CVDex

BRANCH_PROTECTION_SETUP

Finanzas SD – Architecture, Flows & SOPs

Arquitectura, Flujos y Procedimientos

November 10, 2025

1 Branch Protection Configuration Guide

1.1 Overview

This document provides step-by-step instructions for configuring branch protection rules on the `main` branch to enforce quality gates and prevent regressions in the Finanzas module.

1.2 Why Branch Protection?

Branch protection ensures:

- □ All code passes automated tests before merging
- □ Build guards catch configuration errors automatically
- □ Peer review provides human validation
- □ No direct pushes to main (all changes via PR)
- □ Quality gates prevent deployment regressions

1.3 Required: Administrator Access

⚠ **Note**: Branch protection settings can only be configured by repository administrators. If you don't have administrator access, please request it from your repository owner.

1.4 Configuration Steps

1.4.1 Step 1: Navigate to Branch Protection Settings

1. Go to your GitHub repository: `https://github.com/valencia94/financial-planning-u`
2. Click **Settings** (tab at the top)
3. Click **Branches** (in the left sidebar under ”Code and automation”)
4. Find **Branch protection rules** section
5. Click **Add rule** (or edit existing rule for `main`)

1.4.2 Step 2: Configure Rule Pattern

- **Branch name pattern**: `main`
- This will apply the protection rules to the main branch

1.4.3 Step 3: Enable Required Status Checks

□ Check: Require status checks to pass before merging

This ensures that all CI/CD workflows must succeed before a PR can be merged.

Sub-options to enable:
□ **Require branches to be up to date before merging**

- This ensures the PR includes the latest changes from main
- Prevents merge conflicts and integration issues

Required status checks (add all of these):

1. □ `finanzas-quality-gates`
    - From: `.github/workflows/finanzas-pr-checks.yml`
    - Validates: Build guards, environment vars, asset paths
2. □ `unit-and-local`
    - From: `.github/workflows/test-api.yml`
    - Validates: API unit tests, SAM build, local smoke tests

Optional but recommended:

3. □ `build-and-deploy-all` (if you want to require successful deployment validation)
    - From: `.github/workflows/deploy-ui.yml`
    - Validates: Full deployment process

How to add required checks:

- Type the job name in the search box
- The check must have run at least once for it to appear in the list
- Click on each check to mark it as required

1.4.4 Step 4: Enable Required Reviews

□ **Check: Require a pull request before merging**
Sub-options to configure:
□ **Require approvals**: Set to **1** (minimum)

- At least one peer must approve before merging
- Increases to 2 for critical changes (optional)

□ **Dismiss stale pull request approvals when new commits are pushed**

- Ensures reviews are valid for the current code state
- Re-review required after changes

□ **Require review from Code Owners** (optional)

- If you have a CODEOWNERS file
- Ensures domain experts review relevant changes

1.4.5 Step 5: Additional Protection Rules

□ **Require conversation resolution before merging**

- All comments must be resolved before merge
- Ensures discussion points are addressed

□ **Require signed commits** (optional)

- Adds an additional layer of security
- Ensures commit authenticity

□ **Require linear history** (optional)

- Prevents merge commits
- Keeps history clean with rebases/squashes

□ **Include administrators**

- Applies rules to admin users too
- Recommended: Keep checked for consistency

□ **Allow force pushes**: Keep disabled

- Prevents rewriting history on main
- Protects against accidental data loss

□ **Allow deletions**: Keep disabled

- Prevents accidental branch deletion

1.4.6 Step 6: Rules Applied to Everyone

□ **Restrict who can push to matching branches**

- Select: **Restrict pushes that create matching branches**
- This ensures all changes go through PRs
- Even admins must use PRs

1.4.7 Step 7: Save Changes

- Scroll to bottom
- Click **Create** (or **Save changes**)
- Branch protection is now active!

1.5 Verification Steps

After configuring branch protection, verify it's working:

1.5.1 Test 1: Status Checks Required

1. Create a test branch:

```
[] git checkout -b test/branch-protection
```

2. Make a small change (e.g., update README)

```
[] echo "# Test" >> README.md git add README.md git commit -m "test: verify branch protection" git push origin test/branch-protection
```

3. Open a PR to `main`

4. **Expected behavior**:

   - PR shows: "Merging is blocked"
   - Status checks appear in PR (may be queued/running)
   - Green checkmarks appear when checks pass
   - "Merge" button becomes enabled only after all checks pass

1.5.2 Test 2: Required Reviews

1. In the PR from Test 1, **Expected behavior**:

   - PR shows: "Review required"
   - "Merge" button shows: "1 approving review required"
   - Another user must approve before merge

1.5.3 Test 3: Direct Push Blocked

1. Try to push directly to main (should fail):

```
[] git checkout main git pull echo ”# Direct push test” >> README.md git add
README.md git commit -m ”test: direct push” git push origin main
```

2. **Expected behavior**:

```
remote: error: GH006: Protected branch update failed
remote: error: Required status checks must pass before merging
```

1.6 Branch Protection Status Screenshot Locations

Document your branch protection configuration with screenshots:

1. **Branch protection rules page**: Shows enabled rules
   - Save as: `docs/evidence/branch-protection-rules.png`
2. **Required status checks section**: Shows required checks
   - Save as: `docs/evidence/required-status-checks.png`
3. **PR with failing checks**: Shows merge blocked
   - Save as: `docs/evidence/pr-checks-failing.png`
4. **PR with passing checks**: Shows merge enabled
   - Save as: `docs/evidence/pr-checks-passing.png`

1.7 Required Status Checks Reference

Here’s a quick reference of the status checks to configure:

Status Check Name	Workflow File	Purpose
`finanzas-quality-gates`	`finanzas-pr-checks.yml`	Build guards, env validation, asset checks
`unit-and-local`	`test-api.yml`	API unit tests, SAM build, local smoke test

Optional Status Checks:

Status Check Name	Workflow File	Purpose	Blocking
`build-and-deploy-all`	`deploy-ui.yml`	Full deployment validation	⚠ Optional

1.8 Troubleshooting

1.8.1 Issue: Status Check Doesn't Appear in List

Problem: When adding required status checks, the check name doesn't appear. **Solution**:

1. The workflow must run at least once on a PR
2. Create a test PR to trigger the workflow
3. Wait for workflow to complete
4. Return to branch protection settings
5. The check should now appear in the search list

1.8.2 Issue: Can't Enable Branch Protection

Problem: ”You must be an admin to configure branch protection” **Solution**:

- Contact repository owner
- Request admin access, or
- Ask admin to configure using this guide

1.8.3 Issue: Status Checks Always Fail

Problem: Required checks are failing on all PRs **Solution**:

1. Check workflow logs in Actions tab
2. Common issues:
    - Missing repository variables/secrets
    - Environment configuration errors
    - Build errors in code
3. Fix the underlying issue
4. Re-run checks

1.8.4 Issue: Can Merge Without Approval

Problem: Merge button is enabled without review
Solution:

1. Verify ”Require a pull request before merging” is checked
2. Verify ”Require approvals” is set to at least 1
3. Check if you’re bypassing as an admin (disable admin bypass)

1.9 Rollback Plan

If branch protection causes issues:

1.9.1 Temporary Disable (Emergency Only)

1. Go to Settings → Branches
2. Find the `main` branch rule
3. Click **Edit**
4. Uncheck problematic rules temporarily
5. Save changes
6. **Important**: Re-enable after fixing the issue

1.9.2 Adjust Status Checks

If a specific check is causing problems:

1. Go to Settings → Branches → Edit rule
2. Find ”Required status checks”
3. Remove the problematic check
4. Save
5. Fix the workflow
6. Re-add the check when ready

1.10 Best Practices

1.10.1 For Repository Admins

1. **Start with minimal rules** and add more over time
2. **Test changes** with a non-critical branch first
3. **Communicate changes** to the team before enabling
4. **Monitor** the first few PRs after enabling
5. **Be available** to help with initial issues

1.10.2 For Development Team

1. **Run tests locally** before pushing

```
[] ./scripts/build-guards-finanzas.sh BUILD_TARGET=finanzas npm run build npm
run lint
```

2. **Keep PRs small** for faster review and testing

3. **Fix failures quickly** - don’t leave PRs in failed state

4. **Communicate blockers** - if checks are failing due to infrastructure issues

1.10.3 For Code Reviewers

1. **Check that all status checks pass** before approving
2. **Review the actual check outputs**, not just green/red status
3. **Run locally if suspicious**
4. **Don’t approve and merge immediately** - allow time for others to review

---

1.11 Configuration Template

Copy-paste this checklist when configuring:

```
[] ## Branch Protection Configuration Checklist
Branch: `main`
### Required Status Checks - [x] Require status checks to pass before merging - [x] Require branches to be up to date before merging - [x] Required checks: - [x] finanzas-quality-gates - [x] unit-and-local
### Pull Request Requirements - [x] Require a pull request before merging - [x] Require approvals: 1 - [x] Dismiss stale pull request approvals when new commits are pushed - [ ] Require review from Code Owners (optional)
### Additional Rules - [x] Require conversation resolution before merging - [ ] Require signed commits (optional) - [ ] Require linear history (optional) - [x] Include administrators
### Restrictions - [x] Restrict who can push to matching branches - [x] Do not allow bypassing the above settings - [x] Block force pushes - [x] Block deletions
### Verification - [ ] Test PR with passing checks - [ ] Test PR with failing checks - [ ] Verify direct push is blocked - [ ] Screenshot configuration
```

1.12 Support & Questions

For questions about branch protection configuration:

1. **Check**: This guide's troubleshooting section
2. **Review**: GitHub's branch protection documentation
3. **Ask**: Team lead or repository administrator
4. **Test**: Use a test branch/repository first

1.13 Related Documentation

- CI/CD Workflow Setup - Complete workflow documentation
- Build Guards - Automated validation script
- API Testing - API test documentation

Configuration Date: _______________ **Configured By**: _______________ **Verified By**: _______________ **Next Review**: _______________