**Student Assessment Submission and Declaration**

When submitting evidence for assessment, each student must sign a declaration confirming that the work is their own.

|  |  |  |  |
| --- | --- | --- | --- |
| Student name: Ahmad Kallab | | Assessor name: Rawan Baniyounes | |
| Issue date:  **30/3/2024** | Submission date:  **28/04/2024** | | Submitted on: |
| Programme:  Higher National Diploma in Cloud Computing - Cloud Software Development | | | |
| Unit 19\_Software Development Methodologies for the Cloud | | | |
| Assignment number and title: 1. TechBridge Solutions: Pioneering a DevOps Prototype | | | |

Task 1 .A

يمثل الانتقال من نظام تقليدي إلى حل قائم على السحابة مهمة كبيرة لأي مؤسسة ، تتطلب تخطيطا دقيقا وتنفيذا وقدرة على التكيف طوال العملية. في هذا السياق ، يعد اختيار منهجية دورة حياة تطوير البرمجيات (SDLC) المناسبة أمرا بالغ الأهمية لضمان نجاح مشروع الانتقال. سيؤثر اختيار منهجية SDLC على كيفية تخطيط المشروع وتنفيذه وإدارته من البداية إلى النشر وما بعده.

في هذا التقرير ، سوف نستكشف ونقيم العديد من منهجيات SDLC ، مع الأخذ في الاعتبار مدى ملاءمتها للانتقال من النظام الحالي إلى السحابة للخدمات اللوجستية العالمية. سيتم تقييم كل منهجية بناء على قدرتها على مواجهة التحديات والمتطلبات المحددة لمشروع الانتقال ، فضلا عن توافقها مع المراحل التي تنطوي عليها العملية. من بين المنهجيات قيد النظر Agile: و Spiral و Rapid Application Development (RAD) وتطوير التطبيقات المشتركة (JAD) والنماذج الأولية(prototyping model) والشلال(waterfall model). تقدم كل من هذه المنهجيات مزايا واعتبارات مميزة يجب موازنتها مقابل الاحتياجات الفريدة للمشروع.

في نهاية المطاف، ستكون منهجية SDLC المختارة بمثابة إطار عمل يوجه تطوير ونشر نظام إدارة الخدمات اللوجستية القائم على السحابة، مما يضمن التوافق مع أهداف Global Logistics والجداول الزمنية والمتطلبات التشغيلية.

1**.Agile**   
 هو نهج لتطوير البرمجيات يؤكد على المرونة والتعاون والتقدم التكراري. على عكس الطرق التقليدية التي تتبع تسلسلا خطيا من الخطوات ، يسمح Agile بالتكيف المستمر مع المتطلبات المتغيرة والتعليقات من أصحاب المصلحة.  
تعزز Agile تسليم برامج العمل بزيادات صغيرة وتدريجية تعرف باسم التكرارات أو سباقات السرعة. يستمر كل تكرار عادة من أسبوع إلى أربعة أسابيع ويتضمن فريقا متعدد الوظائف يعمل معا لتطوير واختبار وتقديم مجموعة من الميزات أو الوظائف.

تشمل المبادئ الأساسية ل Agile ما يلي:

* التطوير التكراري (Iterative Development): ينقسم المشروع إلى زيادات صغيرة يمكن التحكم فيها ، مما يسمح بالتسليم المتكرر لبرامج العمل.
* تعاون العملاء (Customer Collaboration): يتم تشجيع مشاركة أصحاب المصلحة وتعليقاتهم طوال عملية التطوير لضمان تلبية المنتج لاحتياجاتهم وتوقعاتهم.
* القدرة على التكيف (Adaptability): تستجيب فرق Agile للتغيير ، وتتبنى المتطلبات والأولويات المتطورة لتقديم الميزات الأكثر قيمة بسرعة.
* فرق متعددة الوظائف (Cross-Functional Teams):تتكون فرق Agile من أفراد يتمتعون بمجموعات مهارات متنوعة ، مثل المطورين والمختبرين والمصممين ومالكي المنتجات ، الذين يتعاونون بشكل وثيق لتحقيق الأهداف المشتركة.
* التحسين المستمر (Continuous Improvement): يعد التفكير والتكيف المنتظم جزءا لا يتجزأ من Agile ، حيث تبحث الفرق باستمرار عن طرق لتعزيز عملياتها واتصالاتها وجودة منتجاتها.

تشمل أطر عمل Agile الشائعة Scrum و Kanban و Extreme Programming (XP) ، كل منها يقدم إرشادات وممارسات محددة لتسهيل تطوير Agile. بشكل عام ، تعزز Agile نهجا ديناميكيا وتعاونيا يركز على العملاء لتطوير البرامج ، مما يمكن الفرق من تقديم منتجات عالية الجودة بكفاءة وفعالية.

2.**Spiral**

النموذج الحلزوني هو منهجية لتطوير البرمجيات تجمع بين عناصر كل من التطوير التكراري ونموذج الشلال. تم اقتراحه من قبل Barry Boehm في عام 1986 وهو مناسب بشكل خاص للمشاريع ذات المستويات العالية من عدم اليقين والتعقيد.

في النموذج الحلزوني ، تنقسم عملية التطوير إلى سلسلة من التكرارات أو "اللوالب" ، يتكون كل منها من أربع مراحل رئيسية:

* التخطيط(planning): في هذه المرحلة ، يتم تحديد الأهداف والقيود والبدائل ، ويتم تقييم مخاطر المشروع. يتضمن ذلك جمع المتطلبات وتحديد نطاق المشروع ووضع خطة للتكرارات اللاحقة.
* تحليل المخاطر(risk analysis): تتضمن المرحلة الثانية تقييما شاملا لمخاطر المشروع. يتم تحديد المخاطر وتحليلها وتحديد أولوياتها بناء على تأثيرها المحتمل على نجاح المشروع. ثم يتم تطوير استراتيجيات للتخفيف من هذه المخاطر ، والتي قد تشمل النماذج الأولية أو المحاكاة أو عروض إثبات المفهوم.
* الهندسة(Engineering): في هذه المرحلة يتم التطوير الفعلي للبرنامج. يتم تنقيح المتطلبات ، واتخاذ قرارات التصميم ، وكتابة التعليمات البرمجية واختبارها. يؤدي كل تكرار إلى زيادة المنتج القابل للشحن ، مما يسمح بالتعليقات المبكرة والتحقق من صحة أصحاب المصلحة.
* التقييم(Evaluation): تتضمن المرحلة النهائية من كل دوامة مراجعة التقدم المحرز أثناء التكرار ، وتقييم النتائج مقابل أهداف المشروع ، والتخطيط للتكرار التالي. ويشمل ذلك إجراء مراجعة رسمية للبرنامج، وجمع التعليقات من أصحاب المصلحة، وتنقيح خطة المشروع بناء على الدروس المستفادة.

تتمثل إحدى السمات الرئيسية لنموذج Spiral في تركيزه على إدارة المخاطر والتطوير التكراري. من خلال معالجة المخاطر في وقت مبكر وبشكل مستمر طوال عملية التطوير ، يسعى نموذج Spiral إلى تقليل حالات عدم اليقين في المشروع وزيادة احتمالية نجاح المشروع. بالإضافة إلى ذلك ، يستوعب نموذج Spiral التغييرات في المتطلبات والأولويات ، مما يسمح بالمرونة والقدرة على التكيف استجابة لاحتياجات المشروع المتطورة. ومع ذلك ، يمكن أن يكون النموذج الحلزوني أكثر تعقيدا واستهلاكا للوقت من المنهجيات الأخرى ، مما يتطلب تخطيطا وتنسيقا دقيقين لإدارة الدورات التكرارية بفعالية. بشكل عام ، يقدم نموذج Spiral نهجا منظما ومرنا لتطوير البرمجيات ، مما يجعله مناسبا تماما للمشاريع التي يسود فيها عدم اليقين والتعقيد والمتطلبات المتطورة.

3. التطوير السريع للتطبيقات (RAD) :

التطوير السريع للتطبيقات (RAD) هو منهجية لتطوير البرمجيات تعطي الأولوية للسرعة والمرونة في إنشاء تطبيقات البرمجيات. على عكس مناهج التطوير التقليدية التي تؤكد على التخطيط التفصيلي وجمع المتطلبات المسبقة على نطاق واسع ، تركز RAD على إنتاج نماذج أولية أو إصدارات من التطبيق بسرعة.

يتضمن RAD عادة الخصائص الرئيسية التالية:

* التطوير التكراري(Iterative Development): يقسم RAD عملية التطوير إلى تكرارات أو دورات متعددة ، ينتج عن كل منها نموذج أولي أو إصدار عملي من التطبيق. تسمح هذه التكرارات بالتحسين والتحسين المستمر بناء على ملاحظات المستخدم.
* النماذج الأولية(Prototyping): تعد النماذج الأولية جانبا أساسيا من RAD ، حيث يقوم المطورون بإنشاء إصدارات سريعة ومنخفضة الدقة من التطبيق لإظهار الميزات والوظائف الرئيسية. تعمل هذه النماذج الأولية كأساس لجمع التعليقات وتوجيه المزيد من التطوير.
* مشاركة المستخدم(User Involvement): تؤكد RAD على التعاون الوثيق بين المطورين والمستخدمين النهائيين طوال عملية التطوير. يشارك المستخدمون بنشاط في تقديم ملاحظات حول النماذج الأولية ، مما يساعد على ضمان أن المنتج النهائي يلبي احتياجاتهم وتوقعاتهم.
* القدرة على التكيف(Adaptability): RAD قابل للتكيف بدرجة كبيرة ويستجيب للتغيرات في المتطلبات أو الأولويات. يمكن لفرق التطوير تعديل نهجها بسرعة بناء على التعليقات الواردة من المستخدمين أو التغييرات في نطاق المشروع ، مما يسمح بالتكرار والتحسين السريع.
* التطوير في إطار زمني(Time-boxed Development): غالبا ما تكون مشاريع RAD محددة زمنيا ، مما يعني أن كل تكرار له مدة ثابتة وأهداف محددة لتحقيقها خلال هذا الإطار الزمني. تساعد هذه الملاكمة الزمنية في الحفاظ على الزخم والتركيز ، مما يضمن تقدم التطوير بسرعة وكفاءة.

التطوير السريع للتطبيقات (RAD) هو منهجية تطوير برمجيات رشيقة مثالية للمشاريع التي تتطلب النشر السريع ، والقدرة على التكيف مع احتياجات المستخدم المتطورة ، ودمج التعليقات المتكررة. مناسبة لمناظر السوق الديناميكية ، تعمل RAD على تسريع وقت الوصول إلى السوق من خلال التأكيد على النماذج الأولية السريعة والتطوير التكراري. من خلال الدورات التكرارية ، يعطي RAD الأولوية للميزات بناء على ملاحظات المستخدم ، مما يضمن تقديم وظائف عالية القيمة في وقت مبكر مع تقليل مخاطر إعادة العمل. تعزز مشاركة المستخدم النشطة التعاون وتضمن توافق المنتج النهائي بشكل وثيق مع توقعات المستخدم. في نهاية المطاف ، تمكن RAD الفرق من الابتكار بسرعة ، والاستجابة لديناميكيات السوق المتغيرة ، وتقديم حلول برمجية عالية الجودة تدفع نمو الأعمال.

يبرز التطوير السريع للتطبيقات ((RAD كخيار مثالي للمشاريع التي تتمتع فيها سرعة السوق والمرونة وتعليقات المستخدمين المستمرة بأهمية قصوى. من خلال تبني RAD ، يمكن للفرق الاستفادة من تركيزها على النماذج الأولية السريعة ودورات التطوير التكرارية ومشاركة المستخدم النشطة لتسريع تقديم حلول برمجية قوية. تعزز هذه المنهجية بيئة تمكن فيها التكرارات السريعة الفرق من دمج احتياجات المستخدم المتطورة بسلاسة ، مما يضمن توافق المنتج النهائي بشكل وثيق مع توقعات المستخدم. علاوة على ذلك ، فإن مرونة RAD تمكن الفرق من التكيف بسرعة مع المتطلبات المتغيرة وديناميكيات السوق ، وبالتالي التخفيف من المخاطر وزيادة الاستجابة طوال دورة حياة التطوير. من خلال نهجها التكراري وتركيزها المتمحور حول المستخدم ، لا تعمل RAD على تسريع تسليم البرامج عالية الجودة فحسب ، بل تعمل أيضا على تهيئة بيئة مواتية للابتكار والنجاح المستدام.

4. تطوير التطبيقات المشتركة ( (JAD:

تطوير التطبيقات المشتركة ( (JADهو نهج منظم لتطوير البرمجيات يتضمن ورش عمل أو جلسات تعاونية بين أصحاب المصلحة والمستخدمين النهائيين والمطورين. الهدف من JAD هو جمع المتطلبات بسرعة وتحديد النطاق وتصميم نظام البرنامج أو التطبيق بمشاركة نشطة من جميع الأطراف المعنية. تتضمن جلسات JAD عادة تقنيات مثل العصف الذهني والنماذج الأولية والإرشادات التفصيلية المنظمة لضمان مراعاة جميع وجهات النظر وأن المنتج النهائي يلبي احتياجات وتوقعات المستخدمين. يمكن أن تساعد JAD في تبسيط عملية التطوير وتقليل سوء الفهم وزيادة مشاركة أصحاب المصلحة من خلال تعزيز التواصل المفتوح والتعاون طوال دورة حياة المشروع.

تتضمن المفاتيح الرئيسية لتطوير التطبيقات المشتركة (JAD) ما يلي:

* التعاون: تؤكد JAD على التعاون بين أصحاب المصلحة والمستخدمين النهائيين وفرق التطوير. فهو يجمع بين الأفراد ذوي وجهات النظر والخبرات المختلفة لضمان مراعاة جميع وجهات النظر ذات الصلة أثناء عملية تطوير البرمجيات.
* المشاركة الفعالة: تتطلب جلسات JAD مشاركة نشطة من جميع الأطراف المعنية. هذا يضمن أن يساهم الجميع بمعرفتهم ومتطلباتهم وتوقعاتهم ، مما يؤدي إلى فهم أكثر شمولا لأهداف المشروع وغاياته.
* النهج المنظم: تتبع JAD نهجا منظما لجمع المتطلبات وتحديد النطاق وتصميم نظام البرنامج أو التطبيق. يتضمن هذا عادة ورش عمل أو جلسات منظمة ذات جداول أعمال وأنشطة ومخرجات محددة مسبقا لتوجيه المناقشات وعملية صنع القرار.
* النماذج الأولية التكرارية: غالبا ما تستخدم النماذج الأولية في JAD لتصور المفاهيم والميزات والوظائف والتحقق من صحتها بسرعة. تسمح النماذج الأولية التكرارية لأصحاب المصلحة والمستخدمين النهائيين بتقديم ملاحظات في وقت مبكر من عملية التطوير ، مما يتيح إجراء التعديلات والتحسينات قبل الانتهاء من التصميم.
* الجلسات الميسرة: عادة ما يتم تسهيل جلسات JAD من قبل ميسرين أو مشرفين مدربين يساعدون في توجيه المناقشات وإدارة جدول الأعمال وضمان تحقيق أهداف الجلسات. يساعد الميسرون أيضا في حل النزاعات وتشجيع المشاركة والحفاظ على الجلسات مركزة ومثمرة.
* التوثيق: يعد التوثيق جانبا أساسيا من JAD لتسجيل نتائج الجلسات ، بما في ذلك المتطلبات وقرارات التصميم وبنود العمل وأي معلومات أخرى ذات صلة. تساعد الوثائق التفصيلية على ضمان أن كل شخص مشارك في المشروع لديه فهم واضح للمتطلبات المتفق عليها ومواصفات التصميم.
* مشاركة أصحاب المصلحة: تعزز JAD مشاركة أصحاب المصلحة من خلال إشراك أصحاب المصلحة الرئيسيين والمستخدمين النهائيين في عملية التطوير منذ البداية. من خلال المشاركة الفعالة في جلسات JAD والمساهمة في عملية صنع القرار ، من المرجح أن يشعر أصحاب المصلحة بإحساس بالملكية والالتزام بنجاح المشروع.

يعد تطوير التطبيقات المشتركة (JAD) منهجية مشهورة بطبيعتها التعاونية ، والتي تسرع دورات التطوير وتضمن برامج عالية الجودة مصممة خصيصا لكل من احتياجات الأعمال والمستخدم النهائي. من خلال تنظيم ورش عمل مكثفة، تعزز جاد التعاون بين أصحاب المصلحة وفرق التطوير، مما يؤدي إلى تبسيط العمليات والمواءمة الدقيقة مع الأهداف. من خلال النماذج الأولية التكرارية والتعليقات المستمرة ، تعمل JAD على تحسين جودة البرامج ، مما يجعلها مثالية للمشاريع التي تواجه التعقيد أو التغيير السريع. في نهاية المطاف، تدفع قدرة JAD على التكيف والنهج التعاوني الابتكار وتقدم قيمة ملموسة للمؤسسات.

5. النماذج الاوليه (prototyping) :

* التغذية الراجعة المبكرة(Early Feedback:): تمكن النماذج الأولية أصحاب المصلحة والمستخدمين النهائيين من تصور البرنامج والتفاعل معه في وقت مبكر من عملية التطوير ، مما يسهل التغذية الراجعة المبكرة والتحقق من صحة قرارات التصميم.
* التطوير التكراري(Iterative Development): تسمح الطبيعة التكرارية للنماذج الأولية بالتحسين والتحسين المستمر بناء على التعليقات الواردة خلال كل تكرار ، مما يؤدي إلى منتج نهائي أكثر صقلا وسهولة في الاستخدام.
* الحد من المخاطر(Risk Reduction:): من خلال تحديد المشكلات أو التحديات المحتملة في وقت مبكر ، تساعد النماذج الأولية في التخفيف من المخاطر المرتبطة بالتطوير ، مما يقلل من احتمالية إعادة العمل المكلفة أو التغييرات في وقت لاحق من العملية.
* تحسين الاتصال(Improved Communication): تعمل النماذج الأولية كأداة اتصال ، مما يساعد على سد الفجوة بين أصحاب المصلحة التقنيين وغير التقنيين من خلال توفير تمثيل ملموس لوظائف البرنامج وميزاته.
* مشاركة المستخدم(User Involvement): تشجع النماذج الأولية المشاركة النشطة للمستخدمين النهائيين طوال عملية التطوير ، مما يضمن مراعاة احتياجاتهم وتفضيلاتهم ودمجها في المنتج النهائي.

النماذج الأولية هي بالفعل تقنية قيمة في تطوير البرمجيات. إنه بمثابة أداة حاسمة للفرق للتحقق من صحة الأفكار وجمع التعليقات وتحسين التصميمات بشكل متكرر. من خلال إنشاء نماذج أولية ، يمكن للفرق تصور المفاهيم بشكل ملموس أكثر ، مما يسهل توصيل الأفكار والتماس المدخلات من أصحاب المصلحة والمستخدمين النهائيين. تسمح هذه العملية التكرارية للنماذج الأولية بالتحسين والتحسين المستمرين ، مما يضمن أن الحل البرمجي النهائي يلبي احتياجات المستخدم بشكل فعال. في النهاية ، تلعب النماذج الأولية دورا محوريا في تطوير حلول برمجية ناجحة تتمحور حول المستخدم من خلال تسهيل التعاون وتقليل المخاطر وتحسين الجودة الشاملة للمنتج.

6. waterfall (الشلال) :

نموذج الشلال هو منهجية تطوير برمجيات متسلسلة حيث تتقدم العملية خطيا عبر مراحل متميزة ، حيث تعتمد كل مرحلة على إكمال المرحلة السابقة. يتضمن عادة مراحل مثل تحليل المتطلبات وتصميم النظام والتنفيذ والاختبار والنشر والصيانة. يتم جمع المتطلبات وتوثيقها في المرحلة الأولية ، يليها تصميم النظام التفصيلي ، وتنفيذ البرنامج ، واختبار العيوب ، والنشر في الإنتاج ، والصيانة المستمرة. مع توفير نهج منظم مع مخرجات واضحة لكل مرحلة ، فإن صلابة نموذج الشلال يمكن أن تجعله أقل قابلية للتكيف مع التغييرات أو المتطلبات المتطورة مقارنة بالمنهجيات الأكثر تكرارية أو مرونة. تشمل المراحل النموذجية في نموذج الشلال ما يلي:

* تحليل المتطلبات (Requirements Analysis) : في هذه المرحلة الأولية ، يتم جمع متطلبات المشروع وتوثيقها بالتفصيل. وهذا ينطوي على فهم احتياجات أصحاب المصلحة وتحديد نطاق المشروع.
* تصميم النظام(System Design) : بمجرد تحديد المتطلبات ، يتم التخطيط لبنية النظام وتصميمه. تتضمن هذه المرحلة إنشاء مواصفات تصميم مفصلة لنظام البرمجيات بناء على المتطلبات المجمعة.
* التنفيذ(Implementation) : مع مواصفات التصميم في متناول اليد ، يشرع فريق التطوير في تنفيذ البرنامج وفقا للمتطلبات والتصميم المحددين. تتضمن هذه المرحلة كتابة التعليمات البرمجية ودمج المكونات وبناء نظام البرنامج.
* الاختبار(Testing) : بعد مرحلة التنفيذ ، يخضع البرنامج للاختبار لتحديد وتصحيح أي عيوب أو أخطاء. تتضمن أنشطة الاختبار اختبار الوحدة واختبار التكامل واختبار النظام واختبار قبول المستخدم للتأكد من أن البرنامج يعمل بشكل صحيح.
* النشر(Deployment) : بمجرد اكتمال الاختبار بنجاح ، يتم نشر البرنامج أو إصداره في بيئة الإنتاج ليستخدمه المستخدمون النهائيون. تتضمن هذه المرحلة تثبيت البرنامج وتكوين النظام وتقديم الدعم للانتقال إلى البرنامج الجديد.
* الصيانة(Maintenance) : بعد النشر ، يدخل البرنامج مرحلة الصيانة ، حيث يتم توفير الدعم المستمر والتحديثات والتحسينات حسب الحاجة. تضمن هذه المرحلة بقاء البرنامج قيد التشغيل ومتوافقا مع احتياجات المستخدم المتغيرة.

يتميز نموذج الشلال بتقدمه الخطي ، حيث يتم إكمال كل مرحلة قبل الانتقال إلى المرحلة التالية. في حين أنه يوفر نهجا منظما لتطوير البرمجيات ومخرجات واضحة لكل مرحلة ، فإن صلابته يمكن أن تجعله أقل قدرة على التكيف مع التغييرات أو المتطلبات المتطورة مقارنة بالمنهجيات الأكثر تكرارا أو مرونة.

اذًا ما هو النوع المناسب الذي يجب اختياره لتحقيق متطلبات Global Logistics مع مراعاة الجداول الزمنية والمتطلبات التشغيلية؟

منهجية Agile هي الخيار المثالي لمشروع Global Logistics لتطوير نظام إدارة لوجستيات قائم على السحابة لعدة أسباب مصممة خصيصا لمتطلبات الشركة:

* القدرة على التكيف مع المتطلبات المتطورة: نظرا لأن قائمة المتطلبات بأكملها لم يتم الانتهاء منها بعد ، فإن مرونة Agile تسمح لنا باستيعاب التغييرات والتحسينات طوال عملية التطوير. وهذا يضمن أن الحل البرمجي يتماشى تماما مع احتياجات الأعمال المتطورة والمتطلبات التشغيلية لشركة Global Logistics.
* التسليم السريع لقيمة الأعمال: يعزز Agile تسليم برامج العمل في دورات قصيرة ومتكررة. وهذا يعني أنه يمكننا تحديد أولويات الميزات الأكثر أهمية وتقديمها أولا ، مما يمكن Global Logistics من البدء في تحقيق فوائد النظام الجديد في وقت أقرب وتكرارا بشكل مستمر بناء على التعليقات لمواجهة التحديات الأكثر إلحاحا.
* إدارة المخاطر والتحسين المستمر: يسهل نهج Agile التكراري التحديد المبكر للمخاطر والتخفيف من حدتها من خلال الاختبار المنتظم وحلقات التغذية الراجعة. وهذا أمر بالغ الأهمية، خاصة أثناء الانتقال إلى نظام قائم على السحابة، لأنه يسمح لنا بمعالجة أي تحديات تقنية أو تشغيلية بشكل استباقي، مما يضمن التنفيذ السلس والناجح.
* تعاون أصحاب المصلحة ومشاركتهم: تؤكد منهجيات Agile على التعاون والمشاركة النشطة من أصحاب المصلحة طوال عملية التطوير. من خلال المشاركة المنتظمة مع أعضاء فريق Global Logistics والمستخدمين النهائيين ، يمكننا التأكد من أن الحل يلبي توقعاتهم ، ويعالج نقاط الألم لديهم ، ويعزز في نهاية المطاف خدمة العملاء والكفاءة التشغيلية.
* التوافق مع أهداف الخدمات اللوجستية العالمية: يتماشى تركيز Agile على تقديم القيمة بسرعة وبشكل مستمر تماما مع هدف Global Logistics المتمثل في تحديث عملياتها اللوجستية. من خلال الاستفادة من ممارسات Agile ، يمكننا ضمان بقاء المشروع متوافقا مع الأهداف الاستراتيجية للشركة ، وتحقيق نتائج ملموسة تدفع نمو الأعمال والقدرة التنافسية.

**الان لناخذ بعين الاعتبار المراحل المطلوبة في الشرح:**

* التحقيق في كيفية نشر النظام الحالي باستخدام السحابة و :DevOps

يسمح لنا نهج Agile التكراري بالبدء بزيادات صغيرة يمكن التحكم فيها للمشروع ، مثل ترحيل وحدات أو وظائف معينة إلى السحابة. يتيح لنا ذلك التحقيق في عملية النشر تدريجيا ، ودمج التعليقات وتحسين نهجنا أثناء تقدمنا. من خلال الاستفادة من ممارسات Agile جنبا إلى جنب مع مبادئ DevOps ، يمكننا ضمان الانتقال السلس إلى السحابة مع الحفاظ على التركيز على تقديم قيمة الأعمال في كل مرحلة.

* تقييم المنهجيات الخطية والتكرارية المختلفة المتاحة:

تبرز Agile باعتبارها المنهجية الأكثر ملاءمة نظرا لطبيعتها التكرارية ، والتي تتوافق بشكل جيد مع متطلبات المشروع وأوجه عدم اليقين. على عكس المنهجيات الخطية مثل الشلال ، تمكننا Agile من التكيف مع التغييرات في النطاق أو المتطلبات وتقديم قيمة تدريجية ، وتخفيف المخاطر وضمان التحسين المستمر طوال دورة حياة المشروع.

* تحديد نطاق المشروع:

يسمح لنا نهج Agile التكراري بتحديد نطاق المشروع بشكل تعاوني مع أصحاب المصلحة في Global Logistics ، وتقسيمه إلى مكونات أصغر يمكن التحكم فيها تسمى قصص المستخدم أو الميزات. يتيح لنا ذلك تحديد أولويات الميزات الأكثر قيمة وتقديمها بشكل متكرر ، مما يضمن أننا نعالج الجوانب الأكثر أهمية للمشروع أولا مع الحفاظ على قابليتنا للتكيف مع التغييرات في النطاق مع تقدم المشروع.

* تطوير مقترح إدارة المشروع:

توفر منهجيات إدارة المشاريع الرشيقة ، مثل Scrum أو Kanban ، أطرا لإدارة المشاريع الفعالة ضمن عملية تطوير تكرارية. من خلال الاستفادة من ممارسات إدارة المشاريع الرشيقة ، يمكننا تحديد أدوار ومسؤوليات واضحة ، وتحديد أهداف ومعالم قابلة للتحقيق ، وتعزيز الشفافية والتعاون بين أعضاء الفريق وأصحاب المصلحة.

* إنشاء وثائق فنية مع مراعاة DevOps :

تؤكد منهجيات Agile على برامج العمل على التوثيق الشامل. ومع ذلك ، لا تزال فرق Agile تنتج الوثائق اللازمة ، مثل قصص المستخدمين ومعايير القبول والمواصفات الفنية ، حسب الحاجة لدعم التطوير والتعاون. يتم إنشاء الوثائق بشكل متكرر ومستمر في جميع أنحاء المشروع ، مما يضمن أنها تظل ذات صلة ومحدثة مع الحل المتطور ومتطلبات DevOps Pipeline .

* تنفيذ الانتقال إلى السحابة والتحسين بشكل متكرر:

هي خطوة حاسمة في منهجية Agile . يسمح نهجها التكراري بالتنفيذ التدريجي لاستراتيجيات الترحيل السحابي، مما يضمن أن كل خطوة قابلة للإدارة وتسمح بالتغذيه العائده المستمرة. ومن خلال المراجعات المنتظمة والاستعراضات الاسترجاعية، يمكن للفرق تحديد مجالات التحسين في عمليات وممارسات الانتقال الى السحابه. يضمن هذا التحسين التكراري الانتقال السلس والفعال إلى السحابة، حيث تقوم الفرق بتكييف وتحسين نهجها بناءً على ردود الفعل الواقعية ومتطلبات المشروع المتطورة.

من خلال النظر في المراحل المطلوبة ، تبرز Agile كمنهجية SDLC الأنسب للانتقال من النظام الحالي إلى السحابة للخدمات اللوجستية العالمية. يوفر المرونة والقدرة على التكيف والنهج التكراري اللازم لمعالجة أوجه عدم اليقين وتخفيف المخاطر وتقديم قيمة متزايدة في كل مرحلة من مراحل المشروع.

Task 1.B

1. منهجية دورة حياة تطوير البرمجيات ا**لخطية Linear** Software Development Life Cycle (SDLC) Methodologies)) :

تتبع منهجيات SDLC الخطية ، والمعروفة أيضا باسم المنهجيات التقليدية أو المتسلسلة ، عملية صارمة خطوة بخطوة حيث يتم إكمال كل مرحلة من مراحل دورة حياة التطوير بالتتابع قبل الانتقال إلى المرحلة التالية. منهجية SDLC الخطية الأكثر شهرة هي نموذج الشلال ، والذي يتكون عادة من مراحل متسلسلة مثل جمع المتطلبات وتصميم النظام والتنفيذ والاختبار والنشر والصيانة. تتميز المنهجيات الخطية بنهجها المنظم وتركيزها على التخطيط التفصيلي والتوثيق مقدما. في حين أنها توفر الوضوح والقدرة على التنبؤ ، إلا أنها يمكن أن تكون أقل قدرة على التكيف مع التغييرات أو المتطلبات المتطورة.

2. منهجية دورة حياة تطوير البرمجيات **التكراريه Iterative** Software Development Life Cycle (SDLC) Methodologies)

من ناحية أخرى ، تتبع منهجيات SDLC التكرارية نهجا تكراريا ومتزايدا لتطوير البرمجيات ، حيث يتم تقسيم المشروع إلى دورات أو تكرارات أصغر. يتضمن كل تكرار مجموعة فرعية من نطاق المشروع الكلي ويشمل جميع مراحل SDLC ، بما في ذلك جمع المتطلبات والتصميم والتنفيذ والاختبار والنشر. تتضمن أمثلة المنهجيات التكرارية Agile و Scrum وتطوير التطبيقات السريعة (RAD). تعطي المنهجيات التكرارية الأولوية لتقديم برامج العمل بسرعة ثم تحسينها وتحسينها بشكل متكرر بناء على التعليقات الواردة من أصحاب المصلحة والمستخدمين النهائيين. تتميز بمرونتها وقدرتها على التكيف والتركيز على التحسين المستمر وتعاون العملاء.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **المهنجيه** | **التعريف** | **انواع المنهجيات** | **المزايا** | **العيوب** |
| المنهجيات الخطيه | تتبع منهجيات SDLC الخطية ، والمعروفة أيضا باسم المنهجيات التقليدية أو المتسلسلة ، نهجا منظما خطوة بخطوة حيث يتم إكمال كل مرحلة من مراحل دورة حياة التطوير بالتتابع. | Spiral  Joint Application Development (JAD)  Waterfall | متطلبات واضحة ومحددة جيدًا مقدمًا: تؤكد المنهجيات الخطية، مثل Waterfall، على التخطيط والتوثيق الشامل في بداية المشروع، مما يؤدي إلى فهم واضح لمتطلبات المشروع من قبل جميع أصحاب المصلحة.  جدول زمني وميزانية يمكن التنبؤ بهما: تتيح الطبيعة المنظمة للمنهجيات الخطية تقديرًا أفضل للجداول الزمنية للمشروع وميزانياته، مما يوفر لأصحاب المصلحة إحساسًا بإمكانية التنبؤ والتحكم في تقدم المشروع.  التركيز على التوثيق: تركز المنهجيات الخطية بشدة على التوثيق طوال عملية التطوير، مما يضمن أن جميع متطلبات المشروع ومواصفات التصميم والتقدم المحرز في التطوير موثقة جيدًا للرجوع إليها في المستقبل. | مرونة محدودة لاستيعاب التغييرات: تتبع المنهجيات الخطية عملية متسلسلة صارمة، مما يجعل من الصعب استيعاب التغييرات أو المتطلبات المتطورة بمجرد تقدم المشروع إلى ما بعد مراحل معينة.  مخاطر عالية للتغييرات في المراحل المتأخرة: نظرًا لأن المنهجيات الخطية تؤخر الاختبار والتحقق من الصحة حتى مراحل لاحقة من المشروع، فإن أي تغييرات أو مشكلات يتم تحديدها خلال هذه المراحل يمكن أن تكون أكثر تكلفة وتستغرق وقتًا طويلاً في معالجتها.  الحد الأدنى من مشاركة أصحاب المصلحة حتى مراحل لاحقة: غالبًا ما تتضمن المنهجيات الخطية الحد الأدنى من مشاركة أصحاب المصلحة حتى مراحل لاحقة من المشروع، مما قد يؤدي إلى سوء الفهم أو عدم التوافق بين مخرجات المشروع وتوقعات أصحاب المصلحة. |
| المنهجيات التكراريه | تتضمن منهجيات SDLC التكرارية تقسيم المشروع إلى دورات أو تكرارات أصغر ، مع كل تكرار يشمل جميع مراحل SDLC. | Agile-  Rapid- Application Development  Prototyping- | المرونة في استيعاب التغييرات: تسمح المنهجيات التكرارية، مثل Agile، بتقديم تعليقات وتكيف بشكل متكرر، مما يمكّن فريق المشروع من استيعاب التغييرات أو المتطلبات المتطورة طوال عملية التطوير.  التحسين المستمر: تعطي المنهجيات التكرارية الأولوية لتقديم برامج العمل في دورات متكررة قصيرة، مما يسمح بالتحسين المستمر بناءً على تعليقات أصحاب المصلحة والمستخدمين النهائيين.  المشاركة النشطة لأصحاب المصلحة: تعمل المنهجيات التكرارية على تعزيز التعاون والمشاركة النشطة من جانب أصحاب المصلحة طوال عملية التطوير، مما يضمن أن المنتج النهائي يلبي توقعاتهم واحتياجات العمل. | احتمالية زحف النطاق: بدون الإدارة السليمة، يمكن أن تكون المنهجيات التكرارية عرضة لزحف النطاق، حيث تتوسع متطلبات المشروع إلى ما هو أبعد من النطاق الأولي، مما يؤدي إلى التأخير وزيادة التكاليف.  تعقيد التنسيق: تتطلب المنهجيات التكرارية تعاونًا وتنسيقًا وثيقًا بين الفرق متعددة الوظائف، الأمر الذي قد يكون من الصعب إدارته، خاصة في المشاريع واسعة النطاق أو الفرق الموزعة.  صعوبة تقدير الجداول الزمنية والميزانيات: الطبيعة التكرارية لهذه المنهجيات يمكن أن تجعل من الصعب تقدير الجداول الزمنية للمشروع وميزانياته بدقة، حيث قد تتطور المتطلبات والأولويات بمرور الوقت. |

تتميز منهجيات SDLC الخطية ، مثل نموذج الشلال ، بنهج منظم خطوة بخطوة يعطي الأولوية للتخطيط الشامل والتوثيق مقدما. يضمن هذا النهج متطلبات واضحة ومحددة جيدا في بداية المشروع ، مما يؤدي إلى جداول زمنية وميزانيات يمكن التنبؤ بها. يوفر التركيز على التوثيق طوال عملية التطوير لأصحاب المصلحة فهما شاملا لتقدم المشروع والتسليمات. ومع ذلك ، فإن المنهجيات الخطية لها عيوب ، بما في ذلك المرونة المحدودة لاستيعاب التغييرات ، وارتفاع مخاطر التغييرات في المرحلة المتأخرة ، والحد الأدنى من مشاركة أصحاب المصلحة حتى مراحل لاحقة. يمكن أن تؤدي هذه العوامل إلى تحديات في التكيف مع المتطلبات المتطورة ومعالجة المشكلات بشكل فعال ، مما يؤثر في النهاية على نجاح المشروع.

من ناحية أخرى ، فإن منهجيات SDLC التكرارية ، مثل Agile ، تعطي الأولوية للمرونة والقدرة على التكيف والتحسين المستمر. تتضمن هذه المنهجيات تقسيم المشروع إلى دورات أو تكرارات أصغر ، مع كل تكرار يشمل جميع مراحل SDLC . يسمح هذا النهج التكراري بالتغذية الراجعة والتكيف بشكل متكرر ، مما يمكن فريق المشروع من استيعاب التغييرات أو المتطلبات المتطورة طوال عملية التطوير. تضمن المشاركة النشطة لأصحاب المصلحة في جميع أنحاء المشروع التوافق مع أهداف العمل وتوقعات أصحاب المصلحة. ومع ذلك ، قد تواجه المنهجيات التكرارية تحديات مثل زحف النطاق المحتمل ، وتعقيد التنسيق ، وصعوبة تقدير الجداول الزمنية والميزانيات بدقة. على الرغم من هذه التحديات ، توفر المنهجيات التكرارية فوائد المرونة والتحسين المستمر وتعاون أصحاب المصلحة ، مما يجعلها مناسبة تماما للمشاريع ذات المتطلبات المتطورة والتحديات التقنية المعقدة.

**Task 1.C**

يعد اختيار منهجية دورة حياة تطوير البرمجيات (SDLC) المناسبة أمرا بالغ الأهمية لنجاح المشاريع داخل شركة الخدمات اللوجستية العالمية. يمكن أن يؤثر الاختيار بين المنهجيات التكرارية والخطية بشكل كبير على نتائج المشروع ، بما في ذلك التسليمات ، والالتزام بالجداول الزمنية ، وتخصيص الموارد ، وقابلية التوسع ، والقدرة على التكيف مع التغيير. في هذا التحليل ، سوف نتعمق في الآثار المحتملة لاختيار منهجيات SDLC المختلفة على مخرجات المشروع وإدارته لشركة .Global Logistics Companyسنقوم بتقييم مزايا وقيود كل من النهج التكراري والخطي ، مع الأخذ في الاعتبار المتطلبات والأهداف المحددة للمشاريع اللوجستية للشركة. بناء على هذا التحليل ، سنقدم توصيات قائمة على أسس جيدة لمساعدة الإدارة في اتخاذ قرار مستنير بشأن منهجية SDLC الأكثر ملاءمة لتحقيق أقصى قدر من نجاح المشروع في مجال الخدمات اللوجستية.

1- المنهجيات التكرارية (على سبيل المثال : Agile ، : (Scrum

* المرونة والقدرة على التكيف: توفر المنهجيات التكرارية مثل Agile و Scrum المرونة لاستيعاب المتطلبات المتغيرة والتكيف مع ظروف السوق المتطورة. بالنسبة لشركة لوجستية عالمية ، تعد هذه المرونة أمرا بالغ الأهمية بسبب الطبيعة الديناميكية للصناعة ، حيث يمكن أن تتغير طلبات العملاء واتجاهات السوق بسرعة. يمكن أن تؤدي القدرة على ضبط نطاق المشروع وتقديم تحسينات تدريجية إلى توافق أفضل مع احتياجات العملاء.
* تعزيز التعاون: تؤكد هذه المنهجيات على التعاون الوثيق بين الفرق متعددة الوظائف ، بما في ذلك المطورين ومحللي الأعمال وأصحاب المصلحة. بالنسبة لشركة لوجستية تتعامل مع عمليات سلسلة التوريد المعقدة ، يعد التعاون الفعال أمرا ضروريا لضمان تلبية حلول البرامج للاحتياجات المتنوعة لمختلف الإدارات وأصحاب المصلحة.
* وقت أسرع للوصول إلى السوق: تركز الأساليب التكرارية على تقديم برامج العمل في تكرارات قصيرة ، تتراوح عادة من أسبوعين إلى أربعة أسابيع. يمكن أن تساعد دورة التسليم المتسارعة هذه شركة الخدمات اللوجستية على تقديم ميزات وتحسينات جديدة بسرعة أكبر ، مما يسمح لها بالاستجابة الفورية لمتطلبات السوق واكتساب ميزة تنافسية.
* تخفيف المخاطر: من خلال تقسيم المشروع إلى تكرارات أصغر يمكن التحكم فيها ، تسهل المنهجيات التكرارية التحديد المبكر للمخاطر والتخفيف من حدتها. بالنسبة لشركة لوجستية عالمية ، حيث يمكن أن يكون لفشل النظام أو تأخيره عواقب بعيدة المدى ، فإن الإدارة الاستباقية للمخاطر أمر بالغ الأهمية لضمان سلاسة العمليات ورضا العملاء.

2- المنهجيات الخطية (مثل الشلالwaterfall - ):

* القدرة على التنبؤ والاستقرار: توفر المنهجيات الخطية مثل Waterfall نهجا منظما ويمكن التنبؤ به لتطوير البرمجيات ، مع مراحل متميزة مثل جمع المتطلبات والتصميم والتطوير والاختبار والنشر. يوفر هذا التدفق المتسلسل الاستقرار والمعالم الواضحة لإدارة المشروع ، والتي يمكن أن تكون مفيدة للمشاريع واسعة النطاق ذات المتطلبات المحددة جيدا.
* التخطيط الشامل: تتطلب منهجية الشلال تخطيطا شاملا مقدما ، بما في ذلك وثائق المتطلبات التفصيلية وجداول المشروع. بالنسبة لشركة لوجستية عالمية تتعامل مع مشاريع معقدة ذات تبعيات متعددة ، يعد التخطيط الدقيق أمرا ضروريا لضمان التنفيذ السلس للمهام وتقليل الاضطرابات في العمليات.
* وثائق واضحة: تؤكد منهجيات الشلال على التوثيق في كل مرحلة من مراحل دورة حياة المشروع ، مما يوفر سجلا واضحا للمتطلبات ومواصفات التصميم وحالات الاختبار. يمكن أن تكون هذه الوثائق ذات قيمة لأغراض الامتثال والرجوع إليها في المستقبل ، خاصة في الصناعات عالية التنظيم مثل الخدمات اللوجستية ، حيث تكون إمكانية التتبع والتدقيق أمرا بالغ الأهمية.
* مرونة محدودة: على عكس الأساليب التكرارية ، توفر منهجية الشلال مرونة محدودة لاستيعاب التغييرات بمجرد بدء المشروع. يمكن أن تشكل هذه الصلابة تحديا لشركة لوجستية تعمل في سوق سريع التطور ، حيث قد تتغير المتطلبات بشكل متكرر بسبب التحولات في تفضيلات العملاء أو المتطلبات التنظيمية.

في هذا التحليل ، نتعمق في الآثار المترتبة على اختيار منهجيات SDLC المختلفة ، وهي النهج التكرارية والخطية ، على جوانب المشروع الرئيسية مثل التسليمات ، والالتزام بالجداول الزمنية ، وتخصيص الموارد ، وقابلية التوسع ، والقدرة على التكيف مع التغيير. من خلال استكشاف مزايا وقيود كل منهجية في سياق المشاريع اللوجستية للشركة ، نهدف إلى تقديم رؤى قيمة لمساعدة الإدارة في اتخاذ قرارات مستنيرة لتحقيق أقصى قدر من نجاح المشروع.

|  |  |  |
| --- | --- | --- |
| الجانب | المنهجيات التكرارية | المنهجيات الخطية |
| مخرجات المشروع | تعطي المنهجيات التكرارية ، مثل Agile ، الأولوية لتقديم برامج العمل بشكل تدريجي. وهذا يعني أن مخرجات المشروع يتم تنقيحها وتحسينها باستمرار طوال عملية التطوير ، مما يسمح بردود الفعل المبكرة من أصحاب المصلحة. | تتبع المنهجيات الخطية ، مثل الشلال ، نهجا تسلسليا حيث يتم تحديد مخرجات المشروع مقدما وتطويرها على مراحل. قد يؤدي ذلك إلى دورات تسليم أطول وتأخر التعليقات على التسليمات حتى نهاية المشروع. |
| الالتزام بالجداول الزمنية | تعزز الأساليب التكرارية التكرارات المتكررة ودورات التطوير القصيرة ، والتي يمكن أن تعزز القدرة على الوفاء بالمواعيد النهائية. ومع ذلك ، قد تؤثر التغييرات في المتطلبات أو النطاق أثناء التكرارات على الجداول الزمنية إذا لم تتم إدارتها بشكل فعال. | غالبا ما تتضمن الأساليب الخطية تخطيطا مفصلا مقدما ، بهدف الالتزام الصارم بالجداول الزمنية المحددة مسبقا. ومع ذلك ، قد تتسبب أي انحرافات عن الخطة أو مشكلات غير متوقعة في حدوث تأخيرات في الجدول الزمني العام للمشروع. |
| تخصيص الموارد | تتطلب المنهجيات التكرارية تخصيصا مرنا للموارد لأن المتطلبات والأولويات قد تتغير مع كل تكرار. يجب أن تكون الفرق قابلة للتكيف والاستجابة لاحتياجات المشروع المتغيرة. | تتضمن المنهجيات الخطية عادة تخصيصا أكثر ثباتا للموارد ، حيث يتم وضع خطة المشروع مقدما. وقد يؤدي ذلك إلى تحديات إذا كانت هناك حاجة إلى تعديلات في وقت لاحق من المشروع، مما يتطلب إعادة تخصيص الموارد. |
| قابلية التوسع | غالبا ما تكون الأساليب التكرارية أكثر قابلية للتطوير ، خاصة بالنسبة للمشاريع الكبيرة والمعقدة. يمكن للفرق التوسع أو التقليل بناء على الاحتياجات المتطورة للمشروع ، وإضافة الموارد أو إزالتها حسب الضرورة. | قد تواجه المنهجيات الخطية تحديات قابلية التوسع ، خاصة عند حدوث تغييرات غير متوقعة. قد يكون توسيع نطاق الموارد أكثر تعقيدا بسبب الطبيعة المتسلسلة للعملية. |
| القدرة على التكيف مع التغيير | تتفوق الأساليب التكرارية في القدرة على التكيف مع التغيير ، لأنها تحتضن التحسين التكراري للمتطلبات وتعليقات أصحاب المصلحة المتكررة. يمكن استيعاب التغييرات بسهولة في كل تكرار. | النهج الخطية أقل قدرة على التكيف مع التغيير ، لأنها تتطلب تخطيطا مكثفا مقدما. قد تكون التغييرات المتأخرة في دورة حياة المشروع مكلفة ومدمرة ، مما يؤثر على نتائج المشروع. |

بناء على التحليل المقدم أعلاه ، من الواضح أن اختيار منهجية SDLC يمكن أن يؤثر بشكل كبير على نجاح المشاريع اللوجستية لشركة الخدمات اللوجستية العالمية. بالنظر إلى المتطلبات والأهداف المحددة للشركة ، إلى جانب الآثار المترتبة على المنهجيات المختلفة على نتائج المشروع ، يتم تقديم التوصيات التالية لمساعدة الإدارة في اتخاذ قرار مستنير:

* تقييم خصائص المشروع: تقييم الخصائص المحددة لكل مشروع ، مثل الحجم والتعقيد والنطاق ومستوى عدم اليقين. قد تستفيد المشاريع ذات عدم اليقين العالي والمتطلبات المتطورة من المنهجيات التكرارية مثل Agile أو Scrum ، في حين أن المشاريع ذات المتطلبات المحددة جيدا والنطاق المستقر قد تتوافق بشكل أفضل مع المنهجيات الخطية مثل Waterfall .
* إشراك أصحاب المصلحة: إشراك أصحاب المصلحة الرئيسيين ، بما في ذلك رعاة المشروع والمستخدمين النهائيين وفرق التطوير ، في عملية صنع القرار. فهم تفضيلاتهم ومخاوفهم وأولوياتهم فيما يتعلق بتسليم المشروع والجداول الزمنية والقدرة على التكيف مع التغيير. إن مشاركة أصحاب المصلحة أمر ضروري لنجاح تنفيذ SDLC.
* تقييم توافر الموارد: ضع في اعتبارك توافر الموارد ، بما في ذلك الموظفين المهرة والأدوات والبنية التحتية ، اللازمة لتنفيذ منهجيات SDLC المختلفة بشكل فعال. تقييم ما إذا كانت المؤسسة لديها القدرات والقدرات اللازمة لدعم ممارسات التطوير التكراري، مثل التكرارات المتكررة وبيئات العمل التعاونية وحلقات التغذية الراجعة المستمرة.
* تخفيف المخاطر: تحديد المخاطر المحتملة المرتبطة بكل منهجية SDLC ، مثل تأخير الجدول الزمني ، وتجاوز التكاليف ، وزحف النطاق ، وقضايا الجودة. تطوير استراتيجيات التخفيف من المخاطر وخطط الطوارئ لمعالجة هذه المخاطر بشكل استباقي. على سبيل المثال ، قد تتطلب منهجيات Agile اتصالا وتنسيقا أكثر تكرارا للتخفيف من المخاطر المرتبطة بالمتطلبات المتغيرة.
* المشاريع التجريبية: إجراء مشاريع تجريبية أو إثبات المفاهيم للتحقق من ملاءمة منهجيات SDLC المختلفة في سيناريوهات العالم الحقيقي. اختيار مشاريع تمثيلية بدرجات متفاوتة من التعقيد والحرجية لتقييم فعالية كل نهج في تحقيق النتائج المرجوة. جمع التعليقات من فرق المشروع وأصحاب المصلحة لإبلاغ عملية صنع القرار.
* تبني النهج الهجينة: النظر في اعتماد مناهج SDLC الهجينة التي تجمع بين عناصر كل من المنهجيات التكرارية والخطية للاستفادة من نقاط القوة الخاصة بها والتخفيف من قيودها. على سبيل المثال ، استخدم ممارسات Agile لجمع المتطلبات ومراحل التطوير الأولية ، متبوعة بعمليات تشبه الشلال للتنفيذ والنشر.
* التحسين المستمر: تعزيز ثقافة التحسين المستمر من خلال تشجيع الفرق على التفكير في تجاربهم ومشاركة الدروس المستفادة وتنفيذ تحسينات العملية بشكل متكرر. إنشاء آليات لجمع التغذية الراجعة وقياس مقاييس الأداء وتحديد مجالات التحسين عبر SDLC .

## مقترح ل شركة جلوبال لوجيستيك بالتعاون مع كلية لومينوس الجامعيه التقنيه



School of cloud computing Luminous Technical University College

December 2023

A proposal submitted to Luminous Technical University College

©Ahmad Kallab, 1/4/2024

Supervisor : Eng. Rawan bani younes

Contents

[مقدمة المقترح \* 16](#_Toc165071527)

[1 المقترح 17](#_Toc165071528)

[1.1 تحليل الموقف الحالي للشركه : 17](#_Toc165071529)

[1.2 الأهداف والفرضيات قصيرة وطويلة المدى: 18](#_Toc165071530)

[1.3 تقييم المنهجيه المُتَبَعه: 19](#_Toc165071531)

[1.4 نطاق المشروع: 20](#_Toc165071532)

[1.5 الجدول الزمني للمشروع: 21](#_Toc165071533)

[1.6 ماذا سوف نحقق في نهاية هذا المشروع 22](#_Toc165071534)

[1.7 الميزانيه والمصاريف 23](#_Toc165071535)

[1.8 الختام 25](#_Toc165071536)

# مقدمة المقترح \*

في TechBridge Solutions، شرعنا في مشروع محوري بالتعاون مع Global Logistics، وهي شركة شحن بارزة يقع مقرها الرئيسي في دبي، الإمارات العربية المتحدة. الهدف هو إحداث ثورة في العمليات اللوجستية لشركة Global Logistics من خلال إنشاء نظام إدارة لوجستية متطور قائم على السحابة. باعتباري مهندس DevOps في TechBridge Solutions، أتولى مسؤولية إجراء تحليل شامل ودراسة جدوى ومقترح لتلبية احتياجات الشركة الملحة.

يتضمن نهجنا الخوض في تعقيدات نشر النظام الحالي باستخدام البنية التحتية السحابية وتنفيذ خط أنابيب DevOps لتحسين الكفاءة وقابلية التوسع. بالإضافة إلى ذلك، نقوم بتقييم دقيق لمجموعة من منهجيات تطوير البرمجيات لتحديد النهج الأكثر ملاءمة لهذا المشروع. ومن خلال تحديد نطاق المشروع وتحديد الاستراتيجيات الشاملة لإدارة المشروع، فإننا نهدف إلى ضمان التنفيذ السلس والتسليم في الوقت المناسب.

علاوة على ذلك، يلتزم فريقنا بصياغة الوثائق الفنية التي تتوافق بسلاسة مع DevOps، مما يسهل التكامل السلس والصيانة بعد التنفيذ. في النهاية، لا يسعى حلنا المقترح إلى تحديث العمليات اللوجستية لشركة Global Logistics فحسب، بل يسعى أيضًا إلى تعزيز مستويات خدمة العملاء ودفع التميز التشغيلي في صناعة الشحن التنافسية.

# المقترح

## تحليل الموقف الحالي للشركه :

سياق العمل: تعد شركة Global Logistics، التي يقع مقرها في دبي، الإمارات العربية المتحدة، لاعبًا كبيرًا في لعبة الشحن الدولي. إنهم معروفون بخدماتهم الموثوقة، ولكن مع التطور السريع في صناعة الخدمات اللوجستية، فإنهم بحاجة إلى مواكبة المنافسة للبقاء في المنافسة.

البنية التحتية التكنولوجية: في الوقت الحالي، تستخدم شركة Global Logistics بعض أنظمة المدرسة القديمة جدًا لإدارة عملياتها. فكر في المسارات الورقية وجداول البيانات. إنه يعمل، لكنه ليس متطورًا تمامًا. بالإضافة إلى ذلك، ليس لديهم بيانات في الوقت الفعلي في متناول أيديهم، مما قد يؤدي إلى إبطاء الأمور.

توقعات العملاء: يتوقع العملاء اليوم الكثير من شركات الشحن الخاصة بهم. إنهم يريدون أن يعرفوا بالضبط مكان وجود طرودهم في جميع الأوقات، وأن يحصلوا على تقديرات دقيقة للتسليم، وأن يكون لديهم تواصل واضح طوال العملية. تحتاج شركة Global Logistics إلى تكثيف جهودها لتلبية هذه المطالب.

المشهد التنافسي: هناك الكثير من المنافسة، سواء من شركات الشحن التقليدية أو الأحدث التي تركز على التكنولوجيا. إذا لم تقم شركة Global Logistics بالتحديث، فإنها تخاطر بفقدان العملاء لصالح الشركات التي تقدم المزيد من الراحة والشفافية.

الامتثال التنظيمي: إن العمل في مجال الشحن الدولي يعني التعامل مع الكثير من اللوائح. تحتاج شركة Global Logistics إلى التأكد من اتباعها لجميع القواعد لتجنب الغرامات والدعاوى القضائية والإضرار بسمعتها.

فرص التحسين: على الرغم من التحديات، هناك الكثير من الطرق التي يمكن لشركة Global Logistics أن ترتقي بها إلى مستوى أعلى. ومن خلال تبني الأنظمة المستندة إلى السحابة، واعتماد ممارسات DevOps، واستخدام تحليلات البيانات، يمكنهم تبسيط عملياتهم، واتخاذ قرارات أكثر ذكاءً، وإبقاء عملائهم سعداء.

الأهداف الإستراتيجية: في نهاية المطاف، تريد شركة Global Logistics تحديث عملياتها لتظل قادرة على المنافسة وتلبية توقعات العملاء. وهذا يعني تحسين الكفاءة وتعزيز خدمة العملاء واستخدام التكنولوجيا لدفع النمو.

## الأهداف والفرضيات قصيرة وطويلة المدى:

* تعزيز خدمة العملاء: الهدف الرئيسي الآخر هو رفع مستوى خدمة العملاء التي تقدمها شركة .Global Logistics ستمكن منصة البرمجيات الجديدة من تتبع الشحنات في الوقت الفعلي، وتقديرات التسليم الدقيقة، وتحسين قنوات الاتصال مع العملاء. سيؤدي ذلك إلى قدر أكبر من الشفافية، وزيادة رضا العملاء، وفي نهاية المطاف، تعزيز ولاء العملاء.
* تبسيط العمليات: الهدف الأساسي لهذا المشروع هو تنفيذ منصة برمجية جديدة من شأنها تبسيط عمليات الخدمات اللوجستية العالمية. ومن خلال رقمنة العمليات الرئيسية وأتمتتها مثل إدارة الطلبات وتتبع الشحنات وإدارة المخزون، فإننا نهدف إلى تحسين الكفاءة وتقليل الأخطاء اليدوية وتعزيز الإنتاجية الإجمالية.

من المتوقع أن يكون لتطبيق النظام الأساسي الجديد تأثير تحويلي على عمليات Global Logistics وقدرات خدمة العملاء. ومن خلال الانتقال من العمليات اليدوية الورقية إلى النظام الرقمي القائم على السحابة، نتوقع تحسينات كبيرة في الكفاءة والدقة والاستجابة.

ومع وجود المنصة الجديدة، ستتمكن شركة Global Logistics من أتمتة المهام الروتينية، مثل معالجة الطلبات وإدارة المخزون، مما يسمح للموظفين بالتركيز على المزيد من الأنشطة ذات القيمة المضافة. سيؤدي ذلك إلى أوقات تسليم أسرع، وتقليل الأخطاء، وفي نهاية المطاف، خفض تكاليف التشغيل.

علاوة على ذلك، فإن الرؤية المحسنة التي توفرها منصة البرمجيات ستمكن شركة Global Logistics من توفير مستوى عالٍ من خدمة العملاء. سيتمكن العملاء من الوصول إلى معلومات التتبع في الوقت الفعلي، وتقديرات التسليم الدقيقة، وقنوات الاتصال السلسة، مما يسمح لهم بالبقاء على اطلاع ومشاركة طوال عملية الشحن.

بشكل عام، تتوافق أهداف المشروع مع هدف جلوبال لوجيستكس المتمثل في تحديث عملياتها وتقديم خدمة استثنائية لعملائها. ومن خلال تنفيذ منصة البرمجيات الجديدة، نهدف إلى تحقيق التميز التشغيلي ووضع شركة Global Logistics كشركة رائدة في صناعة الخدمات اللوجستية.

## تقييم المنهجيه المُتَبَعه:

تقييم المنهجيات المختلفة: أجرينا تقييماً شاملاً لمختلف منهجيات تطوير البرمجيات، بما في ذلك النهج الخطي والتكراري. تتبع المنهجيات الخطية، مثل Waterfall، عملية متسلسلة خطوة بخطوة، بينما تتضمن المنهجيات التكرارية، مثل Agile، تطويرًا تدريجيًا وحلقات تغذية راجعة مستمرة.

التوافق مع متطلبات المشروع: تم تحليل كل منهجية لتحديد توافقها مع المتطلبات والأهداف المحددة للمشروع. تعتبر المنهجيات الخطية مناسبة تمامًا للمشاريع ذات المتطلبات الثابتة ونطاق واضح ومحدد مسبقًا. إنها توفر نهجًا منظمًا للتطوير، والذي يمكن أن يكون مفيدًا للمشاريع ذات المواعيد النهائية الصارمة ومتطلبات الامتثال التنظيمية. ومن ناحية أخرى، توفر المنهجيات التكرارية المرونة والقدرة على التكيف، مما يجعلها مثالية للمشاريع ذات المتطلبات المتطورة أو غير المؤكدة. إنهم يعطون الأولوية للتعاون والاستجابة للتغيير وتقديم القيمة للعملاء في التكرارات المتزايدة.

التوصية بالمنهجية الأكثر ملاءمة: بناءً على تحليلنا، نوصي باعتماد منهجية تكرارية، مثل Agile، لتطوير منصة برمجية جديدة للخدمات اللوجستية العالمية. توفر المنهجيات الرشيقة العديد من المزايا التي تتوافق بشكل وثيق مع أهداف المشروع، بما في ذلك:

* المرونة: تسمح Agile بالتطوير التكراري والتكيف المستمر مع المتطلبات المتغيرة، مما يضمن بقاء الحل البرمجي متوافقًا مع الاحتياجات المتطورة للوجستيات العالمية.
* مشاركة أصحاب المصلحة: تعمل Agile على تعزيز التعاون الوثيق بين فريق التطوير وأصحاب المصلحة، مما يتيح الحصول على تعليقات متكررة والتحقق من صحة ميزات البرنامج ووظائفه.
* وقت وصول أسرع إلى السوق: من خلال تقديم برامج العمل في تكرارات قصيرة، تسهل Agile النماذج الأولية السريعة والتسليم المبكر للقيمة، مما يسمح لشركة Global Logistics بتحقيق الفوائد بشكل أسرع والاستجابة بسرعة لمتطلبات السوق.
* ضمان الجودة: تؤكد Agile على الاختبار المستمر وضمان الجودة طوال عملية التطوير، مما يؤدي إلى برامج عالية الجودة مع عدد أقل من العيوب.

## نطاق المشروع:

يشير نطاق المشروع إلى حدود المشروع وتسليماته، ويحدد ما سيتم تضمينه واستبعاده من جهود تطوير البرمجيات. وهو يشمل كلا من المتطلبات الوظيفية، التي تحدد ما يجب أن يفعله النظام، والمتطلبات غير الوظيفية، التي تصف كيفية أداء النظام.

بالنسبة لمشروع نظام إدارة الخدمات اللوجستية القائم على السحابة لشركة Global Logistics، يتضمن النطاق تطوير منصة برمجية من شأنها تبسيط العمليات وتعزيز خدمة العملاء. قد تتضمن المتطلبات الوظيفية ميزات مثل إدارة الطلبات وتتبع المخزون وتتبع الشحنات وإمكانيات إعداد التقارير. قد تتضمن المتطلبات غير الوظيفية معايير الأداء وقابلية التوسع والأمان وسهولة الاستخدام.

قدمت شركة Global Logistics قائمة أولية بمتطلبات المشروع، موضحة توقعاتها والوظائف المطلوبة. تعمل هذه المتطلبات كنقطة انطلاق لجهود التطوير، حيث توفر رؤى قيمة حول احتياجات العمل وأهدافه. ومع ذلك، فمن الضروري أن ندرك أن القائمة الأولية للمتطلبات قد لا تكون شاملة أو نهائية.

يعد الانتهاء من المتطلبات قبل المضي قدمًا أمرًا بالغ الأهمية لعدة أسباب:

* الوضوح والاتساق: يضمن الانتهاء من المتطلبات أن يكون لدى جميع أصحاب المصلحة فهم واضح ومتسق لنطاق المشروع وأهدافه. فهو يقلل من الغموض ويقلل من مخاطر سوء الفهم أو التفسيرات الخاطئة أثناء عملية التطوير.
* التوافق مع احتياجات أصحاب المصلحة: من خلال إشراك أصحاب المصلحة في عملية الانتهاء من المتطلبات، يمكن لفريق المشروع التأكد من أن الحل البرمجي يلبي احتياجات وتوقعات المستخدمين النهائيين والعملاء وأصحاب المصلحة الرئيسيين الآخرين. فهو يسمح بالتعليقات والتحقق من الصحة لمعالجة أي ثغرات أو تناقضات في المتطلبات الأولية.
* خط الأساس للتخطيط والتقدير: توفر المتطلبات النهائية أساسًا ثابتًا لتخطيط المشروع وتخصيص الموارد وتقدير التكلفة. إنها تمكن فريق المشروع من تطوير جدول زمني وميزانية وخريطة طريق واقعية لجهود تطوير البرمجيات، مع الأخذ في الاعتبار نطاق العمل والتسليمات.
* إدارة المخاطر: يساعد الانتهاء من المتطلبات في وقت مبكر من دورة حياة المشروع على التخفيف من مخاطر زحف النطاق، حيث يتم تقديم ميزات أو تغييرات إضافية لاحقًا في عملية التطوير. تقلل المتطلبات الواضحة والمتفق عليها من احتمالية حدوث تغييرات في النطاق وتقلل من التأثير على الجداول الزمنية للمشروع وميزانياته.

## الجدول الزمني للمشروع:

|  |  |
| --- | --- |
| مهام المشروع | الفتره الزمنيه |
| جمع وتحليل المتطلبات مع أصحاب المصلحة وتوثيقها | 1 إبريل 2024 - 7 إبريل 2024 |
| تثبيت المتطلبات مع فريق Global Logistics |  |
| تصميم هندسة البرمجيات ومخطط قاعدة البيانات | 8 إبريل 2024 - 21 إبريل 2024 |
| تطوير مكونات الواجهة الأمامية والخلفية للنظام |  |
| بدء التكامل مع الأنظمة الخارجية وواجهات البرمجة التطبيقية |  |
| نشر النظام في بيئة تجريبية وإجراء اختبارات شاملة لوظائفه | 22 إبريل 2024 - 28 إبريل 2024 |
| معالجة أي مشاكل أو عيوب تم تحديدها أثناء الاختبار |  |
| تطبيق النظام على كافة عمليات Global Logistics |  |
| إجراء جلسات تدريبية للموظفين على استخدام النظام | 29 إبريل 2024 - 12 مايو 2024 |
| استكمال التوثيق الفني والدليل الإرشادي للمستخدم |  |
| رصد أداء النظام وتغذية ردود الفعل من المستخدمين | 13 مايو 2024 - 19 مايو 2024 |
| تحديد المجالات التي يمكن تحسينها وتطويرها |  |
| تنفيذ أي تحديثات أو تحسينات ضرورية |  |
| مراجعة نهائية والتوقيع مع فريق Global Logistics | 20 مايو 2024 - 30 مايو 2024 |
| إعداد وثائق المشروع ومواد التسليم |  |
| إغلاق مهام وموارد المشروع | 1 مايو 2024 – 5 مايو 2024 |

ملاحظة: يمكن تغيير التواريخ بناءً على تقدم المشروع وتوافر أصحاب المصلحة. سيتم إعلام جميع الأطراف ذات الصلة بأي تعديلات على الجدول الزمني بشكل سريع.

## ماذا سوف نحقق في نهاية هذا المشروع

* تحسين الكفاءة التشغيلية: سيعمل نظام إدارة الخدمات اللوجستية الجديد القائم على السحابة على تبسيط عملياتنا، مما يؤدي إلى زيادة الكفاءة في التعامل مع الشحنات وإدارة المخزون ومعالجة الطلبات.
* خدمة عملاء محسنة: من خلال تطبيق هذا النظام، نهدف إلى توفير تجربة أفضل للعملاء من خلال أوقات استجابة أسرع ومعالجة دقيقة للطلبات وإمكانات التتبع المحسنة.
* قابلية التوسع والمرونة: ستسمح لنا طبيعة النظام القائمة على السحابة بتوسيع نطاق عملياتنا حسب الحاجة لاستيعاب نمو الأعمال ومتطلبات السوق المتغيرة. بالإضافة إلى ذلك، فإنه سيوفر المرونة للوصول عن بعد والتكامل السلس مع تطبيقات الأعمال الأخرى.
* اتخاذ القرارات المستندة إلى البيانات: بفضل ميزات إعداد التقارير والتحليلات القوية، سيمكننا النظام من الحصول على رؤى قيمة حول عملياتنا اللوجستية، مما يساعدنا على اتخاذ قرارات مستنيرة وتحسين العمليات لتحقيق أداء أفضل.
* توفير التكاليف: من خلال أتمتة المهام اليدوية وتحسين سير العمل، نتوقع توفير التكاليف من حيث تقليل ساعات العمل وتقليل الأخطاء وتحسين استخدام الموارد.

بشكل عام، سيمكننا هذا المشروع من تحديث عملياتنا اللوجستية، والحفاظ على قدرتنا التنافسية في الصناعة، وتقديم خدمة استثنائية لعملائنا.

## الميزانيه والمصاريف

## 

|  |  |
| --- | --- |
| البند | التكلفه بالدولار |
| الشبكات والمساحات التخزينيه والمثيلات من AWS | 50$ |
| اضافات جينكينز (jenkins) | 20$ |
| خدمات DevOps من Azure | 10$ |
| التكلفه الاجماليه | 100$ |

## الختام

وفي الختام، يمثل نظام إدارة اللوجستيات السحابي المقترح فرصة مثيرة لشركة Global Logistics لإحداث ثورة في عملياتها والارتقاء بخدمة العملاء إلى آفاق جديدة. ومن خلال التحليل الشامل والدراسة المتأنية لمنهجيات تطوير البرمجيات والتخطيط الشامل للمشروع، وضعنا الأساس للتنفيذ الناجح.

ومن خلال الاستفادة من أحدث التقنيات وأفضل الممارسات في تطوير البرمجيات، فإننا نهدف إلى تحقيق العديد من الأهداف الرئيسية. ويشمل ذلك تحسين الكفاءة التشغيلية، وتعزيز خدمة العملاء، وتمكين قابلية التوسع والمرونة، وتسهيل اتخاذ القرارات المستندة إلى البيانات، وتحقيق وفورات في التكاليف.

علاوة على ذلك، يتماشى المشروع المقترح مع الأهداف الإستراتيجية لشركة Global Logistics المتمثلة في تحديث عملياتها اللوجستية والبقاء في المقدمة في مشهد السوق التنافسي. ومن خلال الفهم الواضح لنطاق المشروع، ونهج إدارة المشروع المحدد جيدًا، والتقديرات الواقعية للميزانية، فإننا على أهبة الاستعداد للنجاح.

وبينما نبدأ هذه الرحلة، سيكون التعاون والتواصل أمرًا بالغ الأهمية. ونحن نتطلع إلى العمل بشكل وثيق مع فريق الخدمات اللوجستية العالمية، لمعالجة أي تحديات قد تنشأ، والاحتفال بالإنجازات معًا. معًا، سنعمل على تحويل Global Logistics إلى مؤسسة أكثر مرونة وكفاءة وتركيزًا على العملاء.

وفي الختام، نحن واثقون من أن نظام إدارة الخدمات اللوجستية القائم على السحابة المقترح لن يلبي التوقعات فحسب، بل سيتجاوزها، مما يوفر فوائد ملموسة لشركة Global Logistics وعملائها. نشكرك على النظر في اقتراحنا، ونتطلع إلى الفرصة لتحقيق هذا المشروع.

Task 1.D

الوثائق الفنية: خطوط أنابيب DevOps لنظام إدارة اللوجستيات القائم على السحابة

1**. المقدمة:**

* تم تصميم مسارات DevOps الموضحة في هذه الوثائق لتلبية الاحتياجات المحددة لشركة Global Logistics، وهي شركة شحن دولية تهدف إلى تحديث عملياتها اللوجستية.
* تعمل خطوط الأنابيب هذه كأساس لأتمتة عمليات البناء والاختبار والنشر لنظام إدارة الخدمات اللوجستية القائم على السحابة، مما يضمن الكفاءة والموثوقية وقابلية التوسع في تسليم البرامج.
* ومن خلال تنفيذ أفضل ممارسات DevOps والأدوات الحديثة، تتيح خطوط الأنابيب التعاون السلس بين التطوير والعمليات وأصحاب المصلحة الآخرين، مما يعزز المرونة والابتكار في تطوير البرمجيات.

2**. الإعداد:**

إعداد البنية التحتية:

* مثيلات AWS EC2 ذات المواصفات المناسبة للتشغيل المرحلي والإنتاج وJenkins.
* مجموعات الأمان وتكوينات الشبكة لضمان الاتصال الآمن بين الحالات.

تكوين الادوات:

* تثبيت Jenkins وتكوينه على مثيل Jenkins EC2 المخصص لتنظيم خطوط أنابيب CI/CD.
* دمج Jenkins مع GitHub لإدارة التعليمات البرمجية المصدر، مما يتيح عمليات إنشاء تلقائية يتم تشغيلها بواسطة عمليات تنفيذ التعليمات البرمجية.
* تكوين مستودع Docker Hub لتخزين وإدارة صور Docker، وتسهيل النشر في الحاويات.

3**. النشر:**

خط أنابيب التكامل المستمر ( (CI:

* استخدم Jenkins لأتمتة عملية CI، بما في ذلك تجميع التعليمات البرمجية والاختبار وإنشاء العناصر.
* الاستفادة من Docker في النقل بالحاويات، مما يتيح إنشاءات متسقة وقابلة للتكرار عبر البيئات.
* تنفيذ اختبارات الوحدة واختبارات التكامل وفحوصات الجودة الأخرى لضمان موثوقية كل بناء.

خط أنابيب النشر المستمر ( (CD:

* أتمتة عملية النشر إلى البيئة المرحلية عند اكتمال بناء CI بنجاح.
* استخدم خطوط أنابيب Jenkins لنشر حاويات Docker لتجهيز المثيلات، مما يتيح التكرار السريع والتغذية الراجعة.
* تنفيذ إستراتيجيات النشر باللونين الأزرق والأخضر لتقليل وقت التوقف عن العمل وتخفيف المخاطر أثناء عمليات نشر الإنتاج.

4**. المراقبة:**

الفحوصات:

* تنفيذ فحوصات صحية آلية لمراقبة حالة النظام واستخدام الموارد ومقاييس الأداء.
* استخدام AWS CloudWatch لمراقبة مثيلات EC2 وحاويات Docker وموارد AWS الأخرى، مما يؤدي إلى إطلاق تنبيهات بشأن الحالات الشاذة.

التسجيل والتنبيه:

* تكوين آليات التسجيل المركزية لالتقاط سجلات التطبيق وأحداث النظام والأخطاء.
* إعداد آليات التنبيه باستخدام AWS SNS أو Slack أو خدمات الإشعارات الأخرى لإخطار أصحاب المصلحة بالأحداث الهامة أو تدهور الأداء.

**5. الصيانة:**

مهام الصيانة الروتينية:

* وضع جداول صيانة منتظمة لتحديث أنظمة التشغيل وتبعيات البرامج وتصحيحات الأمان.
* تنفيذ آليات التحديث الآلي لتبسيط عملية الصيانة وتقليل وقت التوقف عن العمل.

النسخ الاحتياطي والتعافي من الكوارث:

* تطوير استراتيجيات النسخ الاحتياطي والتعافي من الكوارث لحماية البيانات والتكوينات الهامة.
* استخدم خدمات النسخ الاحتياطي AWS أو حلول الجهات الخارجية لإنشاء النسخ الاحتياطية وإدارتها، مما يضمن سلامة البيانات وتوافرها في حالات الطوارئ.

**6. الختام:**

تم تصميم هذه الخطوط الأنابيب لتلبية الاحتياجات الخاصة لشركة Global Logistics، مع مراعاة متطلباتها الفنية والتشغيلية. بفضل التطبيق الفعال والشامل لمفاهيم DevOps وأفضل الممارسات، واستخدام الأدوات التكنولوجية المتقدمة، تم تحقيق تحسينات كبيرة في عمليات التطوير والنشر والمراقبة والصيانة. تم تبني نموذج Agile كإطار عمل لتسهيل التعاون والتكيف مع متطلبات العمل المتغيرة بشكل مستمر، مما أدى إلى تسليم مشروع ناجح يلبي توقعات واحتياجات الشركة بفعالية.

Task 1.E

**1. تلبية احتياجات العميل والمستخدم:**

**أهداف عمل العميل:**

* الهدف الأساسي للعميل هو تحديث نظام إدارة الخدمات اللوجستية الخاص به ليظل قادرًا على المنافسة في مشهد الصناعة سريع التطور. ويستلزم ذلك الانتقال من البنية التحتية التقليدية المحلية إلى الحلول المستندة إلى السحابة التي توفر قدرًا أكبر من المرونة وقابلية التوسع والفعالية من حيث التكلفة.
* من خلال اعتماد التقنيات السحابية وممارسات DevOps، يمكّن الحل الموصى به العميل من تحقيق أهداف التحديث الخاصة به. توفر الأنظمة الأساسية السحابية مثل AWS بيئة بنية تحتية قوية كخدمة IaaS) ) يمكنها التوسع حسب الطلب لاستيعاب التقلبات في عبء العمل ومتطلبات الأعمال.
* تعمل منهجيات DevOps، بما في ذلك التكامل المستمر والتسليم المستمر والبنية التحتية كرمز، على تسهيل الوصول إلى السوق بشكل أسرع وتحسين التعاون بين فرق التطوير والعمليات وتعزيز المرونة العامة. يتماشى هذا مع الأهداف الإستراتيجية للعميل المتمثلة في تسريع الابتكار والاستجابة السريعة لتغيرات السوق.

**تحسين تجربة المستخدم النهائي:**

* بالنسبة للمستخدمين النهائيين، يقدم الحل مجموعة من الميزات والوظائف المصممة لتعزيز تجربتهم الشاملة وتبسيط العمليات اللوجستية اليومية. تتيح إمكانات التتبع في الوقت الفعلي للمستخدمين مراقبة حالة الشحنات العابرة وموقعها، مما يوفر قدرًا أكبر من الرؤية والشفافية عبر سلسلة التوريد.
* تعمل الإشعارات والتنبيهات التلقائية على إبقاء المستخدمين على اطلاع بالأحداث المهمة، مثل التأخيرات أو تغييرات المسار أو استثناءات التسليم، مما يمكنهم من معالجة المشكلات بشكل استباقي وتقليل الاضطرابات. يساعد هذا النهج الاستباقي للتواصل على بناء الثقة بين المستخدمين النهائيين ويعزز علاقات أقوى بين العميل وعملائه.
* تسهل الواجهات سهلة الاستخدام وسير العمل البديهي على المستخدمين التنقل في النظام وتنفيذ المهام والوصول إلى المعلومات ذات الصلة بسرعة وكفاءة. وهذا لا يؤدي إلى تحسين رضا المستخدمين فحسب، بل يزيد أيضًا من الإنتاجية ويقلل من احتمالية حدوث أخطاء أو عدم الكفاءة في العمليات اللوجستية.

2**. تخفيف المخاطر:**

**الإستراتيجية الشاملة لإدارة المخاطر:**

* تم تطوير استراتيجية شاملة لإدارة المخاطر لتحديد المخاطر المحتملة وتقييمها ومعالجتها بشكل منهجي طوال دورة حياة المشروع. تم تصنيف المخاطر على أساس خطورتها واحتماليتها وتأثيرها المحتمل على أهداف المشروع.
* يتضمن تحديد المخاطر إشراك أصحاب المصلحة من مختلف الإدارات والتخصصات لجمع الأفكار ووجهات النظر حول التهديدات ونقاط الضعف المحتملة. ويضمن هذا النهج التعاوني تحديد المخاطر بشكل شامل ودقيق.
* ثم تم تقييم المخاطر باستخدام معايير محددة لتحديد أولويات جهود التخفيف. تم إعطاء الأولوية القصوى للمخاطر عالية التأثير التي قد تؤدي إلى تعطيل تقدم المشروع أو نتائجه بشكل كبير، تليها المخاطر المتوسطة والمنخفضة التأثير.

**تدابير التخفيف:**

* تمت معالجة المخاطر عالية التأثير، مثل الثغرات الأمنية وفقدان البيانات، من خلال تنفيذ تدابير أمنية قوية. وشمل ذلك تشفير البيانات الحساسة، وتنفيذ ضوابط الوصول وآليات المصادقة، وعمليات التدقيق الأمني المنتظمة وتقييمات الضعف.
* للتخفيف من مخاطر فشل النظام وضمان التوفر العالي، تم تنفيذ آليات التكرار وتجاوز الفشل على مستويات مختلفة من البنية التحتية. وتضمن ذلك أنظمة تخزين متكررة، وتكوينات خادم متوازنة التحميل، واستراتيجيات نشر متعددة المناطق لتقليل تأثير الانقطاعات المحتملة أو فشل الأجهزة.
* وبالإضافة إلى التدابير الفنية، تم تنفيذ العوامل البشرية مثل برامج التدريب والتوعية للتخفيف من المخاطر المرتبطة بالخطأ البشري أو الإهمال. وشمل ذلك توفير التدريب على أفضل الممارسات الأمنية، وإجراء جلسات توعية أمنية منتظمة، وفرض ضوابط الوصول والأذونات الصارمة.

**المراقبة والصيانة الاستباقية:**

* تم إجراء أنشطة مراقبة منتظمة وصيانة استباقية طوال دورة حياة المشروع لاكتشاف المخاطر ومعالجتها في الوقت الفعلي. وتضمن ذلك تنفيذ أدوات المراقبة وأنظمة التنبيه للكشف عن الحالات الشاذة أو مشكلات الأداء أو الخروقات الأمنية.
* تم تكوين آليات التنبيه الآلية لإخطار أصحاب المصلحة بالمخاطر المحتملة أو الانحرافات عن المعايير المتوقعة، مما يسمح بالاستجابة والحل السريع. بالإضافة إلى ذلك، تم إجراء مهام الصيانة الاستباقية مثل تحديثات البرامج وإدارة التصحيح والنسخ الاحتياطية للنظام بانتظام لتقليل احتمالية وقوع حوادث أمنية أو فشل النظام.

3**. قرارات إدارة المشروع:**

**اختيار المنهجيات الرشيقة:**

* تم اختيار منهجيات Agile، مثل Scrum أو Kanban، لإدارة المشروع نظرًا لطبيعتها التكرارية والتعاونية، والتي تتوافق جيدًا مع بيئة الأعمال الديناميكية للعميل والحاجة إلى المرونة والقدرة على التكيف.
* يسمح Agile بالتسليم المتزايد للميزات والوظائف، مما يمكّن العميل من الحصول على قيمة ملموسة في وقت مبكر من عملية التطوير وتوفير الفرص للتعليقات والتحسينات المستمرة.

**تفصيل قصة المستخدم وتحديد الأولويات:**

* تم تقسيم متطلبات المشروع إلى وحدات أصغر يمكن التحكم فيها تسمى قصص المستخدم، تمثل كل منها جزءًا محددًا من الوظيفة أو الميزة من منظور المستخدم النهائي.
* تم إجراء جلسات تخطيط Sprint لتحديد أولويات قصص المستخدمين هذه بناءً على عوامل مثل قيمة الأعمال والتعقيد والتبعيات. ويضمن هذا معالجة الميزات الأكثر أهمية أولاً وتسليمها في الوقت المناسب.

التسليم التكراري والملاحظات المستمرة:

* اتبع المشروع سلسلة من دورات التطوير التكرارية، المعروفة باسم سباقات السرعة، حيث تستغرق كل دورة عادةً من أسبوعين إلى أربعة أسابيع. أثناء كل سباق، ركز فريق التطوير على تنفيذ مجموعة فرعية من قصص المستخدمين ذات الأولوية.
* تم إجراء مراجعات منتظمة للسباق في نهاية كل سباق لتوضيح العمل المكتمل لأصحاب المصلحة وجمع التعليقات. وقد أتاح ذلك لأصحاب المصلحة تقديم مدخلات بشأن الميزات المقدمة، واقتراح تغييرات أو تحسينات، وضمان التوافق مع أهداف العمل.

**النهج التعاوني وإشراك أصحاب المصلحة:**

* تم التأكيد على التعاون الوثيق بين فريق التطوير وأصحاب المصلحة والمستخدمين النهائيين طوال المشروع. وشمل ذلك التواصل المنتظم وجلسات التعليقات وأدوات التعاون مثل Jira وSlack لتسهيل تبادل المعلومات واتخاذ القرار.
* وقد شارك أصحاب المصلحة والمستخدمون النهائيون بنشاط في المشروع منذ البداية وحتى التسليم، مما يضمن تلبية احتياجاتهم ومتطلباتهم بدقة ومعالجتها. وقد عزز هذا النهج التعاوني الشعور بالملكية والمشاركة من جميع الأطراف المعنية، مما أدى في النهاية إلى حل يلبي احتياجات وتوقعات العميل والمستخدمين على حد سواء.

Task 1.F

مراجعة شاملة لمنهجيات دورة حياة تطوير البرمجيات (SDLC) للمشروع:

1. منهجية waterfall **:**

**أهداف المشروع:**

في سياق مشروع اللوجستيات العالمية، حيث يكون الهدف هو تحديث وتبسيط أنظمة إدارة اللوجستيات، يمكن أن تكون منهجية الشلال مفيدة. ومن خلال نهجه المتسلسل، يتوافق Waterfall جيدًا مع هدف المشروع المتمثل في تقديم نظام مستقر ومحدد جيدًا. ومن خلال التخطيط الدقيق للمتطلبات وتوثيقها مسبقًا، تضمن المنهجية أن يكون لدى فريق المشروع فهم واضح للنطاق والأهداف، وهو أمر بالغ الأهمية لمشروع بهذا الحجم والتعقيد.

**القيود:**

ومع ذلك، قد تواجه منهجية الشلال تحديات في تلبية احتياجات المشروع، خاصة بالنظر إلى الطبيعة الديناميكية لصناعة الخدمات اللوجستية. ونظرًا للمتطلبات المتطورة وظروف السوق، فإن الهيكل الصارم لـ Waterfall قد يعيق قدرة المشروع على التكيف والاستجابة بفعالية. قد تتطلب أي تغييرات أو تحديثات للمتطلبات أثناء عملية التطوير إعادة النظر في المراحل السابقة، مما قد يؤدي إلى تأخيرات وزيادة التكاليف.

**توقعات أصحاب المصلحة:**

يتوقع أصحاب المصلحة في مشروع الخدمات اللوجستية العالمية، بما في ذلك مديري الخدمات اللوجستية والعملاء والقيادة التنفيذية، حلاً لا يلبي احتياجاتهم الحالية فحسب، بل يوفر أيضًا قابلية التوسع والمرونة للنمو المستقبلي. في حين أن منهجية Waterfall توفر إمكانية التنبؤ والتحكم في الجداول الزمنية للمشروع، فقد يقدر أصحاب المصلحة التعليقات المبكرة والمتكررة لضمان توافق الحل مع متطلباتهم المتطورة. قد يُنظر إلى قلة الفرص لمشاركة أصحاب المصلحة في المراحل الأولى من التطوير على أنها قيد من قدرة الشلال على تلبية هذه التوقعات.

**نقاط القوة:**

توفر منهجية الشلال بنية ووثائق واضحة، والتي يمكن أن تكون مفيدة لمشروع مثل Global Logistics الذي يتطلب تخطيطًا وتوثيقًا شاملين للمتطلبات والعمليات وسير العمل. من خلال نهجها القائم على الإنجازات، توفر Waterfall لأصحاب المصلحة خريطة طريق واضحة لتقدم المشروع، مما يساعد على إدارة التوقعات وضمان التوافق مع أهداف العمل.

**نقاط الضعف:**

ومع ذلك، فإن المرونة المحدودة لمنهجية الشلال والتغذية الراجعة المتأخرة قد تشكل تحديات في سياق مشروع اللوجستيات العالمية. قد يؤدي عدم القدرة على استيعاب التغييرات أو التحديثات للمتطلبات أثناء عملية التطوير إلى تقديم حل لا يلبي بشكل كامل الاحتياجات المتطورة لأصحاب المصلحة. بالإضافة إلى ذلك، فإن المشاركة المتأخرة لأصحاب المصلحة في مرحلة الاختبار تزيد من خطر تحديد المشكلات أو التناقضات في وقت متأخر من دورة التطوير، مما قد يؤثر على الجداول الزمنية للمشروع وميزانياته.

**المواءمة مع احتياجات المشروع:**

في حين أن منهجية الشلال توفر البنية والقدرة على التنبؤ، إلا أن حدودها في التكيف مع المتطلبات المتغيرة والمشاركة المتأخرة لأصحاب المصلحة قد لا تتماشى بشكل كامل مع احتياجات مشروع اللوجستيات العالمية. ونظرًا للطبيعة الديناميكية لصناعة الخدمات اللوجستية وتركيز المشروع على الابتكار وخفة الحركة، قد تكون منهجيات SDLC البديلة مثل Agile أو Scrum أكثر ملاءمة لمعالجة المتطلبات والقيود المحددة للمشروع.

2. منهجية Scrum :

**أهداف المشروع:**

يمكن أن تكون منهجية سكروم مناسبة للغاية لمشروع اللوجستيات العالمية، الذي يهدف إلى تحديث أنظمة إدارة اللوجستيات. يسمح أسلوب Scrum التكراري والتزايدي بتسليم برامج العمل في تكرارات قصيرة، تُعرف باسم سباقات السرعة. يتماشى هذا جيدًا مع هدف المشروع المتمثل في تقديم القيمة لأصحاب المصلحة بسرعة وتحسين النظام بشكل مستمر بناءً على التعليقات.

**قيود:**

على الرغم من أن Scrum يوفر المرونة والقدرة على التكيف، إلا أنه قد يواجه قيودًا في المشاريع ذات المواعيد النهائية المحددة أو المتطلبات التنظيمية الصارمة. ومع ذلك، بالنسبة لمشروع اللوجستيات العالمية، حيث يتم إعطاء الأولوية للابتكار والاستجابة، فإن قدرة Scrum على استيعاب المتطلبات المتغيرة وظروف السوق يمكن أن تكون مفيدة.

**توقعات أصحاب المصلحة:**

يتوقع أصحاب المصلحة في مشروع Global Logistics حلاً لا يلبي احتياجاتهم الحالية فحسب، بل يوفر أيضًا المرونة وقابلية التوسع للنمو المستقبلي. يعمل Scrum على تعزيز مشاركة أصحاب المصلحة طوال عملية التطوير، مع توفير فرص منتظمة للتعليقات والتعاون. وهذا يضمن أن الحل يتوافق بشكل وثيق مع توقعات أصحاب المصلحة ويمكن أن يتطور لتلبية المتطلبات المتغيرة.

**نقاط القوة:**

يقدم Scrum العديد من نقاط القوة ذات الصلة بشكل خاص بمشروع اللوجستيات العالمية. ويسمح نهجها التكراري بالتسليم المبكر والمتكرر للميزات القيمة، مما يمكّن أصحاب المصلحة من رؤية التقدم الملموس وتقديم التعليقات طوال عملية التطوير. بالإضافة إلى ذلك، يعمل Scrum على تعزيز فرق التنظيم الذاتي والشفافية، مما يعزز بيئة عمل تعاونية وممكنة.

**نقاط الضعف:**

في حين أن سكروم يشجع المرونة والقدرة على التكيف، فإنه قد يتطلب تحولا ثقافيا داخل المنظمات المعتادة على أساليب إدارة المشاريع الأكثر تقليدية. بالإضافة إلى ذلك، يعتمد Scrum بشكل كبير على التواصل والتعاون الفعالين، الأمر الذي قد يمثل تحديًا للفرق أو المؤسسات الموزعة ذات الهياكل الهرمية.

**المواءمة مع احتياجات المشروع:**

بشكل عام، تتوافق منهجية Scrum بشكل جيد مع احتياجات المشروع اللوجستي العالمي، مع التركيز على التعاون والمرونة والتحسين المستمر. من خلال الاستفادة من ممارسات Scrum مثل التخطيط السريع، والوقوف اليومي، ومراجعات السباق، يمكن لفريق المشروع إدارة المتطلبات بشكل فعال، وتحديد أولويات العمل، وتقديم القيمة بشكل متزايد. إن تركيز Scrum على تقديم برامج العمل بشكل متكرر والتكيف مع التغيير يضمن أن يظل المشروع مستجيبًا لاحتياجات أصحاب المصلحة وديناميكيات السوق.

3. Rapid Application Development:

**أهداف المشروع:**

بينما يركز RAD على النماذج الأولية السريعة والتكرار، قد يكون لمشروع اللوجستيات العالمية متطلبات معقدة تتطلب اتباع نهج أكثر شمولاً وتنظيمًا للتطوير. قد تتضمن أهداف المشروع قابلية التوسع والموثوقية والامتثال لمعايير الصناعة، والتي قد لا تعالجها دورات التطوير السريع لـ RAD بشكل كامل.

**قيود:**

قد لا يكون تركيز RAD على السرعة والمرونة مناسبًا للمشاريع ذات المتطلبات التنظيمية الصارمة أو التي تعتمد على الأنظمة القديمة. قد يواجه مشروع اللوجستيات العالمية قيودًا تتعلق بأمن البيانات والامتثال والتكامل مع البنية التحتية الحالية، والتي قد لا يعالجها نهج النماذج الأولية السريعة الخاص بـ RAD بشكل مناسب.

**توقعات أصحاب المصلحة:**

قد يتوقع أصحاب المصلحة في مشروع الخدمات اللوجستية العالمية درجة عالية من الموثوقية والأداء وقابلية التوسع من الحل البرمجي. في حين أن RAD يعزز مشاركة أصحاب المصلحة والتكرار السريع، فقد يعطي أصحاب المصلحة الأولوية لهذه الجوانب الرئيسية على السرعة، خاصة إذا كان المشروع يتضمن وظائف عمل مهمة أو بيانات حساسة.

**نقاط القوة:**

تقدم RAD العديد من نقاط القوة ذات الصلة بشكل خاص بمشروع اللوجستيات العالمية. ويسمح نهجها التكراري بالتطوير السريع وإنشاء النماذج الأولية، مما يمكّن أصحاب المصلحة من رؤية تقدم ملموس في وقت مبكر من دورة حياة المشروع. بالإضافة إلى ذلك، يعمل RAD على تعزيز التعاون بين المطورين والمستخدمين النهائيين، مما يؤدي إلى حلول تلبي احتياجات المستخدم وتفضيلاته بشكل أفضل.

**نقاط الضعف:**

قد يؤدي تركيز RAD على السرعة والمرونة إلى مقايضات من حيث قابلية الصيانة على المدى الطويل وقابلية التوسع وجودة التعليمات البرمجية. قد يتطلب مشروع اللوجستيات العالمية نهجًا أكثر تنظيمًا للتطوير للتأكد من أن الحل البرمجي يلبي متطلبات الأداء، ويتكامل بسلاسة مع الأنظمة الحالية، ويمكنه استيعاب النمو المستقبلي.

**المواءمة مع احتياجات المشروع:**

بشكل عام، قد لا تتوافق منهجية تطوير التطبيقات السريعة (RAD) بشكل كامل مع احتياجات مشروع الخدمات اللوجستية العالمية، خاصة إذا كان المشروع يتطلب درجة عالية من الموثوقية وقابلية التوسع والامتثال. في حين أن RAD يوفر السرعة والمرونة، إلا أنه قد لا يعالج قيود المشروع وتوقعات أصحاب المصلحة بشكل كافٍ. على هذا النحو، قد يحتاج فريق المشروع إلى النظر في منهجيات SDLC البديلة التي تعطي الأولوية للتخطيط الشامل والاختبارات الصارمة والالتزام بمعايير الصناعة.

4. Joint Application Development:

**أهداف المشروع:**

بينما يركز RAD على النماذج الأولية السريعة والتكرار، قد يكون لمشروع اللوجستيات العالمية متطلبات معقدة تتطلب اتباع نهج أكثر شمولاً وتنظيمًا للتطوير. قد تتضمن أهداف المشروع قابلية التوسع والموثوقية والامتثال لمعايير الصناعة، والتي قد لا تعالجها دورات التطوير السريع لـ RAD بشكل كامل.

**قيود:**

قد لا يكون تركيز RAD على السرعة والمرونة مناسبًا للمشاريع ذات المتطلبات التنظيمية الصارمة أو التي تعتمد على الأنظمة القديمة. قد يواجه مشروع اللوجستيات العالمية قيودًا تتعلق بأمن البيانات والامتثال والتكامل مع البنية التحتية الحالية، والتي قد لا يعالجها نهج النماذج الأولية السريعة الخاص بـ RAD بشكل مناسب.

**توقعات أصحاب المصلحة:**

قد يتوقع أصحاب المصلحة في مشروع الخدمات اللوجستية العالمية درجة عالية من الموثوقية والأداء وقابلية التوسع من الحل البرمجي. في حين أن RAD يعزز مشاركة أصحاب المصلحة والتكرار السريع، فقد يعطي أصحاب المصلحة الأولوية لهذه الجوانب الرئيسية على السرعة، خاصة إذا كان المشروع يتضمن وظائف عمل مهمة أو بيانات حساسة.

**نقاط قوة :**

على الرغم من عيوبها المحتملة، فإن منهجية JAD تجلب العديد من نقاط القوة إلى الطاولة. تتفوق جلسات JAD في تعزيز جمع المتطلبات بسرعة، وتعزيز مشاركة أصحاب المصلحة، وتسهيل النماذج الأولية التكرارية. وفي سياق المشروع اللوجستي العالمي، يمكن لنقاط القوة هذه تسريع عملية صنع القرار، وتعزيز التعاون، وتسريع عملية التطوير. من خلال الاستفادة من كفاءة ومرونة JAD، يمكن لفريق المشروع تحسين متطلبات النظام بشكل متكرر، والتحقق من صحة الحلول المقترحة، ومواءمة توقعات أصحاب المصلحة بشكل فعال. وبالتالي، في حين أن JAD قد لا تتوافق بسلاسة مع جميع جوانب متطلبات المشروع، فإن نقاط قوتها توفر فرصًا قيمة لتبسيط جهود التطوير وتقديم حل يلبي الاحتياجات المتطورة لصناعة الخدمات اللوجستية.

**نقاط الضعف:**

قد يؤدي تركيز RAD على السرعة والمرونة إلى مقايضات من حيث قابلية الصيانة على المدى الطويل وقابلية التوسع وجودة التعليمات البرمجية. قد يتطلب مشروع اللوجستيات العالمية نهجًا أكثر تنظيمًا للتطوير للتأكد من أن الحل البرمجي يلبي متطلبات الأداء، ويتكامل بسلاسة مع الأنظمة الحالية، ويمكنه استيعاب النمو المستقبلي.

**المواءمة مع احتياجات المشروع:**

بشكل عام، قد لا تتوافق منهجية تطوير التطبيقات السريعة (RAD) بشكل كامل مع احتياجات مشروع الخدمات اللوجستية العالمية، خاصة إذا كان المشروع يتطلب درجة عالية من الموثوقية وقابلية التوسع والامتثال. في حين أن RAD يوفر السرعة والمرونة، إلا أنه قد لا يعالج قيود المشروع وتوقعات أصحاب المصلحة بشكل كافٍ. على هذا النحو، قد يحتاج فريق المشروع إلى النظر في منهجيات SDLC البديلة التي تعطي الأولوية للتخطيط الشامل والاختبارات الصارمة والالتزام بمعايير الصناعة.

5. Agile:

**أهداف المشروع:**

تتناسب منهجية Agile بطبيعتها مع المشاريع ذات المتطلبات المتطورة وبيئات العمل الديناميكية، مما يجعلها خيارًا مثاليًا لمشروع الخدمات اللوجستية العالمية. من خلال تقسيم المشروع إلى زيادات صغيرة يمكن التحكم فيها تسمى التكرارات أو السباقات السريعة، يسمح Agile بالتكيف المستمر مع الاحتياجات والأولويات المتغيرة. تتيح هذه المرونة لفريق المشروع الاستجابة بسرعة لاتجاهات السوق وتعليقات العملاء والفرص الناشئة في صناعة الخدمات اللوجستية.

**قيود:**

قد تشكل الطبيعة التكرارية لـ Agile تحديات في المشاريع ذات المواعيد النهائية المحددة أو القيود التنظيمية. ومع ذلك، بالنسبة لمشروع الخدمات اللوجستية العالمية، حيث يعتبر الابتكار والاستجابة وقابلية التوسع أمرًا بالغ الأهمية، فإن فوائد نهج Agile التكيفي تفوق هذه القيود. علاوة على ذلك، فإن تركيز Agile على التواصل والتعاون المنتظم يساعد في تخفيف المخاطر المرتبطة بالعمليات اللوجستية المعقدة والتبعيات التكنولوجية.

**توقعات أصحاب المصلحة:**

تعمل منهجية Agile على تعزيز مشاركة أصحاب المصلحة في جميع مراحل عملية التطوير، وتعزيز الشفافية والثقة ومواءمة التوقعات. في مشروع اللوجستيات العالمية، حيث قد يكون لدى أصحاب المصلحة احتياجات وأولويات متنوعة، توفر احتفالات Agile مثل مراجعات السرعة والمواقف اليومية والتحسينات المتراكمة فرصًا للحوار المستمر وتصحيح المسار. وهذا يضمن أن الحل النهائي لا يلبي المتطلبات الوظيفية لأصحاب المصلحة فحسب، بل يعالج أيضًا أهدافهم الإستراتيجية الأوسع.

**نقاط القوة:**

يتيح النهج التكراري لشركة Agile التسليم المبكر والمتكرر لبرامج العمل، مما يسمح لأصحاب المصلحة بالتحقق من صحة الافتراضات واختبار الفرضيات وتصحيح المسار حسب الحاجة. وهذا يقلل من مخاطر إعادة العمل المكلفة ويضمن بقاء المشروع متوافقًا مع أهداف العمل. بالإضافة إلى ذلك، فإن تركيز Agile على التعاون وفرق التنظيم الذاتي والتحسين المستمر يعزز ثقافة الابتكار والقدرة على التكيف، مما يؤدي إلى خلق القيمة والميزة التنافسية في صناعة الخدمات اللوجستية.

**نقاط الضعف:**

يمكن أن يؤدي النهج التكراري لـ Agile، مع تعزيز القدرة على التكيف والتعاون، إلى زحف النطاق إذا لم يتم إدارته بعناية، مما قد يتسبب في تأخيرات أو مشكلات في الميزانية. قد يكون الحفاظ على مشاركة أصحاب المصلحة والتزامهم عبر الفرق المتفرقة جغرافيًا أو المنظمات الهرمية أمرًا صعبًا. بالإضافة إلى ذلك، قد تواجه فرق التنظيم الذاتي عقبات التنسيق، خاصة في المشاريع ذات التبعيات المعقدة. قد يتطلب اعتماد أجايل على التخطيط التكيفي تحولاً ثقافياً، وربما يواجه مقاومة من أصحاب المصلحة التقليديين. وعلى الرغم من هذه التحديات، فإن التخطيط الدقيق والتواصل وإدارة أصحاب المصلحة يمكن أن يساعد في تخفيف نقاط الضعف هذه وضمان نجاح المشروع.

**المواءمة مع احتياجات المشروع:**

باختصار، توفر منهجية Agile إطارًا مناسبًا تمامًا لمشروع اللوجستيات العالمية، مما يمكّن فريق المشروع من تبني التغيير وتقديم القيمة بشكل تدريجي والاستجابة بفعالية لمتطلبات السوق المتطورة. من خلال إعطاء الأولوية للتعاون مع العملاء والمرونة والتحسين المستمر، تضمن Agile أن الحل النهائي يلبي توقعات أصحاب المصلحة، ويدفع نمو الأعمال، ويقدم فوائد ملموسة لصناعة الخدمات اللوجستية.

6. Prototyping:

**أهداف المشروع:**

قد لا تتماشى منهجية النماذج الأولية بسلاسة مع أهداف مشروع اللوجستيات العالمية بسبب تركيزها على التكرارات السريعة والعروض الملموسة للوظائف. وبينما يهدف المشروع إلى تحديث العمليات اللوجستية، فإن الطبيعة التكرارية للنماذج الأولية قد تؤدي إلى عدم اليقين والتأخير المحتمل في تقديم حل شامل. قد يؤدي التركيز على التحقق المبكر وإشراك أصحاب المصلحة إلى زحف النطاق وتباين الأولويات، مما يعيق قدرة المشروع على تحقيق أهدافه ضمن الإطار الزمني المطلوب.

**قيود:**

قد يشكل النهج التكراري للنماذج الأولية تحديات كبيرة في المشاريع ذات الجداول الزمنية الثابتة أو القيود التنظيمية، مثل مشروع اللوجستيات العالمية. يمكن أن يؤدي الافتقار إلى التخطيط المسبق والتوثيق المتأصل في النماذج الأولية إلى صعوبات في إدارة نطاق المشروع والجداول الزمنية والميزانيات بشكل فعال. علاوة على ذلك، فإن الحاجة إلى مشاركة والتزام أصحاب المصلحة بشكل متكرر قد تؤدي إلى إجهاد الموارد وإعاقة التقدم، لا سيما في المشاريع واسعة النطاق التي تضم مجموعات متنوعة من أصحاب المصلحة ومصالح متنافسة.

**توقعات أصحاب المصلحة:**

قد لا تلبي منهجية النماذج الأولية توقعات أصحاب المصلحة بشكل كامل في سياق مشروع اللوجستيات العالمية، حيث قد يعطي أصحاب المصلحة الأولوية للوضوح والقدرة على التنبؤ وتخفيف المخاطر على التكرار والتجريب السريع. في حين أن النماذج الأولية تشجع المشاركة النشطة وردود الفعل من أصحاب المصلحة، فإن الطبيعة التكرارية للعملية قد تؤدي إلى عدم اليقين والغموض فيما يتعلق بنتائج المشروع والجداول الزمنية. وقد يؤدي ذلك إلى تآكل ثقة أصحاب المصلحة وإعاقة عملية صنع القرار، مما قد يؤثر على نجاح المشروع.

**نقاط القوة:**

تسهل النماذج الأولية الاستكشاف السريع لمفاهيم التصميم ووظائفه، مما يمكّن فريق المشروع من تحديد المشكلات والفرص المحتملة في وقت مبكر من دورة التطوير. من خلال إنشاء نماذج أولية ملموسة يمكن لأصحاب المصلحة التفاعل معها، تساعد النماذج الأولية على سد الفجوة بين المتطلبات والتنفيذ، مما يقلل من مخاطر سوء الفهم أو التفسيرات الخاطئة. بالإضافة إلى ذلك، تعزز النماذج الأولية ثقافة الابتكار والتجريب، وتشجع أصحاب المصلحة على استكشاف أفكار وإمكانيات جديدة لتحسين العمليات اللوجستية.

**نقاط الضعف:**

على الرغم من فوائدها، فإن منهجية النماذج الأولية تنطوي على العديد من نقاط الضعف التي قد تعيق مواءمتها مع احتياجات مشروع اللوجستيات العالمية. قد تؤدي الطبيعة التكرارية للنماذج الأولية إلى متطلبات غير متسقة أو غير مكتملة، مما يؤدي إلى سوء الفهم وإعادة العمل لاحقًا في عملية التطوير. بالإضافة إلى ذلك، فإن الافتقار إلى الوثائق الرسمية والتخطيط المسبق قد يجعل من الصعب إدارة مخاطر المشروع وتبعياته بفعالية. علاوة على ذلك، قد لا تكون النماذج الأولية مناسبة للمشاريع ذات المتطلبات المعقدة للغاية أو ذات المهام الحرجة، حيث يكون التخطيط المسبق الشامل والتحقق من الصحة ضروريين لضمان موثوقية النظام وأدائه.

**المواءمة مع احتياجات المشروع:**

باختصار، في حين أن منهجية النماذج الأولية توفر فوائد من حيث التحقق المبكر وإشراك أصحاب المصلحة، فإن القيود والقيود المتأصلة فيها قد تعيق توافقها مع أهداف مشروع اللوجستيات العالمية. قد يحتاج فريق المشروع إلى النظر بعناية في المقايضات المرتبطة بالنماذج الأولية واستكشاف منهجيات SDLC البديلة التي تقدم نهجًا أكثر تنظيماً لجمع المتطلبات والتخطيط والتنفيذ. سيضمن ذلك أن الحل النهائي يلبي توقعات أصحاب المصلحة والمتطلبات التنظيمية وأهداف العمل بشكل فعال، مع تقليل مخاطر تأخير المشروع وتجاوز التكاليف.

**استنادًا إلى تقييم شامل لمختلف منهجيات دورة حياة تطوير البرمجيات (SDLC)، تبرز Agile باعتبارها النهج الأكثر ملاءمة لمشروع الخدمات اللوجستية العالمية. توفر منهجية Agile إطارًا ديناميكيًا يتوافق جيدًا مع أهداف المشروع وقيوده وتوقعات أصحاب المصلحة.**

**نقاط القوة:**

يمكّن النهج التكراري الذي تتبعه Agile فريق المشروع من تقديم القيمة بشكل تدريجي، مما يسمح لأصحاب المصلحة بالتحقق من صحة الافتراضات واختبار الفرضيات وتقديم التعليقات في وقت مبكر من عملية التطوير. تقلل حلقة الملاحظات التكرارية هذه من مخاطر إعادة العمل المكلفة وتضمن أن الحل النهائي يلبي احتياجات العمل المتطورة. علاوة على ذلك، تعمل Agile على تعزيز التعاون والشفافية، وتمكين الفرق متعددة الوظائف من التنظيم الذاتي والتكيف مع المتطلبات المتغيرة بكفاءة. من خلال تعزيز التحسين المستمر والتعاون مع العملاء، تعمل Agile على تحفيز الابتكار وتعزيز النجاح الشامل للمشروع.

**المواءمة مع احتياجات المشروع:**

يتطلب مشروع اللوجستيات العالمية نهجًا تطويريًا يمكنه استيعاب المتطلبات المتطورة وديناميكيات السوق والتقدم التكنولوجي. إن الطبيعة التكرارية لـ Agile والتركيز على المرونة تجعلها مناسبة تمامًا لتلبية هذه الاحتياجات. ومن خلال تقسيم المشروع إلى زيادات صغيرة يمكن التحكم فيها، يمكّن Agile فريق المشروع من الاستجابة بسرعة للأولويات المتغيرة ومتطلبات السوق. تعتبر هذه القدرة على التكيف ذات قيمة خاصة في صناعة الخدمات اللوجستية، حيث قد تتطور متطلبات العمل بسرعة بسبب عوامل مثل تغيير تفضيلات العملاء، والتحديثات التنظيمية، والضغوط التنافسية.

**قيود:**

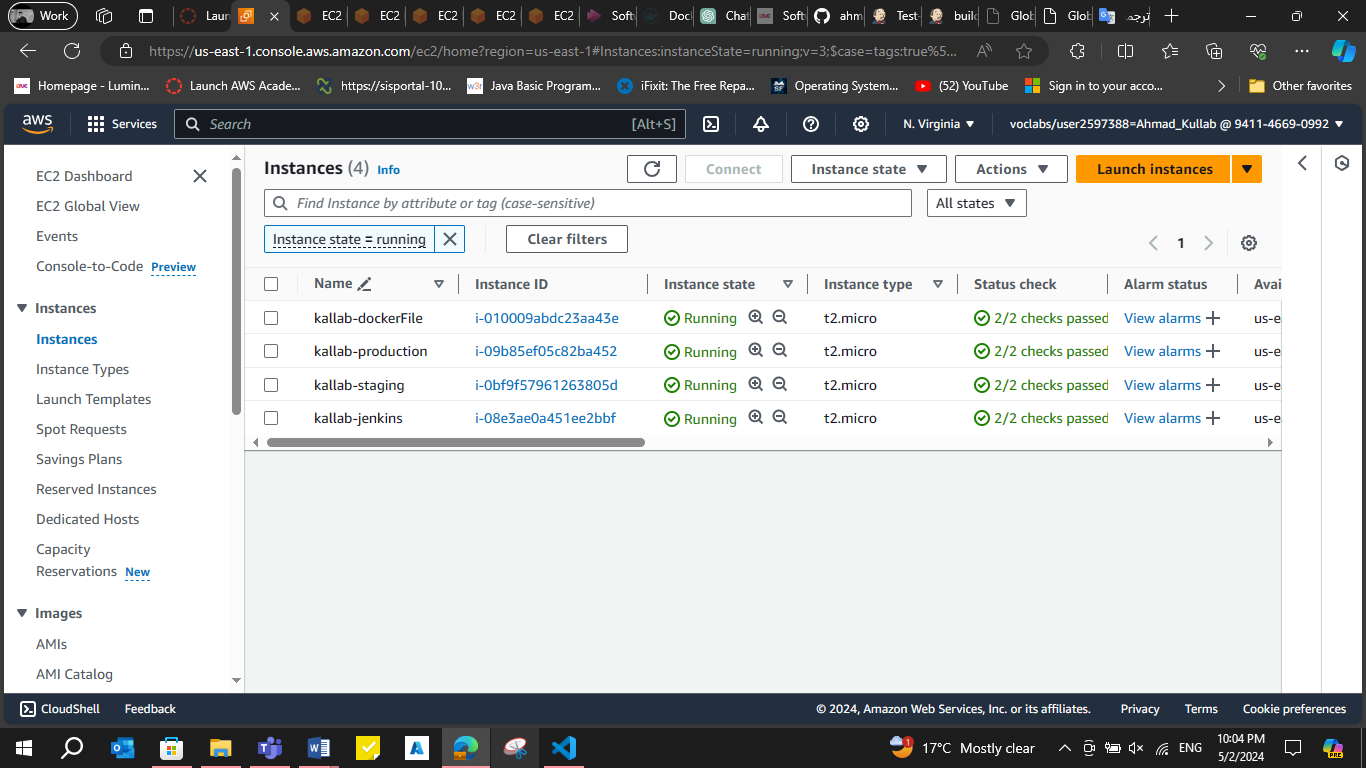
في حين أن Agile يقدم فوائد عديدة، فإنه يقدم أيضًا تحديات يجب معالجتها لضمان نجاح المشروع. أحد العوائق المحتملة هو الحاجة إلى إدارة قوية للمشروع وإشراك أصحاب المصلحة للحفاظ على المواءمة وتحديد أولويات العمل بشكل فعال. بالإضافة إلى ذلك، تتطلب مشاريع Agile مستوى عالٍ من التعاون والتواصل، الأمر الذي قد يكون من الصعب تحقيقه في فرق موزعة أو بعيدة. علاوة على ذلك، قد يؤدي نهج Agile التكراري إلى زحف النطاق إذا لم يتم إدارته بعناية، مما يتطلب مراقبة يقظة وتعديل نطاق المشروع والجداول الزمنية.

**التوصيات:**

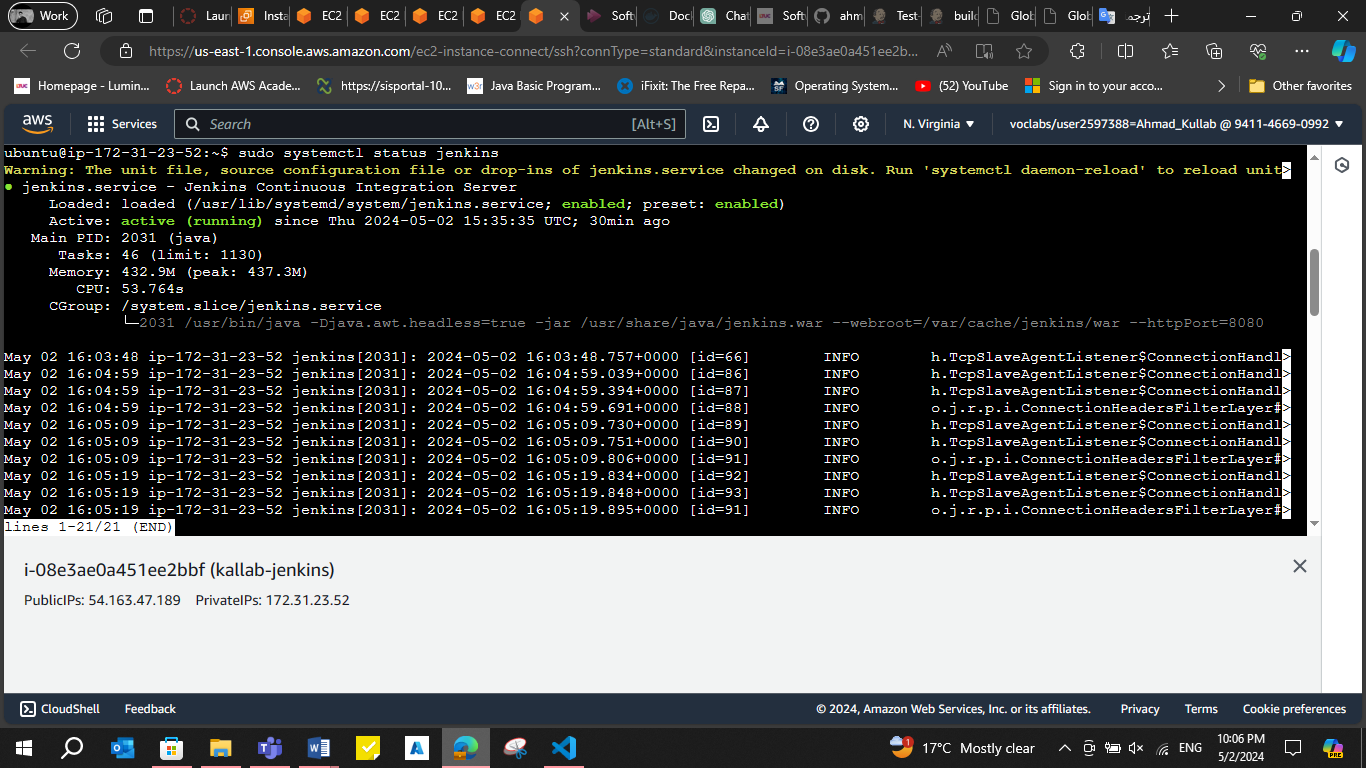
وبالنظر إلى الطبيعة الديناميكية للمشروع، والمتطلبات المتطورة، والحاجة إلى التكيف السريع، تبرز Agile باعتبارها منهجية SDLC الأكثر ملاءمة لمشروع اللوجستيات العالمية. من خلال تبني مبادئ وممارسات Agile، يمكن لفريق المشروع الاستفادة من نقاط قوته في التعاون والمرونة والتحسين المستمر لتقديم حل يلبي توقعات أصحاب المصلحة ويحفز قيمة الأعمال في صناعة الخدمات اللوجستية. ومع ذلك، فمن الضروري معالجة القيود المحتملة بشكل استباقي وتنفيذ ممارسات قوية لإدارة المشاريع لضمان التنفيذ الناجح لمنهجية Agile.

Task 2.A

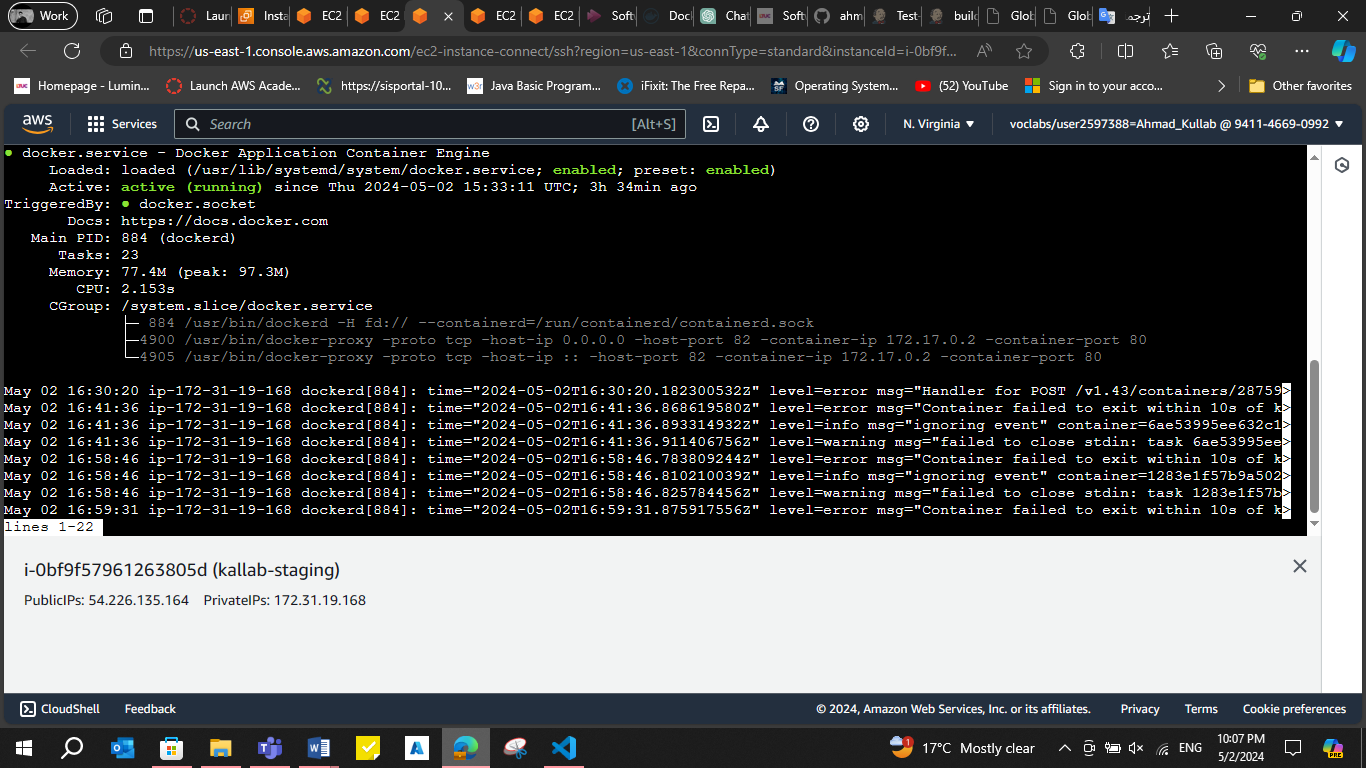
1. إعداد مثيلات AWS EC2: في البداية، قمت بتسجيل الدخول إلى وحدة الإدارة في AWS وانتقلت إلى لوحة معلوما EC2. هناك، بدأت في إنشاء مثيلات EC2 جديدة، وتخصيص مثيلات محددة للإنتاج والتشغيل المرحلي وجينكينز. لقد تأكدت من توافق مواصفات كل مثيل مع متطلبات مشروعي، واختيار أنواع المثيلات وأنظمة التشغيل والتكوينات المناسبة وفقًا لذلك.

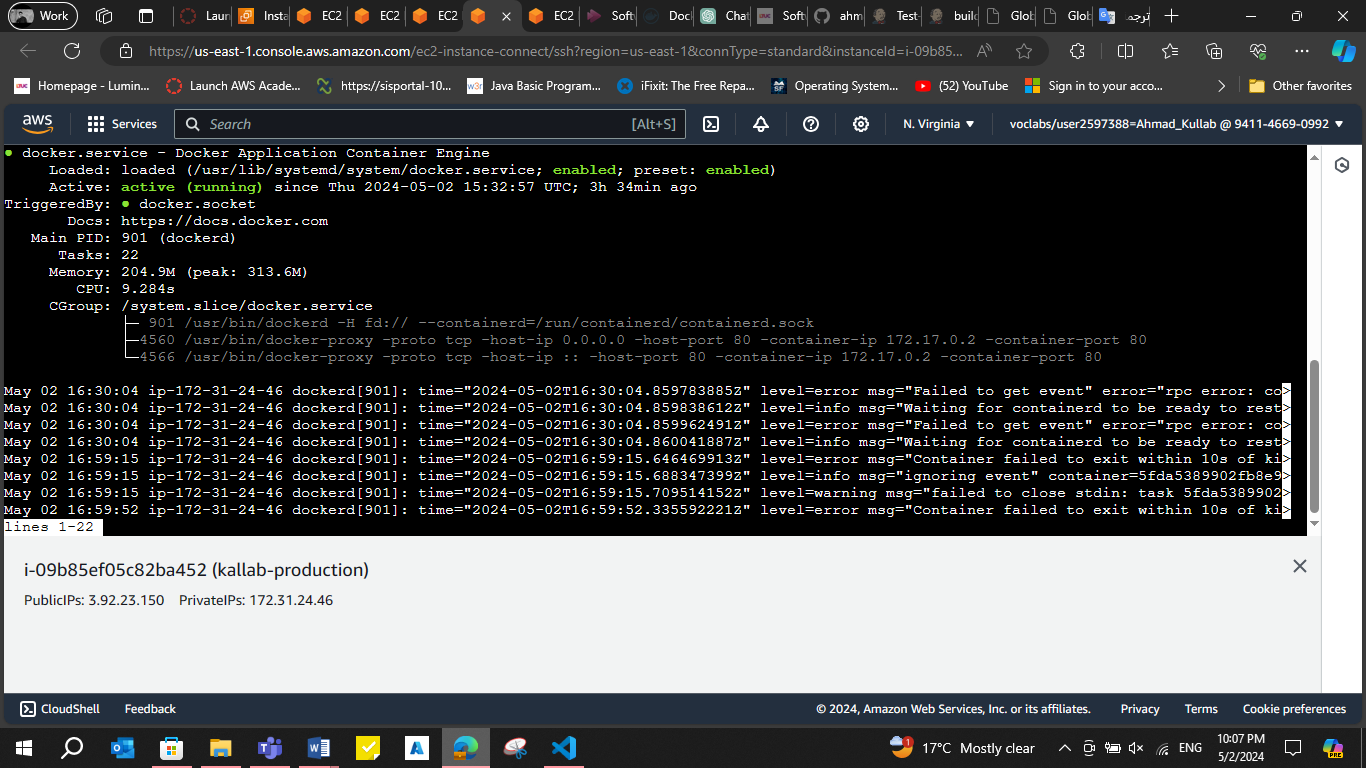


2. تثبيت Jenkins على مثيل :Jenkins EC2بمجرد توفر مثيل Jenkins EC2، قمت بإنشاء اتصال SSH به. بعد ذلك، قمت بتثبيت Jenkins على المثيل. تم تحقيق ذلك إما من خلال استخدام مدير حزم نظام التشغيل أو عن طريق تنزيل ملف Jenkins WAR وتشغيله.

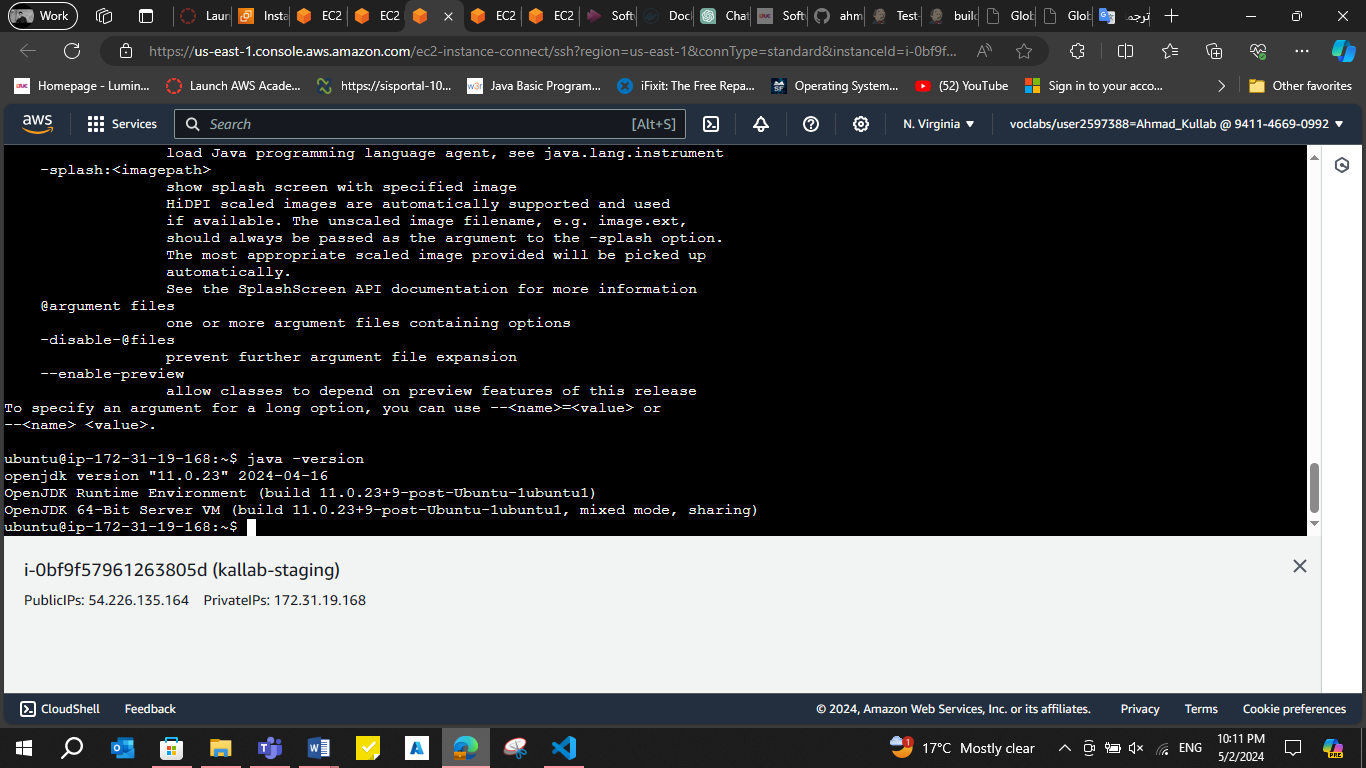


3. تثبيت Docker على مثيلات EC2 للإنتاج والتجهيز: وبالمثل، قمت بالوصول إلى مثيلات EC2 للإنتاج والتجهيز عبر SSH وشرعت في تثبيت Docker على كل مثيل. تتضمن عملية التثبيت هذه استخدام مدير الحزم الذي يوفره نظام التشغيل أو تنفيذ البرنامج النصي لتثبيت Docker الذي تم الحصول عليه من موقع Docker الرسمي.



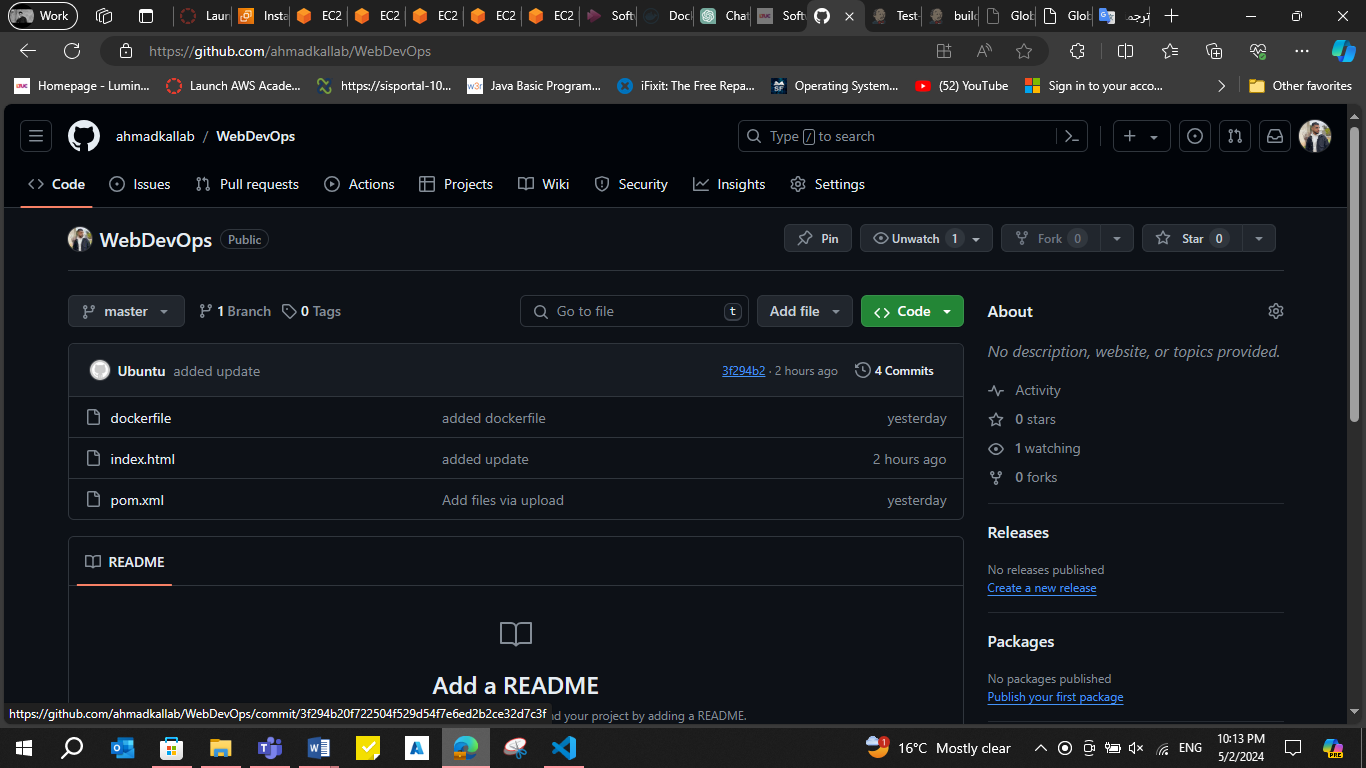


4. تثبيت JDK على جميع مثيلات : EC2 لضمان التوافق مع التطبيقات المستندة إلى Java مثل Jenkins، قمت بتثبيت Java Development Kit (JDK) على جميع مثيلات EC2. كانت هذه الخطوة حاسمة لتسهيل التشغيل السلس لـ Jenkins عبر الحالات. لقد أكملت هذا التثبيت عن طريق SSHing في كل مثيل EC2 وتثبيت JDK باستخدام مدير الحزم المعني.

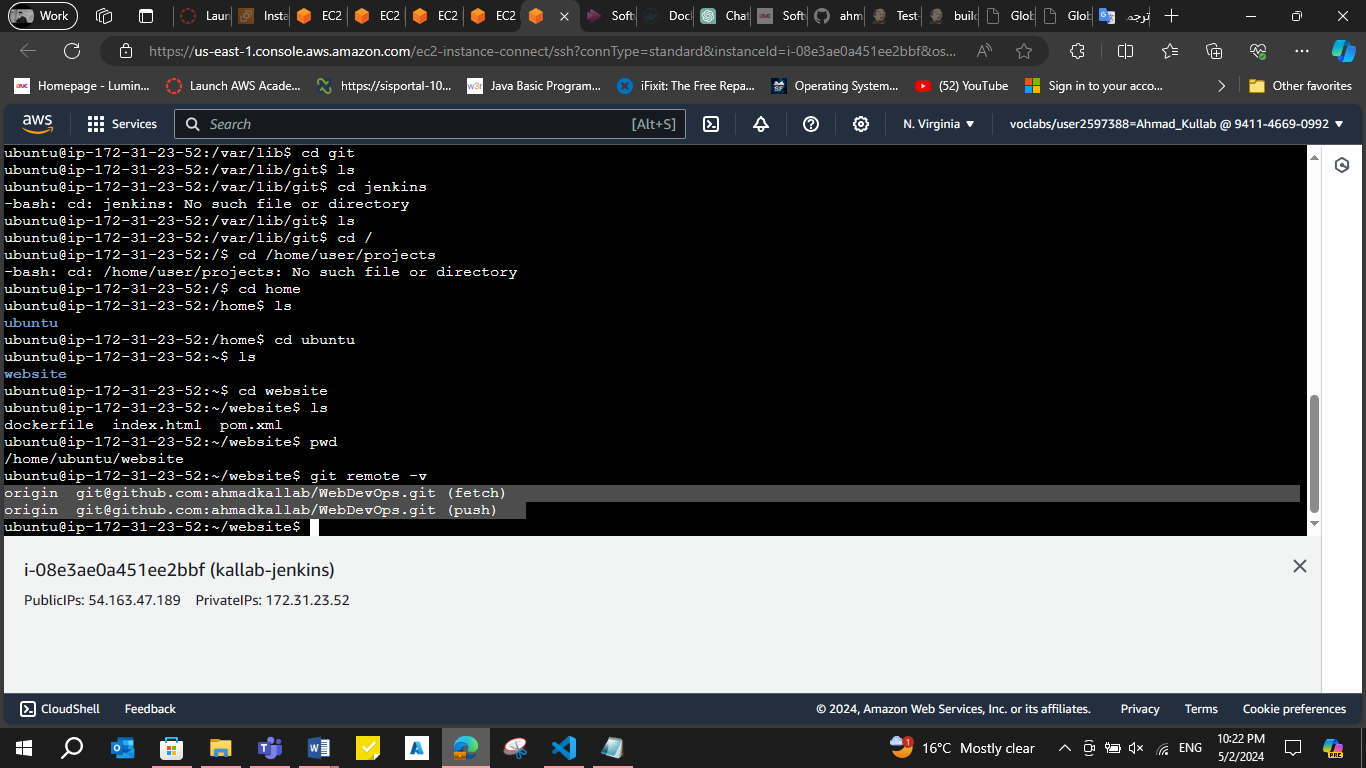




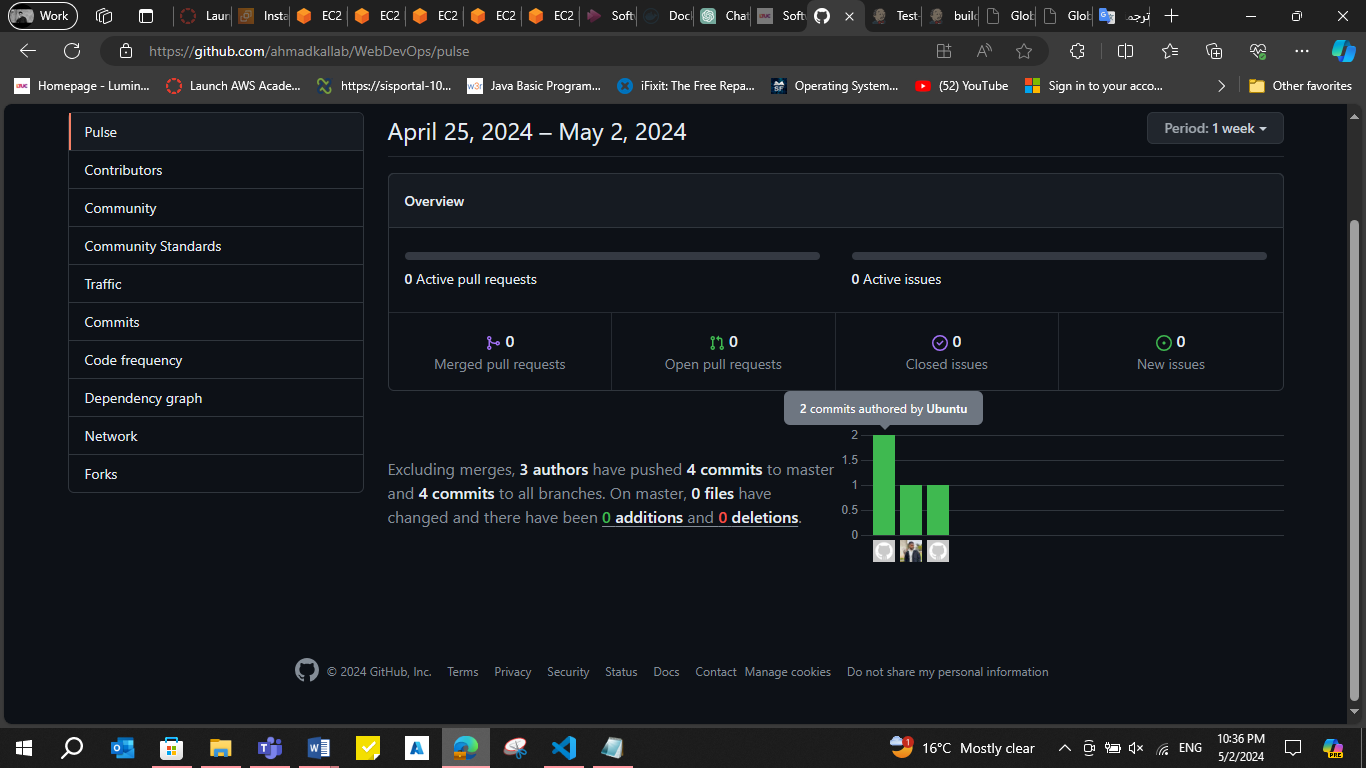
5. إنشاء مستودع على :GitHub بالانتقال إلى GitHub، قمت بتسجيل الدخول إلى حسابي وانتقلت إلى قسم المستودعات. هناك، بدأت في إنشاء مستودع جديد باسم WebDevOps. لقد قدمت التفاصيل الأساسية مثل اسم المستودع والوصف وإعدادات الرؤية لإنشاء المستودع لإدارة قاعدة التعليمات البرمجية لمشروعي.



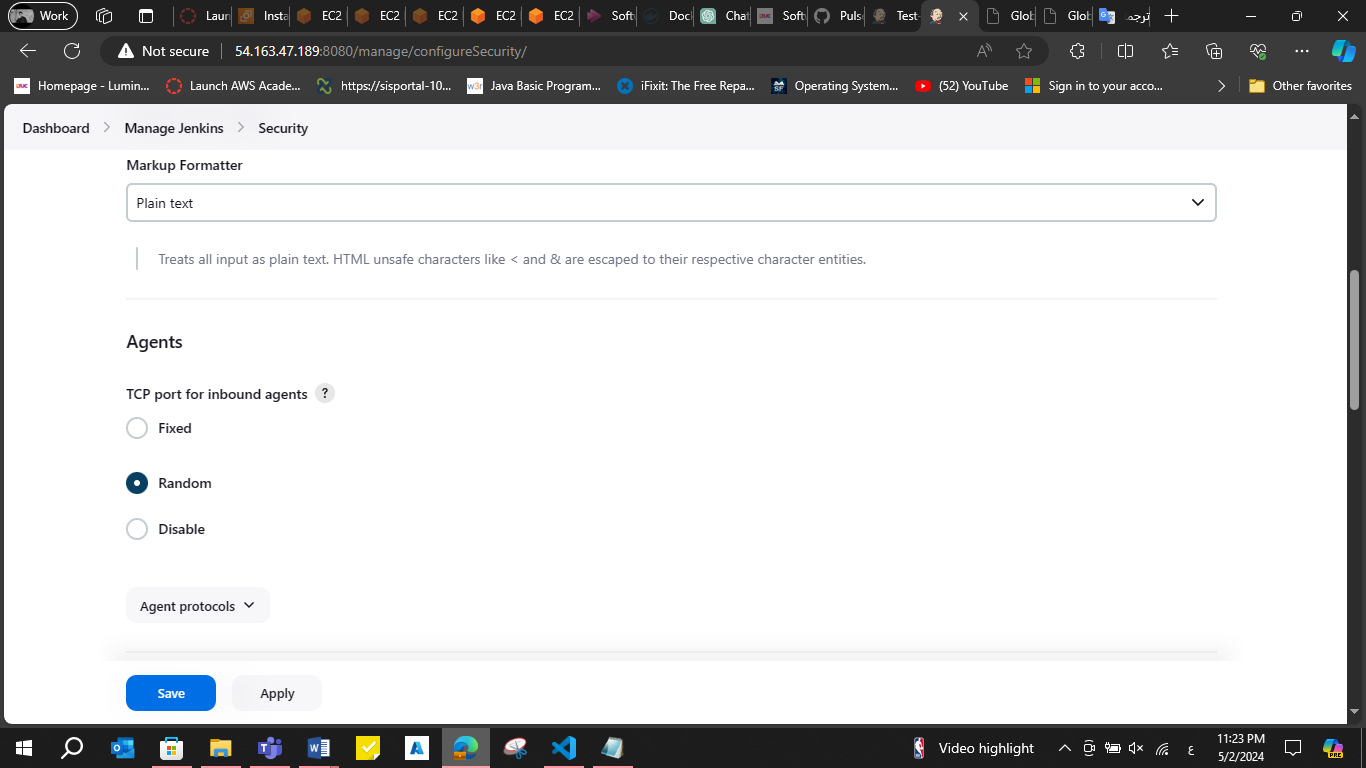
6. ربط مستودع GitHub بـ :Jenkins ضمن لوحة معلومات Jenkins، قمت بتكوين Jenkins لإنشاء اتصال مع مستودع GitHub المنشأ حديثًا. سهّل هذا الاتصال استرجاع التعليمات البرمجية من المستودع، مما مكّن Jenkins من أتمتة عمليات البناء والنشر المختلفة بناءً على قاعدة التعليمات البرمجية المخزنة على GitHub .



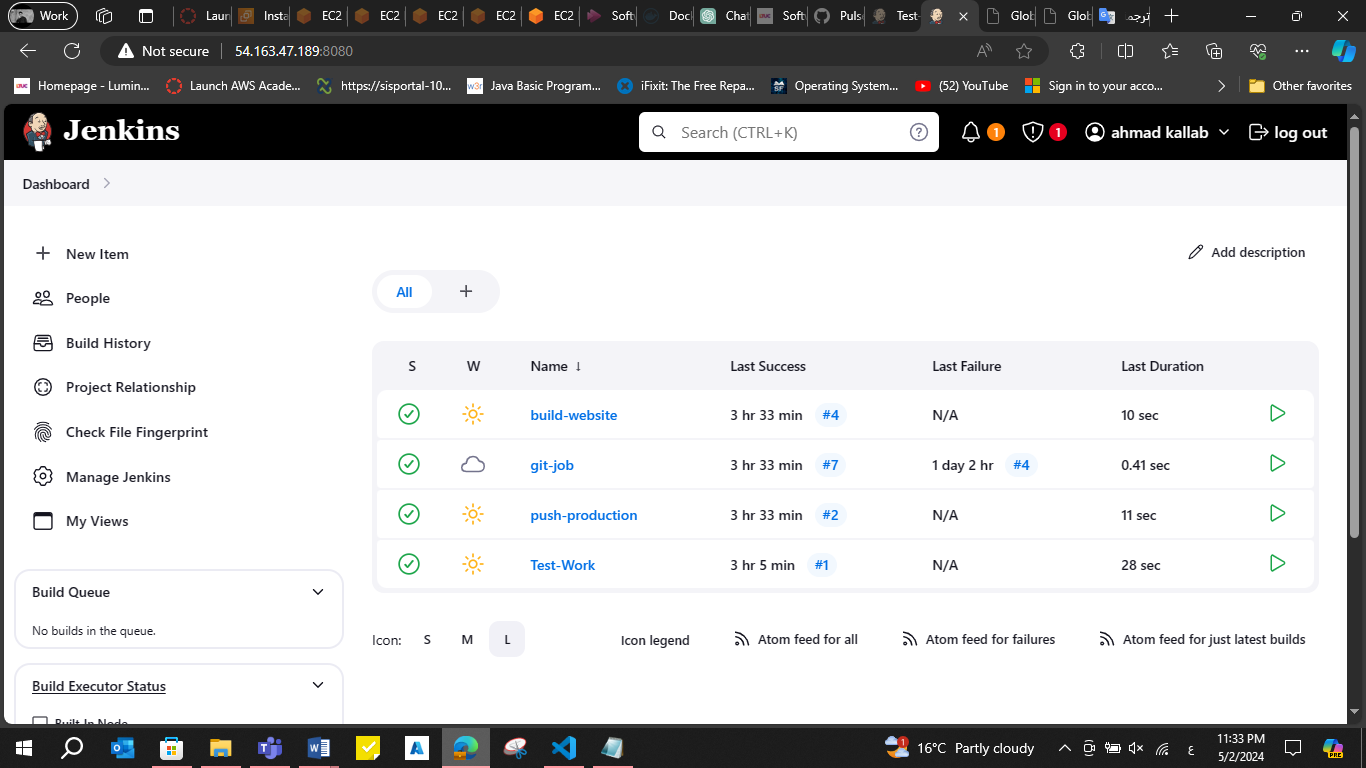
7. دفع الملفات إلى مستودع GitHub: لملء مستودع GitHub بملفات المشروع، انتقلت إلى الدليل الذي يحتوي على ملف Index.html على مثيل Jenkins EC2. باستخدام أوامر Git مثل git add وgit Commit وgit Push، قمت بإضافة ملف Index.html وإلزامه ودفعه إلى مستودع GitHub.



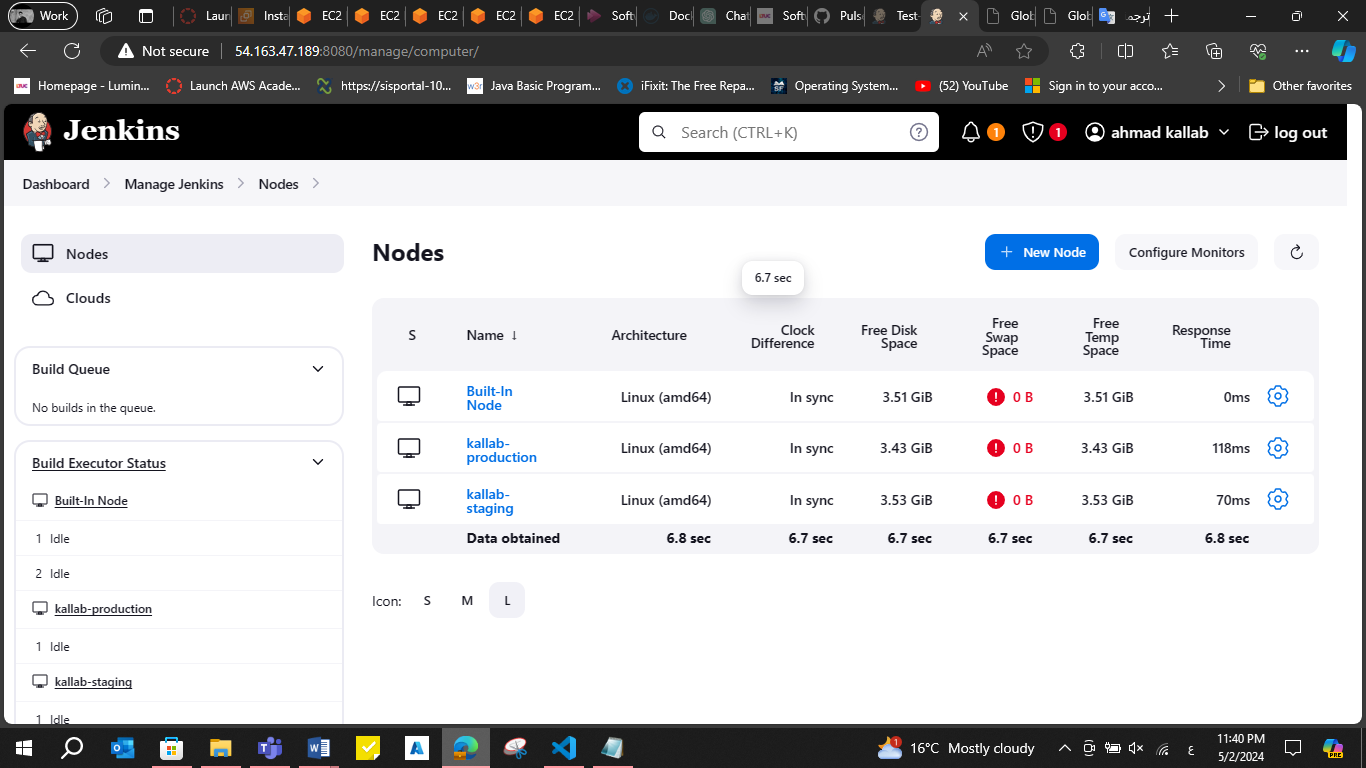
8. تكوين إعدادات الأمان في Jenkins: عند الوصول إلى لوحة معلومات Jenkins، انتقلت إلى قسم "إدارة “Jenkins" لتكوين إعدادات الأمان العامة. هنا، قمت بتخصيص إعدادات الأمان للسماح بالاتصال الآمن بين Jenkins ومثيلات .AWS EC2يضمن ذلك الاتصال الآمن والتحكم في الوصول بين Jenkins ومثيلات EC2 .



9. إنشاء العقد في Jenkins: لتحسين توزيع عبء العمل، قمت بإنشاء عقدتين متميزتين داخل Jenkins، وقمت بتعيينهما على أنهما (التجهيز والإنتاج). تم بعد ذلك ربط هذه العقد بمثيلات EC2 المقابلة لها. وقد مكّن هذا الارتباط جينكينز من توزيع عبء العمل وتنفيذ المهام بشكل فعال عبر العقد المعينة، مما أدى إلى تبسيط سير العمل التشغيلي للمشروع.

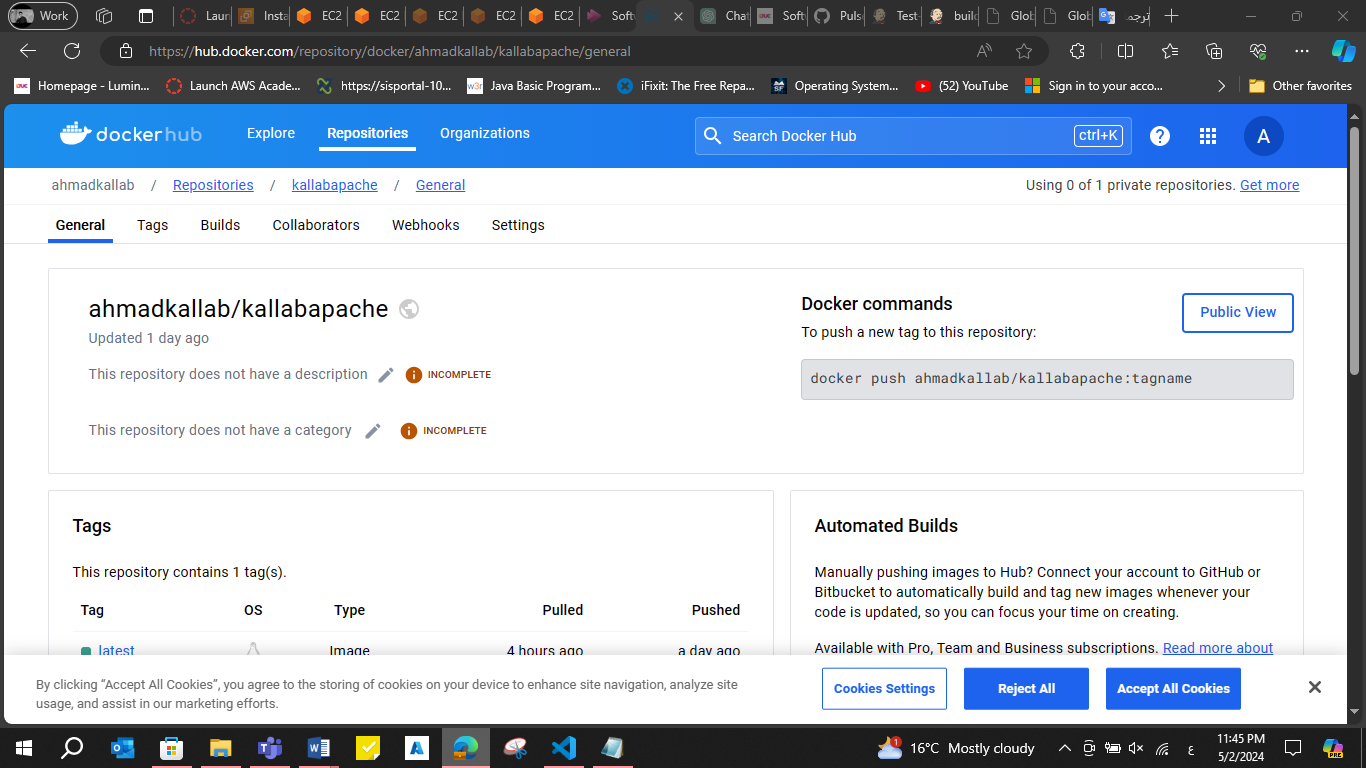


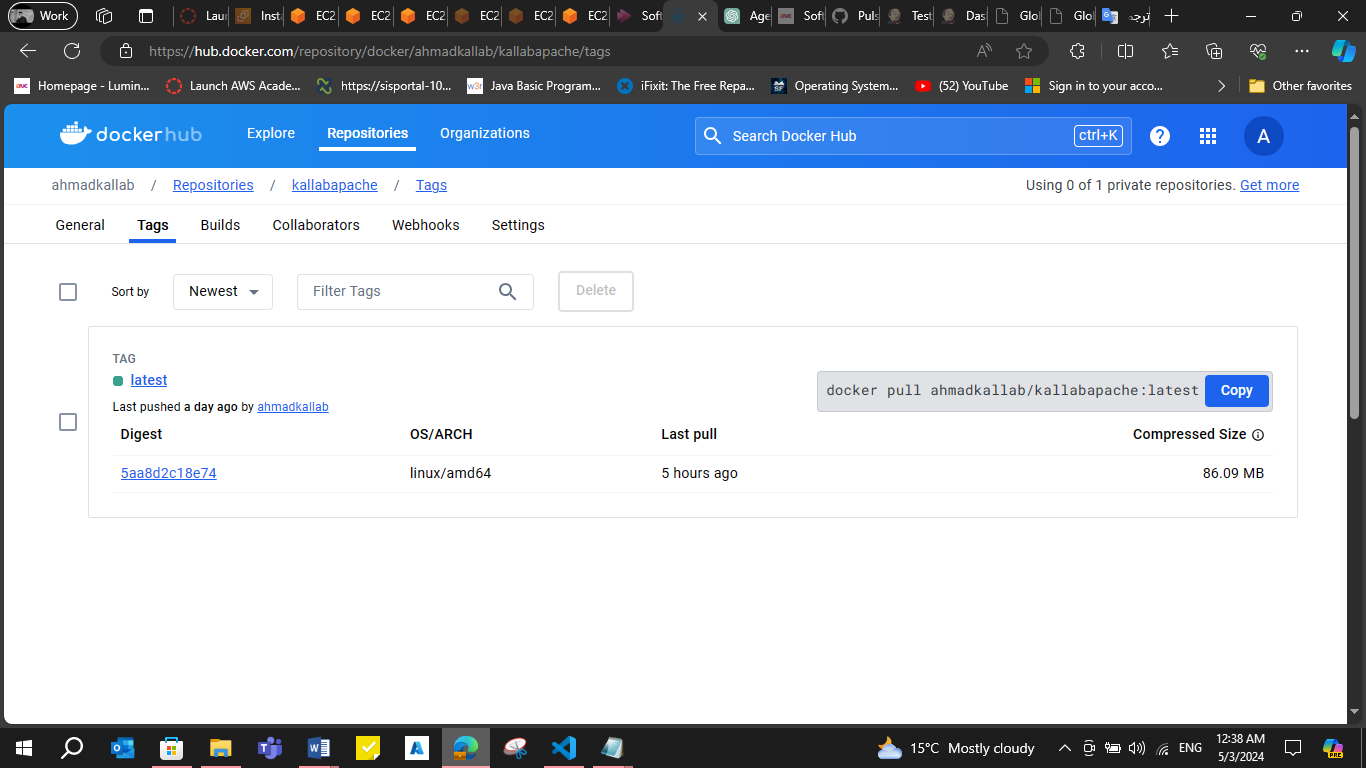
9. إنشاء العقد في Jenkins:لتحسين توزيع عبء العمل، قمت بإنشاء عقدتين متميزتين داخل Jenkins، وقمت بتعيينهما كمرحلتين وإنتاج. تم بعد ذلك ربط هذه العقد بمثيلات EC2 المقابلة لها. وقد مكّن هذا الارتباط جينكينز من توزيع عبء العمل وتنفيذ المهام بشكل فعال عبر العقد المعينة، مما أدى إلى تبسيط سير العمل التشغيلي للمشروع.



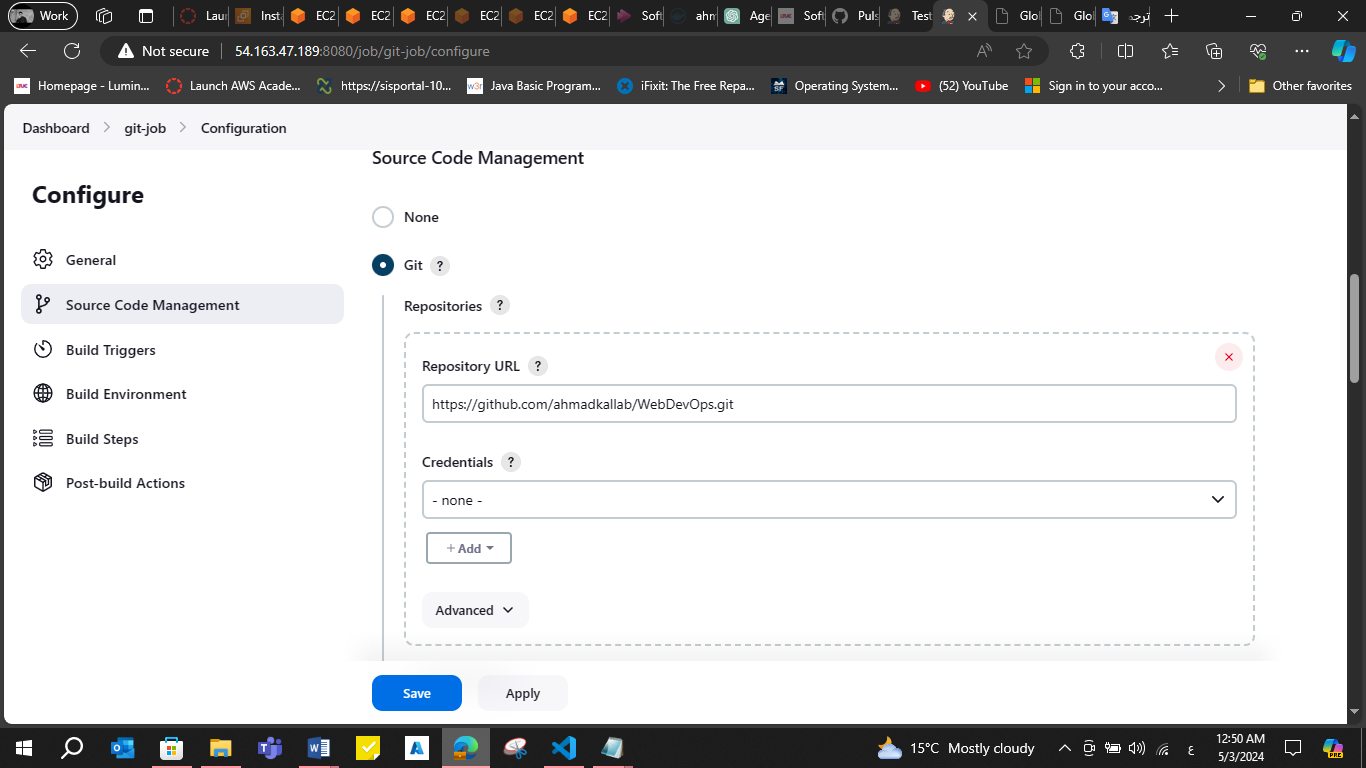
10. إضافة مستودع على Docker Hub: بالانتقال إلى Docker Hub، قمت بتسجيل الدخول وشرعت في إنشاء مستودع جديد لتخزين صور Docker لقد قدمت التفاصيل الأساسية مثل اسم المستودع والوصف وإعدادات الرؤية، وإنشاء مستودع مخصص لإدارة صور Docker المرتبطة بمشروعي.

ربط مستودع Docker Hub بمثيل : EC2بعد إنشاء المستودع على Docker Hub، قمت بإنشاء اتصال بين مستودع Docker Hub ومثيل EC2 المسمى " " dockerfile الذي قمت بإعداده. سمح هذا الارتباط بالتكامل السلس بين مستودع صور Docker ومثيل EC2 المعني، مما يسهل الإدارة الفعالة لصور Docker ونشرها.

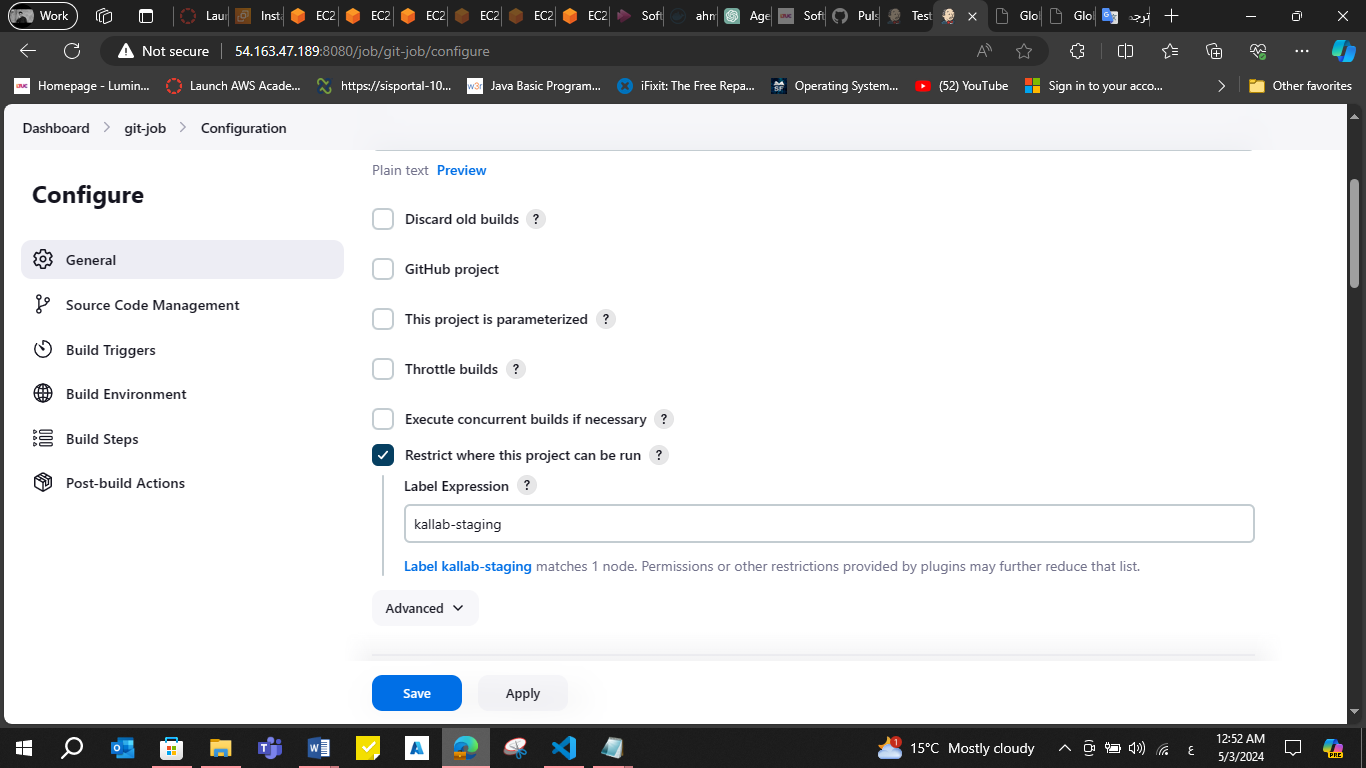




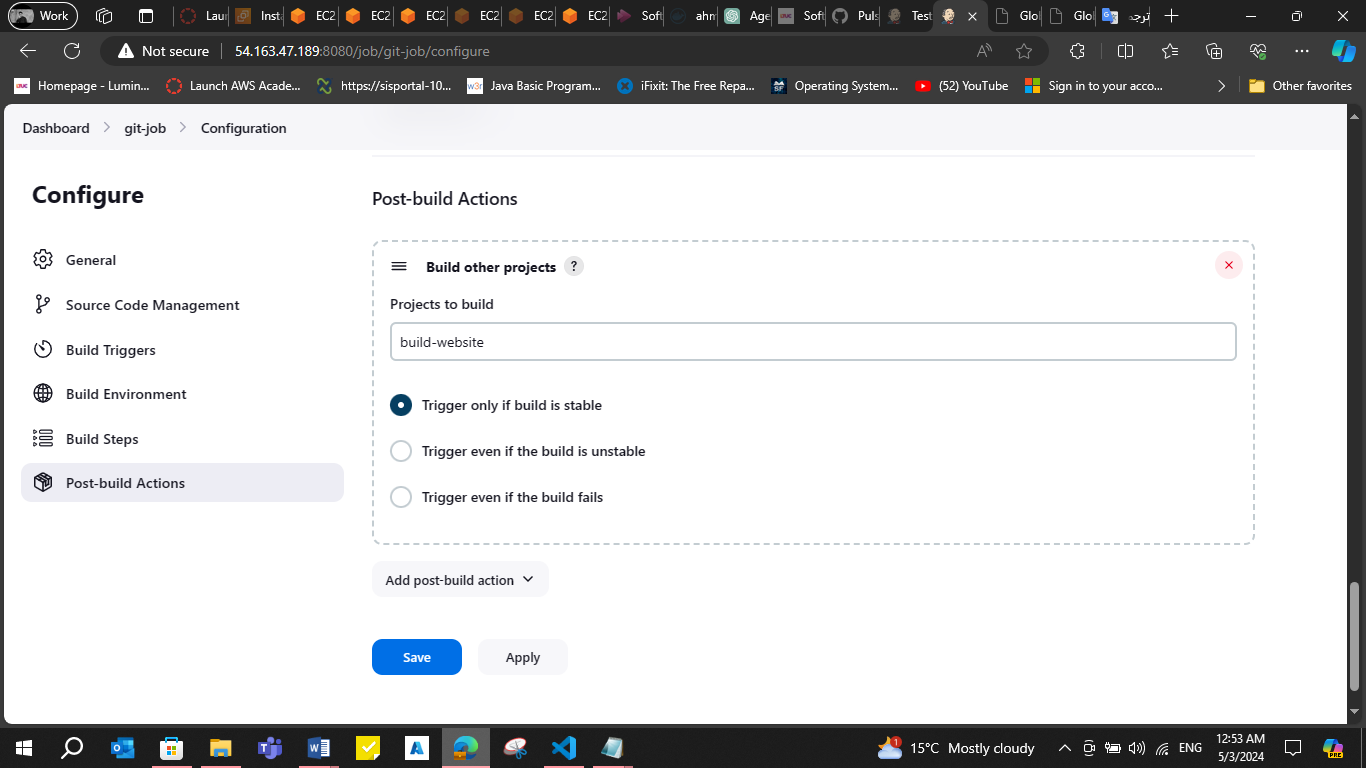
13. تكوين وظيفة Jenkins للبناء والنشر الآلي: بالعودة إلى لوحة معلومات Jenkins، بدأت في إنشاء وظيفة جديدة باسم "" git-job لأتمتة عملية الإنشاء والنشر. لقد قمت بتكوين المهمة لجلب التعليمات البرمجية من مستودع GitHub وتنفيذ خطوات البناء، مثل إنشاء صور Docker ونشرها في البيئة المناسبة.



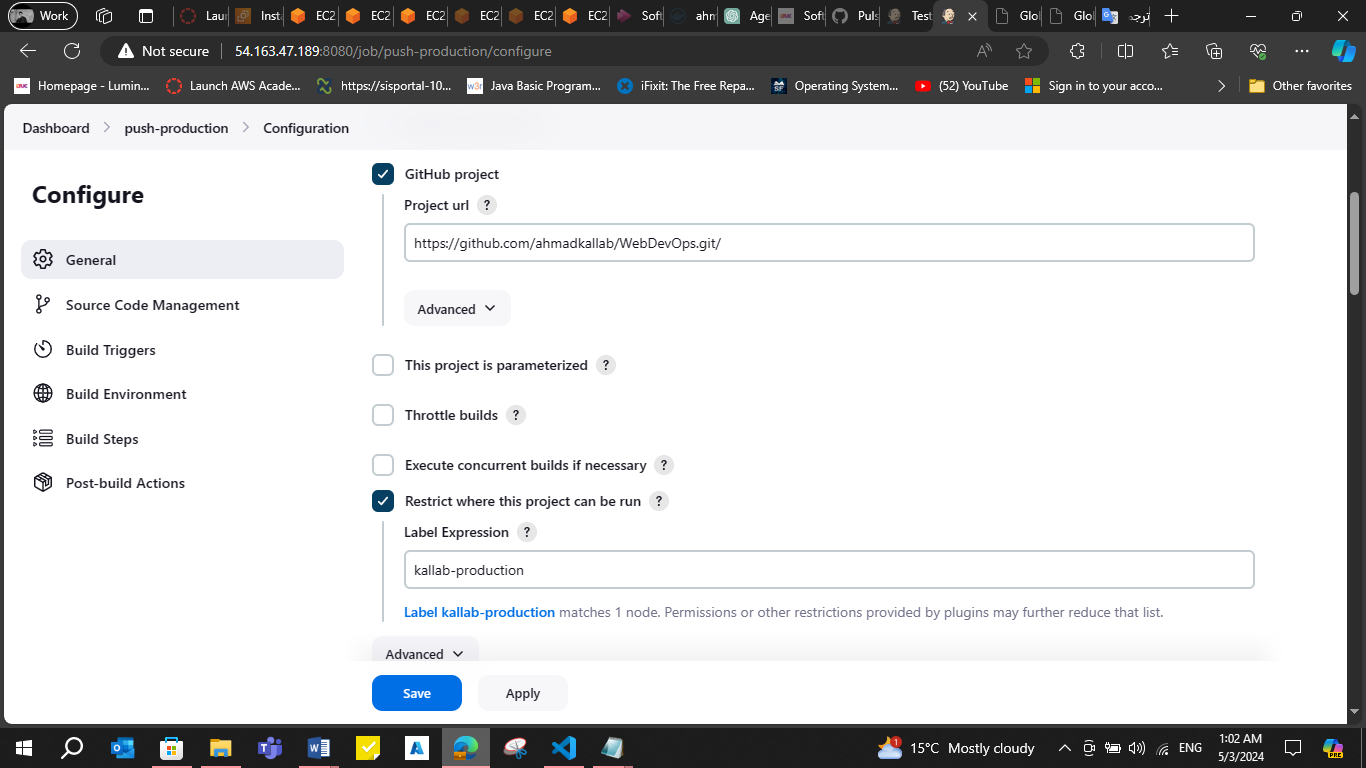
14.إعداد العقد والتسميات في Jenkins: لتحسين استخدام الموارد، تأكدت من إعداد العقد لبيئات التجهيز والإنتاج داخل Jenkins. بالإضافة إلى ذلك، قمت بتسمية هذه العقد وفقًا لذلك لتقييد تنفيذ المهمة بناءً على البيئة المحددة. يضمن هذا التكوين أن يتم تنفيذ المهام المتعلقة بالتشغيل المرحلي فقط على العقد المرحلية والعكس، مما يعزز كفاءة سير العمل وتخصيص الموارد.



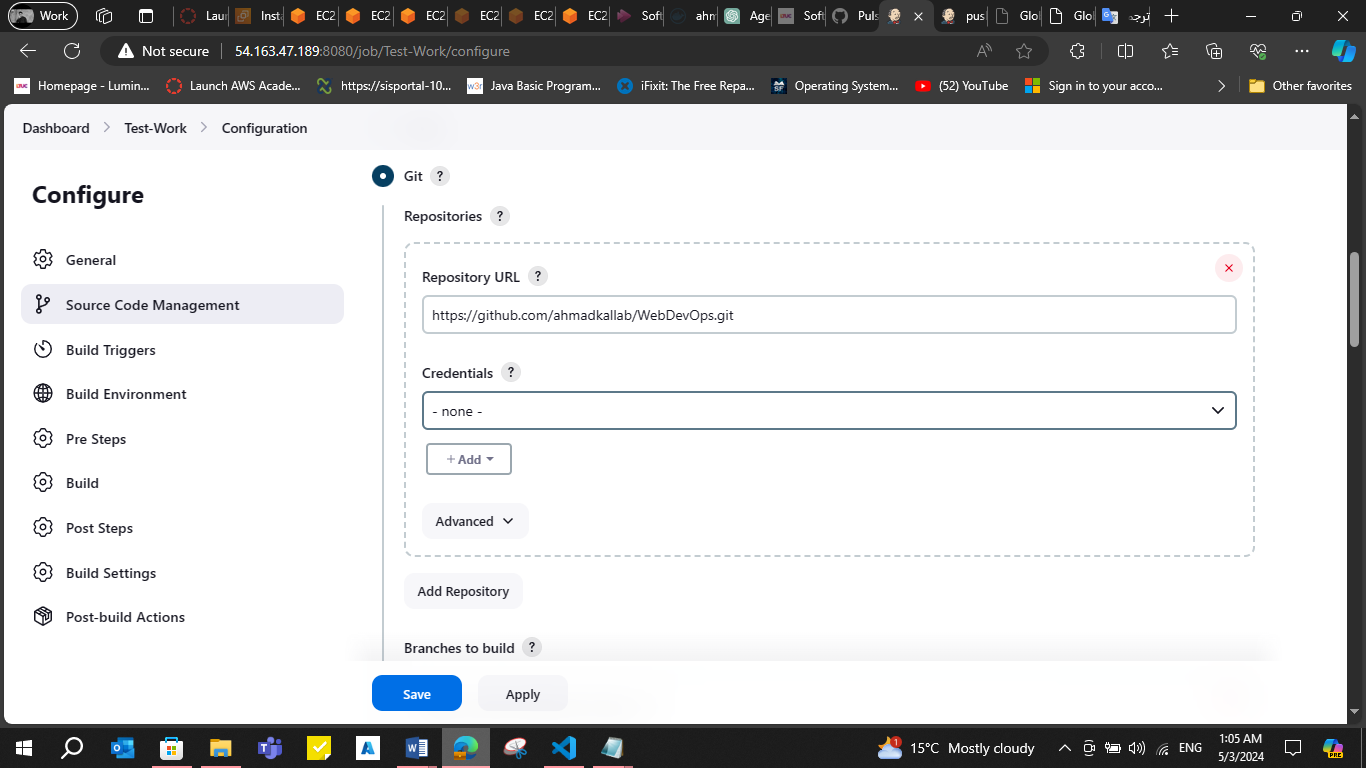
15. إضافة إجراءات ما بعد الإنشاء لنشر موقع الويب: ضمن تكوين وظيفة Jenkins، قمت بتضمين إجراء ما بعد الإنشاء المسمى " " build-websiteلأتمتة نشر موقع الويب بعد عملية الإنشاء الرئيسية. أدى هذا الإجراء إلى تبسيط عملية إنشاء أو تحديث محتوى موقع الويب بعد الإنشاء، مما يضمن التوافق مع أحدث تغييرات التعليمات البرمجية وتعزيز كفاءة النشر.

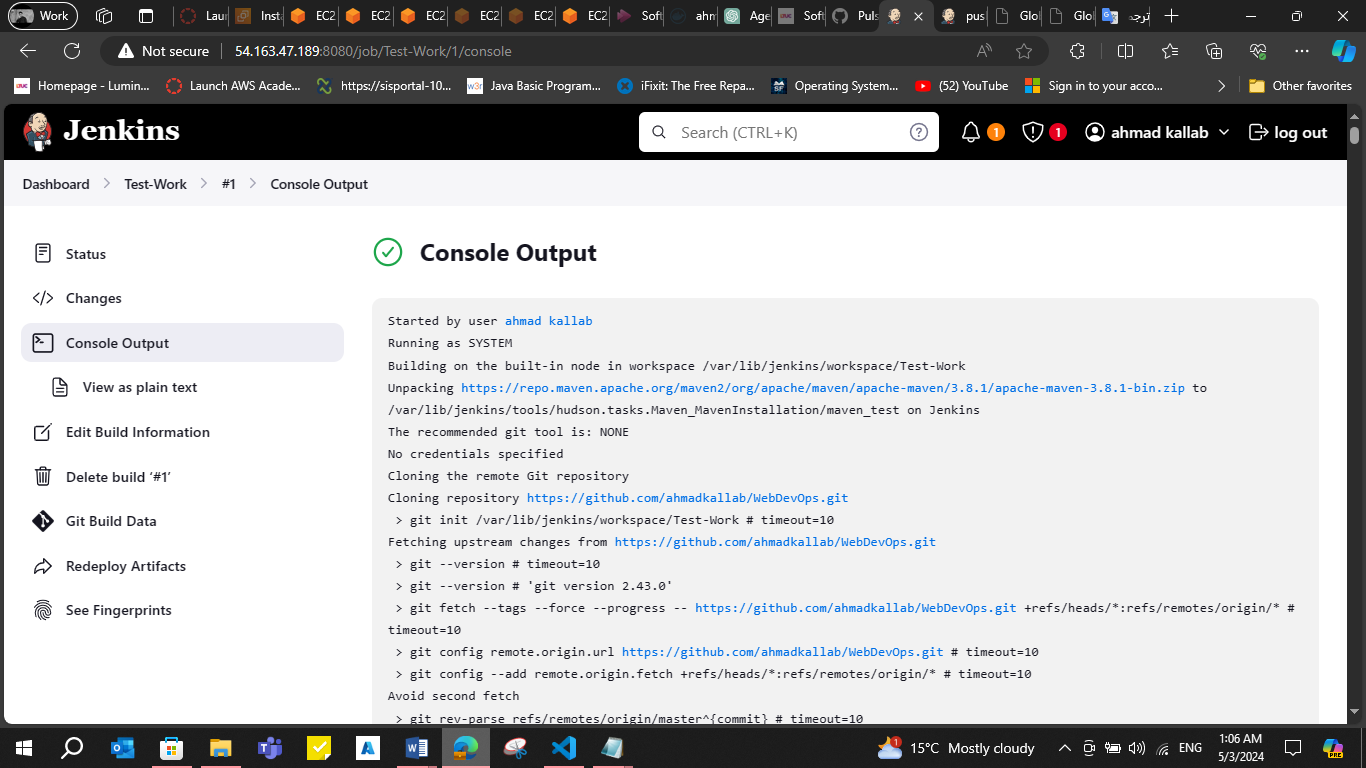


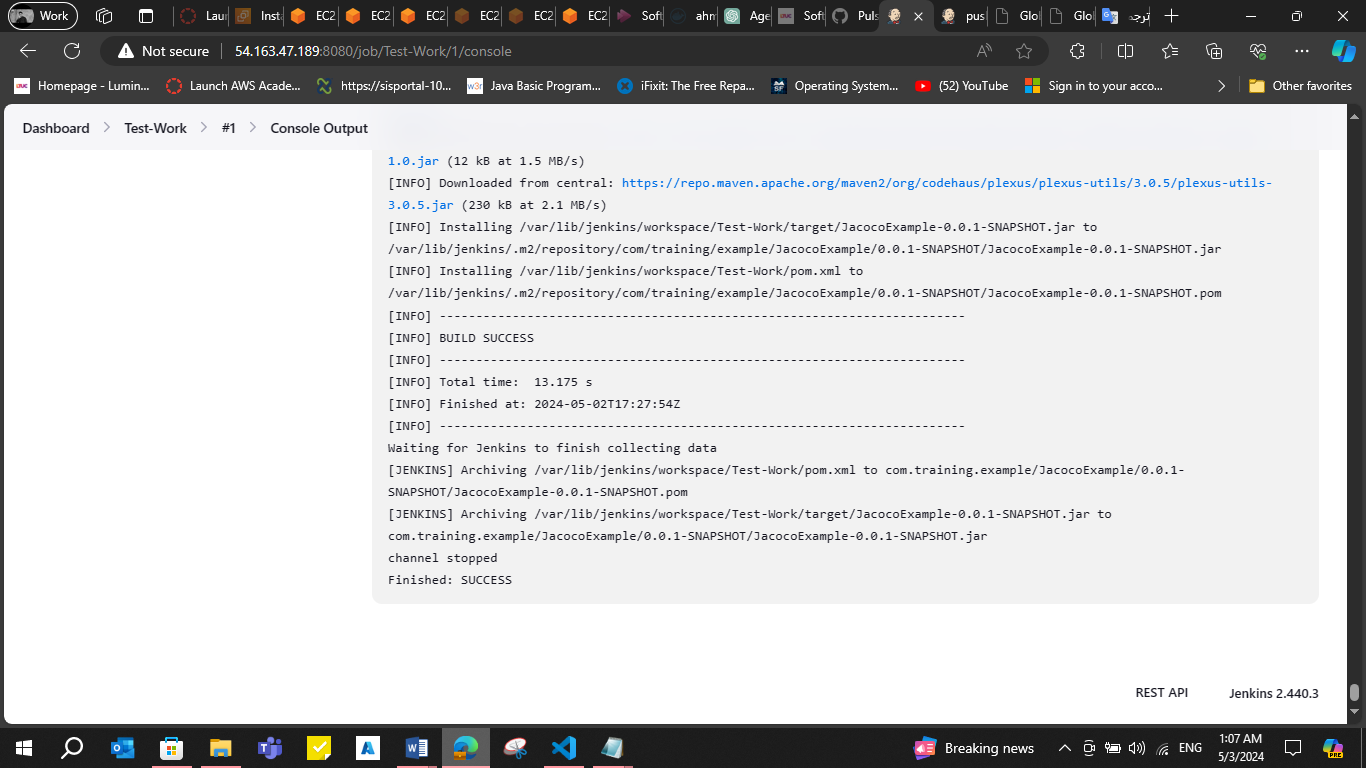
16. الانتهاء من خط أنابيب النشر مع "push-production": لإنهاء مسار النشر، قمت بإنشاء مرحلة خط أنابيب نهائية تسمى "push-production" لتسهيل النشر في بيئة الإنتاج. ضمنت هذه المرحلة نشر الإصدار النهائي للبرنامج للمستخدمين النهائيين، مع الاستفادة من نظام أسماء النطاقات ((DNS إمكانية الوصول العام والوظائف التشغيلية.



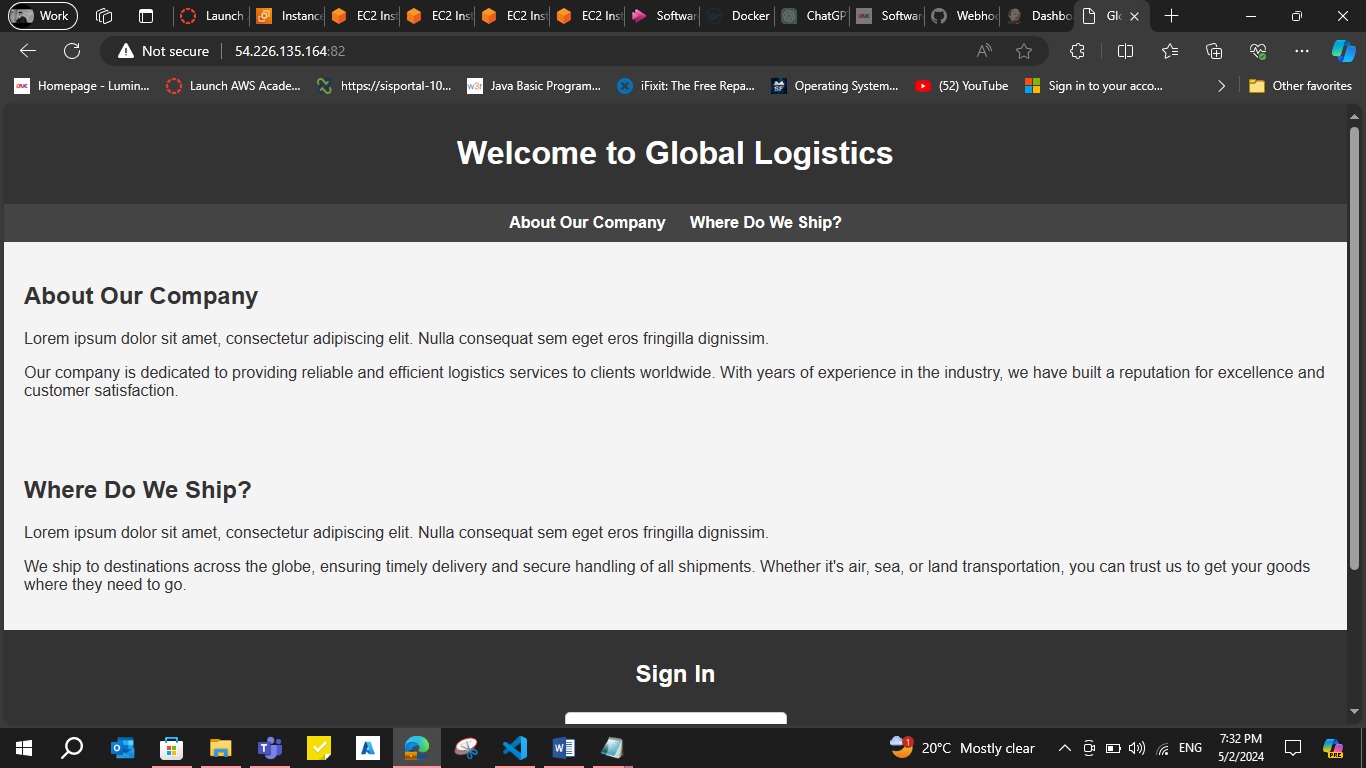
17.الاختبار والتحقق من الصحة: طوال عملية النشر، قمت بإجراء الاختبار والتحقق بدقة في كل مرحلة للتحقق من الأداء الوظيفي والموثوقية. يضمن الاختبار الشامل الاستضافة الصحيحة لموقع الويب، والتنفيذ السليم للتكوينات، واستقرار النظام بشكل عام قبل المضي قدمًا.





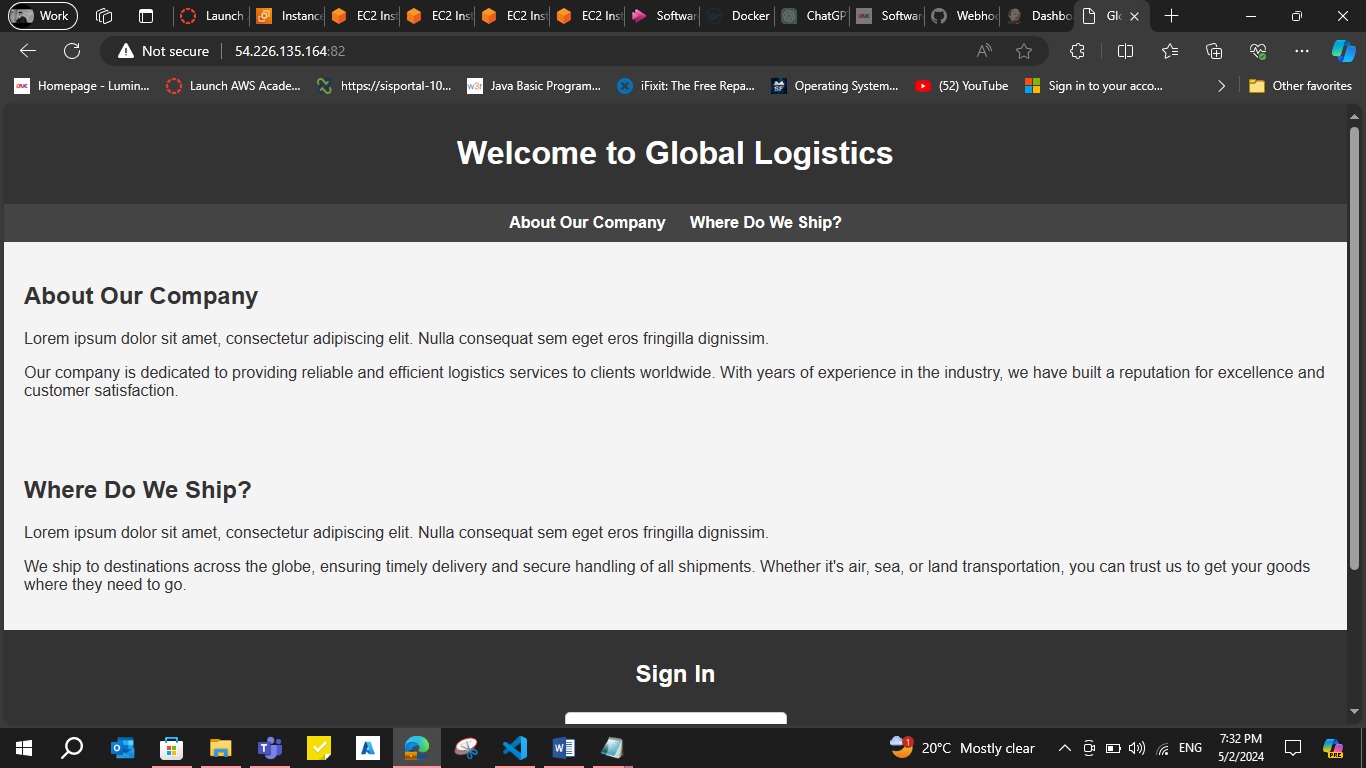


18.الإكمال والانعكاس: مع إنشاء خط النشر وتشغيله، فكرت في تقدم المشروع، وأقرت بالتنفيذ الناجح للأتمتة والعمليات المبسطة. لقد ظللت متيقظًا للتحسينات والتحسينات المستقبلية لتحسين سير عمل النشر وكفاءة المشروع.

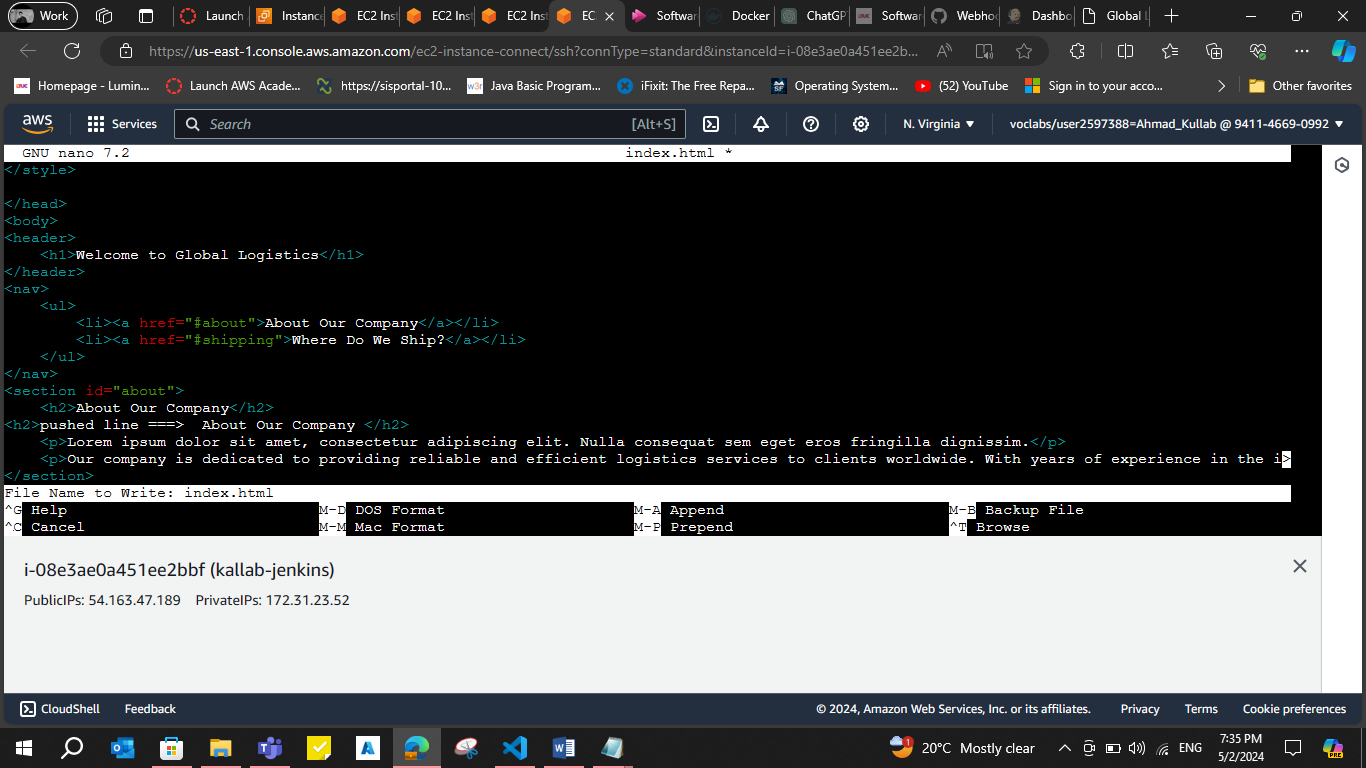


20. وكجزء من الاختبار المستمر والتحقق من الصحة، قمت بإجراء تجارب إضافية لضمان موثوقية وفعالية مسار النشر. يتضمن أحد هذه الاختبارات دفع المحتوى المحدث إلى ملف Index.html الخاص بموقع الويب مباشرةً من مثيل Jenkins EC2. قدم هذا الاختبار دليلاً ملموسًا على التكامل السلس بين Jenkins وبيئة الاستضافة، مما يوضح إمكانات النشر المستمر للمشروع دون مقاطعة اتصال موقع الويب أو الأداء.

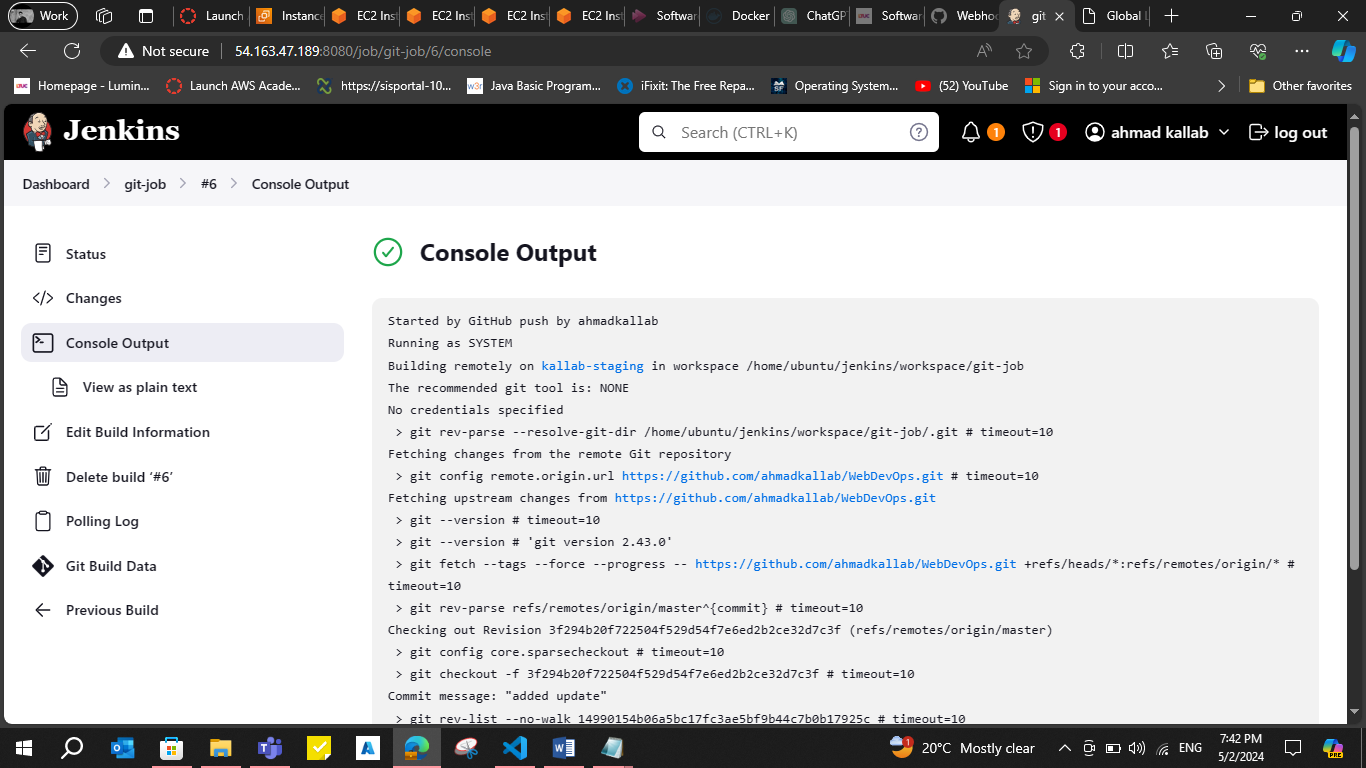
الموقغ قبل دفع المحتوى



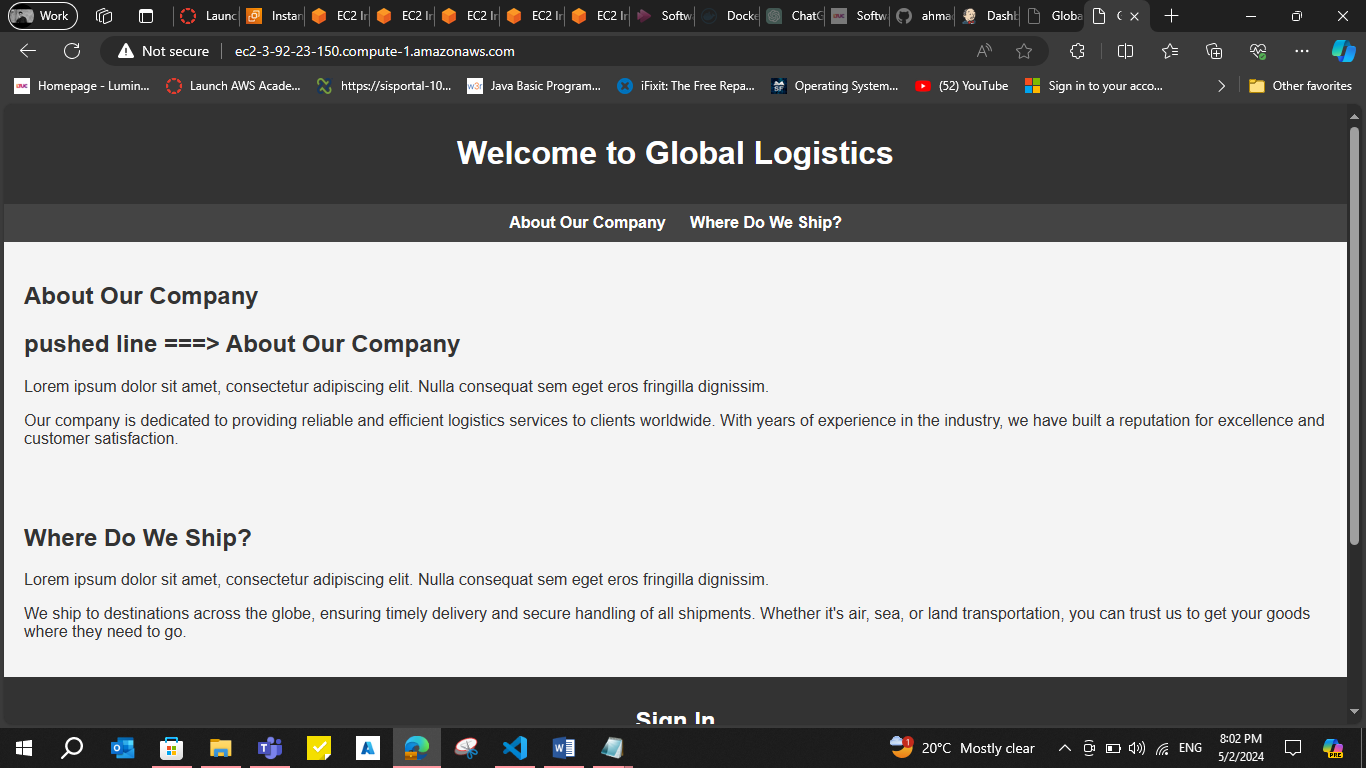
المحتوى الذي اريد دفعه



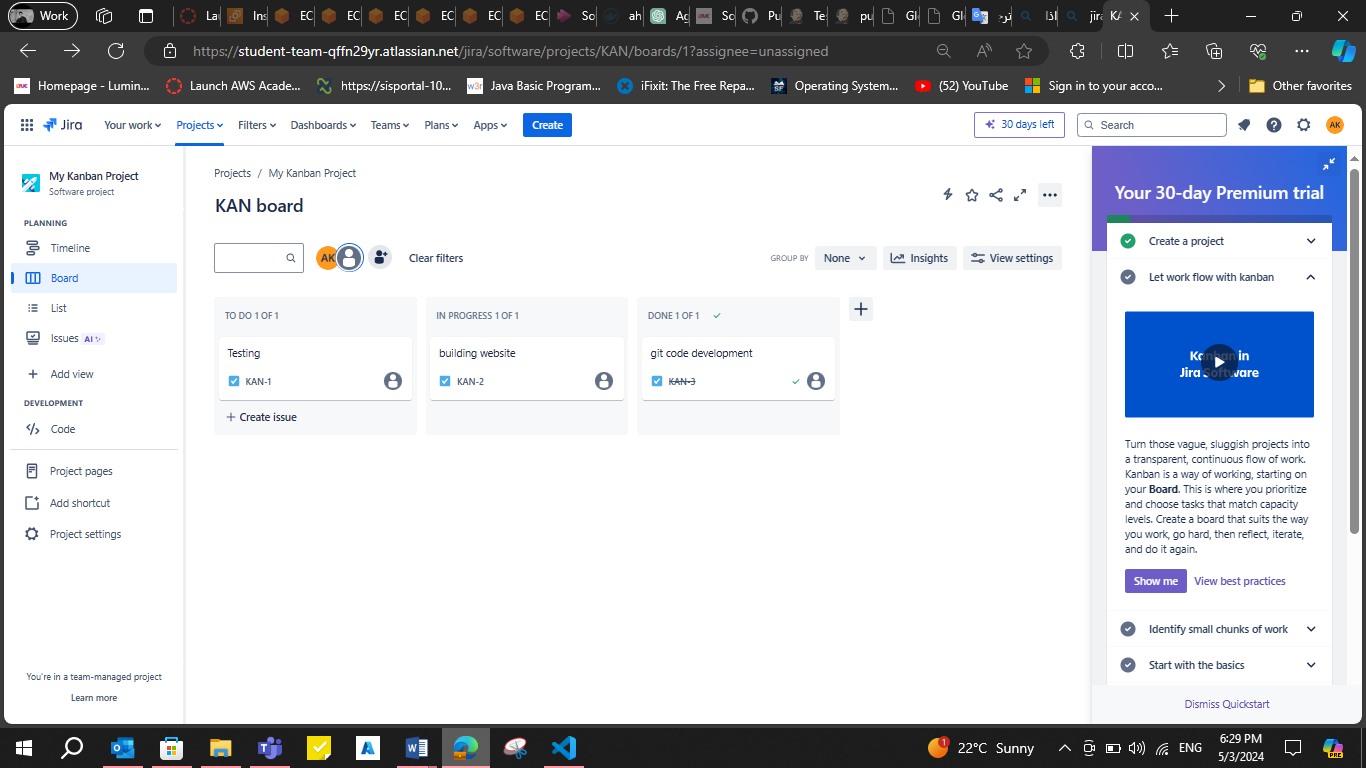
تفاصيل الدفعه على Jenkins



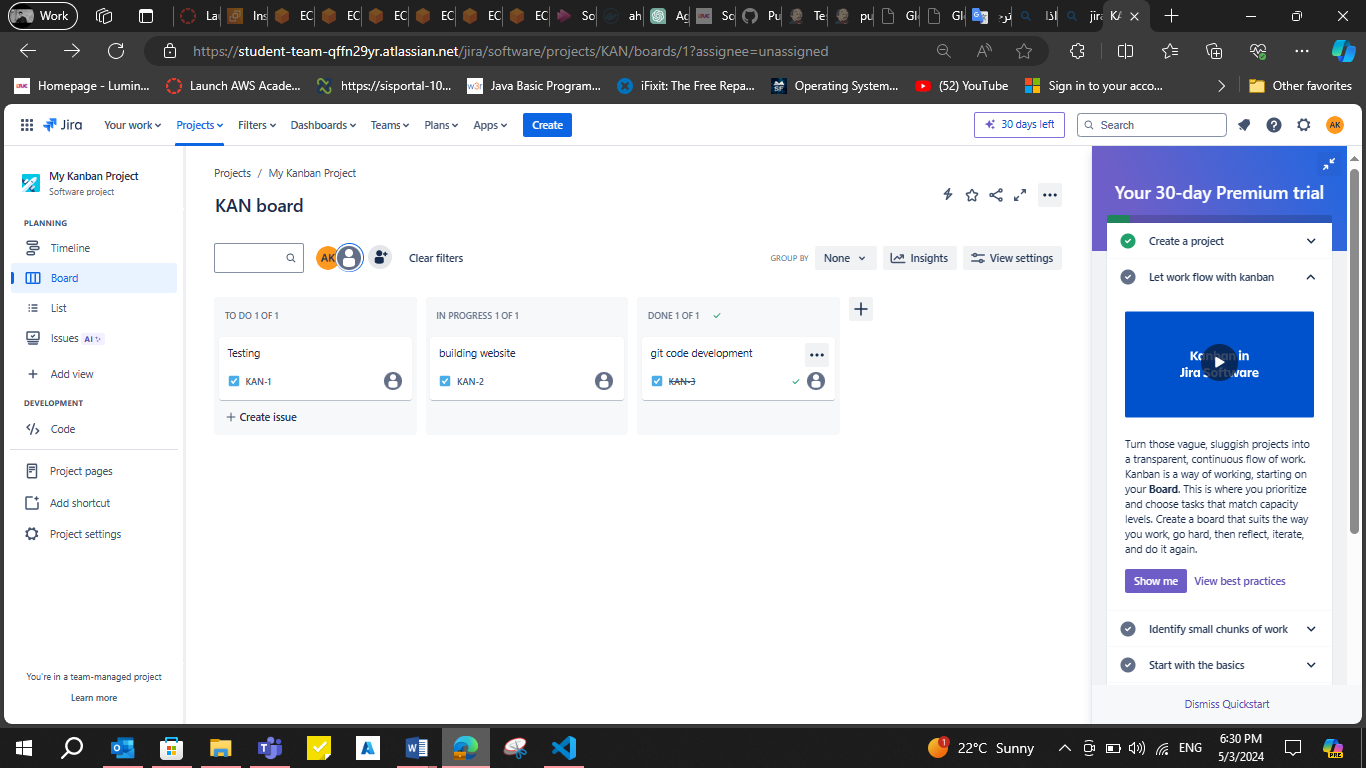
الموقع بعد الدفعه باستخدام جينكينز بالتعاون مع github



17. باستخدام Jira كأداة لإدارة المشروع، تمكنت من الوصول إلى حسابي وبدأت مشروع تطوير برمجيات، واخترت لوحة Kanban لتصور مراحل سير العمل بكفاءة. ضمن قسم "المهام"، قمت بإنشاء issue بعنوان "اختبار" لتمثيل المهام التي تنتظر التنفيذ، مما يضمن رؤية واضحة للأنشطة المعلقة.



18. من خلال التقدم خلال دورة حياة المشروع، قمت بنقل المهام إلى مراحل سير عمل مختلفة داخل لوحة كانبان. في قسم "قيد التقدم"، أضفت مهمة "إنشاء موقع ويب" للإشارة إلى أنشطة التطوير المستمرة، والحفاظ على الشفافية فيما يتعلق بمهام ومسؤوليات المشروع النشطة.



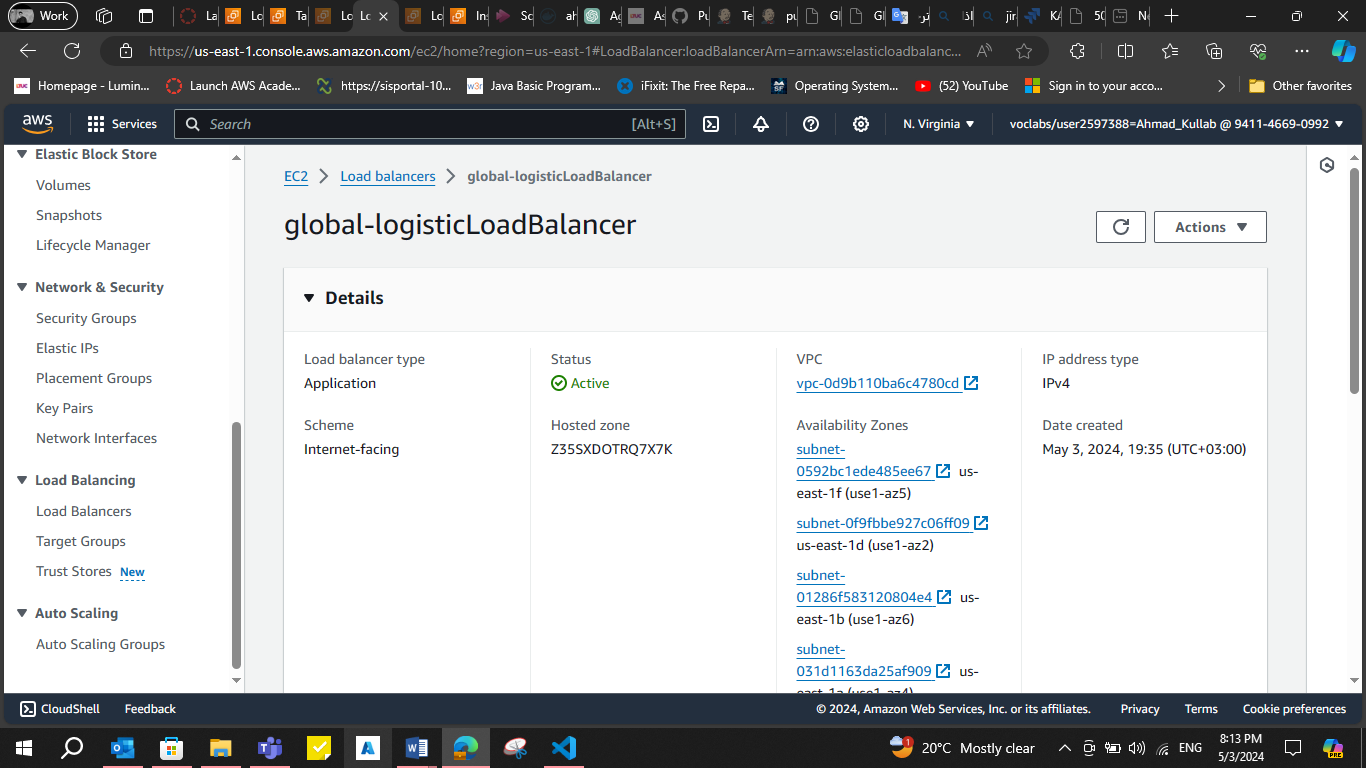
19. مع اكتمال المهام وتحقيق التسليمات، قمت بنقلها إلى قسم "تم" في لوحة كانبان. والجدير بالذكر أنني قمت بنقل المهمة المتعلقة بـ

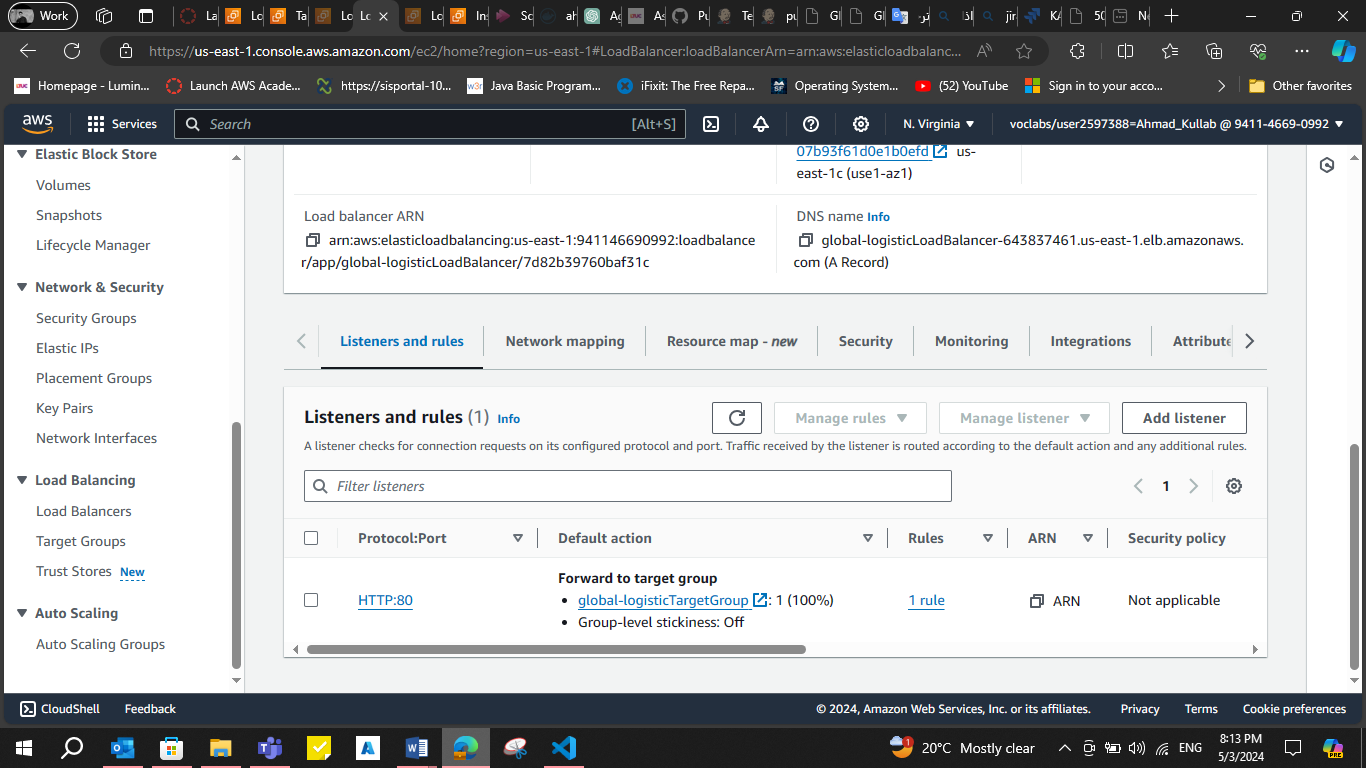
" " Git Code إلى هذا القسم، مما يشير إلى الإكمال الناجح لمهمة التطوير المرتبطة وضمان التتبع الدقيق لتقدم المشروع داخل jira .

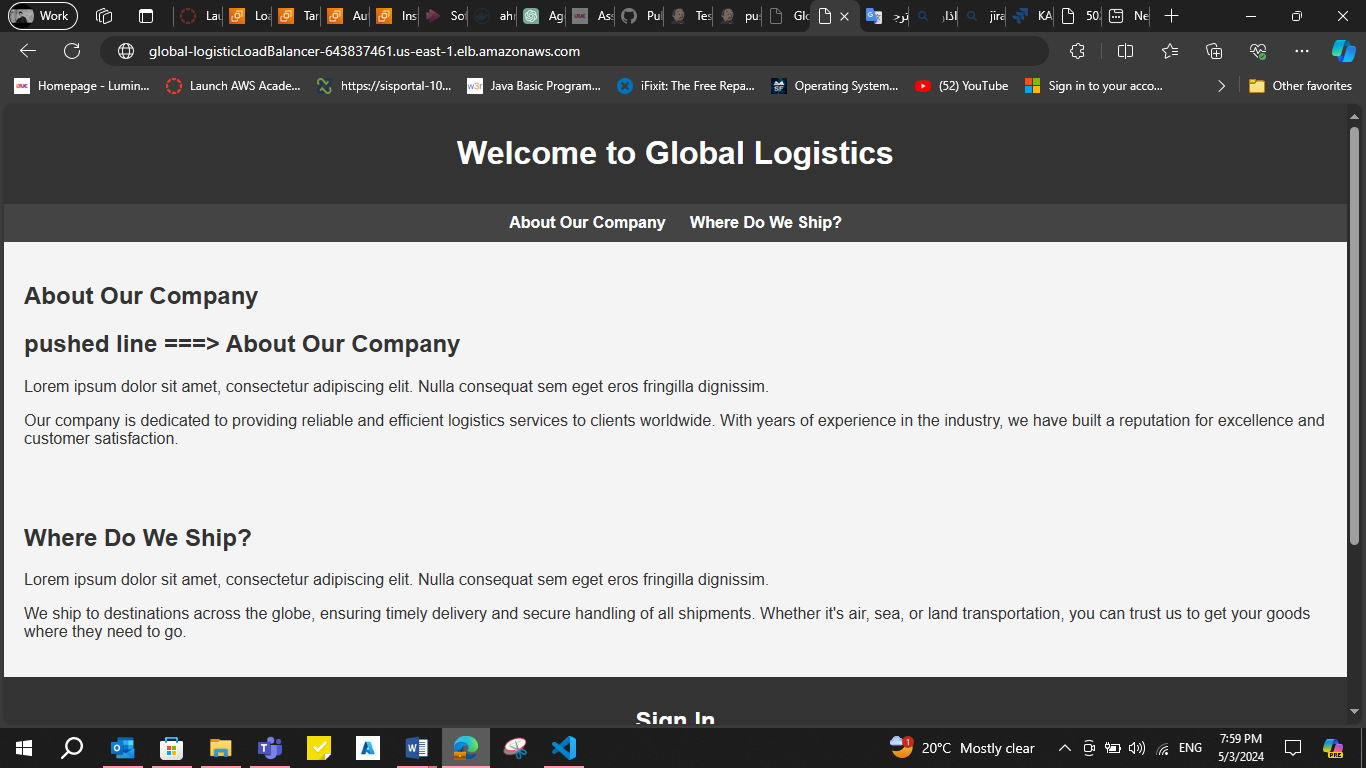


**التحسنيات التي قمت بها بناءً على متطلبات مشروع global logistics**

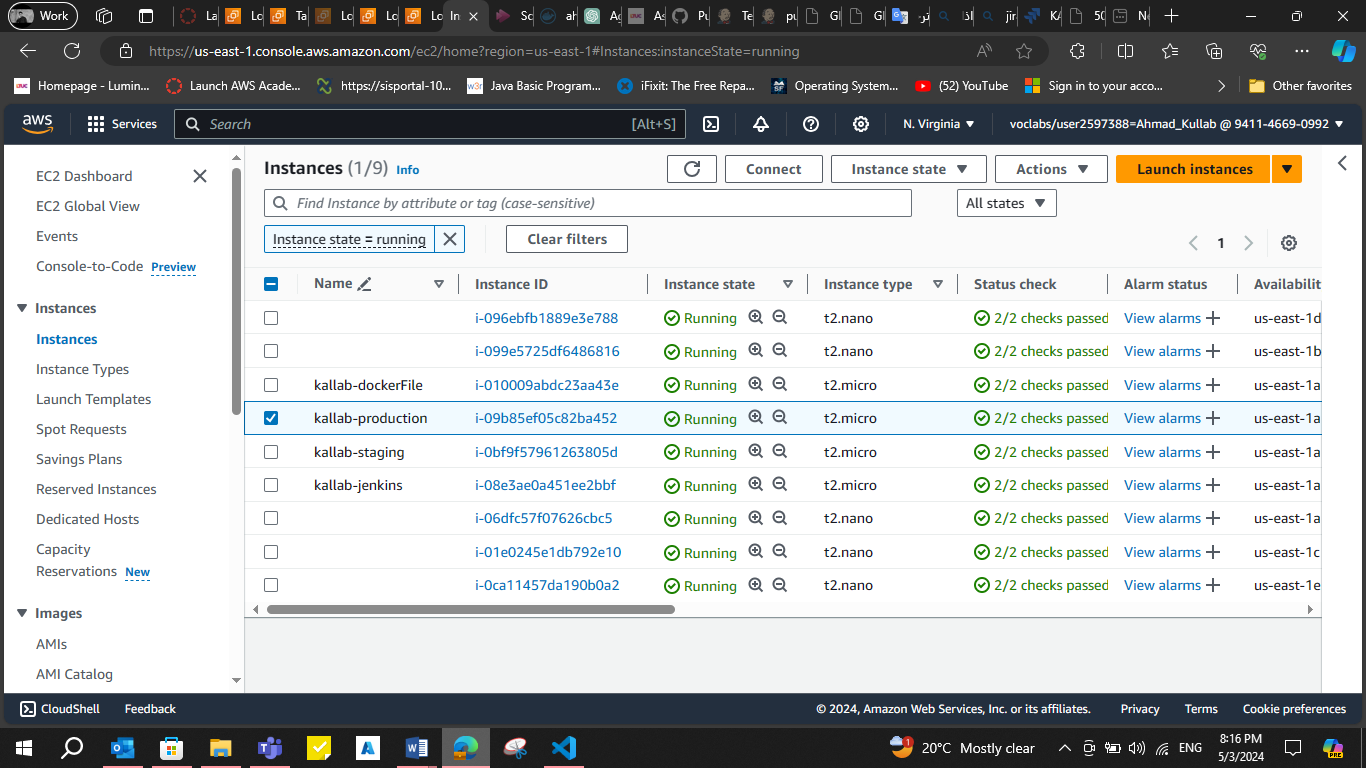
إنشاء موازن التحميل: عند البدء في إنشاء Elastic Load Balancer (ELB)، قمت بتكوين إعدادات موازن التحميل بعناية لتحسين الأداء والموثوقية لموقع الويب. يتضمن ذلك تحديد نوع موازن التحميل المناسب، مثل Application Load Balancer (ALB) أو Network Load Balancer (NLB)، استنادًا إلى متطلبات موقع الويب. على سبيل المثال، إذا كان موقع الويب يتضمن التعامل مع حركة مرور HTTP/HTTPS ويتطلب ميزات توجيه متقدمة، فقد اخترت ALB. بالإضافة إلى ذلك، قمت بتكوين المستمعين لتحديد المنافذ والبروتوكولات التي من خلالها سيستقبل موازن التحميل حركة المرور الواردة. من خلال تكوين إعدادات التحقق من الصحة بعناية، تأكدت من أن موازن التحميل يراقب بشكل مستمر صحة مثيلات EC2 ويوجه حركة المرور فقط إلى المثيلات السليمة، وبالتالي تحسين توفر موقع الويب ومرونته.

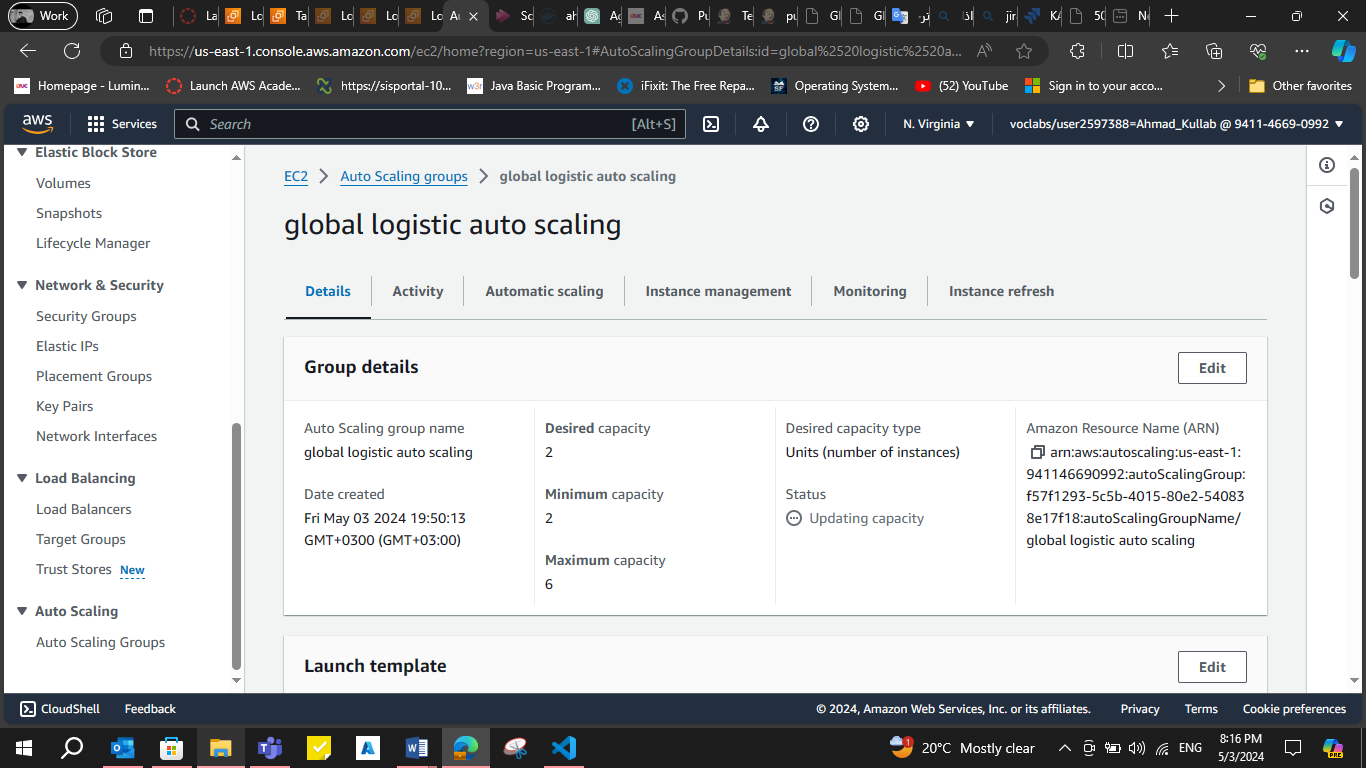


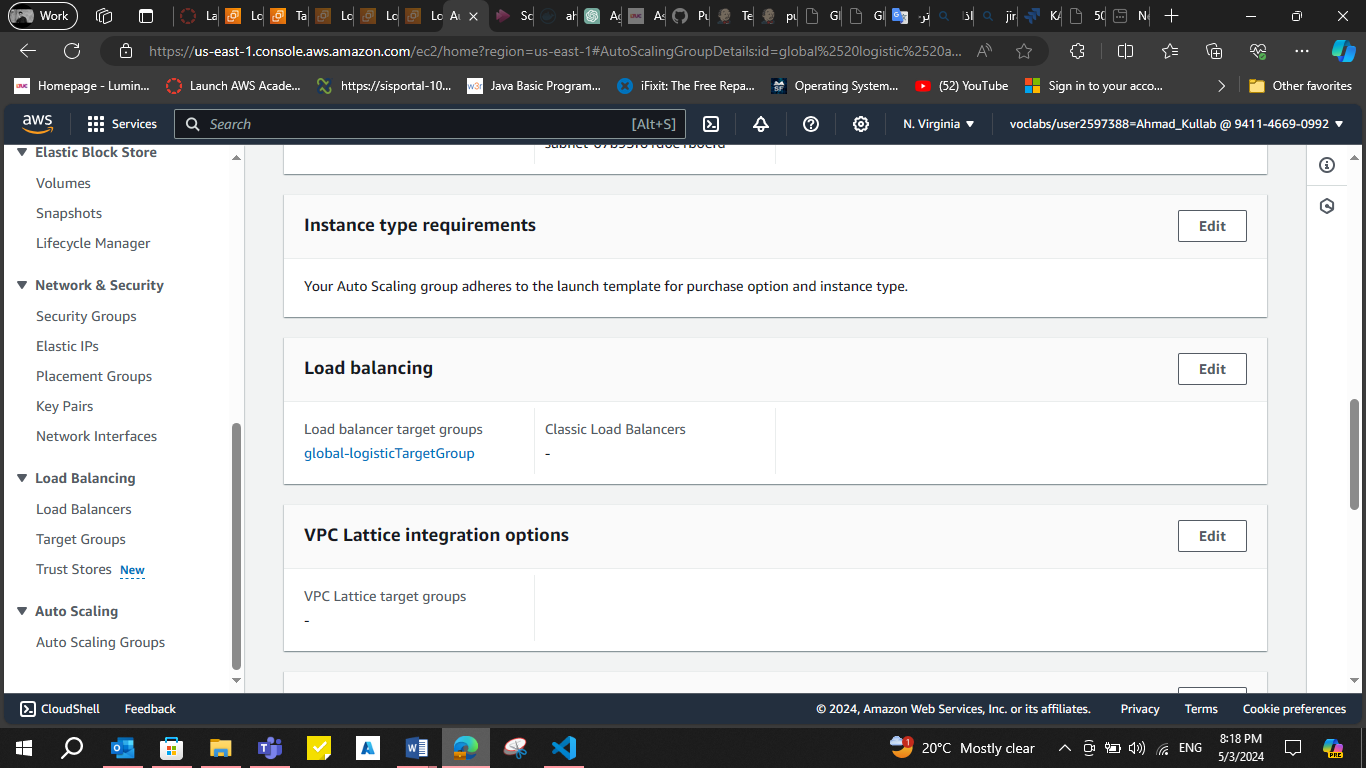


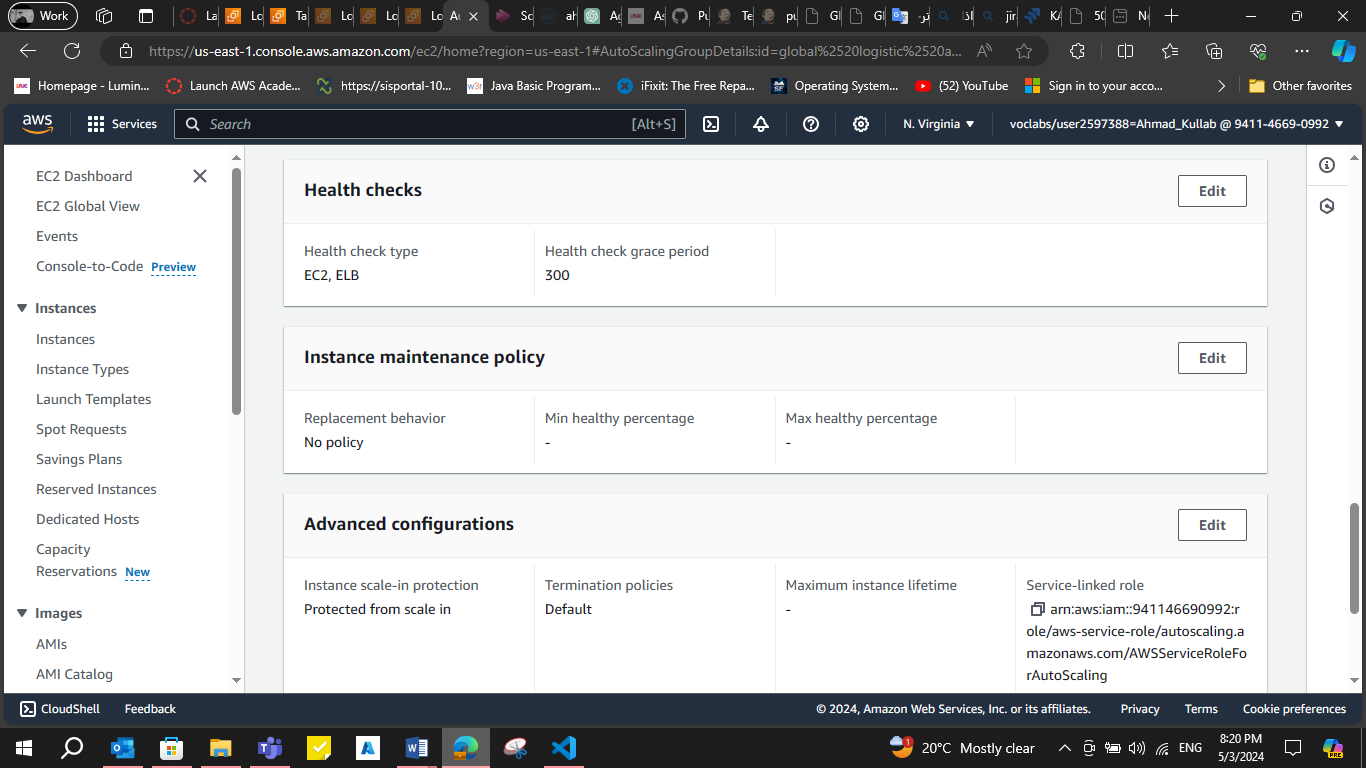


إعداد مجموعة القياس التلقائي: عند إنشاء مجموعة القياس التلقائي، قمت بتصميم تكوين الإطلاق ليناسب الاحتياجات المحددة لموقع الويب. يتضمن ذلك اختيار Amazon Machine Image (AMI) المناسب الذي يحتوي على البرامج والتكوينات اللازمة لاستضافة موقع الويب. على سبيل المثال، إذا كان موقع الويب يتطلب تبعيات برامج معينة أو بيئات تشغيل محددة، فقد تأكدت من تضمينها في تكوين الإطلاق. بالإضافة إلى ذلك، قمت بتكوين أنواع المثيلات بناءً على عبء العمل المتوقع ومتطلبات الأداء لموقع الويب. على سبيل المثال، إذا واجه موقع الويب أنماطًا متقلبة لحركة المرور على مدار اليوم، فقد قمت بتحديد أنواع المثيلات التي توفر توازنًا بين كفاءة التكلفة وقابلية تطوير الأداء. علاوة على ذلك، قمت بضبط سياسات القياس لتفعيل إجراءات القياس بناءً على المقاييس ذات الصلة بأداء موقع الويب، مثل استخدام وحدة المعالجة المركزية، أو عدد الطلبات، أو زمن الاستجابة. سمح ذلك لمجموعة Auto Scaling بضبط عدد مثيلات EC2 ديناميكيًا استجابة للتغيرات في الطلب على حركة المرور، مما يضمن الأداء الأمثل وفعالية التكلفة لموقع الويب. أخيرًا، من خلال ربط مجموعة Auto Scaling بموازن التحميل، قمت بتمكين التكامل السلس بين المثيلات التي تم ضبطها تلقائيًا وموازن التحميل، مما يضمن تسجيل المثيلات الجديدة تلقائيًا مع موازن التحميل والبدء في خدمة حركة المرور دون انقطاع.









**Task 2.B**

يكمن التعاون في جوهر فلسفة DevOps، مع التركيز على التكامل السلس بين الأشخاص والعمليات والتكنولوجيا لتقديم القيمة للعملاء بسرعة وباستمرار. في سياق مشاريع DevOps، يلعب استخدام الأدوات التكنولوجية التعاونية دورًا محوريًا في تسهيل الاتصال الفعال وتعزيز العمل الجماعي وتحفيز الابتكار عبر جميع مراحل دورة حياة تطوير البرمجيات.

من خلال الاستفادة من الأدوات التعاونية مثل Jira وJenkins وGitHub وDocker Hub، يمكن لفرق DevOps تبسيط سير العمل وتعزيز الرؤية وتعزيز التعاون بين أعضاء الفريق متعدد الوظائف. توفر هذه الأدوات منصة مركزية لتخطيط المهام وتتبعها وإدارتها، مما يمكّن الفرق من تحديد أولويات العمل وتخصيص الموارد بكفاءة والاستجابة بسرعة للمتطلبات المتغيرة.

:Jiraباعتبارها أداة قوية لإدارة المشاريع، توفر Jira منصة شاملة لتخطيط وتتبع وإدارة المهام وقصص المستخدمين والملاحم. تتيح لوحات Kanban القابلة للتخصيص وميزات التخطيط الرشيقة للفرق إمكانية تصور سير العمل وتحديد أولويات عناصر العمل والتكيف مع المتطلبات المتغيرة بسرعة. باستخدام Jira، يمكن للفرق تعزيز التعاون والمواءمة والشفافية، مما يضمن أن يكون الجميع على نفس الصفحة ويعملون على تحقيق الأهداف المشتركة. علاوة على ذلك، توفر إمكانات إعداد التقارير والتحليلات الخاصة بـ Jira رؤى قيمة حول تقدم المشروع، مما يمكّن الفرق من تحديد الاختناقات وتحسين العمليات ودفع التحسين المستمر.

Jenkins: Jenkins هو خادم أتمتة متعدد الاستخدامات يلعب دورًا مركزيًا في تمكين ممارسات التكامل والتسليم المستمر (CI/CD). من خلال أتمتة عمليات البناء والاختبار والنشر، تعمل Jenkins على تسريع دورات تسليم البرامج وتقليل الأخطاء اليدوية وتعزيز الكفاءة الإجمالية. تسمح بنيتها الموسعة ونظامها البيئي الواسع للمكونات الإضافية للفرق بالتكامل مع الأدوات والتقنيات المختلفة بسلاسة. تعمل خطوط أنابيب Jenkins، التي تم تعريفها على أنها تعليمات برمجية، على تمكين الفرق من تدوين سير عمل النشر الخاص بهم وتعزيز التعاون وضمان الاتساق عبر البيئات. مع Jenkins، يمكن للفرق تحقيق حلقات ردود فعل أسرع، والتكرار بسرعة، وتقديم حلول برمجية عالية الجودة بثقة.

:GitHubباعتباره منصة رائدة للتحكم في الإصدار، يُحدث GitHub ثورة في تطوير البرامج التعاونية من خلال توفير مستودع مركزي للتعاون والمشاركة في التعليمات البرمجية. تعمل آلية طلب السحب الخاصة بها على تسهيل مراجعة الأكواد البرمجية والتعاون ومشاركة المعرفة بين أعضاء الفريق. تتيح استراتيجيات التفرع مثل GitFlow للفرق إمكانية إدارة تطوير الميزات والإصلاحات العاجلة والإصدارات بكفاءة. يتيح نظام تتبع المشكلات في GitHub للفرق إدارة تقارير الأخطاء وطلبات الميزات والمهام بسلاسة، مما يعزز التعاون والشفافية عبر دورة حياة التطوير. علاوة على ذلك، تعمل GitHub Actions على تمكين الفرق من أتمتة سير العمل مباشرة داخل مستودعاتهم، مما يزيد من تعزيز الإنتاجية والتعاون.

:Docker Hub يعمل Docker Hub بمثابة سجل مركزي لصور Docker، مما يمكّن الفرق من تخزين التطبيقات الموجودة في حاويات ومشاركتها وإدارتها بشكل فعال. باستخدام Docker Hub، يمكن للفرق الاستفادة من الصور المعدة مسبقًا والصور الأساسية والمستودعات الرسمية لبدء رحلة النقل بالحاويات الخاصة بهم. يعمل تكامل Docker Hub مع Docker Desktop وDocker Engine على تبسيط عملية تطوير ونشر التطبيقات المعبأة في حاويات، مما يعزز الاتساق وقابلية النقل وقابلية التوسع عبر البيئات. بالإضافة إلى ذلك، تساعد ميزات الفحص الأمني في Docker Hub الفرق على تحديد نقاط الضعف وتخفيف المخاطر وضمان سلامة صور الحاوية الخاصة بهم.

ومن خلال تسخير قدرات هذه الأدوات التكنولوجية التعاونية، يمكن لفرق DevOps التغلب على الصوامع وتبسيط العمليات وتعزيز ثقافة التعاون والابتكار والتحسين المستمر. تعمل هذه الأدوات على تمكين الفرق من تقديم القيمة للعملاء بسرعة وبشكل موثوق وبجودة لا مثيل لها، مما يؤدي إلى نجاح الأعمال والميزة التنافسية في المشهد الرقمي سريع الخطى اليوم.باختصار، تعد الأدوات التكنولوجية التعاونية أصولًا لا غنى عنها في مشاريع DevOps، حيث تعمل على تمكين الفرق من التعاون بشكل فعال وتبسيط العمليات وتقديم حلول برمجية عالية الجودة بسرعة وسهولة. ومن خلال تبني هذه الأدوات، يمكن لفرق DevOps التغلب على التحديات، ودفع الابتكار، وتحقيق التحسين المستمر، مما يؤدي في النهاية إلى تقديم قيمة أكبر للعملاء وأصحاب المصلحة.

إن استخدام الأدوات التعاونية مثل Jira وJenkins وGitHub وDocker Hub خلال مرحلة تطوير مشروعي له ما يبرره لعدة أسباب مقنعة:

: **Jira** يعتبر Jira بمثابة حجر الزاوية في منهج إدارة المشروع الخاص بي في بيئات Agile، حيث يقدم مجموعة شاملة من الميزات التي تمكن فريقي من تنظيم المهام وتتبع التقدم وتسهيل الاتصال بسلاسة. في Jira، أقوم بإنشاء قصص المستخدم والملاحم والمهام لتمثيل مستويات مختلفة من عناصر العمل، مما يضمن الوضوح والتفاصيل في إدارة المهام. من خلال تحديد أولويات عناصر العمل بناءً على قيمة العمل والإلحاح، أضمن أن فريقي يركز على تقديم ميزات عالية القيمة تتوافق مع أهداف المشروع. توفر لوحات Kanban وScrum القابلة للتخصيص من Jira تمثيلات مرئية للعمل الجاري، مما يمكننا من إدارة المهام بفعالية، وتحديد الاختناقات، وتنفيذ دورات التطوير التكرارية بدقة. بالإضافة إلى ذلك، تسمح لي إمكانات إعداد التقارير القوية في Jira بتحليل مقاييس المشروع ومراقبة أداء الفريق واتخاذ القرارات المستندة إلى البيانات، مما يضمن بقاء فريقي على المسار الصحيح نحو تحقيق أهداف المشروع. بشكل عام، تعمل Jira على تعزيز التعاون والشفافية والمرونة في تنفيذ مشروعي، مما يمكّن فريقي من تقديم حلول برمجية عالية الجودة تلبي توقعات أصحاب المصلحة وتعزز قيمة الأعمال.

**Jenkins**: يعد جينكينز بمثابة حجر الزاوية في خط أنابيب التسليم الآلي للبرامج، مما أحدث ثورة في أسلوبي في عمليات التطوير والنشر. من خلال الاستفادة من Jenkins كخادم أتمتة، أقوم بتنسيق وتبسيط المراحل المختلفة من دورة حياة التطوير، بدءًا من تكامل التعليمات البرمجية واختبارها وحتى النشر والتسليم. من خلال تكوين خطوط أنابيب CI/CD قوية داخل Jenkins، أقوم بإزالة التدخلات اليدوية وأتمتة المهام المتكررة، مثل تجميع التعليمات البرمجية والتعبئة والاختبار والنشر عبر بيئات متنوعة. لا تعمل هذه الأتمتة على تسريع عملية تسليم البرامج فحسب، بل تعمل أيضًا على تحسين الجودة الشاملة وموثوقية التطبيقات المنشورة. بالإضافة إلى ذلك، يتكامل Jenkins بسلاسة مع أنظمة التحكم في الإصدار الشائعة مثل GitHub، مما يتيح إنشاءات قائمة على المشغلات ويضمن التحقق السلس من صحة وتكامل تغييرات التعليمات البرمجية. من خلال جينكينز، أقوم بإنشاء حلقة مستمرة من ردود الفعل، وتسهيل التكرار السريع، والتحسين المستمر، وممارسات التطوير السريعة داخل مشروعي.

:**GitHubيمثل** GitHub منصة تعاونية لا غنى عنها في نظام التطوير البيئي الخاص بي، مما يسهل التحكم في الإصدار والتعاون في التعليمات البرمجية وإدارة المشاريع بكفاءة لا مثيل لها. من خلال الاستفادة من ميزاته القوية، أتعاون بسلاسة مع زملائي المطورين من خلال طلبات السحب، وبدء المناقشات، والتماس التعليقات، وتكرار تغييرات التعليمات البرمجية بشكل تعاوني. تضمن عملية مراجعة الكود الحفاظ على معايير الجودة العالية والالتزام بأفضل الممارسات، وتعزيز ثقافة التحسين المستمر وتبادل المعرفة داخل الفريق. يمكّنني نظام تتبع المشكلات المتكامل في GitHub من تحديد الأخطاء وطلبات الميزات وتحديد أولوياتها ومعالجتها بطريقة منهجية، مما يضمن التقدم السلس للمشروع. علاوة على ذلك، يتيح التكامل السلس لـ GitHub مع أدوات CI/CD مثل Jenkins أتمتة سير العمل، وتبسيط عملية تسليم التعليمات البرمجية مع الحفاظ على جودة التعليمات البرمجية وممارسات النشر المتسقة. في جوهر الأمر، يعد GitHub بمثابة العمود الفقري لمساعي التطويرية، حيث يعزز التعاون والشفافية وإمكانية التتبع عبر دورة حياة تطوير البرمجيات بأكملها.

**:Docker hub**يعتبر Docker Hub بمثابة حجر الزاوية في سير عمل تطوير التطبيقات المعبأة في حاويات، حيث يوفر نظامًا أساسيًا مركزيًا لإدارة صور Docker وتبسيط عملية النشر. من خلال Docker Hub، أستفيد من قوة النقل بالحاويات لتحقيق الاتساق وقابلية النقل عبر بيئات متنوعة، بدءًا من التطوير ووصولاً إلى الإنتاج. يمكّنني المستودع الشامل للمنصة من الصور المعدة مسبقًا والصور الأساسية من تسريع عملية التطوير من خلال الاستفادة من الحلول الحالية وأفضل الممارسات. من خلال تخزين وإدارة صور Docker الخاصة بي على Docker Hub، أضمن الموثوقية وقابلية التوسع في تطبيقاتي الحاوية، مع الاستفادة أيضًا من ميزات الأمان القوية للنظام الأساسي وإمكانيات مسح الصور. علاوة على ذلك، يتكامل Docker Hub بسلاسة مع خطوط أنابيب CI/CD الخاصة بي، مما يتيح لي أتمتة إنشاء ونشر صور Docker، وبالتالي تعزيز كفاءة وموثوقية عملية تسليم البرامج الخاصة بي. بشكل عام، يعمل Docker Hub كأداة لا غنى عنها في مجموعة أدوات التطوير الخاصة بي، مما يتيح لي تبسيط تطوير التطبيقات المعبأة في حاويات ونشرها بسهولة وثقة.

**Task 2.C**

كان عملي متوافقًا بدقة مع احتياجات شركة Global Logistics، مع التركيز على تطوير نظام إدارة لوجستية قائم على السحابة يعمل على تبسيط العمليات وتعزيز خدمة العملاء. من خلال اعتماد نموذج دورة حياة تطوير البرمجيات Agile (SDLC)، تأكدت من أن المشروع يظل مستجيبًا للمتطلبات والأولويات المتطورة لشركة Global Logistics طوال دورة حياته. لقد سهّل نهج Agile التطوير التكراري، ودورات ردود الفعل السريعة، والتحسين المستمر، مما مكننا من تقديم القيمة بشكل تدريجي والتكيف مع ديناميكيات السوق المتغيرة بشكل فعال.

طوال المشروع، قمت بتنسيق إعداد وتكوين مثيلات AWS EC2 بدقة، وتخصيص الموارد بشكل استراتيجي لبيئات الإنتاج والتجهيز وبيئات Jenkins. لقد بدأت في تثبيت الأدوات الأساسية مثل Jenkins وDocker في هذه الحالات، مما يضمن التوافق مع Java Development Kit (JDK) في جميع المجالات. من خلال الاستفادة من وحدة الإدارة في AWS، قمت بإنشاء موازنات التحميل ومجموعات القياس التلقائي، وربطها بسلاسة بمثيلات EC2 لتعزيز قابلية التوسع والموثوقية.

بالانتقال إلى التحكم في الإصدار وإدارة المشاريع، قمت بإنشاء مستودع GitHub باسم WebDevOps، مما يسهل التطوير التعاوني والتحكم في الإصدار. من خلال ربط هذا المستودع بـ Jenkins، قمت بأتمتة عملية دفع تغييرات التعليمات البرمجية إلى بيئة الإنتاج، مما يضمن التكامل والنشر السلس. علاوة على ذلك، قمت بالاستفادة من Docker Hub لتبسيط نشر التطبيقات المعبأة في حاويات، وإنشاء ملف Dockerfile ودفعه إلى مستودع GitHub للتحكم في الإصدار.

داخل Jenkins، قمت بتكوين الوظائف وخطوط الأنابيب بدقة لأتمتة عمليات البناء والاختبار والنشر، مما يزيد من الكفاءة ويقلل التدخل اليدوي. من خلال استخدام الأدوات التعاونية مثل Jira، قمت بتتبع تقدم المشروع وإدارة المهام والتكيف مع المتطلبات المتغيرة في الوقت الفعلي. يضمن هذا النهج الرشيق، المدعوم بالتقنيات التعاونية، التطوير التكراري، ودورات التغذية الراجعة السريعة، والتحسين المستمر.

أدى اعتماد الأدوات التعاونية مثل GitHub وDocker Hub إلى تعزيز عملية التطوير، وتعزيز التعاون في التعليمات البرمجية والشفافية والاتساق. سهّلت ميزات طلبات السحب ومراجعة التعليمات البرمجية من GitHub التعاون بين النظراء وضمان جودة التعليمات البرمجية، في حين قام السجل المركزي لـ Docker Hub بتبسيط إدارة الصور ونشرها.

لقد كان الاستخدام التآزري لأدوات خطوط أنابيب DevOps والتقنيات التعاونية أمرًا محوريًا في تلبية متطلبات الخدمات اللوجستية العالمية. أدى مسار النشر الآلي، المدعوم من Jenkins وGitHub وDocker Hub، إلى تسريع دورات التطوير وتحسين التعاون وضمان تقديم نظام إدارة لوجستية قائم على السحابة عالي الجودة. ومن خلال التواصل الفعال والشفافية وممارسات التطوير المتكررة، نجحنا في تقديم حل يتوافق مع احتياجات وأهداف Global Logistics، مما يضع أساسًا متينًا للتحسينات المستقبلية وقابلية التوسع.

علاوة على ذلك، أدى استخدام الأدوات التعاونية مثل Jira وJenkins إلى تعزيز ثقافة التعاون والمساءلة داخل فريق التطوير. قدمت لوحات Kanban وScrum الخاصة بـ Jira رؤية واضحة لتقدم المشروع، مما مكن أعضاء الفريق من تحديد أولويات المهام وتخصيص الموارد بشكل فعال. من خلال Jenkins، تم دمج الأتمتة في سير عمل التطوير لدينا، مما أدى إلى تقليل الأخطاء اليدوية وتسريع تسليم تحديثات البرامج. من خلال أتمتة عمليات البناء والاختبار والنشر، مكنتنا Jenkins من التركيز بشكل أكبر على الابتكار وبدرجة أقل على المهام المتكررة، مما أدى في النهاية إلى تحسين الإنتاجية ووقت الوصول إلى السوق.

وبالتأمل في فعالية أدوات خط أنابيب DevOps والتقنيات التعاونية المستخدمة، فمن الواضح أنها ساهمت بشكل كبير في نجاح المشروع. أدى التكامل السلس بين مثيلات AWS EC2 وGitHub وDocker Hub وJenkins إلى إنشاء بيئة تطوير متماسكة سهلت التكامل والنشر المستمر. وقد مكننا ذلك من التكرار بسرعة، والرد على التعليقات بسرعة، وتقديم قيمة لشركة Global Logistics في الوقت المناسب. بالإضافة إلى ذلك، أدى اعتماد منهجيات Agile، المدعومة بأدوات تعاونية مثل Jira، إلى تعزيز التخطيط التكيفي والتطوير التطوري والتسليم المبكر، مما يضمن بقاء المشروع متوافقًا مع الاحتياجات المتطورة للوجستيات العالمية. بشكل عام، كان الاستخدام الفعال لأدوات خط أنابيب DevOps والتقنيات التعاونية فعالاً في تقديم حل قوي وقابل للتطوير ومرتكز على العملاء للوجستيات العالمية، مما يضع الأساس للابتكار والنمو في المستقبل.

طوال المشروع، قمت بالاستفادة من مثيلات AWS EC2 لنشر الحل المستند إلى السحابة، مما يضمن قابلية التوسع والموثوقية والأمان. من خلال إعداد مسارات Jenkins للتكامل والنشر المستمر، قمت بأتمتة عمليات البناء والاختبار والنشر، مما يتيح التكرار السريع وتقديم الميزات. وقد سهّل اعتماد حاويات Docker النشر المتسق عبر بيئات مختلفة، في حين كان GitHub بمثابة مستودع مركزي للتحكم في الإصدار والتعاون في التعليمات البرمجية. لقد مكنتنا هذه التقنيات، جنبًا إلى جنب مع ممارسات Agile، من الاستجابة بسرعة للتعليقات، ومعالجة المتطلبات الناشئة، وتقديم حل عالي الجودة مصمم خصيصًا لتلبية احتياجات الخدمات اللوجستية العالمية.

في الختام، يجسد عملي نهجًا شاملاً لتلبية متطلبات الخدمات اللوجستية العالمية من خلال منهجيات Agile وممارسات DevOps. من خلال تبني التعاون والأتمتة والتطوير التكراري، تمكنت من تقديم نظام إدارة لوجستي قائم على السحابة لم يلبي توقعات Global Logistics فحسب، بل تجاوزها أيضًا. يؤكد نجاح المشروع على فعالية نمذجة Agile SDLC والاستخدام الاستراتيجي للأدوات والتقنيات التعاونية في دفع الابتكار والكفاءة ورضا العملاء.

**المراجع:**

[A Complete Guide to SDLC Models | Types, Phases, When to use (geeksforgeeks.org)](https://www.geeksforgeeks.org/sdlc-models-types-phases-use/)

[Top 7 SDLC Methodologies: Phases, Models and Advantages (hackr.io)](https://hackr.io/blog/sdlc-methodologies)

[5 Most Popular Types of SDLC Models | LambdaTest](https://www.lambdatest.com/blog/sdlc-models/)

**Plagiarism**

Plagiarism is a particular form of cheating. Plagiarism must be avoided at all costs and students who break the rules, however innocently, may be penalised. It is your responsibility to ensure that you understand correct referencing practices. As a university level student, you are expected to use appropriate references throughout and keep carefully detailed notes of all your sources of materials for material you have used in your work, including any material downloaded from the Internet. Please consult the relevant unit lecturer or your course tutor if you need any further advice.

**Student Declaration**

|  |
| --- |
| **Student declaration**  I certify that the assignment submission is entirely my own work and I fully understand the consequences of plagiarism. I understand that making a false declaration is a form of malpractice.  Student signature: Date: |