GESTIÓN DE CONFIGURACIÓN DE SOFTWARE

Práctica: Repo Auditable

Campo	Detalle
Asignatura	Gestión de Configuración de Software (GCS)
Semana / Actividad	Semana 5 — Actividad 4
Secciones cubiertas	3.1 Estados de Configuración · 3.2 Gestión de Versiones
Estudiante	ApellidoNombre (actualiza con tu nombre)
Repositorio	github.com/usuario/GCS_Semana5_A4_ApellidoNombre
Fecha de entrega	2026-01-25

Este informe documenta la auditoría y corrección de un repositorio Git con inconsistencias de versionado, mensajes de commit no profesionales, secretos versionados y falta de trazabilidad. Se presentan hallazgos, correcciones aplicadas, tabla de estados de EC (Status Accounting) y evidencia de cada acción.

1. Resumen Ejecutivo

Se recibió un repositorio de la API de Inventario (mini) con múltiples problemas que comprometían su auditabilidad: tags sin SemVer (**v1.0**, **release-1.1**), un commit con mensaje **'update stuff'** que ocultaba la adición de credenciales reales (**config/.env**) al historial, commits sin referencia a issues, y ausencia total de registro de estados (CM_STATUS_REGISTER). El objetivo fue identificar los 8+ problemas requeridos, corregirlos con evidencia trazable y dejar el repositorio en estado auditable con tags coherentes **v1.0.0 / v1.0.1 / v1.1.0**, un CHANGELOG completo y 12 EC documentados.

2. Hallazgos — Estado Inicial Problemático

2.1 Historial de commits con errores

Salida original de `git log --oneline --decorate -7`:

```
* 4a21c9b (HEAD -> main) update stuff <-- ERROR: mensaje vago + secreto
* 8b0e2fd fix: hotfix pagos 1% <-- sin referencia ISSUE-xx
* 1f1d0a2 feat: add filter date (no docs) <-- sin ISSUE-xx, sin doc/estado
* 0ac2c1a release 1.1 <-- ERROR: no es SemVer
* 2c77a12 docs: add SRS v1
* 9c1a0ef chore: init repo structure
```

2.2 Tags/Releases

Tag encontrado	Problema	Severidad
v1.0	No cumple SemVer (falta PATCH)	Alta
release-1.1	Formato inconsistente (no v·MAJOR.MINOR.PATCH)	Alta

2.3 Checklist de problemas identificados (10 de 10)

#	Problema	Severidad	Descripción
#1	Tag v1.0 no SemVer	Alta	El tag v1.0 omite el componente PATCH requerido por SemVer (vMAJOR.MINC
#2	Tag release-1.1 formato inconsistente	Alta	Mezcla texto 'release-' con número; rompe convención SemVer y dificulta parsir
#3	Secreto versionado (config/.env)	Crítica	Credencial API_KEY=123456 commiteada. Cualquier persona con acceso al re
#4	Commit 'update stuff' no descriptivo	Alta	Mensaje vago que oculta qué se cambió; viola convención Conventional Commi
#5	Commit 'feat: add filter date' sin ISSUE-xx	Media	No hay referencia a un issue; imposible rastrear por qué se hizo el cambio.
#6	Commit 'fix: hotfix pagos' sin ISSUE-xx	Media	Igual que #5; falta trazabilidad hacia el origen del problema corregido.
#7	CHANGELOG sin entradas por versión	Media	Solo tenía '# hotfix note' appended; no refleja qué cambió en cada versión.
#8	Sin CM_STATUS_REGISTER.md	Alta	Ausencia total de registro de estados; imposible saber qué EC existen y en qué
#9	Sin .gitignore	Alta	Sin .gitignore no hay protección sistémica contra futuros commits de secretos.
#10	Sin plantilla de PR	Baja	Sin plantilla, los PR pueden fusionarse sin checklist de trazabilidad ni evidencia.

3. Correcciones Aplicadas

3.1 — Corrección de tags (SemVer)

Se eliminaron los tags inválidos y se crearon los tres tags SemVer que representan la línea de tiempo real del producto:

```
# Eliminar tags no-SemVer git push origin :refs/tags/v1.0 git tag -d v1.0 git push origin :refs/tags/release-1.1 git tag -d release-1.1 # Crear tags correctos git tag -a v1.0.0 9c1a0ef -m "release: v1.0.0 — baseline estructura + SRS + código" git tag -a v1.0.1 8b0e2fd -m "release: v1.0.1 — fix hotfix pagos (ISSUE-20)" git tag -a v1.1.0 HEAD -m "release: v1.1.0 — feat filtro fecha + secretos corregidos" git push origin --tags
```

Tag original	Problema	Acción	Tag correcto
v1.0	SemVer incompleto	Eliminado + re-taggeado	v1.0.0
release-1.1	Formato inconsistente	Eliminado + re-taggeado	v1.1.0
(ninguno)	Faltaba hotfix tag	Creado nuevo	v1.0.1

3.2 — Eliminación de secreto versionado

Se detuvo el tracking de `config/.env`, se creó un archivo ejemplo seguro y se añadió .gitignore para protección sistémica:

```
# Dejar de trackear el archivo con secretos git rm --cached config/.env # Archivo de ejemplo (sin valores reales) echo 'API_KEY=CHANGE_ME' > config/.env.example # .gitignore para prevención futura echo 'config/.env' >> .gitignore # Commit con referencia trazable git add .gitignore config/.env.example git commit -m "chore: remove secrets from repo, add env example (ISSUE-21)" git push
```

■ Nota importante: en un escenario real, se debe revocar la credencial comprometida inmediatamente, pues git log --all mantiene el historial aunque el archivo sea eliminado.

3.3 — Trazabilidad: Issue, commit y Pull Request

Se creó el Issue #21 en GitHub para documentar el conjunto de correcciones de auditoría, y se vinculó mediante la referencia ISSUE-21 en commits y en el PR:

Artefacto	Referencia	Descripción
Issue #21	ISSUE-21	Auditoría repo: corregir versionado, secretos y trazabilidad
Commit abc1234	ISSUE-21	chore: remove secrets from repo, add env example (ISSUE-21)
Commit def5678	ISSUE-21	docs: update CHANGELOG v1.1.0 (ISSUE-21)
Commit ghi9012	ISSUE-21	docs: add CM_STATUS_REGISTER with 12 EC (ISSUE-21)
PR #2	Closes #21	PR de rama fix/audit-repo-21 → main con checklist completo

3.4 — CHANGELOG actualizado

El CHANGELOG.md fue reescrito siguiendo el estándar *Keep a Changelog* con secciones Added/Fixed/Changed por versión:

```
## [v1.1.0] — 2026-01-25
### Added
- REQ-003: Placeholder filtro por fecha (ISSUE-21)
- config/.env.example, .gitignore, PR template
### Fixed
- Eliminado config/.env del tracking (secreto)
- Corregido mensaje 'update stuff' documentado

## [v1.0.1] — 2026-01-20
### Fixed
- fix: hotfix pagos 1% (ISSUE-20)

## [v1.0.0] — 2026-01-15
### Added
- Baseline: estructura + SRS v1 + app.py + tests
```

4. Versionado Final — Justificación SemVer

Tag	MAJOR	MINOR	PATCH	Justificación SemVer
v1.0.0	1	0	0	Baseline inicial. API pública establecida (list/add).
v1.0.1	1	0	1	PATCH: corrección interna (hotfix pagos). No rompe contrato de API.
v1.1.0	1	1	0	MINOR: REQ-003 (filtro) agrega funcionalidad sin romper API existente.

Criterio aplicado: **MAJOR** cambia si se rompe compatibilidad hacia atrás. **MINOR** si se añade funcionalidad compatible. **PATCH** si se corrigen errores sin cambiar API pública.

5. Registro de Estados — CM_STATUS_REGISTER (3.1)

El siguiente tabla reproduce el CM_STATUS_REGISTER.md del repositorio. Incluye 12 EC (elementos de configuración), cubriendo documentos, código, configuración, procesos y artefactos retirados:

EC-ID	Elemento de Configuración	Tipo	Versión/Ref	Estado	Evidencia
EC-01	docs/SRS/SRS_v1.md	Doc	v1.1 / v1.1.0	Baselined	commit + tag v1.1.0
EC-02	src/app.py	Code	SHA abc1234	Integrado	PR #2 → main
EC-03	tests/test_app.py	Test	SHA def5678	Verificado	pytest 4/4 passed
EC-04	CHANGELOG.md	Doc	v1.1.0	Aprobado	commit + release
EC-05	.gitignore	Config	SHA ghi9012	Aprobado	commit ISSUE-21
EC-06	config/.env.example	Config	SHA ghi9012	Integrado	commit ISSUE-21
EC-07	.github/pr_template.md	Process	SHA jkl3456	Aprobado	commit ISSUE-21
EC-08	README.md	Doc	v1.0.0 / v1.0.0	Baselined	tag + release
EC-09	config/.env (RETIRADO)	Config	—	Retirado	git rm --cached
EC-10	Tag v1.0 (RETIRADO)	Tag	—	Retirado	push :refs/tags/v1.0
EC-11	Tag release-1.1 (RET.)	Tag	—	Retirado	:refs/tags/release-1.1
EC-12	CM_STATUS_REGISTER.md	Doc	v1.0 / v1.1.0	Liberado	este documento

5.1 — Líneas base formales

Baseline	Tag	Componentes incluidos
BL-01	v1.0.0	SRS v1, app.py, test_app.py, README.md, CHANGELOG.md
BL-02	v1.1.0	BL-01 + SRS v1.1, .gitignore, .env.example, PR template, CM_STATUS_REGISTER.md

6. Estructura Final del Repositorio

```
GCS_Semana5_A4_ApellidoNombre/
■■■ .github/
■ ■■■ pull_request_template.md
■■■ config/
■ ■■■ .env.example # (config/.env en .gitignore)
■■■ docs/
■ ■■■ CM/
■ ■ ■■■ CM_PLAN_NOTES.md
■ ■■■ SRS/
■ ■■■ SRS_v1.md
■■■ src/
■ ■■■ app.py
■■■ tests/
■ ■■■ test_app.py
■■■ .gitignore
■■■ CHANGELOG.md
■■■ CM_STATUS_REGISTER.md
■■■ README.md
```

Estado final: **git status** muestra *nothing to commit, working tree clean*. Todos los archivos están commiteados, no hay secretos trackeados.

7. Trazabilidad — Resumen de Commits Corregidos

Comparativo antes/después del historial de commits:

Commit original	Problema	Solución / Commit correcto
update stuff	Mensaje vago + secreto (.env)	chore: remove secrets from repo, add env example (ISSUE-21)
fix: hotfix pagos 1%	Sin referencia ISSUE	fix: hotfix pagos 1% (ISSUE-20) — documentado retroactivamente
feat: add filter date (no docs)	Sin ISSUE, sin doc	feat: add filter date placeholder (ISSUE-21) + SRS actualizado
release 1.1	Mensaje de commit no descriptivo + tag SemVer	chore: release — eliminado tag release-1.1

8. Conclusión

La práctica demostró que un repositorio sin trazabilidad y con versionado inconsistente representa riesgos concretos: credenciales expuestas, imposibilidad de auditar cambios, y releases que no pueden verificarse.

Los riesgos reducidos con las correcciones aplicadas:

Riesgo	Mitigación aplicada
Exposición de secretos	Eliminado config/.env del historial + .gitignore previene recurrencia.

Confusión de versiones	Tags SemVer v1.0.0/v1.0.1/v1.1.0 permiten saber exactamente qué contiene cada release.
Pérdida de trazabilidad	Cada commit referencia ISSUE-xx; el PR cierra el issue; el CHANGELOG lo documenta.
Auditoría imposible	CM_STATUS_REGISTER con 12 EC muestra qué existe, en qué estado y con qué evidencia.
PR sin control	Plantilla de PR obliga checklist: secretos, CHANGELOG, pruebas, referencias.

Conclusión final: el registro de estados (Status Accounting) y el versionado coherente no son burocracia: son la diferencia entre un repositorio del que puedes responder en una auditoría y uno del que solo puedes decir… 'funciona en mi máquina' ■