الفصل الثالث

بعد أن أكملنا التعامل مع المراحل الثلاثة في Git , الان لنفترض واجهنا واحد من السيناريوهات التالية ماذا سوف نفعل :

السيناريو الأول – لدي مشروع سواء كتابة شفرة أو وثيقة أو تصميم صورة , وعملت بعض التعديلات عليها ولم أنقلها من Working Directory الى Staging Area , وفي اليوم التالي أردت التراجع عن التعديلات التي عملتها بالأمس !

السيناريو الثاني – نفس السيناريو الأول لكنني نقلتها الى Staging Area واردت التراجع عن التعديلات .

السيناريو الثالث – نفس السيناريو الأول لكنني نقلتها الى Staging Area ثم عملت لها Commit .

هذه السيناريوهات سوف تواجهها كثيراً عند العمل على Git وربما حدثت معك حتى عند عدم استخدامك لـ Git وربما حللت المشكلة بجهد ووقت . لنأخذ كل سيناريو ونضع الحلول له .

السيناريو الأول – واحد من اعمال Freelancer اليوم هي الطباعة , بمعنى تعطى لك مجموعة من الوثائق ويطلب منك طباعتها بملف وورد مقابل مبلغ من المال وحسب عدد الكلمات التي سوف تطبعها .

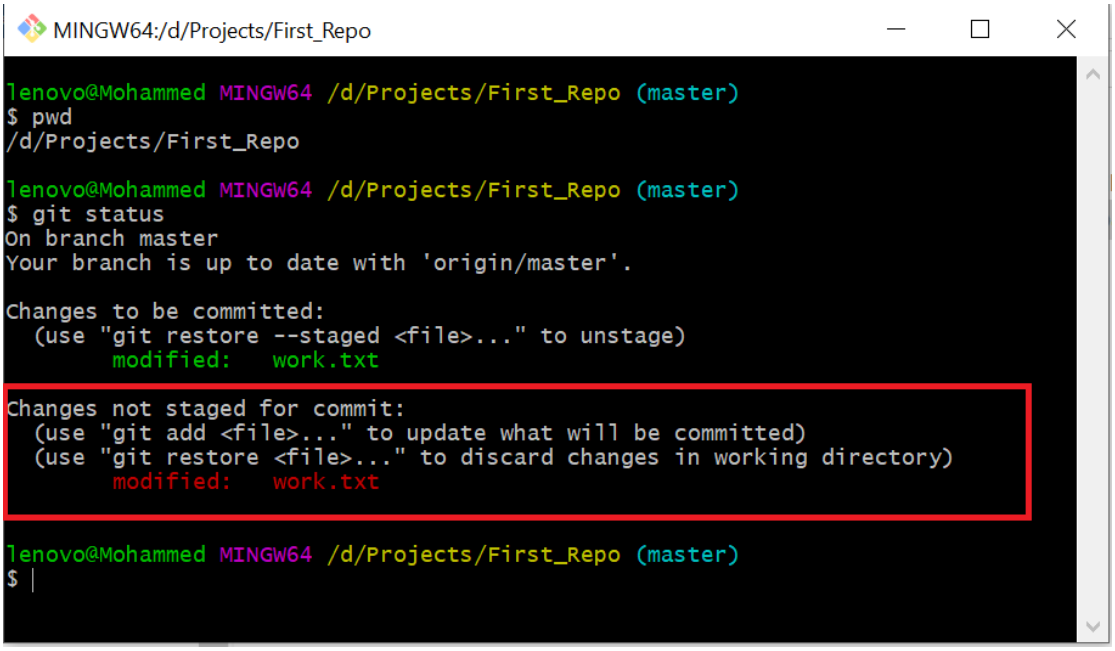
لنفترض طلب منك طباعة مئة وثيقة , وبعد يومين طبعت ما يقارب خمسين وثيقة حسب التسلسل , لكن صاحب العمل طلب منك حذف عشر وثائق (من صفحة 40 الى صفحة 50) واستبدالها بوثائق جديدة , وعملت ذلك لكن في اليوم التالي رجع عليك صاحب العمل وطلب أعادة طباعة الوثائق التي مسحتها ! في هذه الحالة وبدون Git فأنك سوف ترجع تطبعها من جديد اليس كذلك ؟

في مثل هذه الحالات من الوارد ان تحدث معك وليس بالطباعة فقط , وانما في تطوير مشروع برمجي أو تصميم صوري . ولكي تكون لديك مرونة بالعمل وتخطط لكل السيناريوهات التي ممكن ان تواجهها فعليك استخدام Git .

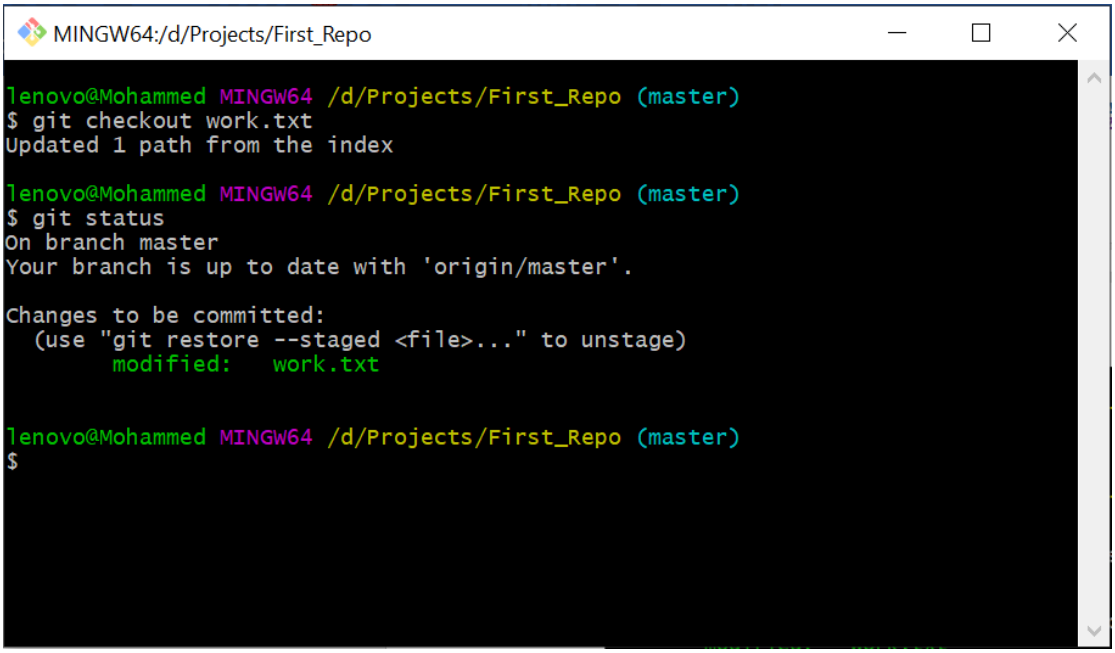
هذا السيناريو يفترض انك تعمل على Git ولكنك أبقيت التعديلات التي أجريتها في مرحلة Working بدون ما تنقلها الى Staging .

الان نطبقه بصورة عملية , اذهب الى مجلد المستودع على مشروعك وأفتح الملف work.txt , وأمسح ما بداخله , ثم أفتح Git Bash وأدخل على مسار المجلد كما في الصورة أدناه :

الان لنرى حالة المشروع من خلال الأمر git status



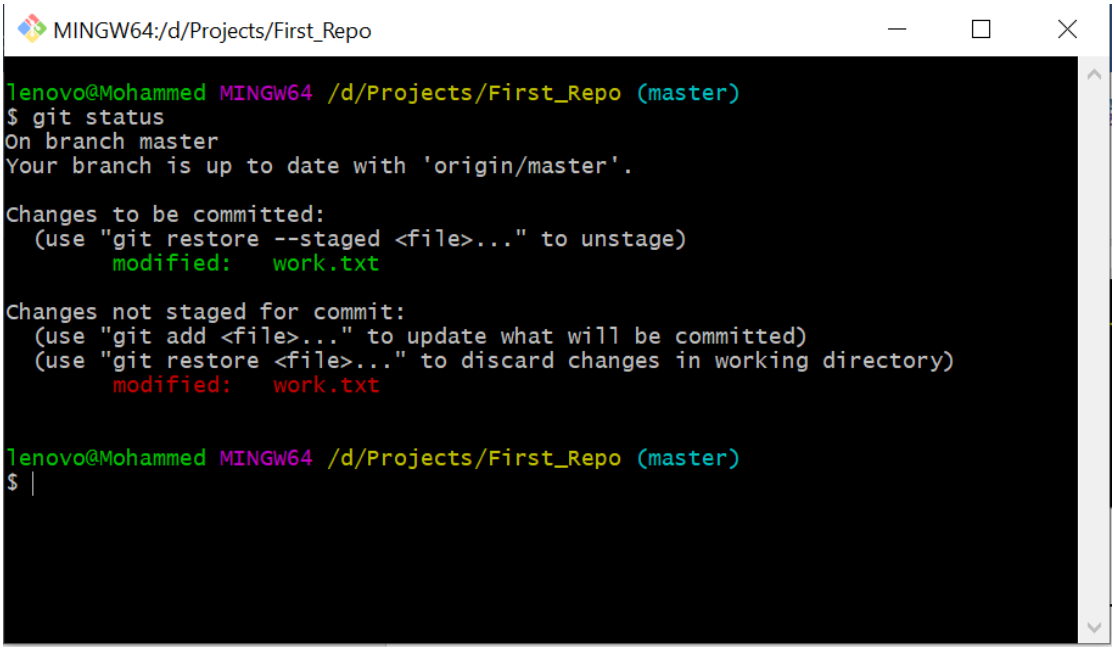
في مثل هذا فأن Git يعطيك مرونة في أسترجاع (Rrecove) الملف قبل اخر تعديل عليه , من خلال الامر git checkout filename



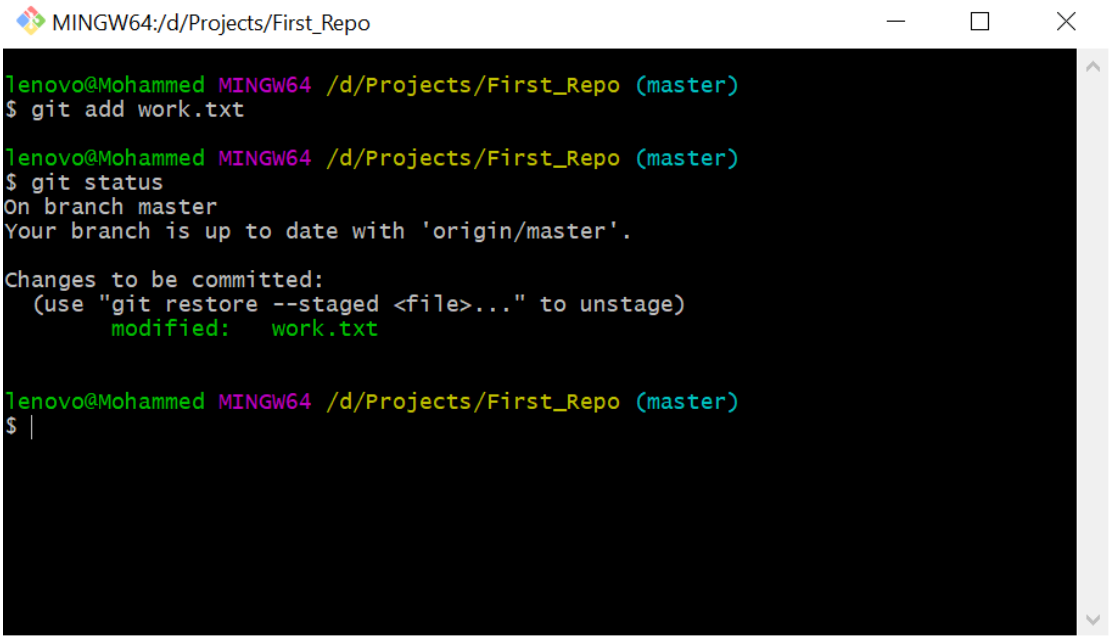
الان لو فتحت الملف work.txt سوف تجده أستعاد المعلومات التي حذفتها .

السـيناريو الثاني :هذا السـيناريو نفس الأول ولكن ماذا لو أننا نقلنا التعديلات الى مرحلة Staging ؟

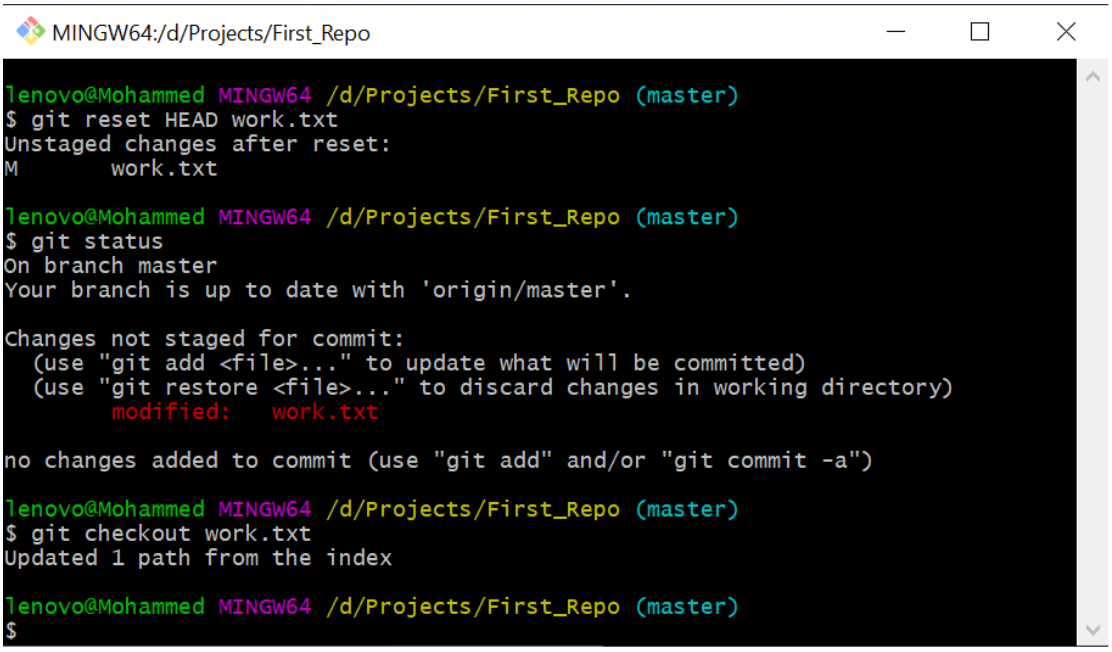
لنمسـح ما بداخل الملف work.txt ونحفظه ثم نذهب الى الGit Bash نسـتعرض حالة مشروعنا :



ثم ننقله الى مرحلة Staging من خلال الامر git add



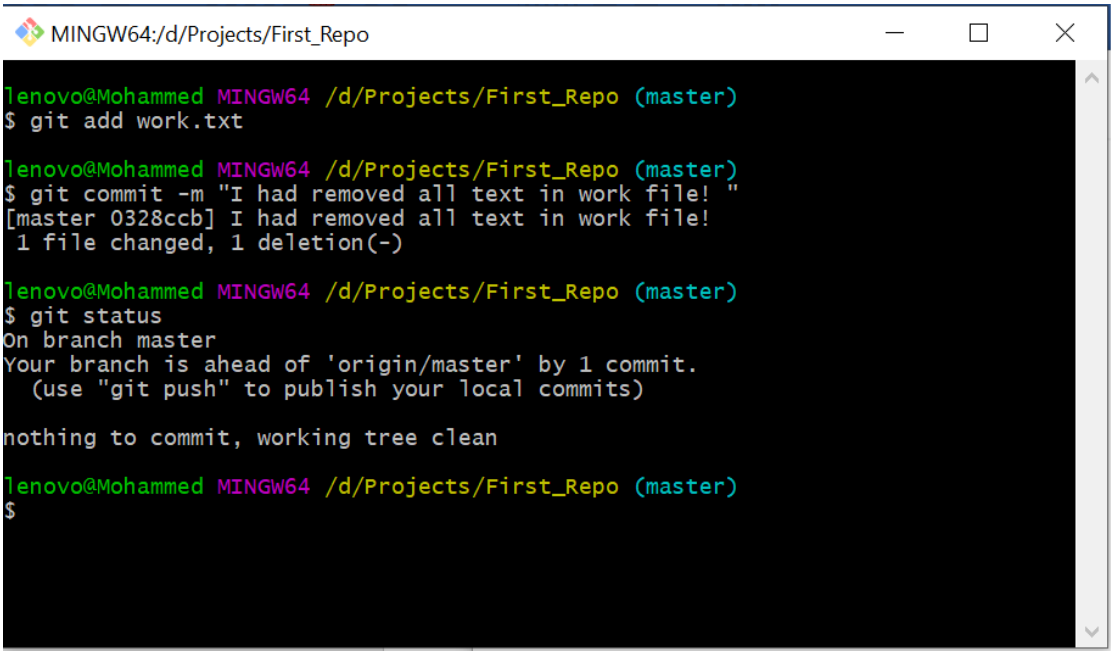
كما تلاحظ التعديلات صـــارت في مرحلة Staging , في مثل هذا الوضـــع لو أردت اســـتعادة محتويات الملف work.txt وهو في هذه المرحلة ! هنا Git يعطيك الأمر git reset HEAD filename ســـوف يرجعه الى مرحلة Working Directory ثم نطبق الامر الســـابق عليه لاسترجاعه قبل التعديلات الأخيرة (مسح المحتويات)



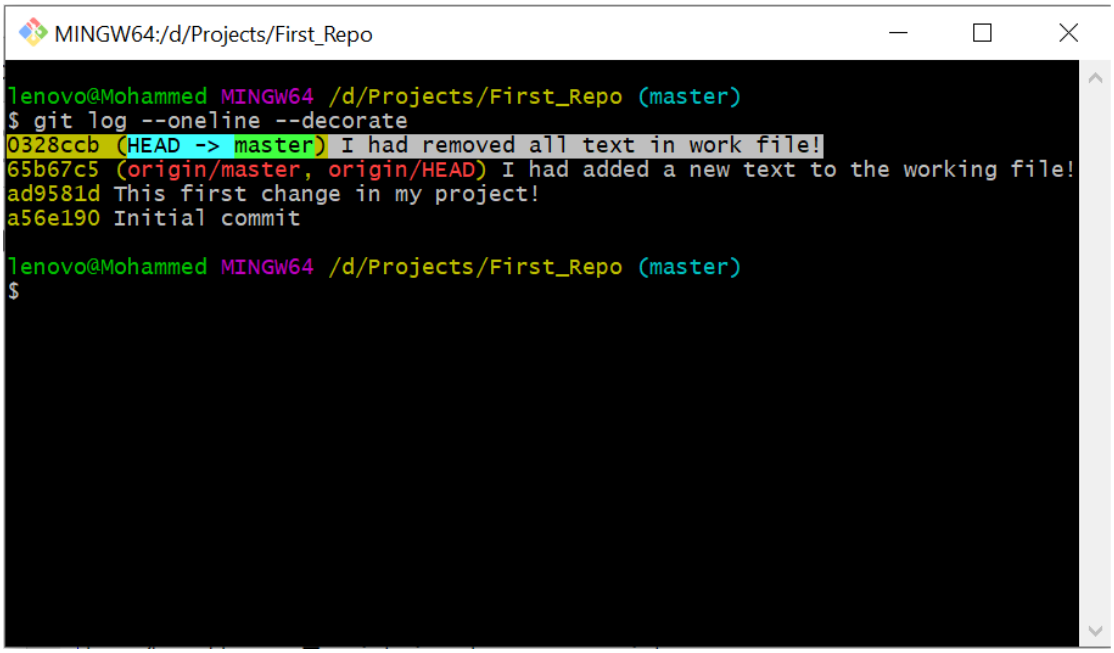
الان لو فتحت الملف work.txt ستجده استعاد المعلومات التي حذفت فيما سبق.

السيناريو الثالث : ماذا لو كان التعديلات في مرحلة Git Directory ؟

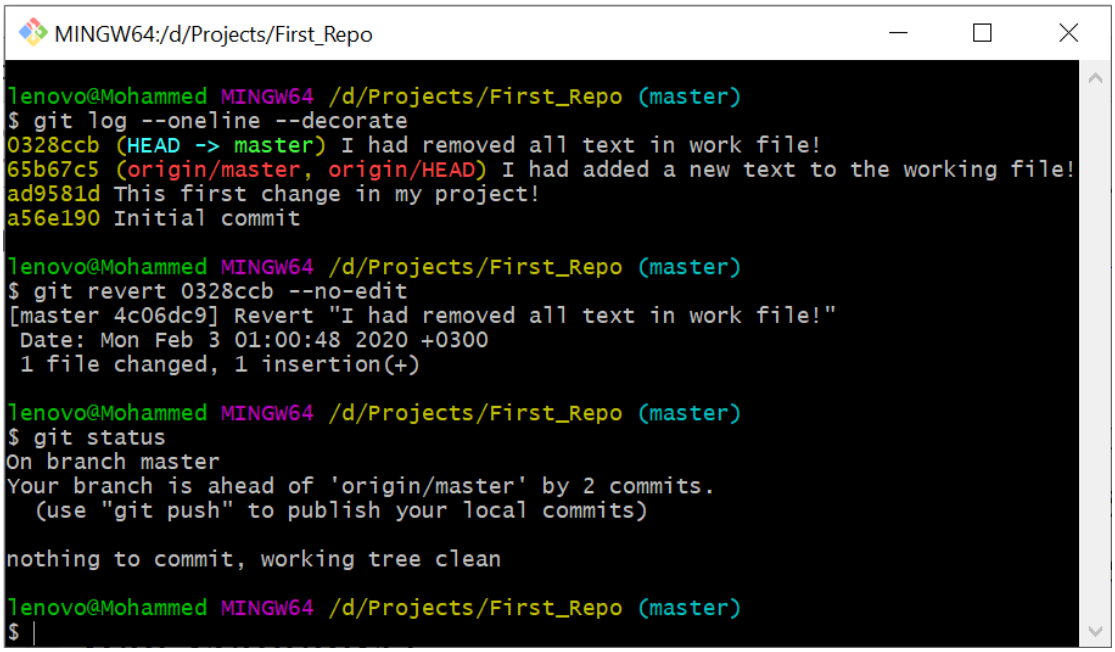
لنجرب نحذف ما بداخل الملف Work.txt وننقله الى مرحلة Staging ثم نعمل له Commit .



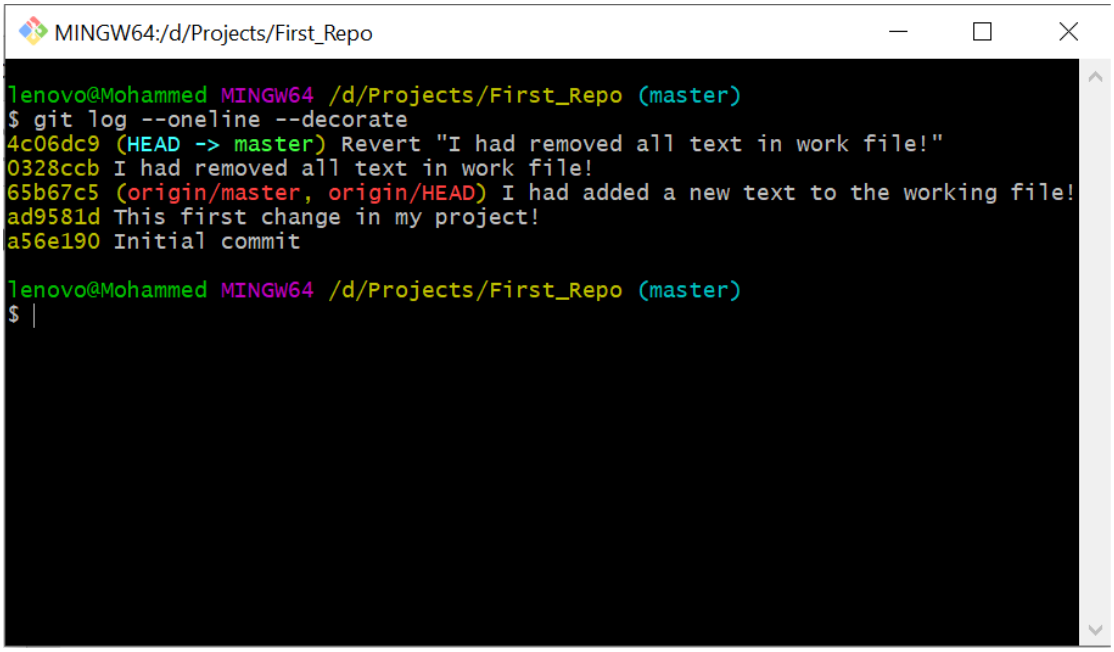
في مثل هذه الحالة سوف نستخدم الأمر git log لنشاهد التعديلات التي صارت :



لاحظ هذا الCommit الذي حذفنا فيه جميع محتويات الملف work.txt , نأخذ المعرف الخاص به (أعلاه هو 0328ccb) ثم نســتخدم الامر git revert كما في الصــورة أدناه حتى نرجعه الى مرحلة Staging :



طيب الان نستخدم الامر git log لنرى ماذا حصل :



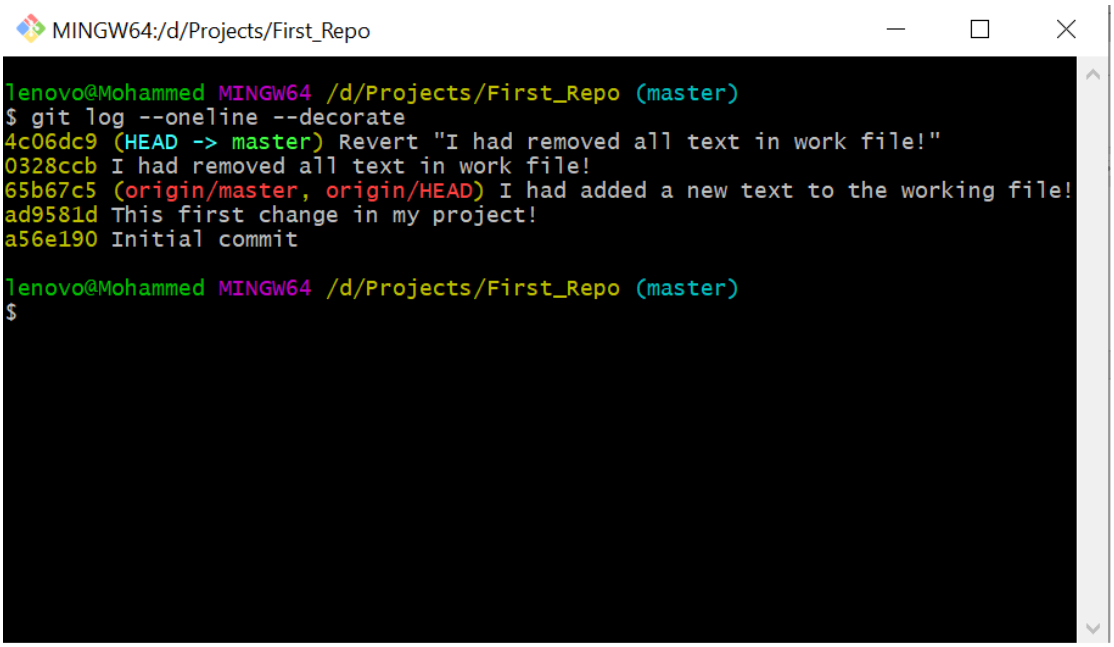
الان افتح الملف وسترى !

عمل أشارة (Tag) للتعديلات :

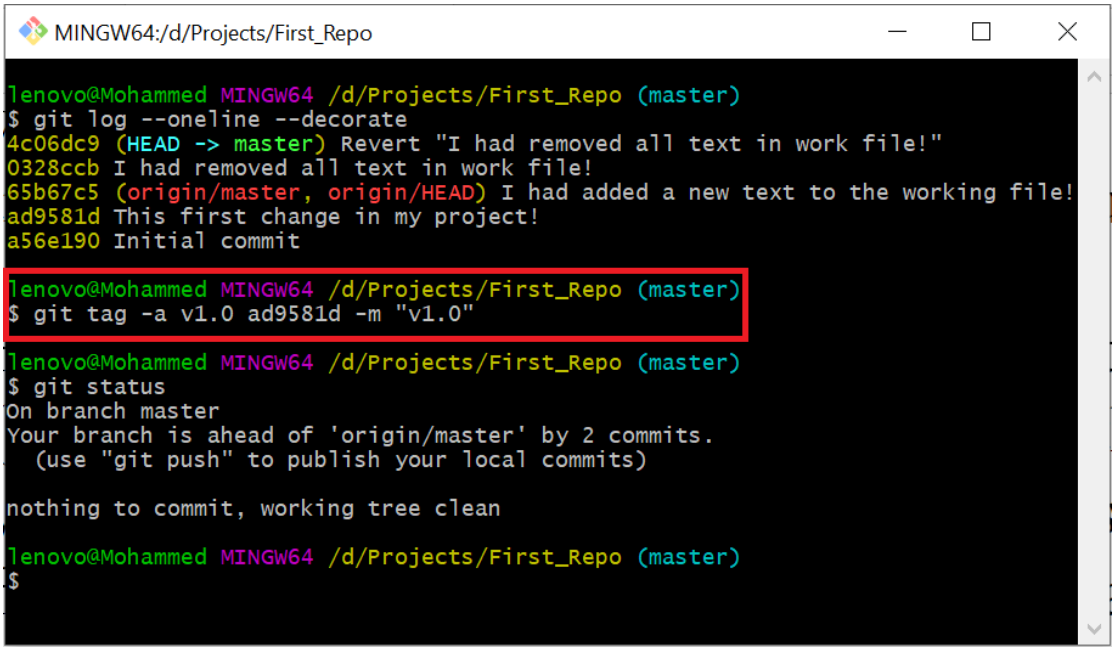
على بسيل الأفتراض انك عملت على مشروع برمجي , وتريد ان تضع إشارة (Tag) لكل تعديل تجريه حتى تستفاد منه في إصدارات التطبيق التي سوف تكملها .

الـ Git يساعدك على أعطاء أشاره (Tag) لكل تعديلات تجريها على المشروع .

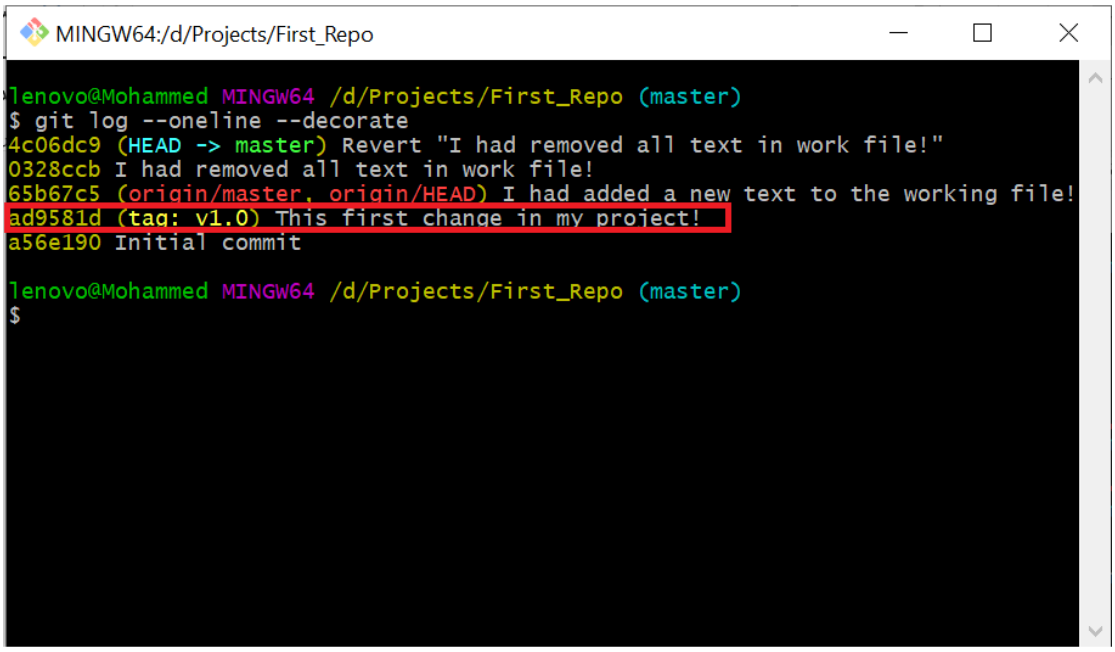
من خلال الامر git tag نستطيع عمل ذلك , لكن ! نحتاج الى المعرف (Commit id) حتى نسند تاغ (اصدار) للتعديلات, وفي هذه الحالة سوف نستخدم الأمر git log لاستخراج المعرف .



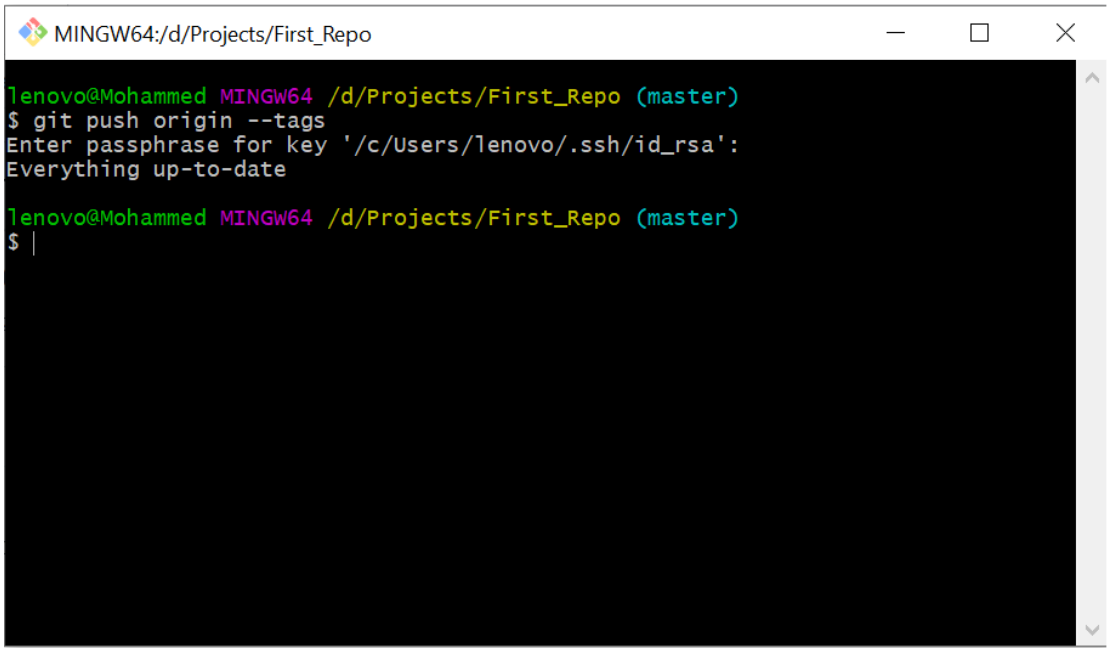
مثلما ترى في الشاشة أعلاه فانه لكل(commit) أعطى معرف (Id) باللون الأصفر , سوف أختار مثلا المعرف (ad9581d) ونص رسالته (This first change in my project!) سوف أعتبره الأصدار رقم (v1.0) . ولتنفيذ أمر الاشارة سوف يكون بهذا الشكل :



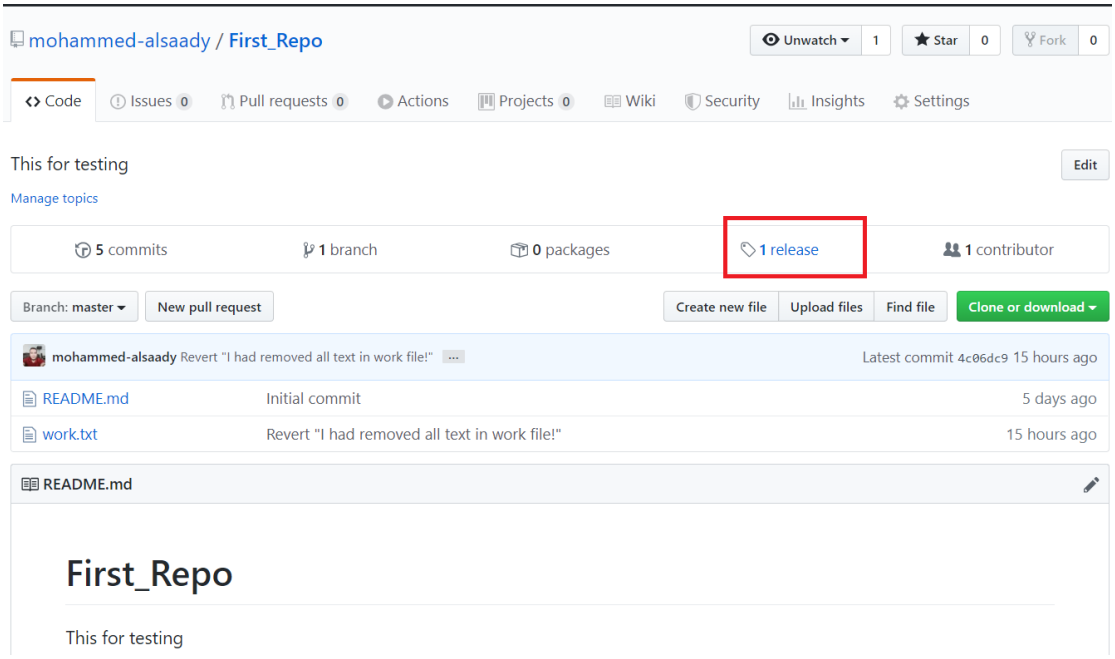
وحتى نقدر نشاهد الأشارة فأننا سوف نستخدم الامر git log :



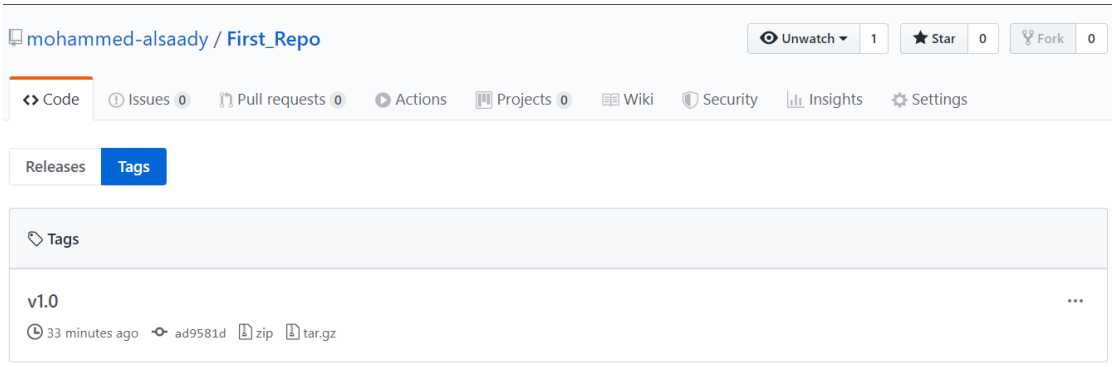
الان بأمكانك رفع التعديلات على الGithub والتي من ضمنها الاشارة من خلال الامر git push :



الان أذهب الى المستودع (First_Repo) على Github وشاهد .



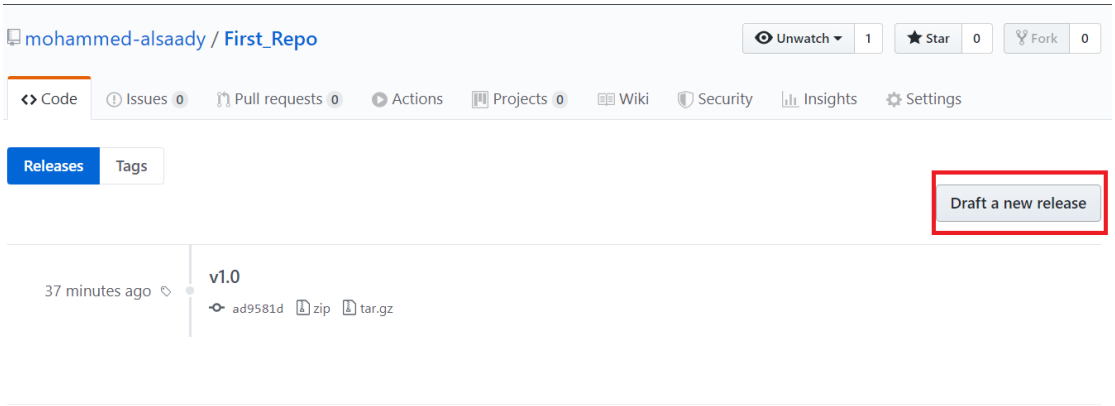
ومن ثم تنقر على الرابط release , بعدها نختار tag حتى يعرض لنا الأشارات التي عملناها لكل Commit .



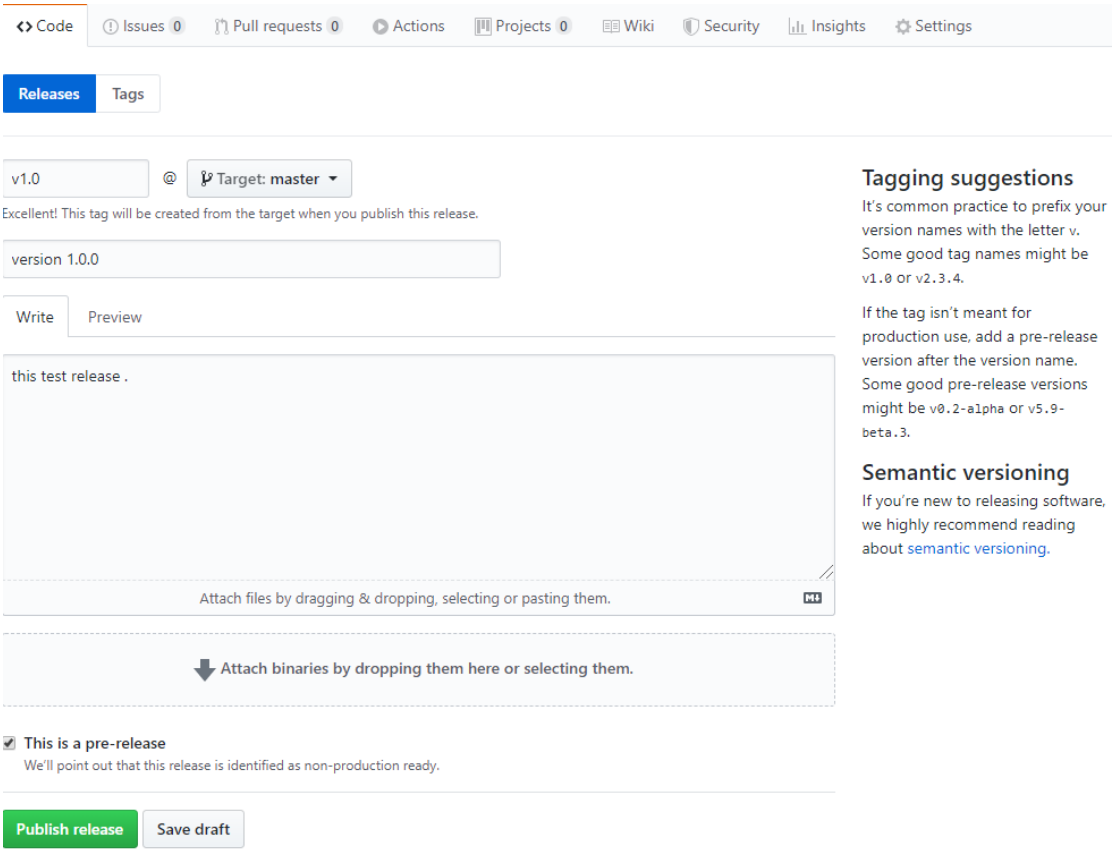
هنا السؤال المطروح , ما الفرق بين Tag و Release ؟ الفرق بسيط جدا , لنفترض أردت تطوير تطبيق موبايل , وهذا التطبيق أكملت التصميم الأول له فهنا تقدر تعطيه (Tag) ثم أكملت قاعدة بياناته وهنا أيضا تقدر تعطيه كذلك (Tag) وبعدها أكملت الشفرات وكذلك تسند له (Tag) , بعد التاغ الثالث صار التطبيق جاهز وبأمكانك تحزيمه (Package) (تحويله من شفرات الى ملف تنصيبي) وبالتالي فأنك سوف تعطيه أصدار مثلا (الإصدار رقم 1.0) وفي الـ Github الأصدار يعتمد على tag أي بمعنى انك يجب ان تعطي تاغ ثم اصدار .

كيف نعمل أصدار (Release) :

نفتح الGithub ونذهب الى المستودع ثم نختار Release وبعدها ننقر على Draft a new release .



سوف تظهر لنا الصفحة التالية ومنها نختار رقم tag و (Branch) الرئيسي , وعنوان الإصدار ووصف للمشروع .كذلك تقدر تضيف ملفات Binary اذا موجودة , وان كان مشروعك ليس جاهز تماما فتقدر تأشر على (pre-release) ثم تنقر على Publish Release :



بعدها سيكون الأصدار جاهز :