Git & GitHub Stories with MCQs

STORY 1: Mayur’s First Team Project on GitHub

Mayur is a BCA student working in a team project to build a Gym Management System.

His teacher asked him to collaborate using GitHub, use branches properly,

handle errors, and submit work using Pull Requests.

SCENARIO-BASED MCQs (SET 2)

Q1. Which command initializes Git in an existing folder?

Answer: git init

Q2. Which command connects local repo to GitHub?

Answer: git remote add origin

Q3. Which command checks modified or staged files?

Answer: git status

Q4. Error: "Author identity unknown" fix?

Answer: Set username and email using git config

Q5. Error: "no upstream branch" fix?

Answer: git push -u origin main

Q6. Merge conflict means?

Answer: Same file edited in two places

Q7. Contribute to repo you don’t own first step?

Answer: Fork repository

Q8. Remove Git tracking but keep files?

Answer: rm -rf .git

Q9. Requirement for Pull Request?

Answer: Two different branches with shared history

Q10. Best workflow?

Answer: checkout main → pull → create feature branch

STORY 2: Sudanva Learns Git & GitHub

Sudanva is a BCA student working on a Fitness Club website project.

He learned cloning, branching, committing, pushing, and debugging Git errors.

MCQs

Q1. Clone repository command?

Answer: git clone

Q2. Create and switch branch?

Answer: git checkout -b feature-ui

Q3. Stage all files?

Answer: git add .

Q4. Save changes with message?

Answer: git commit -m "message"

Q5. Push new branch first time?

Answer: git push -u origin feature-ui

Q6. PR shows unrelated histories reason?

Answer: Branch not created from main

Q7. Error: refusing to merge unrelated histories means?

Answer: Local and remote repos created separately

Q8. Show branches and current branch?

Answer: git branch

Q9. Show complete commit graph?

Answer: git log --oneline --all --graph --decorate

Q10. Best practice?

Answer: Create branch from main before coding

ADVANCED STORY: The Broken Production Night

Mayur and Sudanva broke production after merging feature-payment.

They faced merge conflicts, duplicate commits, and CI/CD failure.

Advanced MCQs

Q1. Safest fix after committing to main?

Answer: Create new branch and follow proper workflow

Q2. Conflict markers mean?

Answer: Conflict between two branch versions

Q3. git reset --hard HEAD~1 does?

Answer: Removes last commit locally

Q4. Best command to analyze history?

Answer: git log --graph --oneline --decorate

Q5. Temporarily save unfinished work?

Answer: git stash

Q6. Fix unrelated histories?

Answer: git merge --allow-unrelated-histories

Q7. Risk of force push?

Answer: Deletes collaborators' commits

Q8. Why avoid working on main?

Answer: Risk of breaking stable production code

Q9. Inspect previous commit safely?

Answer: git checkout

Q10. Most professional workflow?

Answer: Feature branch → PR → Review → Merge