Git Workflow

1. الفروع الرئيسية

- **main:** يحتوي على الكود الجاهز للإنتاج فقط.
- **develop:** يحتوي على أحدث التطويرات المدمجة، ويُعتبر فرع الدمج المؤقت قبل النشر.

2. أنواع الفروع

Feature Branches

- تبدأ من: `develop`
- الاسم: `feature/رقم-التاسك-اسم-الميزة`
- الهدف: تطوير ميزة جديدة
- تدمج إلى: `develop`
- مثال: `feature/123-user-authentication`

Bugfix Branches

- تبدأ من: `develop`
- الاسم: `bugfix/رقم-التاسك-وصف-المشكلة`
- الهدف: إصلاح خطأ أثناء التطوير
- تدمج إلى: `develop`
- مثال: `bugfix/456-fix-login-error`

Release Branches

- تبدأ من: `develop`
- الاسم: `release/رقم-الإصدار`
- الهدف: تجهيز الكود للنشر
- تدمج إلى: `main` و `develop`
- مثال: `release/v1.2.0`

Hotfix Branches

- تبدأ من: `main`
- الاسم: `hotfix/رقم-التاسك-وصف-الإصلاح`
- الهدف: إصلاح مشاكل عاجلة في الإنتاج
- تدمج إلى: `main` و `develop`
- مثال: `hotfix/789-fix-payment-bug`

3. سير العمل اليومي

- قم بفتح التاسك من **Azure DevOps**.
- قم بنسخ رقم التاسك واسم الميزة.
- أنشئ فرعًا جديدًا على Git باسم يحتوي على رقم التاسك مثل: `feature/123-user-authentication` .
- ابدأ العمل على الفرع الجديد.
- عند الانتهاء، افتح Merge Request / Pull Request إلى `develop` .
- اربط Pull Request برقم التاسك في Azure.
- بعد المراجعة والاختبار، يتم الدمج.
- عند اكتمال مجموعة ميزات، يتم إنشاء فرع `release` للنشر.

> **ملاحظة:** تأكد دائمًا من أنك تسحب آخر التحديثات من `develop` قبل البدء في العمل على الفرع الجديد لتجنب التداخلات أو التحديثات المفقودة.

4. نصائح إضافية

- استخدم `git pull --rebase` لتجنب التداخلات.
- قم بتسمية الفروع بشكل واضح ومختصر.
- تأكد من كتابة رسائل الالتزام بشكل واضح.
- قم بمراجعة الكود قبل الدمج.

5. أسئلة شائعة

- **متى يجب استخدام فرع Hotfix؟** عندما تحتاج إلى إصلاح عاجل في الإنتاج.
- **كيف أتعامل مع التداخلات؟** استخدم `git merge` أو `git rebase` لحل التداخلات.

6. موارد إضافية

- وثائق Git الرسمية

7. الخاتمة

هذا هو سير العمل المقترح لاستخدام Git في مشروعك. تأكد من اتباعه لضمان تنظيم الكود وسهولة التعاون بين أعضاء الفريق.