گیت یکی از بهترین ابزارها برای ورژن بندی(کنترل ورژن) است.

و میتوانید پروژه را پابلیک کنید و افراد دیگه آن پروژه را تغییر دهند.

حالا گیت هاپ فضایی است که میتوانیم در هر جایی که باشیم پروژه ها را مشاهده کنیم.

در دسکتاپ راست کلیک کنید و بر روی git bush here کلیک کنید.

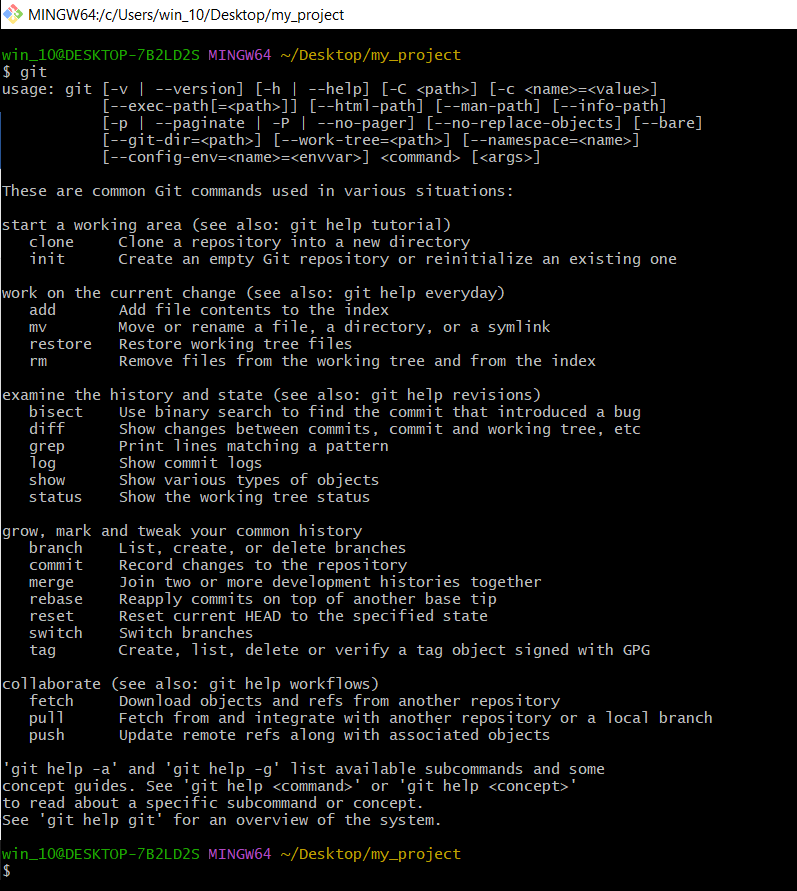
برای دیدن ورژن گیت:

git --version

را در ترمینال باز شده میزنیم.

۲:

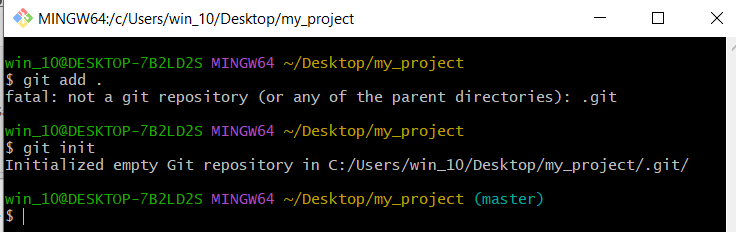
وارد یک فایل میشویم کلیک راست میکنیم و گزینه git bush here را میزنیم تا وارد ترمینال شویم: با دستور git میتوانیم تمام دستورات را با توضیح مشاهده کنیم:



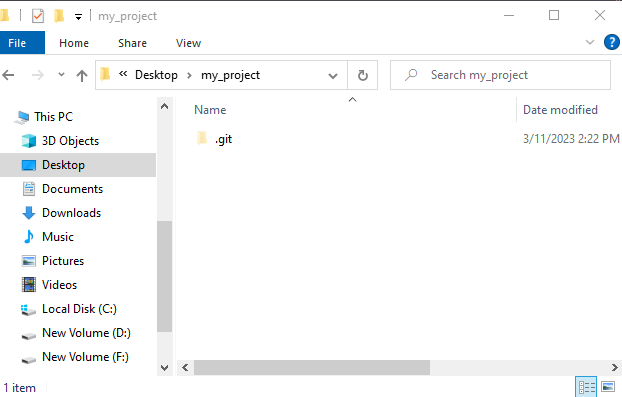
تقریبا با ۹۰ درصد دستورات کار میکنیم.

نیاز به repository داریم که به فارسی به معنی مخزن هست.

برای اضافه کردن repository از دستور زیر استفاده میکنیم:

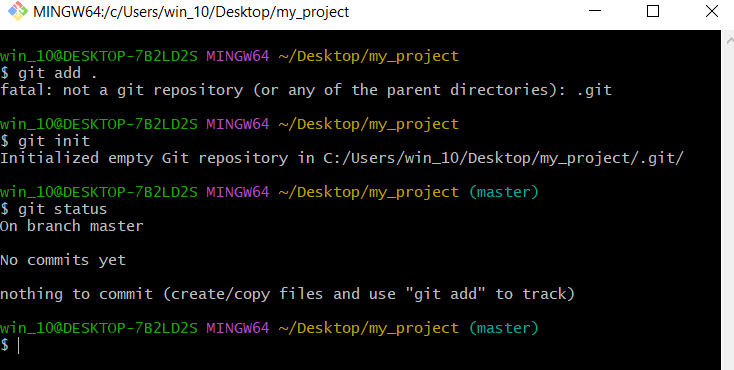


همان طور که مشاهده میکنید یک repository ایجاد شده است برای اینکه فایل آن را نیز مشاهده کنید یک فایل باید ایجاد شده باشد، مانند تصویر زیر:

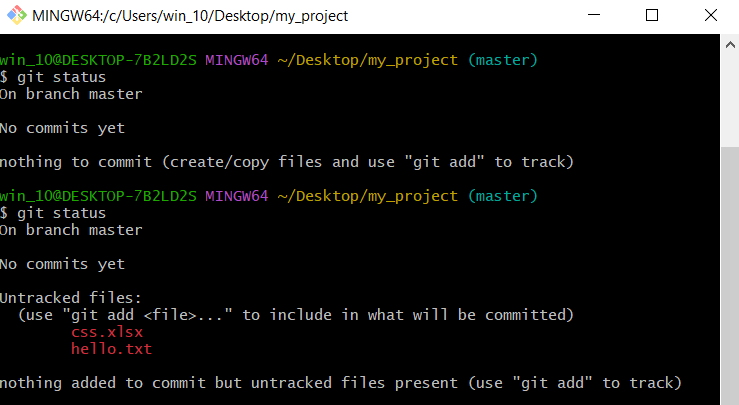


یک فایلhidden .

برای اینکه قدم ها را نگاه کنیم و ببینیم در چه قسمتی هستیم از دستور status استفاده میکنیم.



در پاسخ گفته است که فایل (commits) جدیدی قرار نگرفته است حالا فایلی را قرار میدهیم و دوباره دستور را مینویسیم:

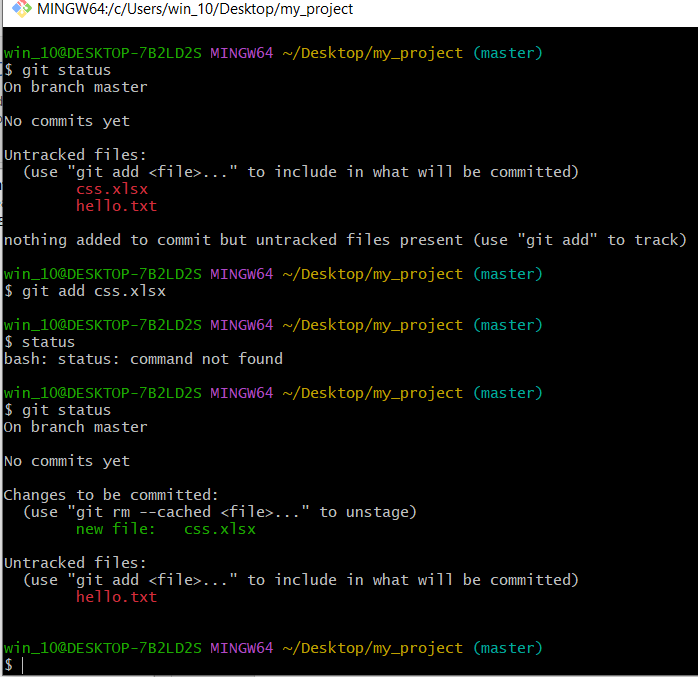


همان طور که مشاهده میکنید دو فایل جدید با نام نمایش داده شده است به رنگ قرمز.

حالا در ادامه گفته است که این فایل ها را add کنید دلیل add کردن هم به این صورت است : چون گیت یک معماری ۳ لایه دارد ، مثال: فایلی که میخواهیم در گیت قرار دهیم را در قسمت راست قرار میدهیم،‌ گیت را در قسمت چپ بین فایل که میخواهد ارسال شود و گیت یک جایی قرار دارد به نام استیج ، به این معنی که کارهایی که میخواهیم انجام دهیم اول در استیج قرار میگیرد یکی از دلایلی که مستقیما وارد repository نشود این است که اگر اشکالی وجود داشت بتوانیم راحت برگردونیم، اول وارد استیج میشه اگر مشکلی داشت خیلی راحت میتونیم برگردونیم و مشکل را حل کنیم.

برمیگردیم به توضیح عکس آخر:

میگه اینجا یکسری فایل هایی دارید که هنوز در استیجتون ثبت نشده برای اینکه این فایل ها را add کنید باید با دستور git add آن را ادد کنیم، باید اسم فایل را نیز بعد از دستور قرار دهیم:

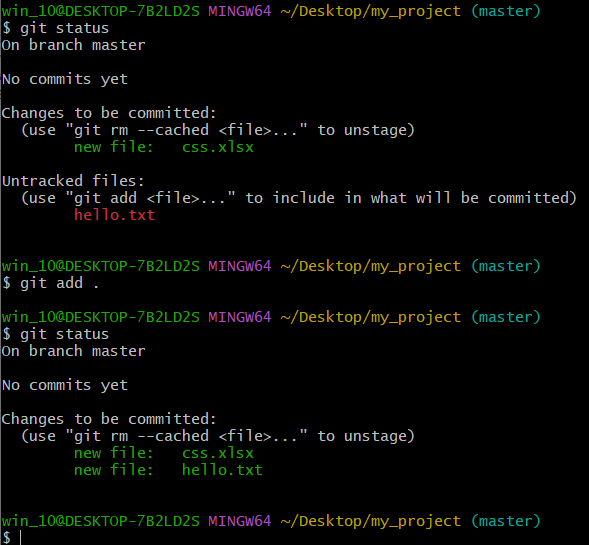


بعد از قرار دادن دستور git add (name file) دستور git status را قرار دادیم تا وضعیت فایل ها را مشاهده کنیم. همان طور که میبینیم فایل css.xlsx قرار گرفته به معنی دیگر داخل استیج ذخیره شده است، و بالای آن نوشته شده است که شما باید آن را commit کنید تا قرارش دهید داخل repository و در ادامه نوشته شده است که فایل هایی وجود دارد که آن را به استیجتون وارد نکردید به این منظور که در repository وجود ندارد و باید آن را داخل repository ثبتش کنید و برای انجام این کار (ثبت فایل ها داخل repository) باید وارد استیج شود و بعد از آن با دستور commit کلا بره داخل repository .

برای قرار دادن تمامی فایل ها از دستور زیر استفاده میکنیم و دیگر اسم فایل ها را نمینویسیم:

Git add .

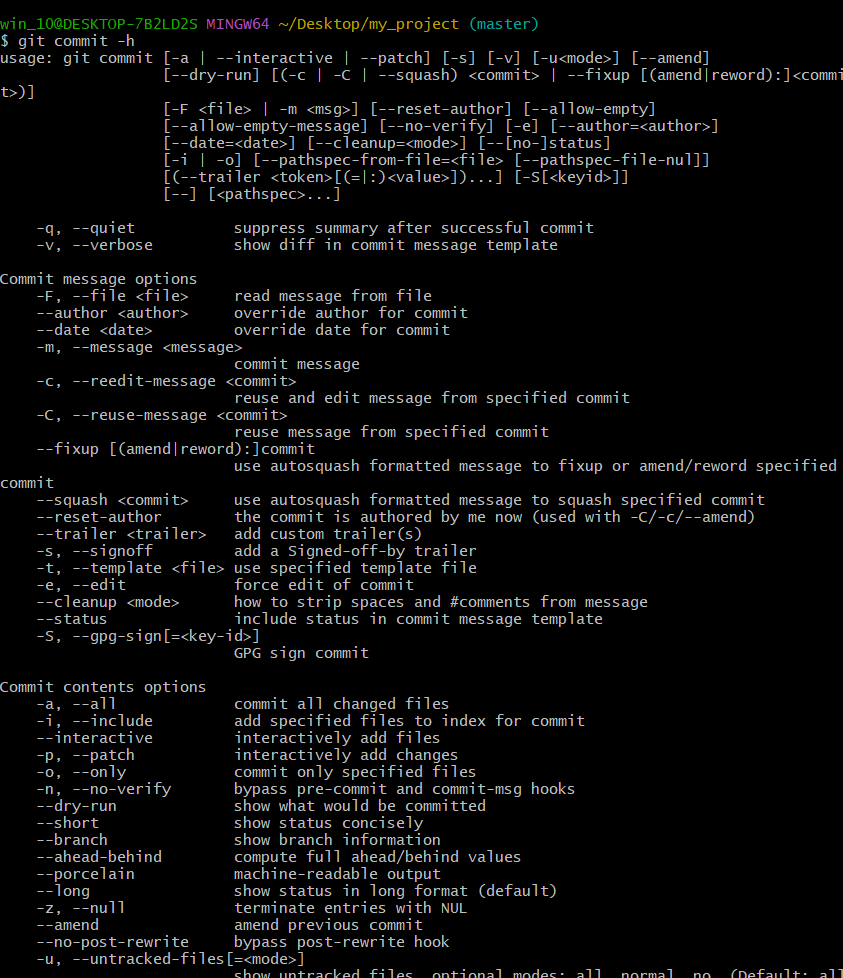
تمامی فایل ها را به استیج ارسال میکند:

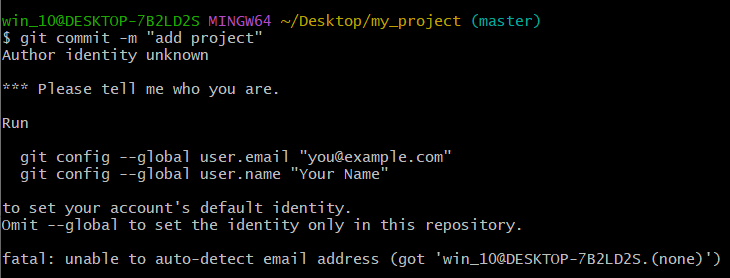


تمام فایل ها اضافه شد.

بعد از این باید commit کنیم برای این کار دستور زیر را میزنیم و وارد قسمت help دستور میشویم:

Git commit –h

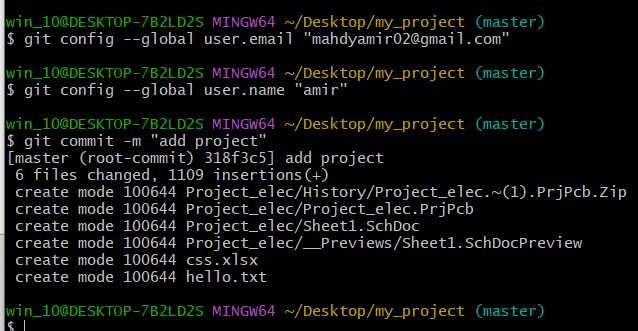


Git commit به تنها کافی نیست و میگوید باید یک پیام نیز داشته باشید که باید به صورت زیر این کار را انجام دهیم نکته بعدی این است که در اینجا باید توضیحاتی را قرار دهیم توضیحاتی که در مورد پروژه است، اجرا دستور: 

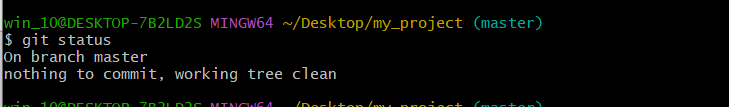
اگر ایمیل خود را اد کرده بودیم تمام فایل ها را کامیت میکرد

حالا برای شناسایی ما ایمیل و رمز ما مورد نیاز است.

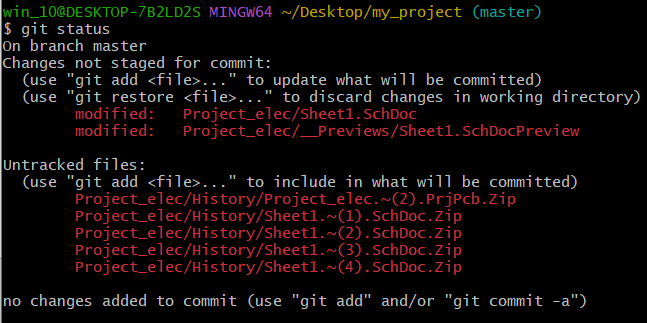
بعد از قرار دادن ایمیل و نام کاربری میتوانیم فایل ها را قرار دهیم:



حالا فایل ها قرار گرفته است و میتوانیم با دستور git status مشاهده کنیم که چه اتفاقی افتاده است:



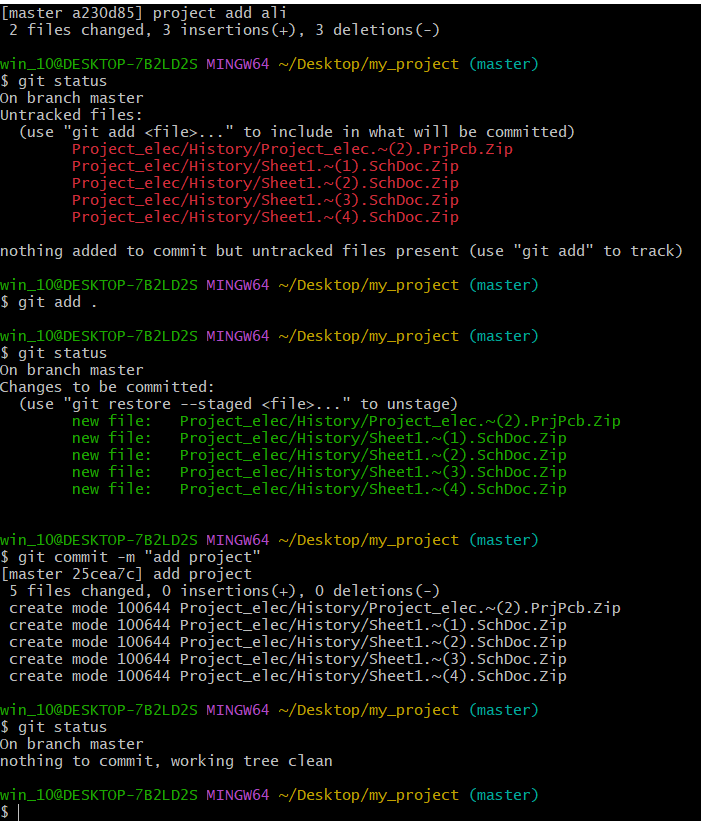
تغییری در فایل آلتیوم میدهیم و دستور git status را میزنیم و میبینیم چه اتفاقی افتاده است:



همان طور که مشاهده میکنید میبینید که گفته تغییری در فایل ایجاد شده و حالا این باید ثبت شود داخل repository .

برای اینکه این تغییر را اضافه کنیم میتوانیم از دستور git add . استفاده کنیم و بعد commit را انجام دهیم یا میتوانیم add و commit را باهمدیگر انجام دهیم.

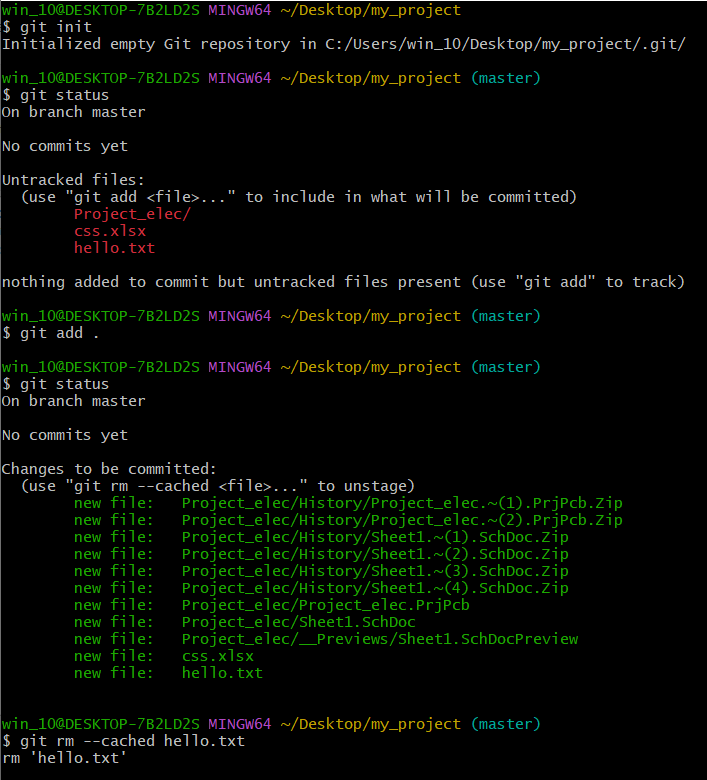
که در دستور زیر مشاهده میکنید.(پیام حتما باید ارسال شود).



۳:

برای اولین بار اضافه کردن فایل ها باید از دستور git add استفاده کنیم.

برای در آوردن فایل از استیج از دستور زیر استفاده میکنیم:



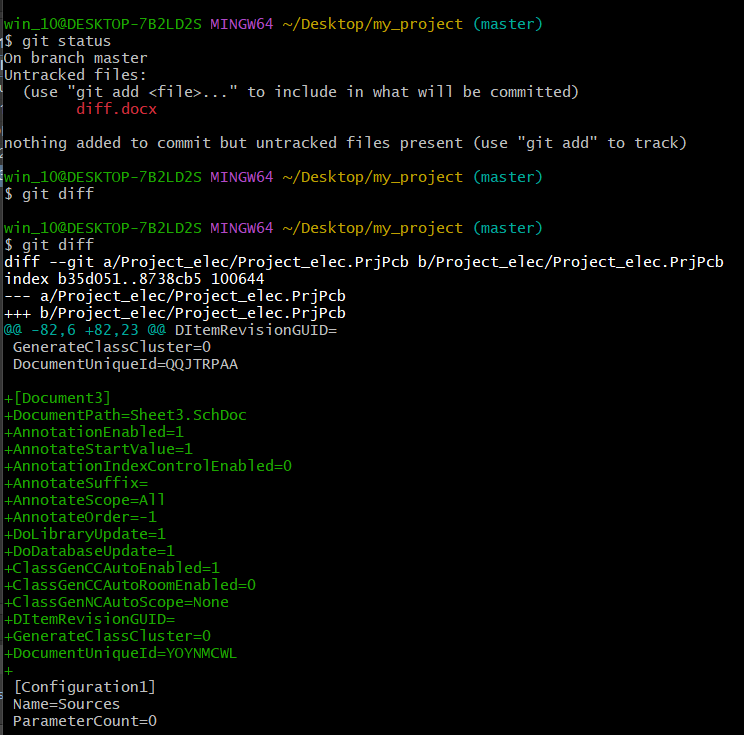
برای حذف کردن تمام فایل ها از دستور زیر استفاده میکنیم:

Git rm –cached –r .

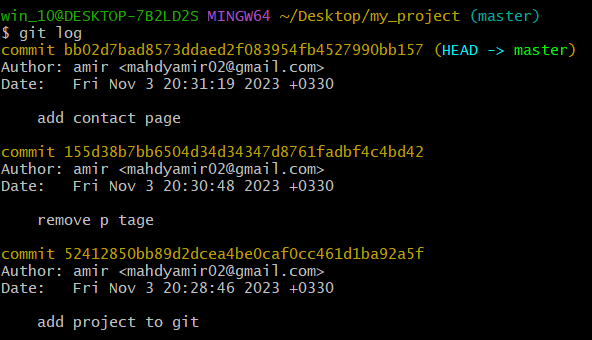
۲ دستور جدید:

یک دستور برای مشاهده کردن تغییر دستورات است :

Git diff



کار بعدی مشاهده کردن commit ها است که در اصل همان commit هایی هستند که داخل repository ذخیره شدند:



هر commit یک عدد خاص دارد و دو تا commit نمیتوانید پیدا کنید که این عدد برابر باشد.

حالا فرض میکنیم این commit را میخواهیم داخل git قرار دهیم.

اطلاعات اول id commit است

اطلاعات دوم در رابطه با کسی است که آن commit را انجام میدهد که اسم و آدرس جیمیل قرار دارد.

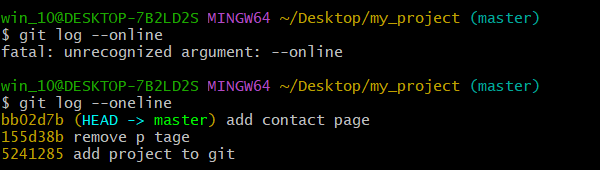
با استفاده از این دستور میتوانید تمامی دستورات مورد نیاز خود را مشاهده کنید:

Git log –help

صفحه اینترنتی باز میشود که تمامی دستورات مربوط به log را انجام میدهد.

یک سری از پیکر بندی اطلاعات و log را مشاهده میکنیم:

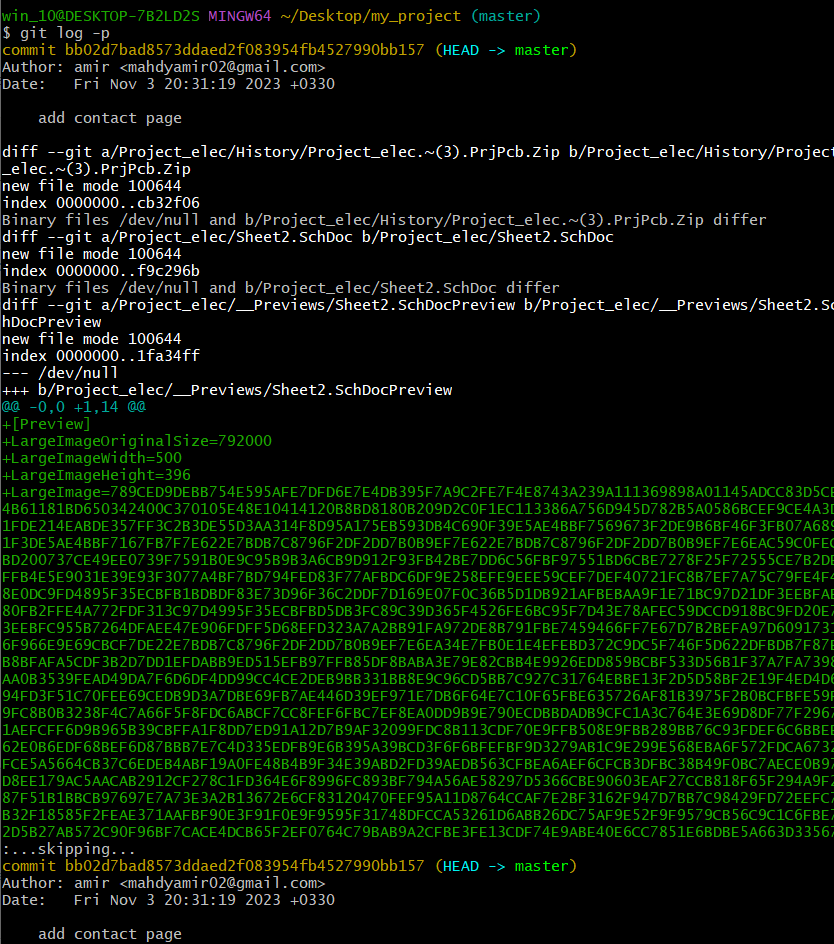
Git log –oneline



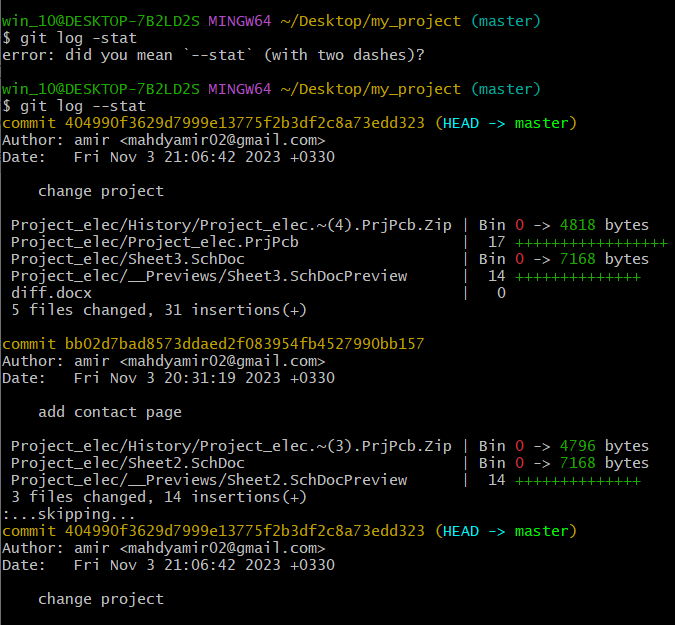
یک مقدار از id commit را و بعد پیام ها را نمایش میدهد.

تغییراتی که داخل آن commit قرارگرفته را نمایش دهد از دستور زیر استفاده میکنیم:

Git log –p



دستور بعدی git log –stat است که اطلاعات به صورت زیر نمایش داده میشود:



۳:

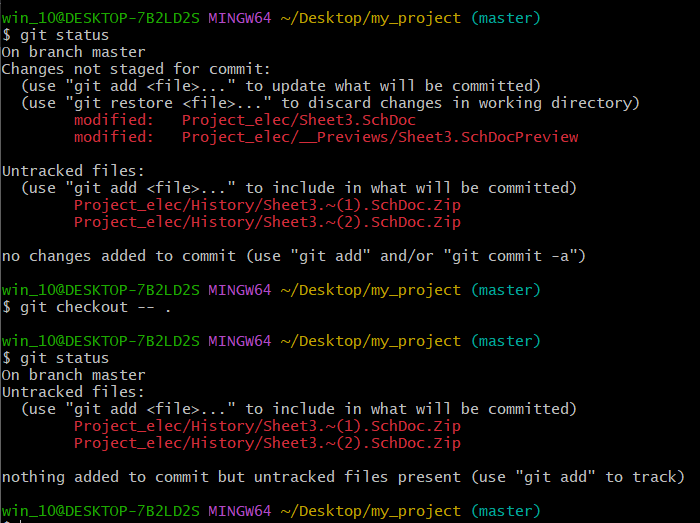
وقتی commit اشتباه انجام شده چه کار کنیم و چگونه تغییر دهیم.

یک سری تغییر در فایل ها قرار بدهیم به طور مثال فایل آلتیوم را تغییر بدهیم بعد برای اضافه کردن آن به استیج چه کار کنیم.

میخواهیم فایل ما برگرده به آخرین حالتی که داشته است از دستور زیر استفاده میکنیم

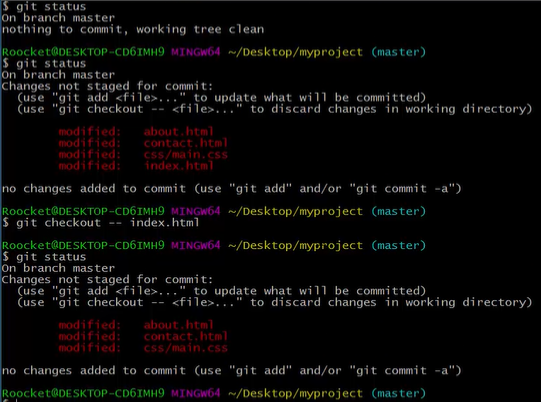
Git checkout -- .

دستورات:



همان طور که مشاهده میکنید تغییرات انجام شده در فایل شماتیک آلتیوم انجام نشد و شماتیک به حالت اولیه قرار گرفته شده است.

برای اینکه فقط فایل مورد نظر به حالت اول برگردد اسم آن را مینویسیم به طور مثال:

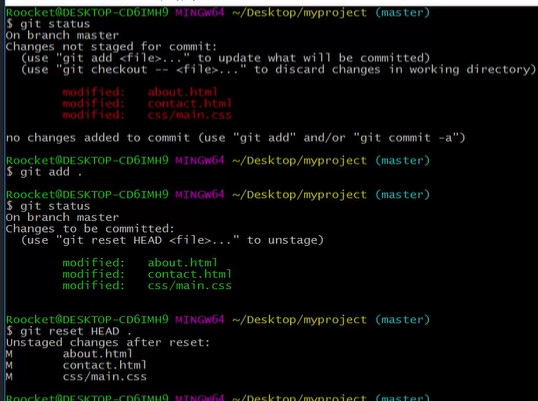


از دستور git checkout –(name file) استفاده میکنیم.

حالا اگر دستور git add. را اجرا کردیم اما متوجه شدیم که نیازی به این تغییرات نداریم چه کاری انجام دهیم :

از دستور git reset HEAD . را انجام میدهیم .

بعد از این دستور این پیغام را میدهد:



میگوید آن چیزی که وارد استیج کردید (در لایه میانی قرار دادید) برگشت داده شد.

و اگر دوباره دستور git status را بزنید میتوانید ببینید که تغییرات هنوز اعمال نشده است.

۵: دو قسمت branch و merge

Branch:

به معنی شاخه است.

مخزن را ایجاد میکنیم و یک شاخه اصلی به صورت خودکار ایجاد میشود که به آن master میگوییم، داخل master برنامه خود را توسعه میدهیم ولی در یک جایی میخواهیم نسخه ۲ را بدهیم بنابراین برای اینکه هم نسخه ۲ را بخواهیم هم نسخه ۱ را بخواهیم توسعه بدهیم.

یک مثال دیگر شما داخل یک شرکت کار میکنید و یک پروژه تعریف میشه و این پروژه بین چند نفر تقسیم میشود هر کدام از افراد کار ها را انجام میدهند و آن را وصل میکنند به پروژه اصلی به این معنی که هرکدام از آن افراد یک شاخه میشوند و وقتی به اتمام رسید آن را merge (ترکیبش) میکند به پروژه اصلی .

اولین دستور:

Git branch -- help

یا میتوانید از دستور زیر به جای دستور بالا استفاده کنید:

Git branch -p

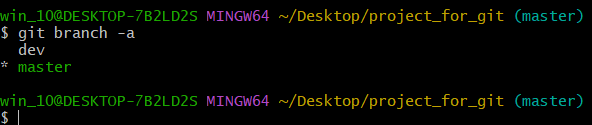
حالا برای ساخت branch چه کار کنم از دستور زیر استفاده میکنیم:

Git branch dev

حالا ما در شاخه master قرار داریم برای اینکه وارد شاخه ای بشویم که الان ساختیم از دستور زیر استفاده میکنیم:

با دستور پایین مشاهده میکنیم که چه شاخه هایی وجود دارد:

Git branch -a



برای سوییچ کردن بین branch ها از دستور زیر استفاده میکنیم:

Git checkout dev

برای حذف شاخه از دستور زیر استفاده میکنیم:

Git branch -d dev

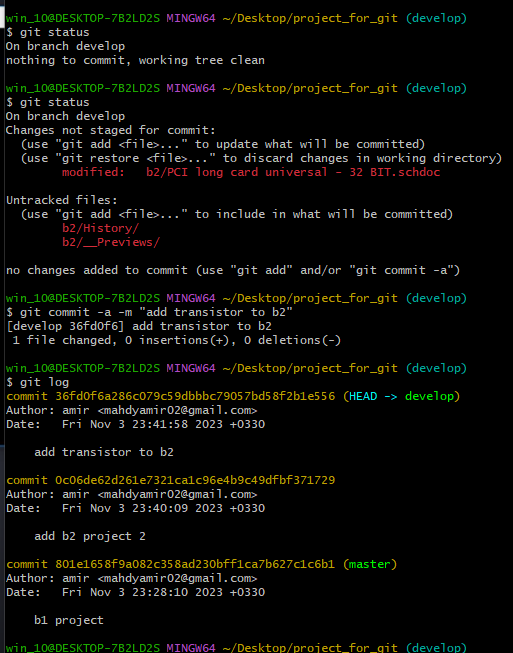
یک فایل جدید میسازیم و در آلتیوم تغییراتی ایجاد میکنیم.

1. git status

در قسمت بعد آن فایل را commit میکنیم:

1. git commit -a -m “add transistor to b2”

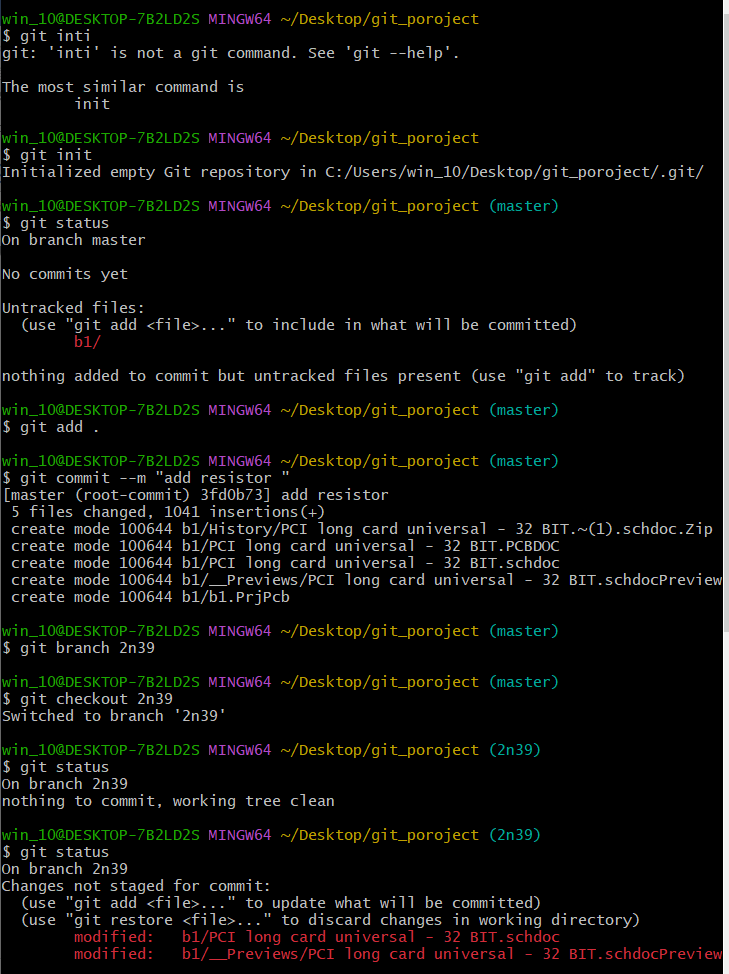
بعد پروژه قرار میگیرد برای مشاهده تغییرات انجام شده از دستور git log استفاده میکنیم تا تغییرات را مشاهده کنیم:(دقیقا به صورت تصویر زیر)

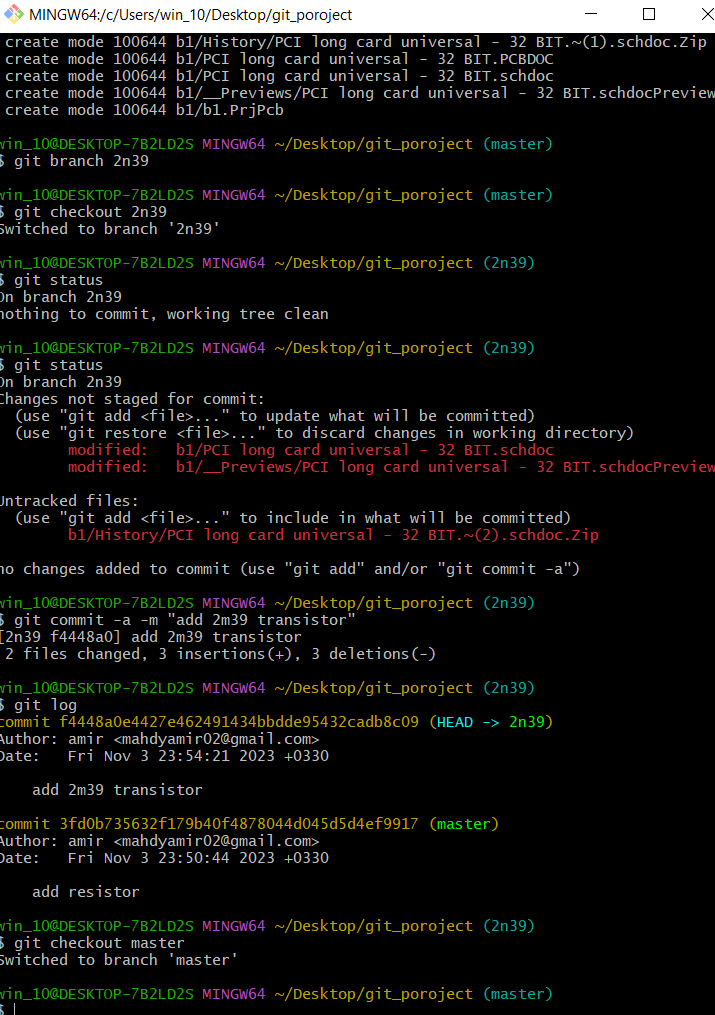


بعد از شاخه فرعی develop خارج میشویم:

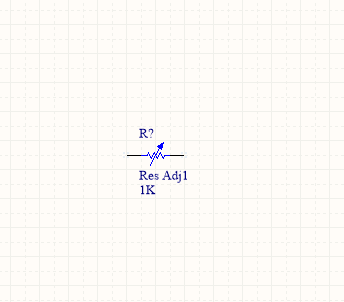
git checkout master

مراحل انجام و ساخت و تغییر شاخه با استفاده از فایل آلتیوم:

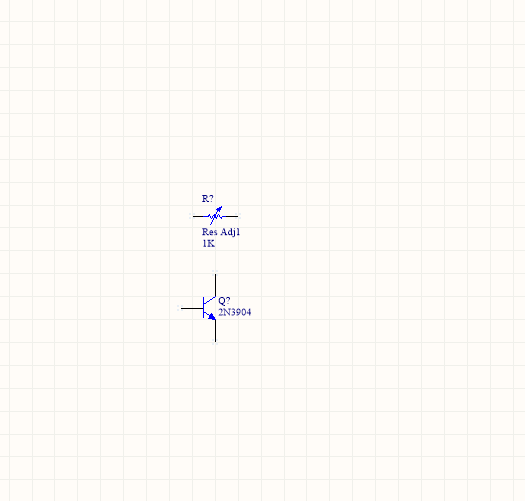




نمایش در آلتیوم: زمانی که در master قرار داریم:



زمانی که در شاخه 2n39 قرار داریم:



همان طور که مشاهده میکنید تغییرات انجام شده است.

حالا میرویم سراغ استفاده کردن از دستور merge به این صورت از این دستور استفاده میکنیم:

این دستور برای این است که تغییراتی که در شاخه فرعی انجام دادی را به شاخه اصلی انتقال بدهید یا هر تغییراتی را به یک شاخه دیگر انتقال بدهید:



برای اینکه راحت تر متوجه بشویم که چه اتفاقی افتاده است در git log از دستور زیر استفاده میکنیم:

git log --groph

for this file for master

data for 2n39