تمرین اول

گزارش

Git برای پروژه مدیریت لیست کارها (To-Do List Manager)

دستور Git Status

دستور `git status` استفاده می‌شود. این دستور نشان می‌دهد Git برای مشاهده وضعیت فعلی پروژه در مخزن شدن آماده هستند یا خیر commit کدام فایل‌ها اصلاح شده، اضافه شده یا حذف شده‌اند و آیا این تغییرات برای.

```
(dev) bardia@Bardias-MacBook-Pro-3 ~/D/SE_lab_1 (feat/add_task_manager)> git status
On branch feat/add_task_manager
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   task_manager.py
```

دستور Git Commit

دستور `git commit` برای ذخیره تغییرات در مخزن استفاده می‌شود. با استفاده از گزینه `-m`، کاربر می‌تواند پیام را شامل توضیحی درباره تغییراتی که اعمال شده‌اند، درج کند commit.

دستور Git Pull

دستور `git pull` برای دریافت و ادغام تغییرات از یک مخزن از راه دور به مخزن محلی استفاده می‌شود. این دستور معمولاً برای به‌روزرسانی مخزن محلی با تغییراتی که توسط اعضای دیگر تیم ایجاد شده‌اند، استفاده می‌شود.

```
(dev) bardia@Bardias-MacBook-Pro-3 ~/D/SE_lab_1 (master)> git pull
Updating 2db25de..9e68101
Fast-forward
 task_manager.py | 24 ++++++++++++++++++--------
 1 file changed, 16 insertions(+), 8 deletions(-)
```

دستور Git Push

دستور `git push` برای آپلود تغییرات مخزن محلی به یک مخزن از راه دور استفاده می‌شود. این دستور معمولاً برای به `origin` آن باید `push` اشتراک گذاری تغییراتی که توسط کاربر با سایر اعضای تیم ایجاد شده‌اند، استفاده می‌شود. برای شاخه نیز در نظر گرفته شود

دستور Git Merge

دستور `git merge` برای ترکیب تغییرات از دو یا چند شاخه به یک شاخه استفاده می‌شود. این دستور برای ادغام تغییرات اعضای دیگر تیم با شاخه محلی کاربر استفاده می‌شود.

دستور Git Checkout

دستور `git checkout` برای تغییر شاخه‌ها یا commit های مختلف در مخزن استفاده می‌شود. این دستور معمولاً برای تغییر به یک شاخه مختلف برای کار بر روی یک قابلیت جدید یا رفع باگ استفاده می‌شود.

```
(dev) bardia@Bardias-MacBook-Pro-3 ~/D/SE_lab_1 (feat/task_manager_str)> git checkout master
Switched to branch 'master'
```

دستور Git Branch

برای مدیریت شاخه‌ها در مخزن استفاده می‌شود. این دستور برای ایجاد، تغییر نام یا حذف `git branch` دستور شاخه‌ها در مخزن استفاده می‌شود.

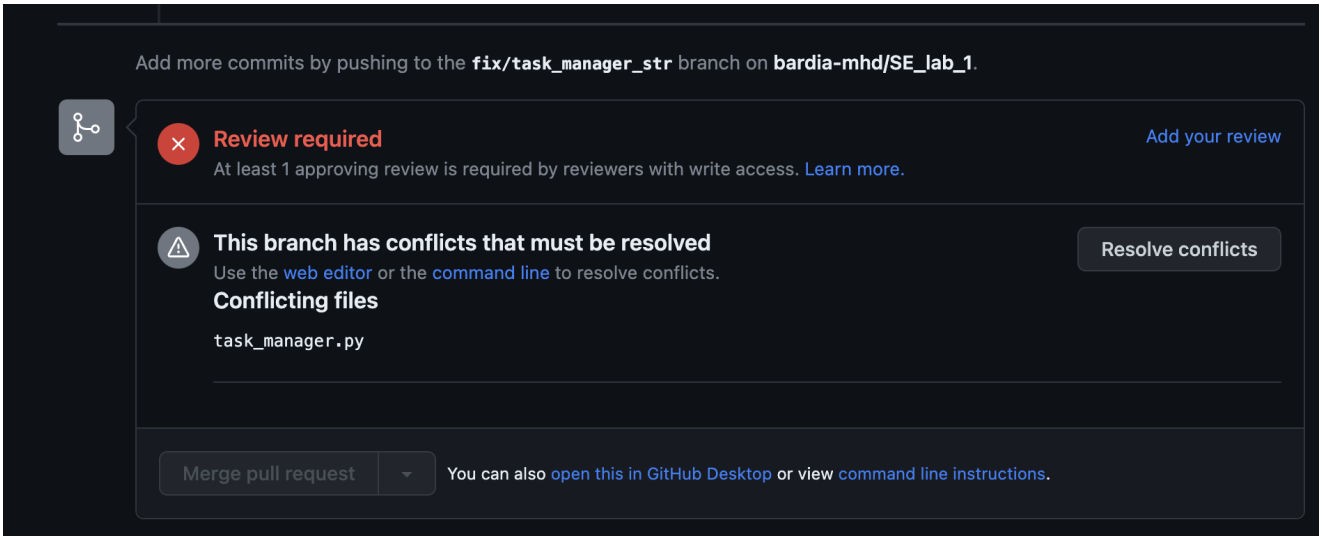
به طور خلاصه روال انجام به صورت زیر است:

1. مخزن را از مخزن اصلی به دستگاه محلی کلون می کنیم، با استفاده از دستور `git clone`.
2. یک شاخه جدید برای قابلیت جدید یا رفع باگ ایجاد مکنیم با استفاده از دستور `git checkout -b`.
3. آماده commit تغییرات را برای انجام `git add` تغییرات را در فایل‌های پروژه اعمال کرده و با استفاده از دستور می کنیم.
4. می کنیم commit تغییرات را با استفاده از دستور `git commit -m`.
5. تغییرات از مخزن اصلی را برای به‌روزرسانی مخزن محلی دریافت می کنیم با استفاده از دستور `git pull`.
6. تداخل را برطرف می کنیم، pull در صورت بروز هرگونه تداخل در هنگام.
7. تغییرات را به مخزن اصلی ارسال می کنیم با استفاده از دستور `git push`.
8. برای ادغام تغییرات با شاخه اصلی ایجاد می کنیم pull یک درخواست.
9. را بررسی و تأیید می کنیم pull درخواست.
10. تغییرات را با شاخه اصلی ادغام می کنیم با استفاده از دستور `git merge`.

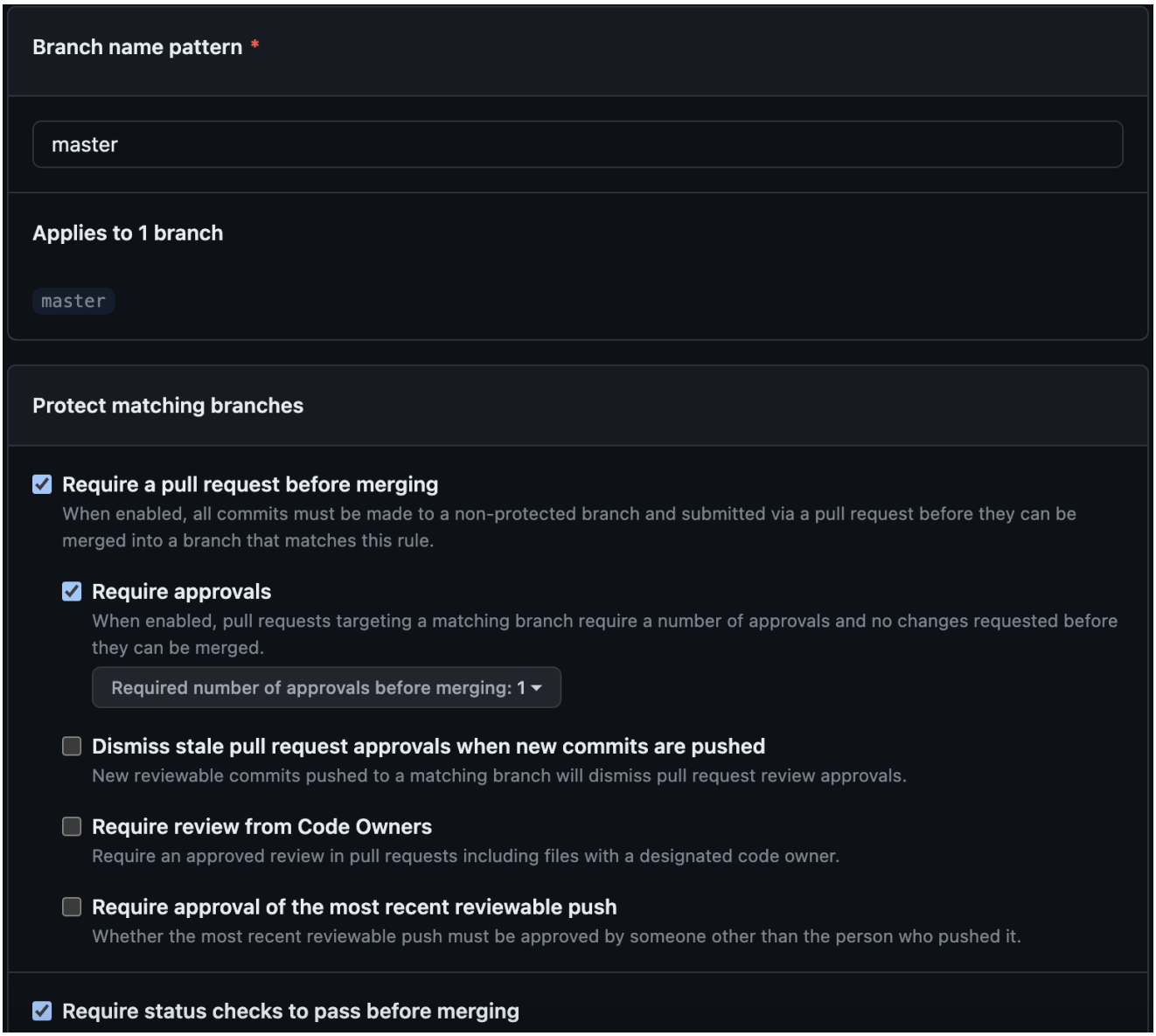
رفع (Merge Conflicts)

می‌شوند و تغییرات متفاوتی در هر (merge) با یکدیگر ادغام Git در یک پروژهدر (branch) هنگامی که دو یا چند شاخه Git رخ دهد. این به معنی این است که (merge conflicts) شاخه ایجاد شده باشد، ممکن است تداخل‌های ادغام نمی‌تواند به طور خودکار تغییرات را با هم دوام بخشد و برای ادغام نیاز به دستکاری دستی دارد.

کرده و برای ادغام تغییرات از commit بهتر است به‌صورت منظم تغییرات خود را، merge conflicts برای جلوگیری از کنیم. همچنین برای رفع protected شاخه‌های جداگانه و با نام معنادار استفاده کنید و شاخه اصلی را merge conflicts که به‌صورت گرافیکی و با استفاده از رابط کاربری آسان‌تر این کار را انجام می‌دهیم GitHub ها از محیط گرافیکی.



بررسی شود و pull request کردن شاخه اصلی: (که در اینجا حتما نیاز است قبل از درخواست ادغام protected نحوه ی جلوگیری می شود master از تغییر مستقیم روی):



سوالات

چیست؟ چه اطلاعاتی در آن ذخیره می‌شود؟ با چه دستوری ساخته می‌شود؟ git. پوشه‌ی

ایجاد می شود. این پوشه شامل Git در یک مخزن «git init» یک پوشه مخفی است که هنگام اجرای دستور «.git» پوشه و سایر اطلاعات مربوط به مخزن remote آدرس مخزن، commit، تمام اطلاعات لازم برای کنترل نسخه، از جمله تاریخچه است template ها و فایل های refs/heads، refs/tag دارای زیردایرکتوری هایی برای «.git» است. پوشه.

چیست؟ atomic pull-request و atomic commit بودن در atomic منظور از

است. این نوع کامیت، تقسیم subtask اتمی به کامیتی گفته می‌شود که نشان‌دهنده یک تغییر یا commit، در گیت ناپذیر است، یعنی می تواند به طور کامل موفق شود یا به طور کامل شکست بخورد، اما نمی تواند تا حدی موفق شود اشاره دارد که نشان دهنده یک pull-request به یک atomic pull-request، یا تا حدی شکست بخورد. به طور مشابه pull-request به‌عنوان تغییرات اتمی آسان‌تر و ایمن‌تر است، زیرا می‌توانیم یک pull-request تغییر واحد است. استفاده از را به طور کامل برگردانیم pull-request، ادغام کنیم و اگر مشکلی پیش آمد commit را به‌عنوان یک request.

را بیان کنید rebase و merge و pull و fetch تفاوت دستورهای.

- دستور `git fetch`: آخرین تغییرات را از یک مخزن remote به مخزن local اما working directory را تغییر نمی دهد اعمال می کند.

- دستور `git pull`: تغییرات را از یک مخزن remote در حالت پیش‌فرض، در شاخه فعلی اعمال می‌کند. `git pull` است. با اسفاده از `git merge FETCH_HEAD` و به دنبال آن `git fetch` نسخه کوتاه شده دو دستور اجرا می کند `git merge` را به جای `git rebase`، دستور `--rebase` فلگ.

- دستور `git merge`: تغییرات از شاخه های مختلف را در شاخه فعلی تلفیق می کند.

- دستور `git rebase`: commit های یک، جدید که دو شاخه را با هم ترکیب می کند به جای ایجاد یک commit شاخه را بر روی دیگری منتقل می کند.

به طور خودکار تغییرات را در شاخه فعلی ادغام `git pull` این است که `git fetch` و `git pull` تفاوت اصلی بین می کند، در حالی که `git fetch` فقط تغییرات را در مخزن محلی شما دانلود می کند. تفاوت بین `git merge` و `git rebase` جدید ایجاد می کند که تغییرات شاخه های مختلف را ترکیب می کند، commit یک `git merge` در این است که کامیت های یک شاخه را روی شاخه دیگر منتقل می کند `git rebase` در حالی که.

تفاوت چهار دستور reset و revert و restore را بیان کنید.

- دستور git reset: این دستور به شما اجازه می‌دهد تا تغییرات استیج شده را بازگردانید و یا به یک کامیت خاص اجرا شود که هر کدام تاثیرات متفاوتی --hard و --mixed، --soft می‌تواند در سه حالت reset. در تاریخچه بروید برای بازگشت به یک نقطهٔ قبلی در تاریخچهٔ مخزن و reset بر روی مخزن شما دارند. به طور کلی، استفاده از حذف تغییرات نامطلوب استفاده می‌شود.

- دستور git revert: این دستور یک کامیت را انتخاب می‌کند و تغییرات آن را خنثی می‌کند. در واقع، با استفاده از شما یک کامیت جدید می‌سازید که تغییرات کامیت قبلی را لغو می‌کند. بنابراین، تاریخچهٔ مخزن شما revert حفظ می‌شود و شما می‌توانید به سادگی به نقطهٔ قبل بازگردید.

- دستور git restore: دستور restore برای بازیابی یک فایل خاص در مخزن Git استفاده می‌شود. با استفاده از این دستور، می‌توانید تغییراتی که در یک فایل انجام شده است، را بازیابی کنید. در این commit فایل را به وضعیت source-- وجود دارد. گزینه staged-- و source-- دستور، گزینه‌های مختلفی مانند بازیابی می‌کند، به این معنی (staging area) فایل را به وضعیت تسک ها staged-- قبلی بازیابی می‌کند. گزینه که تغییراتی که به تسک ها اضافه شده بودند، حذف می‌شوند و فایل به حالت قبلی بازگردانده می‌شود. این restore معرفی شد. با استفاده از reset و checkout دستور در نسخه 2.23 گیت به عنوان جایگزین دستورات شما می‌توانید فایل‌های خود را به نسخهٔ ذخیره شده در استیج یا گذشته بازگردانید. این دستور برای بازگشت فایل‌های تغییر یافته به نسخهٔ قبل استفاده می‌شود.

چه کاری را انجام می‌دهد؟ stash چیست؟ دستور stage منظور از

نام یک مرحلهٔ میانی است که در فرایند کامیت استفاده می‌شود. این مرحله اجازه می‌دهد تا index یا stage، در گیت تغییرات خود را برای کامیت آماده کنید و فقط بخش‌های خاصی از تغییرات اعمال شده از آخرین کامیت را کامیت کنید. نویسندگان گیت تصمیم گرفته‌اند که این مرحله را قابل مشاهده و پایدار کنند، در حالی که در سایر سیستم‌های کنترل نسخه، این مرحله بخش زودگذری از فرایند کامیت است.

یک راه برای ذخیرهٔ تغییرات کامیت نشده (شامل هم تغییرات استیج شده و هم تغییرات استیج نشده) دستور git stash استفاده کنید وقتی می‌خواهید به git stash برای استفادهٔ بعدی و بازگشت آن‌ها از کپی کار شماست. شما می‌توانید از شاخه‌ای دیگر سوئیچ کنید یا عملیات دیگر گیت را بدون کامیت تغییرات فعلی خود انجام دهید. برای ذخیرهٔ تغییرات خود،

را اجرا کنید و پیام دریافت خواهید کرد که تغییرات شما در شاخهٔ شما ذخیره شده‌اند. شما می‌توانید git stash می‌توانید خود را دوباره اعمال کنید stash ،وقتی آماده هستید.

مفهوم snapshot؟ به چه معناست

در گیت snapshot است که شامل تمامی فایل‌ها، پوشه‌ها و تنظیمات مخزن (repository) به معنای یک نسخه از مخزن در گیت به معنای تاریخچه یک کامیت است که شامل تمامی snapshot ،در یک زمان خاص می‌باشد. به طور دقیق‌تر از مخزن را نشان snapshot تغییراتی است که در آن کامیت اعمال شده‌اند. به طور کلی، هر کامیت در گیت یک می‌دهد.

در گیت با یک آیدی شناخته می‌شود که به آن snapshot هر commit hash یا SHA-1 hash می‌گویند. این شناسه یکتا مورد نظر خود دسترسی پیدا کنید و تغییراتی که در آن کامیت اعمال snapshot به شما اجازه می‌دهد تا به راحتی به شده است را مشاهده کنید.

به معنای یک کامیت است که شامل تمامی تغییراتی است که در آن کامیت اعمال snapshot ،به طور خلاصه، در گیت ها امکان مشاهده تاریخچه تغییرات در مخزن snapshot با یک شناسه یکتا شناخته می‌شود. این snapshot شده‌اند و هر را به شما می‌دهند Git.