Jira, JQL, Git & GitHub Commands & Steps Guide

Jira Setup & Navigation

1. 1. Set up Jira Cloud instance:
   - Sign up at https://www.atlassian.com/software/jira
   - Create a project (Scrum/Kanban/Team Managed).
   - Navigate to: Projects > Your Project > Board.
2. 2. Find sprint backlog & active sprint:
   - Projects > Your Project > Backlog (for backlog view).
   - Projects > Your Project > Active Sprint (for sprint view).
3. 3. Configure Kanban board:
   - Projects > Your Project > Board Settings > Columns.
   - Drag statuses to required columns.
4. 4. Create Scrum project:
   - Create project > Scrum > Set backlog and sprint configurations.
5. 5. Create dashboard for issues & sprints:
   - Dashboards > Create Dashboard > Add gadgets (Sprint Burndown, Filter Results, Issue Statistics).
6. 6. Dashboard for sprint progress & status breakdown:
   - Use gadgets like Sprint Health Gadget & Two-Dimensional Filter Statistics.

Jira Issue Management & Filters

7. 7. Create bug & assign:
   - Create > Issue Type: Bug > Link to Epic > Assign Developer.
8. 8. Create critical issue:
   - Create > Priority: High > Assign Developer.
9. 9. JQL for high-priority issues:
   - JQL: priority = High AND project = 'Your Project'.
10. 10. JQL for all open issues assigned to team:
   - JQL: assignee in (currentUser(), team) AND status != Done.
11. 11. Create epic & link stories:
   - Create > Issue Type: Epic > Create Stories > Link to Epic.

12. 12. Create story:
    - Create > Issue Type: Story > Add description, acceptance criteria, and assign.

Jira Sprint & Epic Management

13. 13. Create sprint:
    - Backlog > Create Sprint > Move backlog items to sprint.
14. 14. Close sprint:
    - Active Sprint > Complete Sprint > Move incomplete items to backlog/next sprint.
15. 15. Onboard new users:
    - Settings > User Management > Add Users > Assign Roles.
16. 16. Set up Scrum project with client permissions:
    - Project Settings > Permissions > Customize roles for client.

JQL Search Queries

17. 17. Assigned to you, 'In Progress':
    - JQL: assignee = currentUser() AND status = 'In Progress'
18. 18. Open issues in current sprint:
    - JQL: sprint in openSprints() AND status != Done
19. 19. Bugs in last 7 days:
    - JQL: labels = Bug AND created >= -7d
20. 20. Assigned to you, excluding 'Done':
    - JQL: assignee = currentUser() AND status != Done

Git & GitHub Basics

21. 21. Git setup:
    - git config --global user.name 'YourName'
    - git config --global user.email 'youremail@example.com'
22. 22. Update email:
    - git config --global user.email 'new@email.com'
23. 23. Initialize & link repo:
    - git init
    - git remote add origin https://github.com/username/repo.git

24. 24. Create & push branch:
    - git checkout -b feature-branch
    - git push origin feature-branch
25. 25. Create branch & merge:
    - git checkout -b new-feature
    - (work & commit)
    - git checkout main
    - git merge new-feature
26. 26. Hotfix branch creation:
    - git checkout main
    - git checkout -b hotfix-branch

Git Commit, Merge & Push Operations

27. 27. Stage all, commit, push:
    - git add .
    - git commit -m 'Message'
    - git push
28. 28. Stage specific files:
    - git add file1 file2
    - git commit -m 'Message'
    - git push
29. 29. Add & commit:
    - git add file.txt
    - git commit -m 'Updated file.txt'
30. 30. Merge feature into main:
    - git checkout main
    - git merge feature-branch
31. 31. Merge without fast-forward:
    - git checkout main
    - git merge --no-ff feature-branch
32. 32. Delete branch locally & remotely:
    - git branch -d feature-branch
    - git push origin --delete feature-branch

GitHub Repository Management

33. 33. Create repo & share:
    - Create repo on GitHub > Settings > Collaborators > Add teammate.
34. 34. Repo with README & .gitignore:
    - Create repo > Check 'Add README' & '.gitignore (Java)'
35. 35. License & branch protection:
    - Create repo > Add license > Settings > Branches > Protect main.
36. 36. Invite external developer:
    - Settings > Collaborators > Invite external user.

Git Pull, Merge & Conflict Resolution

37. 37. Pull without losing work:
    - git stash
    - git pull
    - git stash pop
38. 38. Pull & resolve conflicts:
    - git pull
    - Edit conflicting files
    - git add resolved files
    - git commit
39. 39. Pull main into feature branch:
    - git checkout feature-branch
    - git pull origin main