بحجز المواعيد في عيادة الأسنان ومُتطلَّباته التكنولوجية.

2- ما الطرائق التي سأستخدمها في عملية جمع البيانات؟

3- مَنِ الأشخاص الذين يُمكِن الاستفادة منهم في جمع البيانات اللازمة؟

أُدوِّن النتائج التي أتوصَّل إليها، ثمَّ أُناقِشها مع مُعلِّمي/ مُعلِّمتي وزملائي/ زميلاتي.

أبحث

أبحث في المواقع الإلكترونية الموثوقة في شبكة الإنترنت عن تعريف لكلٍّ من مفهوم العملاء ومفهوم مُحلِّلي السوق، ثمَّ أُدوِّن ما أتوصَّل إليه في ملف خاص، ثمَّ أُشارِكه مع الزملاء/ الزميلات في الصف.

المرحلة الثالثة: التصميم (Design)

يستفاد من وثيقة المواصفات الخاصة بمُتطلَّبات البرنامج (SRS) في إعداد تصاميم للنظام، وهي تُضمَّن في وثيقة تُسمّى **مواصفات وثيقة التصميم (Design Document Specification: DDS)**. بعد ذلك يُحدَّد التصميم المناسب للنظام عمليًّا ومنطقيًّا بالاتِّفاق مع العملاء ومُحلِّلي السوق. كذلك تُحدَّد في هذه المرحلة المُدخَلات والمُخرَجات وأجزاء النظام، وتُصمَّم واجهة المُستخدِم وقواعد البيانات الخاصة بالنظام، إضافةً إلى تحديد طريقة عمل النظام، وهي المرحلة التي تسبق برمجة النظام.

مثال:

يتطلَّب مشروع إدارة المهام اليومية تصميم واجهة المُستخدِم وقاعدة البيانات على النحو الآتي:

1. **واجهة المُستخدِم:**
   - أ. إنشاء زِرٍّ يُمكِن للمُستخدِم أنْ يضغط عليه لإضافة مهمة جديدة.
   - ب. إنشاء صفحة لعرض قائمة المهام فيها.
   - ج. إمكانية التعديل على المهام أو حذفها من صفحة قائمة المهام.

2- قاعدة البيانات:

- أ. إنشاء جدول للمهام يحوي عنوان المهمة، ووصفًا لها، وحالتها، والأولوية، والموعد النهائي.
- ب. رسم المُخطَّطات اللازمة لقاعدة البيانات.

نشاط فردي

أُفكِّر في طريقة لتصميم واجهة المُستخدِم في نظام حجز المواعيد في عيادة الأسنان، وآليَّة عمل نظام حجز للمواعيد في العيادة.

أُدوِّن الأفكار التي أتوصَّل إليها، وأرسم المُخطَّطات اللازمة للنظام، ثمَّ أُناقِشها مع زملائي/ زميلاتي في الصف.

المرحلة الرابعة: التطوير (Development)

تمتاز هذه المرحلة بتحويل مُخرَجات مرحلة التصميم إلى صيغة برمجية يُمكِن استخدامها بصورة عملية؛ إذ تتضمَّن هذه المرحلة كتابة الكود (المقاطع البرمجية) الذي يختصُّ بالنظام، ويكون قابلًا للتطوير. كذلك تتضمَّن هذه المرحلة مراجعة المقاطع البرمجية، والعمل على تحسين النظام باستمرار.

نشاط عملي

أُحوِّل مُخرَجات المرحلة السابقة في نظام حجز المواعيد الخاص بعيادة الأسنان إلى برنامج باستخدام لغة البرمجة بايثون (Python)، ثمَّ أُطلِع الزملاء/ الزميلات على البرنامج، ونتبادل معًا الأفكار والمُقترَحات لتحسينه.

المرحلة الخامسة: الاختبار (Testing)

يتمُّ في هذه المرحلة اختبار النظام يدويًّا وآليًّا من قِبَل مُطوِّر النظام؛ للتأكُّد أنَّه يعمل بصورة صحيحة، وأنَّه يُحقِّق الهدف الذي أُنشِئ من أجله.

تهدف هذه المرحلة إلى الحصول على التغذية الراجعة من قبل مُطوِّر النظام والمُستخدِمين ومالك النظام؛ بُغْيَةَ تصحيح الأخطاء (إنْ وُجِدت) والتطوير والتحسين.

نشاط عملي

أختبر البرنامج الخاص بحجز المواعيد في عيادة الأسنان، وذلك بتنفيذه، والتحقُّق من مُخرَجاته. بعد ذلك أُدوِّن ملاحظاتي على البرنامج، ثمَّ أُناقِشها مع الزملاء/ الزميلات بهدف تحسين النظام.

نشاط جماعي

أُنظِّم جلسة نقاش مع زملائي/ زميلاتي في المجموعة للإجابة عن الأسئلة الآتية:

- ما الإجراءات الواجب اتِّخاذها إذا أخفق النظام في أداء المهام المنوطة به بصورة صحيحة؟
- فيمَ يستفاد من الوثائق التي جُمِّعت في المراحل السابقة؟
- كيف يُمكِن الاستفادة من جميع العمليات السابقة في تحسين النظام؟

أُناقِش إجابات الأسئلة مع زملائي/ زميلاتي في المجموعة، ثمَّ أعرضها أمام أفراد المجموعات الأُخرى بهدف التوصُّل إلى استنتاجات مُشتَرَكة.

المرحلة السادسة: النشر والصيانة (Publishing and Maintenance)

تتضمَّن هذه المرحلة نشر النظام؛ أيْ جعله متاحًا للاستخدام في بيئة حقيقية، ويكون ذلك ضمن عدد من المراحل؛ للتأكُّد أنَّ النظام يعمل بسلاسة وسهولة ويُسْر على النحو المُخطَّط له. كذلك تتضمَّن هذه المرحلة إخضاع النظام للصيانة الدورية؛ لضمان تنفيذه جميع المهام المنوطة به بصورة صحيحة.

إضاءة

توجد ثلاث طرائق لاستخدام النظام، هي:

1. **الاستخدام المباشر**: تمتاز هذه الطريقة بالتحوُّل إلى النظام الجديد مباشرة، وإلغاء النظام القديم.
2. **الاستخدام المُتزامِن**: تمتاز هذه الطريقة باستخدام النظام الجديد، جنبًا إلى جنب مع النظام القديم؛ للتأكُّد أنَّه يُنفِّذ جميع المهام المنوطة به على النحو الصحيح.
3. **الاستخدام المرحلي المُتدرِّج**: تُستخدَم هذه الطريقة إذا كان النظام ضخمًا وكبيرًا، وذلك باستعمال نظام فرعي جديد فقط، والإبقاء على بقيَّة الأنظمة الفرعية مع النظام القديم لحين التحقُّق من صلاحيته، ثمَّ يتمُّ الانتقال إلى نظام فرعي آخر، وهكذا.

نشاط جماعي

يُعَدُّ التوثيق والتدريب والدعم من أهمِّ العمليات التي تضمن استمرارية عمل النظام.

أُناقِش – بالتعاون مع أفراد مجموعتي – المراحل التي يُمكِن أنْ تحدث فيها عمليات التوثيق والتدريب والدعم، ثمَّ أُشارِك أفراد المجموعات الأُخرى في ما توصَّلنا إليه من نتائج.

نشاط فردي

أُطبِّق مراحل تطوير النظام على نظام بسيط لتسجيل الطلبة في إحدى المدارس، وأستخدم برمجية سكراتش (Scratch) أو برمجية بايثون (Python) في تطوير البرنامج الخاص بالنظام. بعد ذلك أُدوِّن الخطوات التي استخدمتها في ذلك، ثمَّ أُشارِك أفراد المجموعات الأُخرى في ما توصَّلْتُ إليه من خلال مجموعتي.

النماذج الخاصة بدورة حياة تطوير النظام (SDLC Models):

يوجد لدورة حياة تطوير النظام كثير من النماذج التي تختلف في ما بينها من حيث آلية التطبيق، والإيجابيات والسلبيات. وفي ما يأتي أهمُّ هذه النماذج:

1- نموذج الشلّال (Waterfall):

يُعَدُّ هذا النموذج الأساس لبقيَّة النماذج؛ ذلك أنَّه يتكوَّن من مراحل بسيطة أساسية، وهو يُناسِب المشروعات التي تكون فيها المُتطلَّبات واضحة ومُحدَّدة، لكنَّه لم يعد مُستخدَمًا اليوم بسبب عدم مرونته.

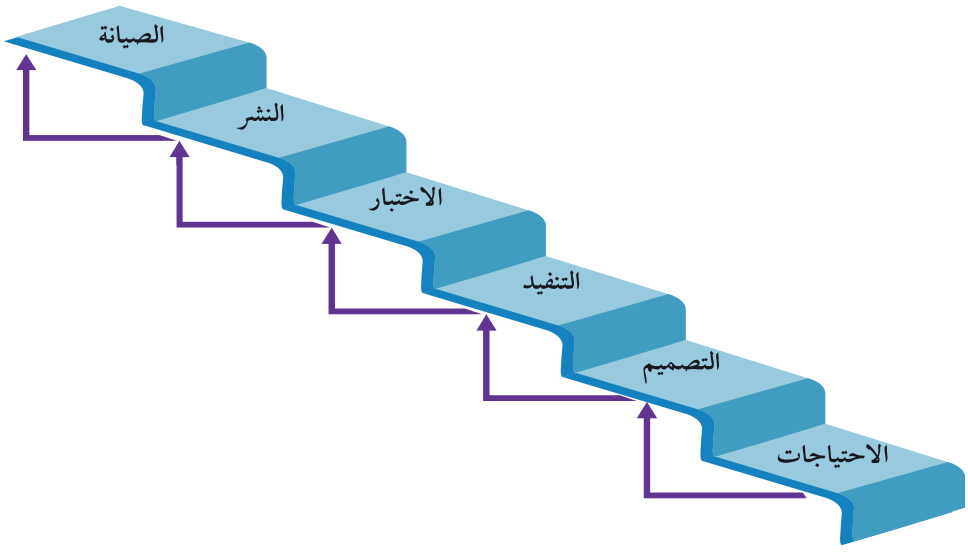
مزايا نموذج الشلّال (Waterfall):

يُمكِن إجمال مزايا نموذج الشلّال في ما يأتي:

1. **التسلسل الخَطِّي**: يمتاز نموذج الشلّال بأنَّه خَطِّي ومُتسلسِل؛ إذ تنتهي كل خطوة بصورة كاملة قبل الانتقال إلى الخطوة التي تليها.
2. **التوثيق الجيِّد**: يراعى في نموذج الشلّال التوثيق الجيِّد لجميع مراحل المشروع؛ لضمان سَيْر العمل بصورة صحيحة، وتمكين الفريق من تحقيق أهداف المشروع على نحوٍ واضح ودقيق.
3. **الجودة العالية**: تخضع جميع مراحل المشروع في نموذج الشلّال لاختبارات خاصة، فضلًا عن ملاحظة جودة كل مرحلة ومتابعتها؛ للتأكُّد أنَّها تعمل وَفق ما هو مُخطَّط له.
4. **التخطيط الدقيق**: يرتكز نموذج الشلّال في عمله على التخطيط الدقيق للمشروع، ويشمل ذلك تحديد نطاق المشروع، والجداول الزمنية، والمُنتَجات النهائية التي تخضع للمراقبة والمتابعة طوال دورة حياة المشروع.

مراحل نموذج الشلّال (Waterfall):

يتألَّف نموذج الشلّال من ست مراحل يُبيِّنها الشكل (1-3).



الشكل (1-3): مراحل نموذج الشلّال (Waterfall).

مُحدِّدات استخدام نموذج الشلّال (Waterfall):

1. **الجمود وعدم المرونة**: ليس من السهل في نموذج الشلّال إجراء تغييرات على المُتطلَّبات أو التصميم بعد الانتهاء من المرحلة ذات الصلة.
2. **عدم ملاءمة المشروعات الديناميكية**: لا يُناسِب نموذج الشلّال المشروعات التي قد تتغيَّر فيها المُتطلَّبات بمرور الوقت، وتُكتشَف فيها الأخطاء مُتأخِّرًا.
3. **التأخُّر في تقديم المُنتَج النهائي**: لا يُمكِن للمُستخدِمين الحصول على أيِّ نتائج إلّا بعد انتهاء جميع المراحل.
4. **صعوبة التعامل مع المشروعات الكبيرة والمشروعات المُعقَّدة**: قد يؤدّي استخدام نموذج الشلّال في المشروعات الكبيرة والمشروعات المُعقَّدة إلى ظهور مشكلات في عمليتي التتبُّع والتنفيذ إذا كانت التفاصيل كثيرة ومُتشعِّبة.

أُناقِش

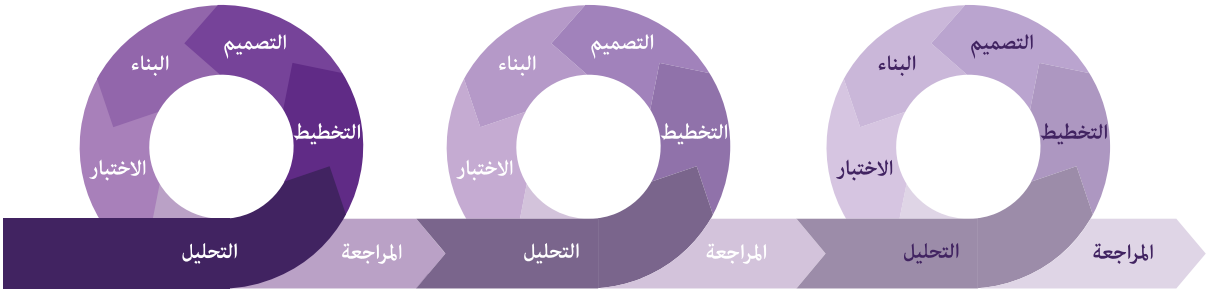
أُناقِش أفراد مجموعتي في الحالات التي يُمكِن فيها استخدام نموذج الشلّال، مُبيِّنين أسباب ذلك، ثمَّ نعرض ما نتوصَّل إليه من نتائج على أفراد المجموعات الأُخرى، ونعمل على مناقشتها معًا.

2- النموذج الرشيق (Agile):

يُقسَّم المشروع بحسب هذا النموذج إلى مجموعة من التكرارات (أو المراحل) الصغيرة التي تُسمّى (Iteration) أو (Sprint). وهذه التكرارات تتبع مسارًا خَطِّيًّا، وفيها ينتهي كل تكرار (أو مرحلة) بانتهاء المُدَّة الزمنية المُحدَّدة له، التي قد تمتدُّ من أسبوع واحد إلى أربعة أسابيع، لتبدأ بعدها مراجعة ما تحقَّق من مُتطلَّبات، وتلقّي التغذية الراجعة؛ لتعرُّف ما يجب إدخاله من تعديلات في المشروع أوَّلًا بأوَّل، وتحديد المُتطلَّبات التي يتعيَّن نقلها إلى المرحلة التالية؛ ما يُسهِم في منح النظام مرونة أكثر عند تطويره، ويجعله أكثر قدرة على التكيُّف مع المُتغيِّرات. ولهذا يُطلَق على النموذج الرشيق اسم (Change Driven)؛ أيْ مُوجَّه بالتغيير، فهو يُستخدَم في الحالات التي تتطلَّب تكيُّفًا سريعًا مع كل مُتغيِّر، علمًا بأنَّ لهذا النموذج العديد من المنهجيات التي تقوم جميعها على المبدأ نفسه بالرغم من وجود اختلافات بسيطة في ما بينها.

مراحل النموذج الرشيق (Agile):

يَمُرُّ كل تكرار (Sprint) في هذا النموذج بخمس مراحل، هي: البناء، والتصميم، والتخطيط، والتحليل، والاختبار. أنظر الشكل (1-4).



الشكل (1-4): مراحل النموذج الرشيق (Agile)، وخطوات كل مرحلة.

مبادئ النموذج الرشيق (Agile):

يعتمد هذا النموذج على مجموعة من المبادئ التي تُعزِّز التعاون والتطوير المستمر؛ لضمان تحقيق جميع مُتطلَّبات العميل واحتياجاته بفاعلية وسرعة. ومن أبرز هذه المبادئ:

- **الحفاظ على اتِّصال وثيق بالعميل،** والحرص على إشراكه في كل مرحلة من المراحل؛ لضمان فهم واضح لمُتطلَّباته واحتياجاته، والوقوف على مستوى التقدُّم في سَيْر العمل، وإعادة تقييم المُتطلَّبات والاحتياجات.
- **التوجُّه نحو نشر البرامج بصورة مُتكرِّرة بدلًا من الاعتماد على التوثيق الشامل،** وتسليم إصدارات مُتزايِدة خلال مُدَد زمنية قصيرة (بضعة أسابيع).
- **وجوب استخدام فِرَق عمل يَتَّصِف أفرادها بالتميُّز والكفاءة والقدرة على التواصل الفاعل في ما بينهم،** إضافةً إلى عقد اجتماعات دورية؛ لمناقشة مستوى التقدُّم الذي تحقَّق، وضمان التنسيق بين الأطراف جميعًا.

خصائص العمليات في النموذج الرشيق (Agile):

تمتاز العمليات في هذا النموذج بخصائص عِدَّة، أبرزها:

- **المرونة:** يُمكِن للعمليات في النموذج الرشيق أنْ تتكيَّف مع المُتغيِّرات الفنية والمُتغيِّرات البيئية للنظام.
- **التطوير التدريجي:** تخضع العمليات في النموذج الرشيق للتطوُّر والتحسين بصورة تدريجية مستمرة.
- **التفاعل مع العميل:** تتيح العمليات في النموذج الرشيق استخدام ملاحظات العميل في تعديل النظام البرمجي وَفقًا لمُتطلَّباته واحتياجاته.
- **السرعة:** تُسلَّم التعديلات الخاصة بالعمليات في وقت قصير لتحقيق قيمة مضافة بسرعة.

أُناقِش

أُناقِش زملائي/ زميلاتي في الحالات التي يُمكِن فيها استخدام النموذج الرشيق، مُبيِّنين أسباب ذلك، ثمَّ نبحث معًا عن إجابات للأسئلة الآتية:

- هل يُمكِن تطبيق النموذج الرشيق على جميع أنواع المشروعات؟
- ما التحدِّيات والمُعوِّقات التي قد تَحول دون استخدام هذا النموذج في المشروعات الكبيرة والمشروعات المُعقَّدة؟
- كيف يُمكِن ضمان الاستفادة الفاعلة من هذا النموذج في حال كان تفاعل العملاء محدودًا؟

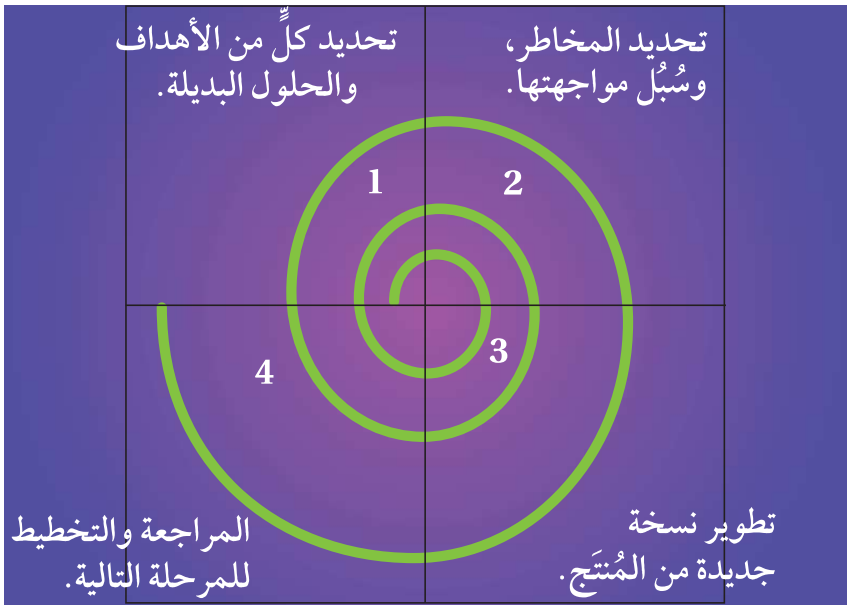
أُشارِك زملائي/ زميلاتي في آرائي المُتعلِّقة بهذا النموذج، وأستمع إلى وِجهات نظرهم المختلفة، مُقدِّمين أمثلة واقعية لدعم النقاش وإثرائه.

3- نموذج الحلزون (Spiral):

يُعَدُّ هذا النموذج واحدًا من أشهر النماذج المُستخدَمة في المشروعات، وهو يمتاز بشكله الحلزوني، ووجود حلقات مُتعدِّدة فيه، يختلف عددها من مشروع إلى آخر، وتُمثِّل كلٌّ منها مرحلة من مراحل النموذج. يتبع نموذج الحلزون نهجًا تكراريًّا كما هو الحال في النموذج الرشيق (Agile)، لكنَّه يُوفِّر - في الوقت نفسه- نهجًا مُنظَّمًا لإدارة المخاطر في المشروعات المُعقَّدة التي تتطلَّب دراسة دقيقة للمخاطر التقنية والمخاطر التشغيلية.

مراحل نموذج الحلزون (Spiral):

يُبيِّن الشكل (1-5) المراحل التي يَمُرُّ بها نموذج الحلزون.



الشكل (1-5): مراحل تطوير البرمجية باستخدام نموذج الحلزون (Spiral).

مواجهة المخاطر في نموذج الحلزون (Spiral):

تُعرَّف المخاطرة بأنَّها أيُّ عامل قد يُؤثِّر سلبًا في نجاح المشروع.

تُستخدَم في نموذج الحلزون استراتيجية فاعلة للتعامل مع المخاطر ومواجهتها؛ لضمان نجاح المشروع. وتتمثَّل أهمية هذه الاستراتيجية في ما يأتي:

1. **التحديد المستمر للمخاطر:** يُمكِن تحديد المخاطر المُحتمَلة في كل مرحلة من مراحل النموذج بصورة دورية؛ ما يساعد على اتِّخاذ الإجراءات اللازمة لمعالجة هذه المخاطر في الوقت المناسب.
2. **توفير النماذج الأوَّلية:** يتمثَّل ذلك في إنشاء نموذج أوَّلي لكل مرحلة من مراحل تطوير البرمجيات؛ ما يتيح الكشف المُبكِّر عن المشكلات والمخاطر والمُعوِّقات، وإيجاد الحلول المناسبة لها قبل الانتقال إلى المرحلة التالية.
3. **تحييد المخاطر المعروفة مُسبَّقًا:** تتيح هذه الاستراتيجية التعامل الفاعل مع المخاطر التي حُدِّدت قبل البَدْء بتطوير البرمجيات، علمًا بأنَّ النماذج الأوَّلية قد لا تكون كافية لمواجهة المخاطر غير المُتوقَّعة التي قد تظهر أثناء عملية التنفيذ.

يُعَدُّ نموذج الحلزون (Spiral) مناسبًا للاستخدام في الحالات الآتية:

1. **المشروعات الضخمة**: يُستخدَم نموذج الحلزون في المشروعات الكبيرة التي تتطلَّب تخطيطًا وتنفيذًا دقيقين، واختبارًا مستمرًّا للمخاطر في مراحل مُتعدِّدة.
2. **الإصدارات المُتكرِّرة**: يُستخدَم نموذج الحلزون عند الحاجة إلى إصدار نسخ مُتكرِّرة من المُنتَج بصورة دورية.
3. **النماذج الأوَّلية**: تتطلَّب عملية تطوير البرمجيات أحيانًا إنشاء نموذج أوَّلي؛ لفهم المُتطلَّبات اللازمة، أو تقييم الحلول المُمكِنة.
4. **تقييم المخاطر**: قد يكون تحليل المخاطر وتقييمها جزءًا أساسيًّا من عملية التطوير، كما في المشروعات العسكرية، والمشروعات المالية، والمشروعات الطبية.
5. **المُتطلَّبات المُعقَّدة والغامضة**: يُفضَّل استخدام نموذج الحلزون في المشروعات ذات المُتطلَّبات غير الواضحة أو المُعقَّدة التي قد تتغيَّر مع تقدُّم سَيْر العمل في المشروع.

نشاط جماعي

في ما يأتي مجموعة من الحالات العملية لمشروعات مختلفة. أقرأ كل حالة بتدبُّر ورَوِيَّة، ثمَّ أُحلِّلها بالتعاون مع أفراد مجموعتي؛ لتحديد النموذج الأنسب لتطوير النظام مع التعليل:

1- شركة مُتخصِّصة في تطوير البرمجيات تعمل على تنفيذ مشروع يتضمَّن تصميم نظام لإدارة الامتحانات المدرسية، علمًا بأنَّ مُتطلَّبات النظام واضحة وثابتة منذ بَدْء المشروع، ولا يُتوقَّع تغييرها أثناء عملية التطوير.

2- فريق مُتخصِّص في تطوير تطبيقات الهواتف الذكية يعمل على إنشاء تطبيق للتسوُّق الإلكتروني، وهو مشروع يتطلَّب تكرارًا مستمرًّا لتجربة الوظائف الجديدة بناءً على آراء المُستخدِمين الذين يختبرون التطبيق في كل مرحلة من مراحل التطوير.

3- شركة طيران ترغب في تطوير نظام لحجز تذاكر الطيران يَتَّسِم بالتعقيد الكبير، ويشمل العديد من المخاطر المُتعلِّقة بالأمان والدِّقَّة. وهذا المشروع يتطلَّب تقييم المخاطر وتذليلها بصورة مُتكرِّرة قبل الانتقال إلى المراحل التالية.

4- شركة ناشئة تعمل على تطوير مُنتَج تجريبي جديد بناءً على أفكار غير مُكتمِلة، مع توقُّع حدوث تغييرات مُتكرِّرة بناءً على ملاحظات العملاء عند الاستخدام.