هـيـئـة الحكومة الرقمـيـة
Digital Government Authority

الدليل الاسترشادي
لمنهجيـــة دورة حيـــاة تطويـر النظـم (SDLC)

يونيو 2024

نـــوع الوثيقـــــة: **دليل استرشادي**
تصنيف الوثيقـــة: **عـام**
رقـــم الإصـــــدار: **1.0**
رقـــم الوثيقـــــة: **DGA-1-2-5-229**

المحتويات

1. مقدمة

تعـي هيئـة الحكومـة الرقميـة أهميـة إقـرار التنظيمـات وتحـديثها باسـتمرار لـمواكبة المتطلبـات الحاليـة والمسـتقبلية، وللمسـاهمة بشـكل رئيسـي فـي تعزيـز الأداء الرقمـي داخل الجهات الحكومية، والرفع من جودة الخدمات المقدمة، وتحسين تجربـة المسـتفيد من تلك الخدمات، بما يتوافق مع الرؤية الطموحة للمملكة 2030، ويتـواءم مـع التوجهـات الاستراتيجية للحكومة الرقمية التي تؤكد على أهمية تـوفير بيئـة تنظيميـة فعّالـة ومرنـة تتكيـف مـع التغييـرات المسـتقبلية. وتمهّـد الهيئـة الطريـق للجهـات الحكوميـة لتـوفير خدمات حكومية رقمية ذات جودة وكفاءة عالية تساهم في رفع العوائد الاستثمارية والرفع من قيمة الاقتصاد الوطني.

وانطلاقاً مـن دور الهيئة في دعم وتمكين الجهـات الحكوميـة من خلال إرشادها بتطبيـق أفضـل الممارسـات فـي المجـالات المتعلقـة بالحكومـة الرقميـة، ولتحقيـق أهـداف رؤيـة المملكة 2030 والأهـداف الاسـتراتيجية للتحـول الرقمـي الحكـومي، أعـدت الهيئـة "الـدليل الاسترشادي لمنهجية دورة حياة تطوير النظم (SDLC)"؛ لإرشاد الجهات الحكومية إلى تبني المنهجيات الرقمية لمواجهـة التحـديات، وتحسـين الأداء لرفـع جـودة الخـدمات الرقميـة، وتعزيزاً لرضا المستفيدين، وتحقيقاً للتميز والاستدامة الرقميـة. وتتضـمن الوثيقـة نظـرة عامة إلى منهجية ونماذج دورة حياة تطـوير الـنظم (SDLC)، والأدوات والخطـوات المتبعـة لتطوير وتشغيل البرمجيات.

2. أهداف الدليل الاسترشادي

يساهم الدليل الاسترشادي في تحقيق ما يلي:

- دعم الجهات الحكومية من خلال إرشادها بخطوات استخدام منهجية SDLC (Software Development Life Cycle).
- تعزيز تبني منهجية (SDLC).
- تحسين أداء الجهات الحكومية في مجال تطوير النظم والبرمجيات.
- رفع جودة الخدمات والمنتجات الحكومية الرقمية.

3. نطاق الدليل الاسترشادي

يسهم الدليل في فهم منهجية دورة حياة تطوير النظم (SDLC) وخطوات تبنيها عبر تسليط الضوء على الجوانب التالية:

- منهجية دورة حياة تطوير النظم (SDLC).
- أهداف منهجية دورة حياة تطوير النظم (SDLC).
- مراحل تطبيق دورة حياة تطوير النظم (SDLC).
- النماذج الداعمة لتطبيق منهجية دورة حياة تطوير النظم (SDLC).
- توصيات عامة لتبني منهجية تطوير النظم (SDLC)
- الاستمرارية وقياس الأثر.

4. الفئات المستهدفة

يستهدف هذا الدليل الخبراء والممارسين والموظفين التقنيين الذين يعملون على تطوير الأنظمة والبرمجيات في الجهات الحكومية.

5. بيان الدليل الاسترشادي

5.1 منهجية دورة حياة تطوير النظم (SDLC)

هي نهج لتطوير البرمجيات، وتحتوي على العمليات والخطوات المستخدمة لبنائها، وتتناول عدة مفاهيم مثل: تحليل المتطلبات والتصميم والتطوير والاختبار والنشر والتشغيل، كما أن من أهم نتائجها: دعم التعاون الفعّال بين مطوّري البرمجيات في عملية بناء الحل التقني الذي يلبي احتياجات الأعمال، والذي بدوره سيساهم في رفع مستوى الجودة وخفض التكلفة.

5.2 أهداف منهجية دورة حياة تطوير النظم (SDLC)

تعتبر منهجية دورة حياة تطوير النظم (SDLC) منهجية نوعية من ناحية شموليتها وقيمتها. فتطبيقها يؤدي إلى تحقيق خمسة أهداف رئيسية:

تطوير أسرع

يساهم تطبيق المنهجية في تقليل الوقت المستهدف لتطوير النظم والتطبيقات.

ضمان نجاح تطوير البرمجيات

اتباع المنهجية يضمن نجاح التطوير من خلال التركيز على فهم وتحليل الغرض من الحل التقني ومتطلباته بشكل ممنهج.

تكلفة أقل

استخدام المنهجية بشكل صحيح يساعد في رفع الكفاءة وتقليل تكاليف التطوير.

جودة أعلى

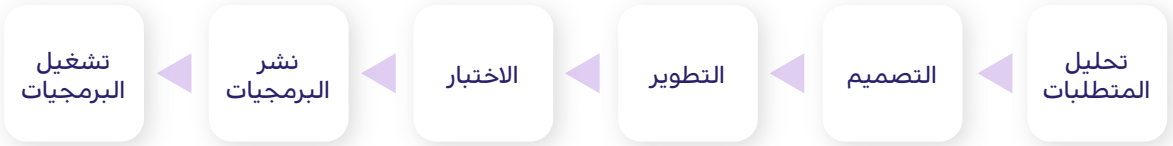
اتباع المنهجية في التطوير يعمل على رفع جودة المخرجات.

دعم التعاون الفعّال

تشجع المنهجية على التعاون بين مطوّري البرمجيات في عملية إيجاد حل تقني ملائم يلبي احتياجات الأعمال من خلال عملية منظمة.

5.3 مراحل تطبيق منهجية دورة حياة تطوير النظم (SDLC)

تتكون منهجية دورة حياة تطوير النظم الشكل (1) من ست مراحل أساسية:



الشكل (1) مراحل تطبيق منهجية دورة حياة تطوير النظم (SDLC)

تحليل المتطلبات

وذلك من خلال دراسة الاحتياج الحالي، والتخطيط للنظام أو البرنامج الجديد ومتطلبات العمل عليه.

التصميم

تصميم النظام أو البرنامج المقترح، وهندسة الحل التقني.

التطوير

بناء النظام أو البرنامج الجديد، وتحويل التصور إلى برمجيات حاسوبية (CSCI).

الاختبار

يتم اختبار جميع جوانب النظام أو البرنامج الجديد، والتأكد من تحقيقها للتصور الأولي للنظام.

نشر البرمجيات

يتم نشر النظام أو البرنامج في بيئة الإنتاج للبدء بالاستخدام.

تشغيل البرمجيات

ويتم فيها الصيانة والمراقبة والتحسين لضمان الاستمرارية.

وفيما يلي تفصيل وشرح مراحل تطبيق منهجية دورة حياة تطوير النظم (SDLC):

5.3.1 تحليل المتطلبات

عملية وصف سلوك وخصائص وسمات النظام المطلوب إنشاؤه:

متطلبات الأعمال Business Requirements (BR)

توضحّ غايات الأعمال (النتائج الرئيسية الأوسع نطاقاً التي يجب تركيز الجهود والإجراءات نحوها)، وأهداف الأعمال (الخطوات القابلة للقياس لتحقيق الغايات)، وأدوار الأعمال واحتياجات الأقسام (متطلبات الأنشطة التي تنفذها المنظمة)، وعمليات الأعمال (الأنشطة والمهام اللازمة لتحقيق أهداف المنظمة).

متطلبات المستخدمين Users Requirements (UR)

توضحّ احتياجات المستخدم والأنشطة المتوقعة للمستخدم ضمن الحل التقني، شاملة تجربة المستخدم التي تحرك تفاعل المستخدم وإمكانية الوصول (تتناول الجوانب المميزة)، وإمكانية الاستخدام (فعالية التصميم) من خلال حصر احتياجات المستخدمين.

المتطلبات الوظيفية Functional Requirements (FR)

توضِّح القدرات والمزايا التشغيلية المتوقعة للحل التقني، وتتضمن:

- **الميزات وقصص المستخدم (User Story)**: المتطلبات المكتوبة من منظور مستخدم نهائي يصف احتياجاته.
- **حالات الاستخدام**: متطلبات التفاعل مع الحل التقني لتلبية احتياجات المستخدم النهائي.
- **المتطلبات العامة (Epic)** : متطلب عام للأعمال التي يمكن تقسيمها إلى متطلبات المشروع.

المتطلبات الفنية Technical Requirements (TR)

تصف الجوانب الفنية للحل التقني الذي يجب اعتماده، ويشمل الموثوقية وقابلية التوسع والأمان والأداء والتكامل وإمكانية التشغيل التفاعلي والمعايير والبُنى والواجهات والنشر والنقل والبيئة والسلامة والعوامل البشرية، وتغطي جميع القيود الفنية التي تعيق بناء الحل التقني وتصميمه ونشره وتشغيله.

متطلبات الأنظمة (System requirements)

تركز على الجانب الخاص بالأنظمة والتطبيقات.

وهناك أربعة أنشطة رئيسية في عملية تحليل المتطلبات السابقة، وهي كما يلي:

تحليل احتياجات المستخدمين

تقدير مدى احتمالية تلبية احتياجات المستخدم المحدّدة باستخدام البرمجيات وتقنيات الأجهزة الحالية، وقد تتم من خلال عدة أنواع من الدراسات منها دراسة تحليل الأثر من تنفيذ هذه المتطلبات، وكذلك التأثير على النظم الحالية (Impact analysis).

استنتاج المتطلبات

استخلاص المتطلبات من خلال ملاحظة الأنظمة الحالية والمناقشات مع المستخدمين المحتملين، وعقد ورش عمل حول المتطلبات، وإنشاء قصص مصورة (Storyboards)، وما إلى ذلك.

مواصفات المتطلبات

تحويل المعلومات التي تم جمعها إلى "وثائق متطلبات" (متطلبات الأعمال ومتطلبات المستخدمين والمتطلبات الوظيفية والمتطلبات الفنية).

التحقّق من المتطلبات

التأكد من اكتمال الوثائق المُنشأة ودقتها وفقًا لاحتياجات المستخدم، ويمكن أيضا القيام بدراسة جدوى تنفيذ المتطلبات (Feasibility assessment).

5.3.2 التصميم

عملية تصور للحل التقني تعكس المتطلبات التي تم جمعها وتحليلها، حيث يعرّف الحل التقني بأنه مجموعة من المكوّنات المرتبة على مستويات (الشكل 2)، وفقا لما يلي:

مستوى تجربة المستخدم

تشمل إدارة التفاعل مع المستخدم النهائي عبر المتصفح أو التطبيقات باستخدام مجموعة من الخدمات التي يشملها مستوى الخدمة.

مستوى الخدمة

تشمل أنماط البنية الملائمة، وتحديد آليات التواصل لاستنتاج المزايا التشغيلية.

مستوى طبقة الأعمال وسير العمل

المستوى الذي تُدار من خلاله المزايا التشغيلية.

مستوى البيانات والتحليل

تشمل التحليل والبيانات الناتجة من العمليات.

مستوى البنية التحتية

يشمل جميع الأجهزة والشبكات والأدوات غير البرمجية للحل التقني.

المستويات المساندة

يوجد مستويات إضافية مساندة (مستوى التكامل، مستوى المعرفة، مستوى الأمان...) والتي تعتبر ذات قيمة مضافة، وتدعم بناء حل تقني كامل وقوي وملائم.

الشكل (2) مكونات ومستويات الحل التقني

وتشمل علميات التصميم ما يلي:

هندسة الحل التقني

والتي تركز على تحليل البرمجيات إلى طبقات ووحدات أساسية ووحدات فرعية، وتسلط الضوء على سلوكها وتفاعلها.

تصميم الحل التقني

يركز على حل المشاكل بالاعتماد على حلول تصميمية يُمكن إعادة استخدامها وصيانتها، وتتمتع بالمرونة والقوة والانتشار لضمان حل مشاكل التصميم الشائعة.

وتتضمن عملية التصميم العديد من الأنشطة لتحقيق هندسة وتصميم حلول تقنية تتسم بالاتساق والشمولية.

وفيما يلي توضيح لأنشطة عملية التصميم الرئيسية:

تصميم تجربة وواجهة المستخدم

والتي تغطي جميع جوانب تفاعل المستخدم النهائي والحل التقني، وهو يعزز القرارات المتخذة بشأن دراسة التفاعلات والتنقل وإمكانية الوصول للخدمات حسب فئة وصلاحية المستخدم.

التصميم الهندسي

هو ما يحدد القرارات الخاصة ببنية البرمجيات وطبقاتها ووحداتها وكيفية تصميم الواجهات بين الوحدات النمطية (Modules).

تصميم المكوّنات

تتم من خلال تخصيص الخدمات وعرض الواجهات على المكوّنات الأخرى.

تصميم بنية البيانات

يتم من خلاله تصميم نماذج وبُنى البيانات.

تصميم الخوارزميات

يتم من خلاله تغطية القرارات الخاصة بالخوارزميات لاستيفاء الخدمات التي يقدّمها كل مكوّن أو جزئية من النظام.

تصميم البرمجيات

هو عملية متكررة لا تقل عن ثلاث دورات تُنتج عدداً من التصاميم على النحو التالي:

- التصميم العام (HLD) : يُعنى التصميم العام غالبًا بطبيعة المكوّنات وآلية تفاعلها وفقًا لهندسة الحل التقني وتصميمه.
- التصميم الشامل (LLD): يُعدّ التصميم الشامل تنقيحًا أوليًّا للتصميم العام، ويقدّم التفاصيل والتعريفات للمنطق الفعلي لكل مكوّن، ويتعامل مع أدوات البناء ولغات البرمجة وأطر العمل والمكتبات والمنتجات، وغيرها من الأدوات (من المكوّنات المُحتفظ بها)، وبُنى البيانات وخوارزميات الأداء والأمان.
- التصميم التفصيلي (DLD) : هو مرحلة ثانية لتنقيح التصميم، ويتعامل مع معالجة الأخطاء وقرارات تنفيذ الخوارزميات والتسلسلات المنطقية وعمليات تحسين بنية البيانات.

5.3.3 التطوير

تطوير البرمجيات هي عملية تحويل تصميم البرمجيات إلى عنصر تكوين برمجيات حاسوبية (CSCI) فعّال وموثق ومتكامل وجاهز لدخول مرحلة اختبار البرمجيات، ويتشارك في عملية التطوير مطوّرو البرمجيات ومصممو الرسوم الذين يعملون معًا لبناء عنصر تكوين برمجيات حاسوبية مطابق للمتطلبات، وتتضمن عملية التطوير ما يلي:

- تحسين التصميم التفصيلي للتعامل مع المزيد من قرارات تنفيذ الخوارزميات وبنية البيانات.
- إجراء العملية الإبداعية لتصميمات الرسوم التي تلبي متطلبات المستخدم (تجربة المستخدم وواجهة المستخدم، وغيرها) وتتوافق مع قرارات تصميم البرمجيات.
- كتابة الشفرة المصدرية ووحدات التحويل البرمجي وملفات التكوين وموارد البرمجيات، ومشاركتها مع المطورين الآخرين باستخدام أدوات إدارة الشفرة المصدرية.
- كتابة الشفرة البرمجية للأتمتة، وذلك لتسريع عملية التطوير، أو تمكين التطوير المستمر.
- كتابة شفرة البرنامج النصي للتفاعل مع بيئات أنظمة التشغيل، وقواعد البيانات، والخوادم، وغيرها من مكوّنات الحل التقني.
- تحديد المكتبات وأطر العمل والأدوات والمنتجات المتوافقة مع قرارات التصميم، والاستفادة منها.
- تصميم اختبارات الوحدة (Unit tests) للتحقّق من جودة الشفرة مقابل المتطلبات وقرارات التصميم.
- كتابة الوثائق الفنية باستخدام الأساليب القائمة على الأكواد.
- اختبار وحدة الأدوات المبنية عن طريق إجراء مجموعات اختبار الوحدة وحالات اختبار الوحدة باستخدام أطر اختبار الوحدة.
- دمج مكونات البرمجيات وتجميعها ضمن مكونات وتطبيقات ومنتجات وحلول أكبر.
- كتابة اختبارات التكامل الآلية، وإجراؤها لتأهيل مجموعات الوحدات النمطية (Modules) للبرمجيات المدمجة منطقيًّا، والتأكد من تأثير الدمج على السلوك الفردي لكل وحدة من الوحدات النمطية (Modules).
- تجميع الأدوات ضمن حزمة قابلة للنشر والتركيب و/ أو التنفيذ، ويمكن مشاركتها كعنصر تكوين برمجيات حاسوبية (CSCI).
- إعداد النماذج الأولية واختبار مكوّنات البرمجيات لحل تعقيدات التنفيذ ومخاطره، أو بناء إثبات المفهوم (POC).

- إصلاح المشاكل والأخطاء في البرمجيات الحالية، وبناء إصدارات أو حزم إرسال جديدة للنشر في بيئة التشغيل.
- تحديد وإدارة الإصدارات، ويشمل ذلك: إصدارات الملفات، وإصدارات الوحدة النمطية، وإصدارات إنشاء تسلسل وحدة التحويل البرمجي.
- متابعة تعديلات وإصدارات الشفرة المصدرية من خلال أدوات دورة حياة تطوير النظم، مثل: قوائم المهام، وقوائم وأدوات متابعة المشاكل، وإضافات الميزات الوظيفية.

5.3.4 الاختبار

اختبار البرمجيات هو عملية التحقّق مما إذا كان البرنامج الذي تم تطويره في المرحلة السابقة، يطابق المتطلبات المتوقعة أم لا، وعملية ضمان خلوه من الأخطاء البرمجية. وتكون عمليات الاختبار في بيئات مختلفة عن بيئة الإنتاج الرئيسية، ولكن يجب أن تكون مطابقه لها، وتكون مخصصة للاختبار، وتشكّل عملية الاختبار جزءًا من عملية شاملة لضمان الجودة ومراقبتها للعمليات التي تضمن التوافق مع المتطلبات، ويشمل اختبار البرمجيات الأنشطة الرئيسية التالية:

تخطيط الاختبار ومراقبته

يتضمن إنشاء وثيقة تصف النهج العام وأهداف الاختبار.

تحليل الاختبار وبناء خطواته:

يتضمن هذا الإجراء المزيد من التحليل لتصميم خطوات الاختبار عن طريق:

- مراجعة أساس الاختبار (المتطلبات ومواصفات التصميم وتحليل مخاطر المنتج والبنية والواجهات).
- تحديد ظروف الاختبار اللازمة لتطبيق الاختبار.
- تصميم ظروف بيئة الاختبار، وتحديد البنية التحتية والأدوات المطلوبة.
- تصميم حالات الاختبار (Test Cases).

تنفيذ الاختبار

يتضمن إجراء الاختبار المحدّد يدويًّا أو باستخدام أداة اختبار آلية، ويشمل ذلك الأنشطة التالية:

- تنفيذ حالات الاختبار التي تم تصميمها وتحديثها وترتيب أولوياتها.
- إعادة تنفيذ الاختبارات التي فشلت سابقًا.
- تسجيل نتائج تنفيذ الاختبار، شاملة الاختبارات التي فشلت سابقًا.
- مقارنة النتائج الفعلية بالنتائج المتوقعة.

مراقبة الاختبار

تتضمن مقارنة التقدّم الفعلي بخطة الاختبار، وإعداد التقارير بالحالة، بما يشمل حالات الانحراف عن خطة الاختبار.

إعداد معايير التوقف

تتضمن اتخاذ القرار بشأن العملية التي تحدّد توقيت التوقف عن إجراء الاختبار، وإعداد تقارير الاختبار، وتقييم مدى الحاجة لإجراء المزيد من الاختبارات.

إعداد معايير إنهاء واكتمال الاختبار

بناء معايير اكتمال الاختبار وجاهزية البرمجيات للتنفيذ.

وتتكون عمليات الاختبار من اختبارات وظيفية وغير وظيفية، موضحة على النحو التالي:

الاختبار الوظيفي

يتعلق بالمتطلبات الوظيفية، ويُعنى بتأهيل وحدات البرمجيات المبنية وعنصر تكوين البرمجيات الحاسوبية (CSCI) مقابل المتطلبات الوظيفية ومتطلبات المستخدم.

الاختبار غير الوظيفي

يعنى بتأهيل وحدات البرمجيات المبنية وعنصر تكوين البرمجيات الحاسوبية (CSCI) مقابل المتطلبات الفنية.

ويُمكن إجراء الاختبارات الوظيفية وغير الوظيفية يدويًّا أو بشكل آليّ أو شبه آليّ موضحة على النحو التالي:

الاختبار اليدوي

ويتضمن الاستخدام الشخصي للبرمجية كما يفعل المستخدم النهائي، مع اتباع خطوات محدّدة في حالات الاختبار ومجموعات الاختبار لمقارنة النتائج المتوقعة بالنتائج الفعلية، واكتشاف حالات الفشل والمشاكل المحتمل تكرارها.

الاختبار الآلي

كتابة البرنامج النصي، أو تهيئة أدوات الاختبار للتشغيل الآلي مقابل عنصر تكوين البرمجيات الحاسوبية، ومحاكاة تفاعل المستخدم أو التطبيق في حالات اختبار مصممة ومحدّدة مُسبقًا.

الاختبار شبه الآلي

يشمل كلا من: الأدوات الآلية، والاختبار اليدوي.

5.3.5 نشر البرمجيات

عملية إعداد أحد التطبيقات والأنظمة للعمل والتشغيل في بيئة محدّدة، ويتضمن التثبيت والتهيئة لضمان التشغيل الأمثل للبرمجية، كما يتضمن الدمج والنقل والتحديث والتعطيل وإلغاء التثبيت وتعقب الإصدار وتشغيل العديد من الإصدارات لنفس البرمجية في آنٍ واحد، ويشمل ذلك التحقق من توافقها مع البنية التحتية للشبكة والأجهزة.

في بداية عملية النشر، يتم إعداد نسخة من بيئة الإنتاج، وتُسمى "ما قبل الإنتاج"، ويجري عليها عملية التحقّق من النشر وعمليات الإصدارات الجديدة للمنتج قبل التطبيق الفعلي على بيئة الإنتاج. وفي حال فشل النشر، يتم القيام بنشاط "التراجع عن العملية"، والذي يشمل عملية إعادة نشر آخر إصدار كان يعمل من المنتج.

وهناك عدة انواع لنشر البرمجيات:

النشر الداخلي (On-premise Deployment)

يشير إلى عمليات النشر الداخلية في الجهة مع التحكم الكامل في إدارة البرنامج والتكامل مع المكوِّنات الأخرى، وتتيح عمليات النشر الداخلية إمكانية الوصول سواءً عبر الإنترنت أو في وضع عدم الاتصال، ومن المهم في هذا النوع من النشر مراعاة الاعتبارات الخاصة بالشبكات والأمان.

النشر السحابي (Cloud Deployment)

يشير إلى عمليات النشر الخارجية مع التحكم الكامل في إدارة البرنامج والتحكم المحدود في التكامل مع المكوِّنات الأخرى على حسب نوع الخدمة، ويتم الوصول للبرنامج أو النظام عبر الإنترنت مع مراعاة متطلبات المُستضيف. ويتولى المستضيف التعامل مع الاعتبارات الخاصة بالشبكات والأمان وقابلية الوصول. وينقسم النشر السحابي إلى أنواع مختلفة، موضحة على النحو التالي:

- تقديم البنية التحتية كخدمة (IaaS: Infrastructure as a service)
- تقديم الوظائف كخدمة (FaaS: Function as a service)
- تقديم البرمجيات كخدمة (SaaS: Software as a service)
- تقديم البيانات كخدمة (DaaS: Data as a service)
- تقديم المنصة كخدمة (PaaS: Platform as a service)

والنشر السحابي يوفر المزيد من المرونة، وقابليّة التكيُّف، والأمان، وإمكانية الإدارة، وقابلية التوسع.).

النشر الافتراضي (Virtualized Deployment)

يشير إلى النشر في البيئات الافتراضية بصورة مستقلة عن البنية التحتية المادية.

النشر على الحاويات (Containerized Deployment)

يشير إلى نشر التطبيقات والمنتجات والحلول بجميع تبعياتها البرمجية، ويشمل ذلك الشبكات وتكوينات نظام التشغيل وأقسام القرص للمساعدة في تشغيل البرمجية في بيئة معزولة تمامًا (حاوية).

5.3.6 تشغيل البرمجيات

يمثل تشغيل البرمجيات مجموعة من الأنشطة لتحسين الكفاءات التشغيلية للبرامج، وزيادة الإنتاجية، وخفض التكاليف التشغيلية للأعمال عن طريق تذليل العقبات التشغيلية وأتمتة المهام المتكررة، والتي تحتوي على تفاصيل متشعبة. ومن بين أنشطة التشغيل الشائعة ما يلي:

تشغيل الحلول

وهي الدور الرئيسي للعملية التشغيلية، وتشمل: تحسين أداء الحل التقني، وقابلية الاستخدام.

إدارة بيئة التشغيل

تغطي البنية التحتية ونظام التشغيل والتكوين والشبكات والتخزين. وتنقسم إدارة بيئة التشغيل إلى عدة أنواع، موضحة على النحو التالي:

- **إدارة البنية التحتية**: تشمل إعداد البنية التحتية المطلوبة وتهيئتها لعمليات للنشر.
- **إدارة الإصدارات**: تشمل إدارة تعيين الإصدار وخصائص التكوين للبرمجية المراد نشرها.
- **إدارة الشبكات**: تشمل أي تكوين مرتبط بالشبكات مثل عناوين IP الافتراضية لقابلية الوصول العالية.
- **إدارة التخزين**: تشمل تكوين الخدمات المرتبطة بالتخزين وإدارتها.
- **إدارة خدمات الدليل**: تشمل تقسيم البيانات المخزنة للهوية، والمستخدمين وجزئيات البرنامج لدعم عمليات التحكم بصلاحيات الوصول.
- **دعم المستخدمين**: تشمل عمليات التفاعل والتواصل المباشر مع المستخدمين لاكتشاف مشاكل الاستخدام والأخطاء البرمجية المرتبطة بتكوين البنية التحتية أو التي تتطلب تدخلاً في عملية التطوير، ويتم تتبع هذا التفاعل من خلال مكتب الخدمة المساعدة.

التخفيف من حدة الكوارث

إزالة آثار المخاطر ومصادر الخطر أو الحد منها من خلال اتخاذ التدابير الاستباقية، مثل: إجراء النسخ الاحتياطي وقاعدة البيانات والتكوين، وذلك قبل وقوع الكارثة أو الحدث الطارئ، وبعد ذلك، اتخاذ تدابير استعادة الإصدار الأحدث من النسخ الاحتياطي لإتاحة التشغيل الطبيعي للحل التقني.

مراقبة الحل

ويشمل ذلك المراقبة المستمرة للتمكن من توقع أي عيب في تشغيل البرمجية، ويشمل ذلك:

- مراقبة الأداء: وهي مراقبة استخدام الموارد وزمن استجابة البرمجية للتحقّق مما إذا كان ذلك في نطاق السلوك المتوقع أم لا، وتفسير اتجاهات استخدام الموارد لتوقع الإجراءات اللازم اتخاذها لتجنب أي تعطل للخدمة.
- مراقبة السجلات: مراقبة مُخرجات السجل المفصل للحل أو التطبيق لاكتشاف أي حدث غير طبيعي بشأن خطأ متكرر أو غير معالج تكتشفه البرمجية.
- مراقبة البيانات: مراقبة كمية البيانات التي يستخدمها الحل لتخطيط توسعات القرص الممكنة.
- مراقبة الأمان والتهديدات: الاستفادة من الآليات والنظم والأدوات الأمنية لمواجهة أي تهديد أمني.

الإبلاغ والتنبيه والتحذير

عند مواجهة الحل أو التطبيق أو بيئته سلوكًا غير متوقع، فإنه يجب اتخاذ إجراء من الجهة المختصة (المطورين والأمن والمختبرين).

5.4 النماذج الداعمة لتطبيق منهجية دورة حياة تطوير النظم (SDLC)

هناك عدد متزايد من النماذج الداعمة لتطبيق دورة حياة تطوير النظم، ويشكل اختيار النموذج الأنسب فارقاً كبيرًا في تحقيق نتائج ناجحة عند قياسها من ناحية التكلفة، أو الالتزام بالتسليم في الموعد المحدد، أو تحقيق رضا العميل، أو سلامة وجودة الشفرات البرمجية.

ويُمكن تصنيف أنواع النماذج الداعمة لتطبيق دورة حياة تطوير النظم إلى نوعين رئيسين (الشكل 3) موضحة على النحو التالي:

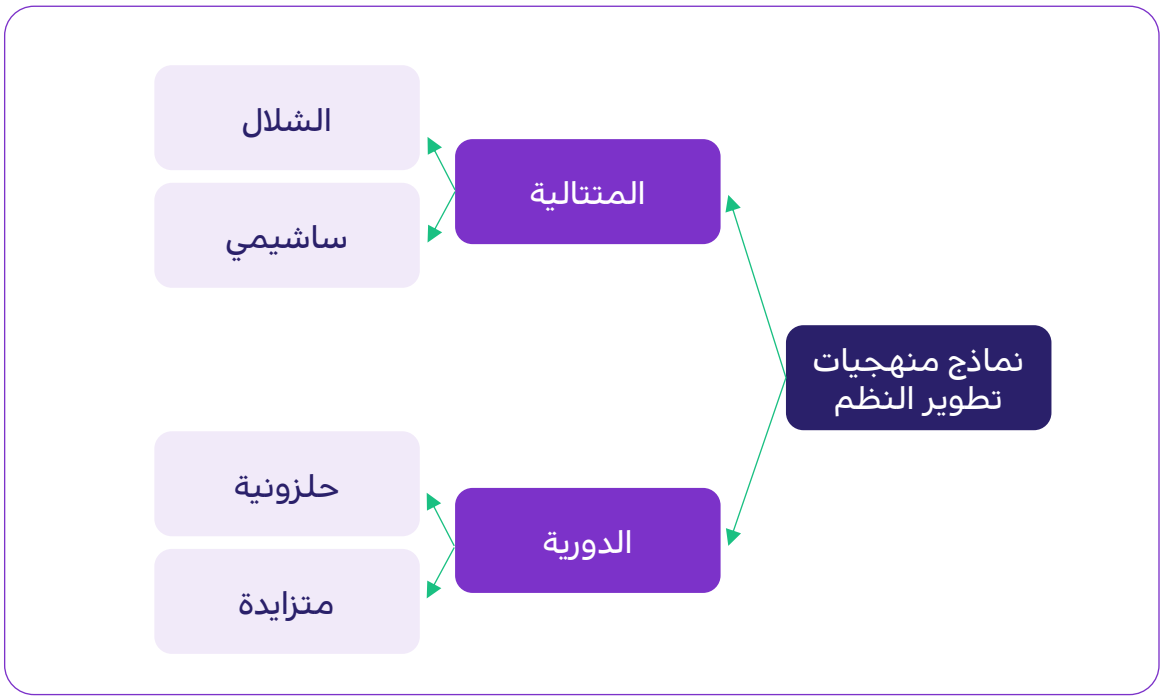
النماذج المتتالية (Sequential Models):

نماذج تقسّم أنشطة تطوير البرمجيات متتالية وخطيّة، وتعتمد كل خطوة من خطوات النشاط على نتائج النشاط السابق لها. ومن أكثر المنهجيات الشائعة للنموذج المتزايد هو منهجية الشلال (Waterfall) ونموذج (Sashimi).

النماذج الدورية

ترتب الأنشطة في النماذج الدورية بشكل متتابع شاملة أنشطة تحليل المتطلبات وتطوير التصميم والنشر والعمليات التشغيلية لإنتاج دورة تكرارية، وتُكَرّر الدورة في هذا النموذج حتى اكتمال العمل. وتنقسم النماذج الدورية إلى النموذجين التاليين:

- **النماذج الحلزونية (Spiral Model)**: الدورات التكرارية التي تركّز على إدارة المخاطر، وتقوم بتحسين المكوِّنات بناءً على تلك المخاطر.
- **النماذج المتزايدة (Incremental Model)**: التطوير المتزايد للبرمجية حيث تتم عملية الانتهاء والتحقق من كل جزئية أضيفت، ثم يتم دمجها في البرنامج ككل قبل الانتقال إلى الدورة التالية. ومن المنهجيات الشائعة للنموذج المتزايد منهجية التطوير المرن (Agile)، والتي تعتمد على الزيادات الصغيرة ضمن دورات صغيرة لفترات زمنية قصيرة (أسابيع قليلة).



الشكل (3) النماذج المستخدمة في منهجية تطوير النظم

يتم اختيار النموذج الملائم الذي يناسب خصائص وقيود كل مشروع على أفضل وجه، فيأخذ في الاعتبار مختلف العناصر وحالات الاستخدام لاختيار المنهجيات التابعة للنماذج المناسبة للعمل وفقا لما يلي:

تستخدم منهجية الشلال (Waterfall) في الحالات التالية

- عندما يكون الأرجح أن المتطلبات محددة بشكل جيد ولن يتم تغييرها.
- عند إمكانية جمع كل التفاصيل (أو معظمها) قبل البدء في العمل.
- في حال إتقان فريق العمل التعامل مع التقنية المستخدمة في العمل.
- في حالة محدودية نطاق العمل وجدوله الزمني، بما يتيح التحكم في أي تحوّل ناتج عن تحسين المتطلبات أو التغيير أو كلاهما، والحد منه، وجعله في نطاق التكلفة المقبولة.

تستخدم منهجية التطوير المرن (Agile) في الحالات التالية

- في حال قدرة أصحاب المصلحة واستعدادهم للمشاركة في العمل بصورة متكررة.
- إذا كانت المتطلبات الرئيسية محددة ولكن غير واضحة التفاصيل.
- في حال إمكانية تغيّر المتطلبات خلال فترة تنفيذ العمل.
- في حال اقتراح النظر في تقنية جديدة أو بيئات تشغيل لم يصل العمل فيها إلى مستوى الإتقان.
- في حال أن الفريق يتكون من عدد قليل من الأفراد.
- عند الحاجة لعرض سير عمل البرمجية على أصحاب المصلحة بصورة متكررة.
- إذا كانت الأولوية في العمل تركز على الالتزام بمدّة الإنجاز أكثر من كونه قائماً على المخرجات والمميزات.
- في حال كون حجم بنود العمل صغيرة وسهلة التقدير، مثل: المسائل البسيطة، أو طلبات التحسين الصغيرة.

5.5 توصيات عامة لتبني منهجية تطوير النظم (SDLC)

يمكن استخدام القائمة أدناه كمرجع عام للمتطلبات التي يجب استكمالها لضمان تنفيذ المنهجية بشكل شامل، وتحديد المتطلبات الأساسية:

- تحديد جميع متطلبات العمل.
- تحديد جميع متطلبات المستخدمين وأصحاب المصلحة.
- تحديد جميع المتطلبات الوظيفية.
- تحديد جميع المتطلبات الفنية.
- تحديد جميع المتطلبات الأمنية.
- الحصول على مصادقة وتعميد أصحاب المصلحة على جميع المتطلبات.
- تحديد جميع متطلبات البرمجيات والأجهزة، وتوضيحها بالتفصيل للتطوير والاختبار والنشر والمراقبة.
- تحديد جميع معايير القبول.
- خضوع الجدول الزمني للمشروع والميزانية ومواصفات متطلبات البرمجيات لمراجعة فريق المشروع والجهة الراعية وأصحاب المصلحة المعنيين.
- اعتماد التقديرات والخطط وتوافقها مع استراتيجية الأعمال.
- توثيق التصميم (التصميم العام والتصميم الشامل والتصميم التفصيلي) وتعميمه.
- توافق الشفرة المصدرية مع القواعد والمعايير للجهة.
- خضوع أي شفرة مصدرية لاختبارات الوحدة واختبارات التكامل.
- وجود إصدارات للشفرة والتوثيق وأي معلومات مرتبطة بالبرمجيات للشفرة المستخدمة.
- تأكيد إجراء جميع الاختبارات اللازمة شاملة اختبار واجهة المستخدم الآلية، واختبار الحِمل والإجهاد، وأي اختبارات آلية أخرى بشكل آلي.
- الاختبارات التي يتعذر إجراؤها آليًّا فقط هي التي تُجرى يدويًّا.
- إمكانية مراقبة النظام والتطبيق وجميع المكوِّنات والموارد الأساسية.

5.6 الاستمرارية وقياس الأثر

لضمان التبني الصحيح لمنهجية تطوير النظم (SDLC)، هناك بعض المؤشرات الرئيسية التي يمكن قياسها قبل وأثناء وبعد التبني لضمان اعتماد فعّال ومفيد، ويمكن تصنيف المؤشرات إلى ثلاثة أصناف رئيسية، وهي: مؤشرات تتعلق بالجانب المالي، ومؤشرات تخص العملاء، ومؤشرات أداء داخلية، وذلك لتحقيق الآثار التالية:

- **سرعة الوصول إلى السوق**: من خلال قياس الوقت المستهلك في تطوير النظم قبل وبعد تبني المنهجيات وتتبع التقدم.
- **نجاح أكبر**: يقاس النجاح من خلال رضا العملاء، وعليه فإن اتباع منهجيات منظمة لتطوير النظم سيساهم في تحقيق رضا المستفيدين والمستخدمين لهذه الأنظمة.
- **تكلفة أقل**: قياس التكاليف لتطوير النظم قبل وبعد تبني المنهجيات؛ للتأكد من تحقيق الوفورات المطلوبة.
- **جودة أعلى**: من خلال قياس نتيجة الجودة النهائية للنظم بعد تطبيق المنهجيات.

وفيما يلي بعض مؤشرات الأداء التي يمكن للجهات الحكومية قياسها ومتابعتها لقياس فاعلية تطبيق المنهجية، علما بأنه يمكن تغيير النسب حسب حجم الجهات والمشاريع:

مؤشر الأداء الرئيسي	الوصف	أداء من الدرجة الأولى	أداء متوسط	أداء ضعيف
نسبة الرضا	نسبة رضا المستفيد النهائي من النظام	أكثر من 90 %	90 – 70 %	أقل من 70 %
نسبة التغيير	عدد عمليات التغيير التي تمت على النطاق مقارنة بنطاق العمل الأساسي	أقل من 10 %	20 – 10 %	أكثر من 20 %
معدل الفشل	نسبة العمليات الخاطئة في النظام	0-15%	15-30%	أكثر من 30%
عدد تذاكر الدعم	عدد تذاكر الدعم التي تم رفعها في فترة زمنية محددة بعد استخدام النظام	أقل من 10 %	20 – 10 %	أكثر من 20 %

6. جدول التعريفات

يُقصد بالألفاظ والعبارات الآتية – أينما وردت في هذه الوثيقة – المعاني المبينة أمام كلٍّ منها، ما لم يقتض السياق خلاف ذلك:

المصطلح	التعريف
الهيئة	هيئة الحكومة الرقمية.
التحول الرقمي	تحويل نماذج الأعمال وتطويرها بشكل استراتيجي، لتكون نماذج رقمية مستندة على بيانات وتقنيات وشبكات الاتصالات.
الحكومة الرقمية	دعم العمليات الإدارية والتنظيمية والتشغيلية داخل القطاعات الحكومية -وفيما بينها- لتحقيق التحول الرقمي، وتطوير وتحسين وتمكين الوصول بسهولة وفاعلية للمعلومات والخدمات الحكومية.
الجهات الحكومية	الوزارات والهيئات والمؤسسات العامة والمجالس والمراكز الوطنية، وما في حكمها.
دورة حياة تطوير النظم System Development Life Cycle (SDLC)	منهجية لتطوير النظم والبرمجيات تتكون من مراحل متعددة ومتتابعة.
عنصر تكوين برمجيات حاسوبية (CSCI)	مجموعة من البرامج أو أيّ من أجزائها المنفصلة التي تلبي وتنفذ المتطلبات الوظيفية التي تم طلبها وتحديدها من قبل أصحاب المصلحة.
الوحدات النمطية (Modules)	أحد مكونات البرنامج، أو جزء من برنامج يحتوي على إجراء واحد أو أكثر. يمكن لواحدة أو أكثر من الوحدات المطورة بشكل مستقل تشكيل البرنامج كاملا، وتخدم كل وحدة عمليات فريدة ومنفصلة.
أنماط البنية	وصف لهيكل البرنامج، وتصميم لأنواع الترابطات بين مكوناته.
التطوير المرن (Agile)	عملية لتطوير البرمجيات يتم من خلالها إنشاء وتطوير البرمجيات من خلال جهود التعاون داخل فريق عمل معين ومتعدد الوظائف.
النماذج المتتالية	أنواع من نماذج دورة حياة تطوير النظم، وهي تقسم عملية التطوير حسب الأنشطة المميزة لتطوير البرمجيات بطريقة متتالية وخطية، وتعتمد كل أداة من أدوات النشاط على الأدوات الناتجة عن النشاط السابق له.
نموذج الشلال (Waterfall)	نموذج لإدارة المشاريع يركز على التقدم الخطّي من بداية المشروع إلى نهايته. وتركز هذه المنهجية على المراحل الأولى بشكل أكبر، مثل: التخطيط الدقيق، والتوثيق التفصيلي، والتنفيذ المتتالي.
نموذج (Sashimi)	أحد نماذج تطوير النظم الذي تتداخل فيه المراحل المتتابعة حيث تبدأ مرحلة قبل انتهاء المرحلة التي تسبقها تماما، بخلاف نموذج الشلال، وذلك لإعطاء مجال للمراجعة والتصحيح لأي مشاكل أو نواقص تظهر عند بدء مرحلة جديدة.
متطلبات الأعمال (Business Requirements)	توضّح غايات الأعمال، وأهداف الأعمال، وأدوار الأعمال، واحتياجات الأقسام، وعمليات الأعمال.
متطلبات المستخدمين (User Requirements)	متطلبات توضّح احتياجات المستخدم والأنشطة التي يتوقعها مستخدم النظام ضمن الحل، وتشمل تجربة المستخدم التي تحدد تفاعل المستخدم وإمكانية الوصول وإمكانية الاستخدام والتضمين والغايات والقيود.
حالات الاختبار (Test Cases)	وصف لسيناريو محتمل حدوثه على المكون أو النظام، مع توضيح المدخلات لكل حالة اختبار، والمخرجات أو النتائج المتوقعة منه.
بيئة الإنتاج (Production Environment)	المساحة التي ينشر عليها أحدث إصدار من البرنامج، ويكون في حالة صالحة للعمل وخالية من الأخطاء ومتاحاً عندما يحتاجه المستخدم.
بيئة التقسيم المرحلي (Staging Environment)	نسخة طبق الأصل تقريبًا من بيئة الإنتاج، تُستخدم لاختبار البرامج. يتم تصميم بيئات التقسيم المرحلي لاختبار الشفرات والبنيات والتحديثات لضمان الجودة في بيئة شبيهة بالإنتاج قبل نشر التطبيق.
نشر البرمجيات	عملية إعداد أحد تطبيقات البرمجية للعمل والتشغيل في بيئة محدّدة، ويتضمن التثبيت والتهيئة لضمان التشغيل الأمثل للبرمجية.
القصص المصورة (Storyboards)	أداة مستخدمة في التحليل المرن للأعمال؛ لإنشاء نماذج مرئية لقصص المستخدم والمساعدة في تحديد المشاكل والمخاطر المحتملة. يتم استخدام القصص المصورة (Storyboarding) لوصف مهمة أو سيناريو أو قصة من حيث كيفية تفاعل أصحاب المصلحة مع الحل.
الحاويات	المحاكاة الافتراضية على مستوى نظام التشغيل التطبيقات، عبر موارد متعددة في الشبكات بحيث يمكن تشغيل تطبيقات البرامج في مساحات مستخدم معزولة تسمى "الحاويات" في أي بيئة سحابية أو غير سحابية.

7. جدول الاختصارات

الاختصار	المعنى
SDLC	Software Development Life Cycle
BR	Business Requirements
CSCI	Computer Software Configuration Item
DaaS	Data as a Service
DLD	Detailed Level Design
FaaS	Function as a Service
FR	Functional Requirements
HLD	High Level Design
IaaS	Infrastructure as a Service
LDAP	Lightweight Directory Access Protocol
LLD	Low Level Design
PaaS	Platform as a Service
POC	Proof of Concept
SaaS	Software as a Service
TR	Technical Requirements
UR	User Requirements

هيئة الحكومة الرقمية
Digital Government Authority

DGA.GOV.SA