ब्रान्च व्यवस्थापन (Branch - Management)  
अब तपाईंले केही ब्रान्चहरू बनाइसकेपछि, मर्ज गरिसकेपछि र मेटाइसकेपछि, आउनुहोस् ब्रान्च व्यवस्थापनका उपकरणहरूको बारेमा जानौं, जुन तपाईंले ब्रान्चहरू प्रायः प्रयोग गर्न थालेपछि उपयोगी हुन्छन्।

git branch कमाण्डले ब्रान्च सिर्जना र मेटाउनु बाहेक पनि धेरै काम गर्न सक्छ। यदि तपाईंले यसलाई कुनै पनि argument बिना चलाउनुभयो भने, यो तपाईंका वर्तमान ब्रान्चहरूको साधारण सूची देखाउँछ:

$ git branch

iss53

\* master

Testing

**ध्यान दिनुहोस्**: \* चिन्हले master ब्रान्चलाई अगाडि राखेको छ। यसले तपाईंले हाल कुन ब्रान्चमा काम गरिरहनु भएको छ (HEAD जुन ब्रान्चमा सूचित गर्छ) भन्ने देखाउँछ। यसको मतलब यदि तपाईंले यस बिन्दुमा commit गर्नुभयो भने, master ब्रान्च तपाईंको नयाँ कामसँग अघि बढ्नेछ।

यदि तपाईंले प्रत्येक ब्रान्चको अन्तिम commit हेर्न चाहनुहुन्छ भने, तपाईं git branch -v चलाउन सक्नुहुन्छ:

$ git branch -v

iss53 93b412c Fix javascript issue

\* master 7a98805 Merge branch 'iss53'

testing 782fd34 Add scott to the author list in the readme

--merged र --no-merged जस्ता उपयोगी विकल्पहरूले यो सूचीलाई फिल्टर गर्न सक्छन्। यी विकल्पहरूले ती ब्रान्चहरू देखाउँछन् जुन तपाईंले अहिलेको ब्रान्चमा मर्ज गरिसक्नु भएको छ वा मर्ज नगरेको छ।

उदाहरणका लागि, यदि तपाईंले हेर्न चाहनुहुन्छ कि कुन ब्रान्चहरू पहिले नै मर्ज भइसकेका छन् भने, तपाईं git branch --merged चलाउन सक्नुहुन्छ:

$ git branch --merged

iss53

\* master

**किनभने तपाईंले iss53 लाई पहिल्यै मर्ज गरिसक्नुभएको छ, यो सूचीमा देखिन्छ।**  
सूचीमा भएका ब्रान्चहरू, जसको अगाडि \* छैन, ती प्रायः सुरक्षित रूपमा मेटाउन सकिन्छ git branch -d कमाण्डको प्रयोग गरेर। तपाईंले ती ब्रान्चहरूको काम पहिले नै अर्को ब्रान्चमा समावेश गरिसक्नुभएको हुन्छ, त्यसैले तपाईंले केही पनि गुमाउनुहुने छैन।

यदि तपाईं हेर्न चाहनुहुन्छ कि कुन ब्रान्चहरूमा अझै काम मर्ज हुन बाँकी छ, तपाईं git branch --no-merged चलाउन सक्नुहुन्छ:

$ git branch --no-merged

testing

**यसले तपाईको अर्को ब्रान्च देखाउँछ।**  
किनभने यसमा काम बाँकी छ र मर्ज गरिएको छैन, यसलाई git branch -d प्रयोग गरेर मेटाउने प्रयास गर्दा यो असफल हुन्छ:

$ git branch -d testing

error: The branch 'testing' is not fully merged.

If you are sure you want to delete it, run 'git branch -D testing'.

यदि तपाईं साँच्चै उक्त ब्रान्च मेटाउन चाहनुहुन्छ र त्यसमा भएको काम गुमाउन तयार हुनुहुन्छ भने, तपाईं यसलाई -D फ्ल्यागको साथ जबरजस्ती मेटाउन सक्नुहुन्छ, जसलाई सोधिएको सन्देशले स्पष्ट गर्छ।

माथि वर्णन गरिएका **--merged** र **--no-merged** विकल्पहरूले, यदि कुनै कमिट वा ब्रान्च नामलाई तर्क (argument) का रूपमा दिइएको छैन भने, क्रमशः तपाईको हालको ब्रान्चमा के मर्ज गरिएको छ वा मर्ज गरिएको छैन भनेर देखाउँछन्।

तपाईं सधैं एउटा अतिरिक्त तर्क प्रदान गरेर अन्य ब्रान्चको सन्दर्भमा मर्जको अवस्था सोध्न सक्नुहुन्छ, त्यस ब्रान्चलाई पहिले चेकआउट नगरी नै।  
जस्तै, master ब्रान्चमा मर्ज नगरिएको के छ भनेर हेर्न:

$ git checkout testing

$ git branch --no-merged master

यस कमाण्डले master ब्रान्चमा अझै मर्ज नगरिएका ब्रान्चहरूको सूची देखाउँछ, जुन तपाईंसँग testing ब्रान्च चेकआउट गर्दा पनि हेर्न सकिन्छ।

Changing a branch name - - -

मानौं तपाईंको एउटा ब्रान्चको नाम bad-branch-name छ र तपाईं यसलाई corrected-branch-name मा परिवर्तन गर्न चाहनुहुन्छ, सम्पूर्ण इतिहास (history) बचाइराख्दै। साथै, तपाईंले यो ब्रान्चको नाम रिमोट (GitHub, GitLab, वा अन्य सर्वर) मा पनि परिवर्तन गर्न चाहनुहुन्छ।  
यसका लागि तलका चरणहरू अनुसरण गर्नुहोस्:

**१. लोकल ब्रान्चको नाम परिवर्तन गर्नुहोस्**

लोकल ब्रान्चको नाम परिवर्तन गर्न git branch --move कमाण्ड प्रयोग गर्नुहोस्:

$ git branch --move bad-branch-name corrected-branch-name

यो कमाण्डले bad-branch-name लाई corrected-branch-name मा परिवर्तन गर्छ, तर यो परिवर्तन हाललाई केवल लोकल रूपमा लागू हुन्छ।

**२. रिमोटमा नयाँ ब्रान्च अपलोड गर्नुहोस्**

तपाईंको रिमोटमा नयाँ ब्रान्च corrected-branch-name देखाउन, तलको कमाण्ड प्रयोग गर्नुहोस्:

$ git push --set-upstream origin corrected-branch-name

**३. ब्रान्चहरू जाँच गर्नुहोस्**

अब तपाईंको ब्रान्चहरूको सूची हेर्न git branch --all चलाउन सक्नुहुन्छ:

$ git branch --all

\* corrected-branch-name

main

remotes/origin/bad-branch-name

remotes/origin/corrected-branch-name

remotes/origin/main

यहाँ देख्न सकिन्छ कि corrected-branch-name अब रिमोटमा उपलब्ध छ। तर, bad-branch-name ब्रान्च पनि अझै रिमोटमा छ।

**४. रिमोटमा पुरानो ब्रान्च मेटाउनुहोस्**

रिमोटबाट पुरानो ब्रान्च bad-branch-name हटाउन, तलको कमाण्ड प्रयोग गर्नुहोस्:

$ git push origin --delete bad-branch-name

**नतिजा**

अब bad-branch-name ब्रान्च मेटाइनेछ, र corrected-branch-name ब्रान्चले यसको ठाउँ लिनेछ।

**अन्तिम स्थिति**

तपाईंको रिमोट र लोकल ब्रान्च अब पूर्ण रूपमा नयाँ नाम corrected-branch-name मा अद्यावधिक हुनेछ।

Points to remember ---

मास्टर, मेन, मेनलाइन, वा डिफल्टजस्ता ब्रान्चको नाम परिवर्तन गर्दा यसले तपाईंको रिपोजिटरीले प्रयोग गर्ने **इंटेग्रेशनहरू**, **सर्भिसहरू**, **हेल्पर युटिलिटीहरू**, र **बिल्ड/रिलिज स्क्रिप्टहरू**लाई प्रभावित गर्न सक्छ।  
त्यसैले, यो परिवर्तन गर्नु अघि, तलका सुझावहरूलाई ध्यानमा राख्नुहोस्:

**१. सहयोगीहरूसँग सल्लाह गर्नुहोस्**

ब्रान्चको नाम परिवर्तन गर्नु अघि, तपाईंको टीमका **सहयोगीहरू**सँग परामर्श गर्नुहोस्। सबैसँग सहमति हुनुपर्छ किनकि यसले सबैजना प्रभावित हुन सक्छ।

**२. रिपोजिटरीमा पुरानो नामका रेफरेन्सहरू खोज्नुहोस्**

रिपोजिटरीमा जहाँ जहाँ पुरानो ब्रान्च नामको प्रयोग भएको छ (जस्तै, कोड, स्क्रिप्ट, र कन्फिगरेसन फाइलहरूमा), तिनीहरूलाई अपडेट गर्नुहोस्।

**उदाहरण:**

* **डिप्लोयमेन्ट स्क्रिप्टहरूमा पुरानो ब्रान्च नाम**
* **CI/CD पाइपलाइनमा सेटिङहरू**
* **कोडमा हार्डकोडेड रेफरेन्सहरू**

**३. ब्रान्चको नाम परिवर्तन गर्ने प्रक्रिया**

यदि सबै कुरा तयार छ भने, ब्रान्चको नाम परिवर्तन गर्न तलका चरणहरू अनुसरण गर्नुहोस्:

**i. लोकल ब्रान्च नाम परिवर्तन गर्नुहोस्**

$ git branch -m master main

**ii. रिमोट ब्रान्च नाम परिवर्तन गर्नुहोस्**

रिमोटमा पुरानो ब्रान्चलाई नयाँ नामले बदल्न:

$ git push origin -u main

$ git push origin --delete master

**४. सर्भिसहरू र इंटेग्रेशनहरूमा अपडेट गर्नुहोस्**

* सुनिश्चित गर्नुहोस् कि तपाईंको सबै **बाह्य सर्भिसहरू** (जस्तै, Jenkins, GitHub Actions) मा नयाँ ब्रान्च नाम सही रूपमा कन्फिगर भएको छ।
* **API रेफरेन्सहरू**लाई पनि परिमार्जन गर्नुहोस्।

**५. क्लोन गरिएको रिपोजिटरीलाई अपडेट गर्नुहोस्**

जसले यो रिपोजिटरी प्रयोग गरिरहेका छन्, तिनीहरूलाई पुरानो ब्रान्च नामको सट्टा नयाँ नाम क्लोन गर्न आग्रह गर्नुहोस्।

**६. पुन: परीक्षण गर्नुहोस्**

सबै अपडेटपछि, इंटेग्रेशन, बिल्ड, र रिलीज प्रक्रियाहरूलाई पूर्ण रूपमा परीक्षण गर्नुहोस् ताकि कुनै पनि समस्या नहोस्।

**नोट:** कुनै पनि परिवर्तन गर्नु अघि तपाईंको कामको **ब्याकअप लिनु** महत्त्वपूर्ण छ।

----------------------------------------------------------------------------------------------------------------------------------------------

तपाईंको लोकल मास्टर ब्रान्चलाई मेनमा परिवर्तन गर्ने तरिका

**१. लोकल मास्टर ब्रान्चको नाम परिवर्तन गर्नुहोस्**

लोकल मास्टर ब्रान्चलाई मेनमा परिवर्तन गर्न यो आदेश चलाउनुहोस्:

$ git branch --move master main

अब लोकल मास्टर ब्रान्च रहेन, यो मेन ब्रान्चको रूपमा पुनःनामकरण गरिसकिएको छ।

**२. नयाँ मेन ब्रान्चलाई रिमोटमा पठाउनुहोस्**

रिमोटमा मेन ब्रान्च देखाउन यो आदेश प्रयोग गर्नुहोस्:

$ git push --set-upstream origin main

**३. वर्तमान अवस्था हेर्ने**

ब्रान्चहरूको अवस्था हेर्न:

$ git branch --all

\* main

remotes/origin/HEAD -> origin/master

remotes/origin/main

remotes/origin/master

**नोट:**

* लोकल मास्टर ब्रान्च मेनमा रूपान्तरण भइसकेको छ।
* रिमोटमा मेन ब्रान्च उपलब्ध छ।
* पुरानो मास्टर ब्रान्च अझै रिमोटमा छ, जसलाई अन्य सहयोगीहरूले आफ्नो कामको आधारको रूपमा प्रयोग गर्न जारी राख्छन्।

**४. परिवर्तनपछि गर्नुपर्ने थप कामहरू**

मेन ब्रान्चलाई पूर्ण रूपमा कार्यान्वयन गर्नका लागि तपाईंले तलका कामहरू गर्न आवश्यक छ:

* **निर्भर प्रोजेक्टहरू अपडेट गर्नुहोस्:**  
  कुनै पनि प्रोजेक्ट वा स्क्रिप्ट जसले मास्टर ब्रान्चलाई प्रयोग गर्छ, तिनलाई मेन ब्रान्चको नाम अनुसार परिमार्जन गर्नुहोस्।
* **टेस्ट रनर कन्फिगरेसन फाइलहरू परिमार्जन गर्नुहोस्।**
* **बिल्ड र रिलिज स्क्रिप्ट परिमार्जन गर्नुहोस्।**
* **रिपोजिटरी होस्टमा सेटिङ परिवर्तन गर्नुहोस्:**
  + डिफल्ट ब्रान्चलाई मास्टरबाट मेनमा परिवर्तन गर्नुहोस्।
  + मर्ज नियमहरू अपडेट गर्नुहोस्।
  + अन्य सेटिङहरू, जुन ब्रान्च नामसँग मिल्दछन्, तिनलाई परिमार्जन गर्नुहोस्।
* **डक्युमेन्टेशन अपडेट गर्नुहोस्:**  
  जहाँ जहाँ मास्टर ब्रान्चको रेफरेन्स छ, तिनलाई मेन ब्रान्चको नाममा बदल्नुहोस्।
* **ओपन पुल रिक्वेस्टहरू क्लोज वा मर्ज गर्नुहोस्:**  
  पुरानो मास्टर ब्रान्चलाई लक्षित गर्ने कुनै पनि पुल रिक्वेस्टहरू क्लोज वा मर्ज गर्नुहोस्।

**५. पुरानो मास्टर ब्रान्च मेटाउनुहोस्**

सुनिश्चित भएपछि कि मेन ब्रान्चले मास्टर ब्रान्चकै रूपमा काम गर्दछ, मास्टर ब्रान्चलाई रिमोटबाट मेटाउन यो आदेश प्रयोग गर्नुहोस्:

$ git push origin --delete master

अब तपाईंको रिपोजिटरी पूर्ण रूपमा मेन ब्रान्चमा रूपान्तरण हुनेछ।

--------------------------------------- END ---------------------------------------------