**GUIA PARA PROYECTO BLOCKCHAIN**

**1) Definir alcance y requisitos**

* Objetivo en una oración.
* Roles: Admin, Emisor (institución), Estudiante, Verificador.
* Casos de uso: dar/quitar permisos a emisores, emitir certificado, revocar, verificar.
* Reglas clave: **no transferible**, metadatos verificables, registro de revocación.

**2) Diseño de arquitectura mínima**

* Un contrato principal (ERC-721 + AccessControl + lógica soulbound).
* Estructura de datos del certificado: issuer, holder, degree, major, fecha, CID/IPFS, hash del archivo, revoked.
* Decisión de estándar soulbound (propio, EIP-5192 o ERC-5484).
* Diagrama simple de componentes (contrato ↔ IPFS ↔ dApp).

**3) Setup del proyecto (repo y tooling)**

* Repo mono (contracts + tests + scripts + frontend + docs).
* Hardhat + Ethers + TypeScript.
* OpenZeppelin (ERC-721, AccessControl).
* Prettier + Solhint + Git hooks (lint/format al commit).
* .env para RPC/keys (secrets fuera del repo).

**4) Modelo de permisos**

* Roles: DEFAULT\_ADMIN\_ROLE y ISSUER\_ROLE.
* Flujos: otorgar/remover ISSUER\_ROLE, quién puede revocar.
* Eventos: IssuerGranted, IssuerRevoked.

**5) Lógica soulbound**

* Bloquear transferFrom, safeTransferFrom, approve, setApprovalForAll.
* Si usás EIP-5192: locked(tokenId)=true.
* Tests que aseguren que no hay forma de transferir ni aprobar.

**6) Emisión y revocación**

* issueCertificate(holder, degree, major, cid, hash) solo emisor autorizado.
* revoke(tokenId) solo la institución que lo emitió.
* Eventos: Issued(tokenId, issuer, holder), Revoked(tokenId, issuer).

**7) Metadatos y archivos**

* IPFS para JSON + PDF/JPG del diploma.
* Guardar **CID** y **hash** en el contrato.
* Estructura mínima del JSON (name, description, attributes, image/document).
* Política de actualizaciones: inmutable (salvo flag de revocado on-chain).

**8) Seguridad base**

* Reentrancy (aunque baja probabilidad, documentar).
* Validaciones de inputs (no address 0, strings requeridos).
* Pausas de emergencia (opcional Pausable).
* Pruebas de fuzzing básicas (rangos razonables).

**9) Suite de tests (contratos)**

* Cobertura: permisos, emisión válida, no transferible, revocación, lectura de datos, eventos.
* Tests negativos: emisor no autorizado, token inexistente, revocar dos veces, transferencias prohibidas.
* Métrica de coverage y objetivo (p.ej. >90%).

**10) Scripts de despliegue y uso**

* deploy:local, deploy:testnet.
* Script para **agregar emisor** y **emitir** N certificados de ejemplo.
* Script para **revocar** y verificar estado.

**11) Despliegue en testnet**

* Elegir red (Sepolia).
* Variables de entorno (RPC, private key).
* Verificación del contrato (Etherscan/Blockscout).
* Guardar addresses por red en docs/addresses.json.

**12) Frontend mínimo (verificación pública)**

* Página “Verificar certificado”: input tokenId o address → muestra datos on-chain + metadatos IPFS + estado de revocación.
* Panel de emisor: conectar wallet, emitir, revocar (solo si tiene rol).
* Conexión con wagmi/viem + botón de wallet (RainbowKit).
* Manejo de errores y loaders.

**13) Flujo de verificación para terceros**

* Guía en docs: cómo un verificador valida (pasos on-chain + hash del PDF).
* Enlace directo tipo /verify/:tokenId.
* Copiar/pegar checksum y CID.

**14) CI/CD del repo**

* GitHub Actions: lint + test + coverage en cada PR.
* (Opcional) Deploy automático del frontend (Vercel/Netlify).
* Badge de status y coverage en README.

**15) Gas y performance**

* Revisión de estructuras (usar uint64 para fecha, bytes32 para hash).
* Eventos parcimoniosos (sin strings pesados).
* Chequeo de costos de mint y revoke.

**16) Documentación del proyecto**

* README con: pitch corto, features, diagrama, estándares usados, cómo correr tests, deploy y demo.
* docs/ con: casos de uso, modelo de datos, decisiones de diseño, amenazas y mitigaciones.
* Capturas/GIF del frontend.

**17) Demo y muestras**

* Crear 2–3 instituciones demo y emitir 4–6 certificados.
* Subir 1 PDF/JPG de ejemplo a IPFS por certificado.
* Publicar direcciones y tokenIds en README para probar al toque.

**18) Licencia y cumplimiento**

* SPDX headers en contratos.
* Licencia del repo (MIT/Apache-2.0).
* Aclarar que es **demo** y no sustituto legal.

**19) Roadmap y “nice to have”**

* SBT consensual (ERC-5484) con aceptación del holder.
* Listado on-chain de emisores “verificados”.
* Firma fuera de cadena de la institución + verificación on-chain.
* Soporte multi-título por address (consulta por holder).
* Indexación con The Graph (búsquedas rápidas).
* Multi-red / L2 (Base, Arbitrum, Optimism).
* Exportar certificado verificable (QR con tokenId + CID + hash).

**20) Revisión final (audit-style)**

* Pasar Slither/Surya y documentar findings.
* Repasar controles de acceso y test negativos.
* Chequear mensajes de error claros y coverage.
* Último pass de UX (verificador puede validar en 30s).