**गिटलाई सर्भरमा सेटअप गर्ने - Getting Git on a Server**

अब हामी गिट सेवा चलाउने प्रक्रिया कभर गर्नेछौं, जुन तपाईँको सर्भरमा विभिन्न प्रोटोकलहरूको प्रयोग गरेर सेटअप गर्न सकिन्छ।

**सामान्य जानकारी**

यो गाइड विशेष गरी लिनक्स-आधारित सर्भरहरूको लागि बनाइएको छ। यद्यपि, यी चरणहरूलाई macOS वा Windows सर्भरमा पनि अनुकूल गर्न सकिन्छ। यदि तपाईँ उत्पादन सर्भर सेटअप गर्दै हुनुहुन्छ भने, अतिरिक्त सुरक्षा उपाय र अपरेटिङ प्रणाली-विशिष्ट उपकरणहरू आवश्यक हुन सक्छन्।

**चरण १: बेयर रिपोजिटरी तयार गर्नुहोस्**

बेयर रिपोजिटरी एउटा त्यस्तो गिट रिपोजिटरी हो जसमा **वर्किङ डिरेक्टरी हुँदैन**। यो मुख्यत: सहकार्यको लागि प्रयोग गरिन्छ।

**बेयर रिपोजिटरी कसरी बनाउने:**

1. **असExisting Repository लाई Clone गर्नुहोस्**  
   git clone कमाण्डसँग --bare विकल्प प्रयोग गरेर, तपाईँको प्रोजेक्टलाई बेयर रिपोजिटरीमा परिवर्तन गर्नुहोस्:

git clone --bare my\_project my\_project.git

उदाहरणमा आउने आउटपुट:

Cloning into bare repository 'my\_project.git'...

done.

1. **यसले के गर्छ?**  
   यो कमाण्डले नयाँ **my\_project.git** नामको डिरेक्टरी बनाउँछ, जसमा सबै गिट डेटा हुन्छ तर कुनै वर्किङ डिरेक्टरी हुँदैन।
2. **वैकल्पिक विधि:**  
   तपाईं cp कमाण्ड प्रयोग गरेर .git फोल्डर मात्र प्रतिलिपि गरी पनि यसलाई बनाउन सक्नुहुन्छ:

cp -Rf my\_project/.git my\_project.git

यद्यपि, यो विधि git clone --bare जस्तो राम्रो हुँदैन। यसमा सानो फरकहरू हुन सक्छन्, जस्तै कन्फिगरेसन फाइलहरू।

**चरण २: सर्भरमा बेयर रिपोजिटरी होस्ट गर्नुहोस्**

बेयर रिपोजिटरी बनाएपछि, यसलाई सर्भरमा राख्ने स्थानमा सार्नुहोस्।

mv my\_project.git /path/to/your/server/

**चरण ३: सर्भर एक्सेस सेटअप गर्नुहोस्**

तपाईं कुन प्रोटोकल (SSH, HTTPS, वा Git) प्रयोग गर्न चाहनुहुन्छ, त्यसअनुसार सेटअप गर्नुहोस्:

* **SSH** को लागि, SSH कीज र अनुमतिहरू सेटअप गर्नुहोस्।
* **HTTPS** को लागि, Apache वा Nginx जस्ता वेब सर्भरमा Git एकीकरण सेटअप गर्नुहोस्।
* **Git Protocol** को लागि, git-daemon कन्फिगर गर्नुहोस्।

Putting the Bare Repository on a Server

सर्भरमा गिटको बेयर रिपोजिटरी राख्नका लागि केही साधारण चरणहरू पूरा गर्नु पर्छ। अब हामी यो प्रक्रिया विस्तृत रूपमा तर सरल शैलीमा बुझ्नेछौं। पहिलो कदम भनेको तयार गरिएको बेयर रिपोजिटरीलाई सर्भरमा अपलोड गर्नु हो। मानौं तपाईँसँग **git.example.com** नामको सर्भर छ, जसमा तपाईँलाई SSH पहुँच छ, र तपाईँले सबै गिट रिपोजिटरीहरू /srv/git नामको डिरेक्टरीमा राख्ने योजना बनाउनु भएको छ। यसका लागि, तपाईँले निम्न कमाण्ड प्रयोग गरेर आफ्नो बेयर रिपोजिटरी सर्भरमा स्थानान्तरण गर्न सक्नुहुन्छ:

scp -r my\_project.git user@git.example.com:/srv/git

यस कमाण्डले तपाईँको **my\_project.git** नामको रिपोजिटरीलाई सर्भरको /srv/git नामक डिरेक्टरीमा कपी गर्छ।

रिपोजिटरी सर्भरमा राखिसकेपछि, अब अन्य प्रयोगकर्ताहरू जससँग उक्त सर्भरमा SSH पहुँच छ, उनीहरूले तपाईँको रिपोजिटरीलाई आफ्नो कम्प्युटरमा क्लोन गर्न सक्छन्। क्लोन गर्नका लागि उनीहरूले तलको कमाण्ड चलाउन सक्दछन्:

git clone user@git.example.com:/srv/git/my\_project.git

यस कमाण्डले सर्भरमा रहेको रिपोजिटरीलाई प्रयोगकर्ताको स्थानीय मेसिनमा प्रतिलिपि बनाउँछ।

अब, यदि कोही प्रयोगकर्ताले सर्भरमा रहेको **/srv/git/my\_project.git** नामको डिरेक्टरीमा लेखन पहुँच पाउँछन् भने, उनीहरूले **push** पनि गर्न सक्छन्। यो लेखन पहुँचलाई अझ राम्रो बनाउन र साझा प्रयोगका लागि अनुकूल बनाउन, तपाईँले रिपोजिटरीलाई साझा मोडमा इनिसियलाइज गर्नुपर्छ। यो गर्नका लागि, सर्भरमा लगइन गरेर तलको कमाण्डहरू चलाउनुहोस्:

ssh user@git.example.com

cd /srv/git/my\_project.git

git init --bare --shared

यो कमाण्डले गिट रिपोजिटरीमा स्वचालित रूपमा समूह लेखन अनुमति थप्छ, जसले सबै प्रयोगकर्ताहरूलाई सहकार्य गर्न सहज बनाउँछ।

यस प्रक्रियाले देखाउँछ कि गिट रिपोजिटरीलाई सर्भरमा राखेर सहकार्यका लागि तयार पार्न कति सजिलो छ। कुनै पनि सर्भरमा SSH-योग्य खाता र एउटा बेयर रिपोजिटरी राख्न सक्ने ठाउँ भएमा, तपाईँ र तपाईँका सहकर्मीले एउटै प्रोजेक्टमा सजिलै सहकार्य गर्न सक्नुहुन्छ।

यद्यपि, यदि तपाईँ थप सुरक्षा चाहनुहुन्छ वा सार्वजनिक पहुँच सक्षम गर्न चाहनुहुन्छ भने, पछि अन्य सुधारहरू पनि थप्न सकिन्छ, जस्तै सार्वजनिक रूपमा पढ्न मिल्ने पहुँच थप्ने वा वेब UI सेटअप गर्ने। तर व्यक्तिगत वा सानो समूहमा काम गर्न, यो प्रक्रिया पर्याप्त छ।

सानो सेटअपहरू (Small Setups) - - -

यदि तपाईँ सानो टिममा हुनुहुन्छ वा आफ्नो संगठनमा गिटलाई परीक्षण गरिरहनु भएको छ र केवल केही विकासकर्ताहरू छन् भने, तपाईँको गिट सर्भर सेटअप निकै सरल हुन सक्छ। सानो सेटअपमा, सबैभन्दा जटिल पक्षमध्ये एक भनेको **प्रयोगकर्ता व्यवस्थापन** हो।

यदि तपाईँ चाहनुहुन्छ कि केही प्रयोगकर्ताहरूले केही रिपोजिटरीहरूमा मात्र **पढ्ने पहुँच (read-only)** पाउन र अरूले **पढ्न र लेख्न (read/write)** दुबै गर्न सकून्, यो अवस्था व्यवस्थापन गर्न अलिक कठिन हुन सक्छ।

**समाधान:**

1. **SSH-आधारित पहुँच:**
   * यदि तपाईँसँग थोरै मात्र प्रयोगकर्ताहरू छन् भने, **SSH**-आधारित पहुँच प्रयोग गर्नु राम्रो विकल्प हो।
   * तपाईँले प्रत्येक प्रयोगकर्ताको लागि SSH कुन्जी सेटअप गर्न सक्नुहुन्छ।
   * उदाहरणका लागि, प्रत्येक प्रयोगकर्ताको सार्वजनिक कुन्जीलाई **/home/git/.ssh/authorized\_keys** फाइलमा थप्न सकिन्छ।
2. **साझा मोड सेटअप:**
   * सबै विकासकर्ताहरूले एकै समूहमा काम गर्न सकून् भनेर **साझा मोड (shared mode)** मा गिट रिपोजिटरी इनिसियलाइज गर्नुहोस्।
   * कमाण्ड:

git init --bare --shared

* + यसले प्रयोगकर्ताहरूको समूहमा लेखन पहुँच सक्षम बनाउँछ।

1. **फाइल अनुमतिहरू व्यवस्थापन:**
   * यदि केही प्रयोगकर्ताहरूलाई मात्र लेखन पहुँच दिनु छ भने, तपाईँले सर्भरमा फाइल अनुमतिहरूलाई म्यानुअल रूपमा व्यवस्थापन गर्न सक्नुहुन्छ।
   * उदाहरणका लागि, **/srv/git/repository.git** डिरेक्टरीमा केवल निश्चित प्रयोगकर्ताहरूलाई लेखन पहुँच दिने गरी फाइल अनुमति सेटअप गर्नुहोस्।

**उदाहरण:**

मानौं तपाईँको सर्भर git.example.com मा **/srv/git/project.git** नामको रिपोजिटरी छ। तपाईँले दुई प्रकारका प्रयोगकर्ताहरू व्यवस्थित गर्न सक्नुहुन्छ:

* **Read-only प्रयोगकर्ता:**  
  यी प्रयोगकर्ताहरूले केवल रिपोजिटरी क्लोन गर्न सक्छन्:

git clone user@git.example.com:/srv/git/project.git

* **Read/Write प्रयोगकर्ता:**  
  यी प्रयोगकर्ताहरूले थप परिवर्तनहरू पुश गर्न सक्छन्:

git push user@git.example.com:/srv/git/project.git

**सानो समूहका लागि सुझाव:**

यदि तपाईँको टोली सानो छ भने, SSH-आधारित पहुँच र सरल फाइल अनुमतिहरू नै पर्याप्त हुन्छ। जटिल परिदृश्यहरू जस्तै LDAP वा विशेष भूमिका व्यवस्थापनको आवश्यकता नपर्न सक्छ। यसले तपाईँको सेटअपलाई सरल र व्यवस्थित बनाउँछ।

SSH पहुँच - - -

यदि तपाईँसँग त्यस्तो सर्भर छ जहाँ तपाईँका सबै विकासकर्ताहरूले SSH पहुँच पाइसकेका छन् भने, आफ्नो पहिलो गिट रिपोजिटरी सेटअप गर्न यो सबैभन्दा सजिलो विकल्प हो। यस्तो अवस्थामा, तपाईँलाई धेरै मेहनत गर्नुपर्दैन, किनकि सर्भरको फाइल प्रणाली अनुमतिहरू नै पहुँच नियन्त्रण व्यवस्थापन गर्न पर्याप्त हुन्छ।

**सर्भरमा SSH पहुँच व्यवस्थापन**

यदि तपाईँको सर्भरमा सबै विकासकर्ताहरूको लागि अलग-अलग खाता छैन तर लेखन पहुँच चाहिन्छ भने, तपाईँले SSH पहुँचको सेटअप गर्नुपर्ने हुन्छ।  
तल विभिन्न तरिकाहरू दिइएको छ:

**1. प्रत्येक प्रयोगकर्ताको खाता सिर्जना गर्नु (Accounts for All Users)**

यो सरल तर केही झन्झटिलो प्रक्रिया हो।

* हरेक विकासकर्ताको लागि **adduser** (वा **useradd**) प्रयोग गरेर नयाँ खाता सिर्जना गर्नुहोस्।
* नयाँ प्रयोगकर्ताहरूको लागि अस्थायी पासवर्ड सेटअप गर्नुहोस्।

sudo adduser developer\_name

sudo passwd developer\_name

फाइदा:

* अलग-अलग खाता हुने हुँदा अनुमतिहरू व्यवस्थापन गर्न सजिलो हुन्छ।  
  बाधा:
* साना टोलीका लागि उपयुक्त भए पनि ठूला टोलीमा झन्झटिलो।

**2. साझा 'git' खाता प्रयोग गर्नु (Shared 'git' Account)**

साझा खाता प्रयोग गर्दा तपाईँले हरेक प्रयोगकर्ताको **पब्लिक SSH कुन्जी** सङ्कलन गर्नुपर्ने हुन्छ।

* **'git' खाता सिर्जना गर्नुहोस्:**

sudo adduser git

* हरेक प्रयोगकर्ताको सार्वजनिक कुन्जी ~/.ssh/authorized\_keys मा थप्नुहोस्।

mkdir /home/git/.ssh

nano /home/git/.ssh/authorized\_keys

उदाहरणका लागि, authorized\_keys फाइलमा यसरी थप्नुहोस्:

ssh-rsa AAAAB3NzaC1yc2... user1\_key

ssh-rsa AAAAB3NzaC1yc2... user2\_key

यसले सबै प्रयोगकर्ताहरूलाई **'git' खाता** मार्फत सर्भरमा पहुँच दिनेछ।

फाइदा:

* सरल सेटअप, साना समूहका लागि उपयुक्त।
* सबै प्रयोगकर्ताहरूको पहुँच एउटै स्थानमा नियन्त्रण गर्न सकिन्छ।  
  बाधा:
* प्रयोगकर्ताहरूले सर्भरमा साझा खाता प्रयोग गर्दा सुरक्षा कमजोर हुन सक्छ।

**3. LDAP वा अन्य प्रमाणीकरण प्रणाली प्रयोग गर्नु**

यदि तपाईँसँग LDAP जस्तो केन्द्रीय प्रमाणीकरण प्रणाली छ भने, SSH सर्भरलाई त्यसमा प्रमाणित गर्न प्रयोग गर्न सक्नुहुन्छ।

* LDAP अथवा कुनै केन्द्रीय प्रमाणीकरण प्रणालीमा प्रयोगकर्ता खाता व्यवस्थापन गर्नुहोस्।
* प्रत्येक प्रयोगकर्तालाई SSH पहुँच दिनुहोस्।

फाइदा:

* ठूलो समूह र कर्पोरेट सेटअपका लागि उपयोगी।  
  बाधा:
* सेटअप गर्न प्राविधिक ज्ञान आवश्यक।

**उदाहरण**

मानौं तपाईँले साझा 'git' खाता प्रयोग गर्नुभयो। तपाईँको टोलीमा दुई विकासकर्ताहरू छन्, **Alice** र **Bob**।

1. Alice र Bob ले तपाईँलाई आफ्नो **पब्लिक SSH कुन्जी** पठाउँछन्।
2. यी कुन्जीहरू ~/.ssh/authorized\_keys मा थप्नुहोस्।
3. अब उनीहरूले यसरी पहुँच गर्न सक्छन्:

git clone git@git.example.com:/srv/git/project.git

**साना समूहका लागि सुझाव**

साझा 'git' खाता वा सरल SSH पहुँच सेटअप, साना समूहहरूका लागि उपयुक्त छ। सुरक्षा बढाउन प्रत्येक प्रयोगकर्ताको लागि SSH कुन्जी प्रयोग गर्दा, सर्भरको पहुँच नियन्त्रण व्यवस्थित गर्न सजिलो हुन्छ।

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