25 most commonly ask mid-level interview question focus on devOps senrio base regarding Git and answered it details in easy English and Bangla

Git

Focusing on Git in a DevOps context, here are 25 commonly asked mid-level interview questions with scenario-based aspects. The answers are provided in simplified English and Bangla for better understanding.

### 1. How do you manage branches in a project with multiple teams working on different features?

\*\*English:\*\* Use a branching strategy like Git Flow or Feature Branch Workflow. Each team works on their feature branches and merges back to the main branch upon completion after thorough testing and code review.

\*\*Bangla:\*\* Git Flow অথবা Feature Branch Workflow এর মতো একটি ব্রাঞ্চিং কৌশল ব্যবহার করুন। প্রতিটি দল তাদের ফিচার ব্রাঞ্চগুলিতে কাজ করে এবং পরীক্ষা এবং কোড পর্যালোচনা শেষে মূল ব্রাঞ্চে মার্জ করে।

### 2. How do you handle a merge conflict in a critical feature branch?

\*\*English:\*\* First, communicate with the team to understand changes. Use `git merge` or `git rebase` to start the merge, then manually resolve conflicts in the code. Test changes before committing to ensure stability.

\*\*Bangla:\*\* প্রথমে, পরিবর্তনগুলি বুঝতে দলের সাথে যোগাযোগ করুন। `git merge` অথবা `git rebase` ব্যবহার করে মার্জ শুরু করুন, তারপর কোডে ম্যানুয়ালি সংঘর্ষগুলি মীমাংসা করুন। স্থিতিশীলতা নিশ্চিত করতে কমিট করার আগে পরিবর্তনগুলি পরীক্ষা করুন।

### 3. How do you ensure that the `main` branch always remains stable?

\*\*English:\*\* Implement a CI/CD pipeline that runs automated tests for every commit or pull request to the `main` branch. Only merge changes that pass all tests and code reviews.

\*\*Bangla:\*\* প্রতিটি কমিট বা পুল রিকোয়েস্টের জন্য অটোমেটেড পরীক্ষা চালানো একটি CI/CD পাইপলাইন বাস্তবায়ন করুন `main` ব্রাঞ্চে। শুধুমাত্র সমস্ত পরীক্ষা এবং কোড পর্যালোচনা পাস করা পরিবর্তনগুলি মার্জ করুন।

### 4. Describe how you would revert a feature that caused issues in production after deployment.

\*\*English:\*\* Use `git revert` to create a new commit that undoes the problematic changes without altering the project history. Deploy the revert commit to production to restore stability.

\*\*Bangla:\*\* প্রকল্পের ইতিহাস পরিবর্তন না করে সমস্যাযুক্ত পরিবর্তন

গুলি বাতিল করার জন্য `git revert` ব্যবহার করে একটি নতুন কমিট তৈরি করুন। স্থিতিশীলতা পুনরুদ্ধার করতে প্রোডাকশনে রিভার্ট কমিট ডিপ্লয় করুন।

### 5. How do you squash multiple commits into one for a clean history before merging a feature branch?

\*\*English:\*\* Use `git rebase -i` followed by the commit range. Mark the commits you want to squash with 's' and then complete the rebase. This combines the commits into one.

\*\*Bangla:\*\* `git rebase -i` ব্যবহার করে কমিট রেঞ্জের পরে অনুসরণ করুন। আপনি যে কমিটগুলি স্কোয়াশ করতে চান তা 's' দিয়ে চিহ্নিত করুন এবং তারপর রিবেস সম্পন্ন করুন। এটি কমিটগুলিকে একত্রিত করে।

### 6. What strategy do you use to keep your feature branch up to date with the `main` branch?

\*\*English:\*\* Regularly perform `git pull origin main` from your feature branch to merge in the latest changes from `main`, ensuring your branch is up-to-date.

\*\*Bangla:\*\* নিয়মিত `main` ব্রাঞ্চ থেকে সর্বশেষ পরিবর্তনগুলি মার্জ করতে আপনার ফিচার ব্রাঞ্চ থেকে `git pull origin main` সঞ্চালন করুন, নিশ্চিত করুন যে আপনার ব্রাঞ্চ আপ-টু-ডেট আছে।

### 7. How do you handle a hotfix in your workflow?

\*\*English:\*\* Create a hotfix branch from the `main` branch, make the necessary fixes, and then merge back into both `main` and the development branch. Deploy the hotfix to production immediately after testing.

\*\*Bangla:\*\* `main` ব্রাঞ্চ থেকে একটি হটফিক্স ব্রাঞ্চ তৈরি করুন, প্রয়োজনীয় মেরামত করুন, এবং তারপর `main` এবং ডেভেলপমেন্ট ব্রাঞ্চ উভয়ে মার্জ করুন। পরীক্ষা করার পরে অবিলম্বে প্রোডাকশনে হটফিক্স ডিপ্লয় করুন।

### 8. How do you review a pull request before merging?

\*\*English:\*\* Check out the pull request branch locally, review the code changes, run tests, and look for any potential issues. Provide feedback or approve the pull request if everything looks good.

\*\*Bangla:\*\* পুল রিকোয়েস

্ট ব্রাঞ্চটি লোকালি চেক আউট করুন, কোড পরিবর্তনগুলি পর্যালোচনা করুন, পরীক্ষা চালান, এবং যে কোনও সম্ভাব্য সমস্যা খুঁজুন। সবকিছু ভাল দেখালে প্রতিক্রিয়া প্রদান করুন অথবা পুল রিকোয়েস্ট অনুমোদন করুন।

### 9. Explain how you would use tags in Git.

\*\*English:\*\* Use tags in Git to mark specific points in history as important, typically for releases. For example, use `git tag v1.0.0` to mark a version release.

\*\*Bangla:\*\* রিলিজের জন্য সাধারণত, ইতিহাসের নির্দিষ্ট বিন্দুগুলিকে গুরুত্বপূর্ণ হিসাবে চিহ্নিত করতে Git-এ ট্যাগ ব্যবহার করুন। উদাহরণস্বরূপ, একটি সংস্করণ রিলিজ চিহ্নিত করতে `git tag v1.0.0` ব্যবহার করুন।

### 10. How do you integrate Git with a CI/CD pipeline?

\*\*English:\*\* Configure the CI/CD tool to trigger builds and tests upon a new commit or pull request in Git. This ensures code is automatically built, tested, and deployed.

\*\*Bangla:\*\* CI/CD টুলটি গিটে নতুন কমিট বা পুল রিকোয়েস্টের উপর বিল্ড এবং পরীক্ষা ট্রিগার করতে কনফিগার করুন। এটি নিশ্চিত করে যে কোড স্বয়ংক্রিয়ভাবে বিল্ড, পরীক্ষা এবং ডিপ্লয় করা হয়।

Continuing with the Git-focused DevOps scenario-based questions, here are 10 additional ones with answers in simplified English and Bangla.

### 11. How do you manage large files in Git repositories?

\*\*English:\*\* Use Git Large File Storage (LFS) to manage large files without bloating the repository. It stores large files separately and keeps lightweight pointers in the repository.

\*\*Bangla:\*\* রিপোজিটরিটি বড় না করে বড় ফাইলগুলি পরিচালনা করতে Git Large File Storage (LFS) ব্যবহার করুন। এটি বড় ফাইলগুলিকে আলাদাভাবে সংরক্ষণ করে এবং রিপোজিটরিতে হালকা পয়েন্টার রাখে।

### 12. How do you automate Git workflows with hooks?

\*\*English:\*\* Use Git hooks, scripts that run automatically before or after Git commands like commit or push, to automate workflows like code formatting or running tests before a commit.

\*\*Bangla:\*\* কমিট বা পুশের মতো Git কমান্ডের আগে বা পরে স্বয়ংক্রিয়ভাবে চালিত হয় এমন স্ক্রিপ্ট Git হুক ব্যবহার করে কোড ফরম্যাটিং বা কমিটের আগে পরীক্ষা চালানোর মতো ওয়ার্কফ্লো অটোমেট করুন।

### 13. How do you squash commits in a feature branch before merging?

\*\*English:\*\* Use `git rebase -i HEAD~[number of commits]` to combine multiple commits into a single commit, improving the commit history's readability.

\*\*Bangla:\*\* কমিটের ইতিহাসের পাঠযোগ্যতা উন্নত করতে একাধিক কমিটকে একক কমিটে একীভূত করতে `git rebase -i HEAD~[কমিটের সংখ্যা]` ব্যবহার করুন।

### 14. How do you handle accidental commits to the wrong branch?

\*\*English:\*\* Use `git cherry-pick [commit-hash]` to apply the commit to the correct branch, and then revert or reset the commit in the wrong branch.

\*\*Bangla:\*\* সঠিক ব্রাঞ্চে কমিট প্রয়োগ করতে `git cherry-pick [commit-hash]` ব্যবহার করুন, এবং তারপর ভুল ব্রাঞ্চে কমিটটি বাতিল বা রিসেট করুন।

### 15. How do you perform a dry run of a Git command?

\*\*English:\*\* Use the `--dry-run` option with Git commands to simulate the command execution without making any actual changes.

\*\*Bangla:\*\* কোনো আসল পরিবর্তন না করে কমান্ড নির্বাহের অনুকরণ করতে Git কমান্ডের সাথে `--dry-run` অপশ

ন ব্যবহার করুন।

### 16. How do you revert to a previous commit in Git without losing changes?

\*\*English:\*\* Use `git reset --soft [commit-hash]` to move the HEAD pointer to a previous commit without discarding changes in the working directory.

\*\*Bangla:\*\* কর্মরত ডিরেক্টরিতে পরিবর্তনগুলি বাদ না দিয়ে পূর্ববর্তী কমিটে HEAD পয়েন্টার সরানোর জন্য `git reset --soft [commit-hash]` ব্যবহার করুন।

### 17. What is a `git fetch` vs. `git pull`, and when do you use each?

\*\*English:\*\* `git fetch` updates your remote-tracking branches without merging changes into your current branch, while `git pull` fetches and then merges the changes. Use `fetch` to review changes before merging with `pull`.

\*\*Bangla:\*\* `git fetch` আপনার বর্তমান ব্রাঞ্চে পরিবর্তনগুলি মার্জ না করে আপনার রিমোট-ট্র্যাকিং ব্রাঞ্চগুলি আপডেট করে, অন্যদিকে `git pull` পরিবর্তনগুলি ফেচ করে এবং তারপর মার্জ করে। `pull` দিয়ে মার্জ করার আগে পরিবর্তনগুলি পর্যালোচনা করতে `fetch` ব্যবহার করুন।

### 18. How do you find a bug introduced into the codebase using Git?

\*\*English:\*\* Use `git bisect` to perform a binary search on your commit history to quickly identify the commit that introduced the bug.

\*\*Bangla:\*\* কোডবেসে বাগ চালু করা কমিটটি দ্রুত শনাক্ত করতে আপনার কমিট ইতিহাসে একটি বাইনারি অনুসন্ধান সম্পাদনের জন্য `git bisect` ব্যবহার করুন।

### 19. How do you clean up local branches that have been merged into the main branch?

\*\*English:\*\* Use `git branch --merged` to list merged branches, then `git branch -d [branch-name]` to delete each locally.

\*\*Bangla:\*\* মার্জ করা ব্রাঞ্চগুলি তালিকাভুক্ত করতে `git branch --merged` ব্যবহার করুন, তারপর প্রতিটি লোকালি মুছতে `git branch -d [branch-name]` ব্যবহার করুন।

### 20. How do you create a tag for a release in Git?

\*\*English:\*\* Use `git tag [tag-name] [commit-hash]` to create a tag at a specific commit, marking a significant point like a release.

\*\*Bangla:\*\* একটি রিলিজের মতো গুরুত্বপূর্ণ বিন্দু চিহ্নিত কর

These questions and answers aim to provide a foundation for understanding Git in DevOps scenarios, applicable for mid-level positions.