NAO ID: 3306

08/09/2025

***Github and Digital Respostory Management***

***SPRINT 2***

IN-MEXICO PROGRAM BACKEND DEVELOPER CERTIFICATION

Valeria Anel Duque Salgado

**Nao id: 3306**

**08/09/2025**

2025

|  |  |
| --- | --- |
| User Story | Requirements |
| I want to create a new branch from the Master branch  So that I can work on a separate set of changes without affecting the main codebase | Must be able to create a branch named "A" and another named "B" from Master.  Branches should be visible in local and remote repositories.  Branch creation should not affect Master. |
| As a developer, I want to create a docs.txt file in branch "A" and a different docs.txt in branch "B"  So that I can add branch-specific documentation | The docs.txt file in branch A must be unique to that branch.  The docs.txt file in branch B must be different from branch A.  Files must be committed and pushed to their respective branches. |
| As a developer, I want to create a PR for branch "A" and branch "B"  So that my changes can be reviewed before merging into Master | PRs must be created for both branches in GitHub.  PRs must not be approved immediately.  PRs should include a clear title and description. |
| As a developer, I want to resolve conflicts between branch A and branch B  So that both branches can be merged safely into Master | Identify files causing conflicts.  Merge changes correctly without losing data.  Test changes after resolving conflicts to ensure integrity. |
| As a developer, I want to approve both PRs and merge them into Master  So that all changes from branches A and B are incorporated into the main codebase | PRs must be approved in GitHub.  Merges must not override important changes.  Master branch must reflect all changes from both branches after merging. |

|  |  |  |  |
| --- | --- | --- | --- |
| Requirements | Stages | Time estimation | Deliverables |
| Ability to create branches from Master.  Branches A and B must exist in both local and remote repositories. | 1. Checkout Master branch. 2. Create branch A (git checkout -b A). 3. Push branch A to remote (git push origin A). 4. Repeat steps for branch B. | 15 minutes | Branch A and Branch B created and visible on GitHub. |
| Each branch must contain its own version of docs.txt.  Files must be committed and pushed correctly.  Stages | 1. Switch to branch A. 2. Create docs.txt with unique content. 3. Stage, commit, and push changes. 4. Switch to branch B and repeat with different content. | 20 minutes | Unique docs.txt file in branch A.  Different docs.txt file in branch B. |
| PRs must be created in GitHub for both branches.  PRs must include a clear title and description.  PRs should not be approved yet. | 1. Open GitHub. 2. Create PR for branch A → Master. 3. Create PR for branch B → Master. 4. Add proper descriptions to both PRs. | 15 minutes. | Two open PRs: A → Master, B → Master. |
| Detect merge conflict in docs.txt.  Ensure no important content is lost.  Push resolved version back to repository. | 1. Attempt to merge PRs (GitHub flags conflicts). 2. Checkout conflicting branch locally. 3. Manually resolve docs.txt conflict. 4. Commit resolved file and push changes. 5. Verify in GitHub that conflicts are cleared. | 30–40 minutes. | Conflict-free version of docs.txt.  Updated branches ready to merge. |
| PRs must be reviewed and approved.  Merging should not overwrite or discard changes.  Master must reflect both updates. | 1. Review PR for branch A. 2. Approve and merge PR A. 3. Review PR for branch B. 4. Approve and merge PR B. 5. Validate Master branch reflects both sets of changes. | 20 min | Both PRs approved and merged.  Master branch updated with all changes. |