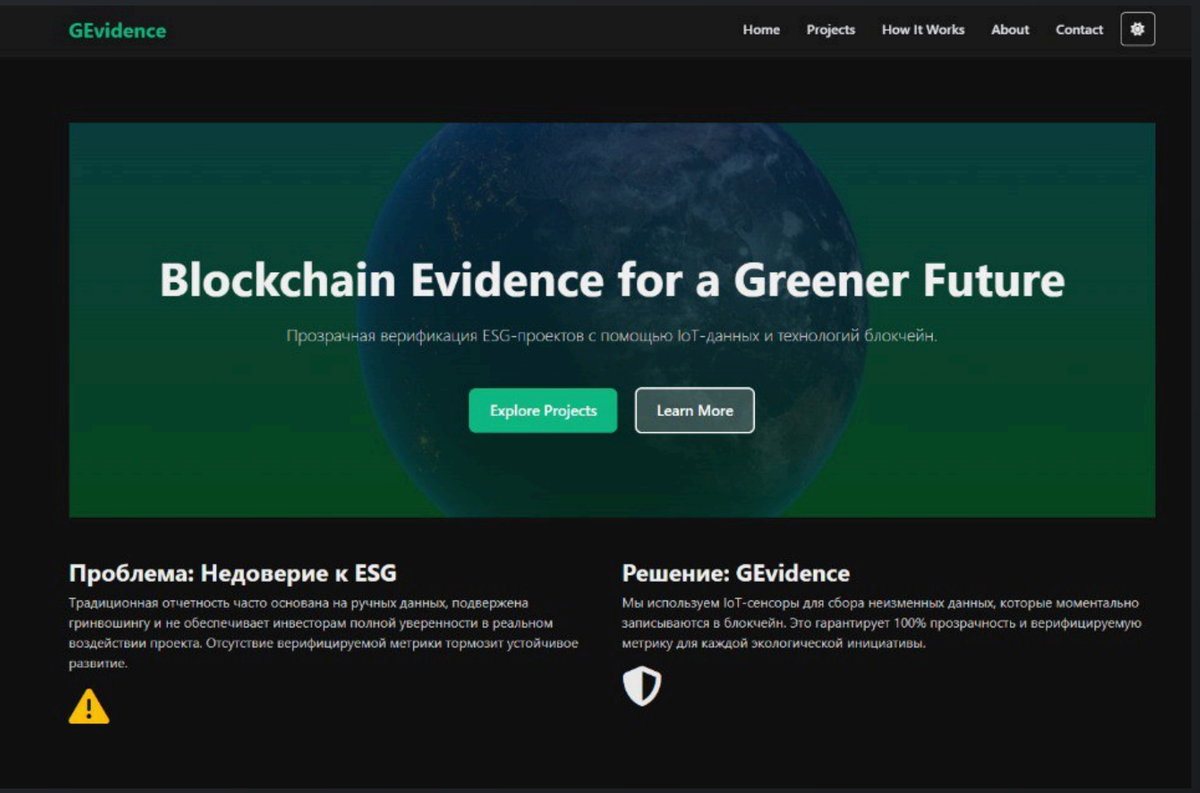
HANOVER

GEVIDENCE

SE - 2430

PROOF OF GREEN

Team: Ali, Anton, Madias



OVERVIEW OF THE APPLICATION ARCHITECTURE

ARCHITECTURE LAYERS

1) Frontend (Web UI)

- Static pages: index.html, dashboard.html, create-campaign.html, campaign.html, investments.html, offcycle.html
- UI logic in JavaScript (folder js/ + script.js)
- The UI connects to MetaMask and calls smart contracts via ethers.js.

2) Web3 connection (MetaMask + ethers.js)

- MetaMask provides the user wallet + transaction signing
- ethers.js is used to:
  - read on-chain state (evidence status, campaign progress, token balances)
  - send transactions (create evidence, donate, finalize, mint NFT, off-cycle requests)

3) Smart Contracts (Solidity / Hardhat)

- Contracts are in contracts/
- Deployment scripts are in scripts/deploy/
- Deployed addresses are stored in deployments/<network>.json
- ABI + addresses are exported to shared/ for easy frontend integration.

Data model (on-chain vs off-chain)

- The full evidence file is not stored on-chain (expensive and unnecessary)
- On-chain we store:
  - dataHash (bytes32) → immutable proof of content
  - uri (string) → where to view the file (IPFS/HTTP)
- This gives transparency + integrity, while keeping gas usage reasonable.

DESIGN & IMPLEMENTATION DECISIONS

KEY ARCHITECTURAL DECISIONS

ROLE-BASED ACCESS CONTROL

- Decision: Centralized RoleManager contract using OpenZeppelin's AccessControl
- Rationale: Flexible permission management for admins, companies, verifiers, and IoT operators
- Implementation:
  - ADMIN_ROLE: Full platform control
  - COMPANY_ROLE: Submit evidence and create campaigns
  - VERIFIER_ROLE: Review and verify evidence
  - IOT_OPERATOR_ROLE: Operate and validate IoT data

REWARD TOKEN ECONOMY

- Decision: ERC-20 tokens minted dynamically based on ETH contributions (1000 tokens/ETH)
- Rationale: Incentivizes participation and provides governance token for future DAO
- Implementation: Conversion via RewardMath library using fixed rate

OFF-CYCLE VERIFICATION WITH STAKING

- Decision: Investors can stake reward tokens to trigger ad-hoc verification checks
- Rationale: Prevents spam, creates skin-in-the-game, enables continuous monitoring
- Mechanism: Stakes returned if approved, sent to treasury if rejected

CAMPAIGN-EVIDENCE LINKING

- Decision: Explicit cross-module linking via GEvidenceRegistry
- Rationale: Maintains integrity across crowdfunding, NFT, and off-cycle modules
- Benefits: Single source of truth for evidence state

FRONTEND TO BLOCKCHAIN

1.

WALLET CONNECTION

- MetaMask integration → User signs with private key
- No backend servers required (decentralized)

2.

CONTRACT INTERACTION STACK

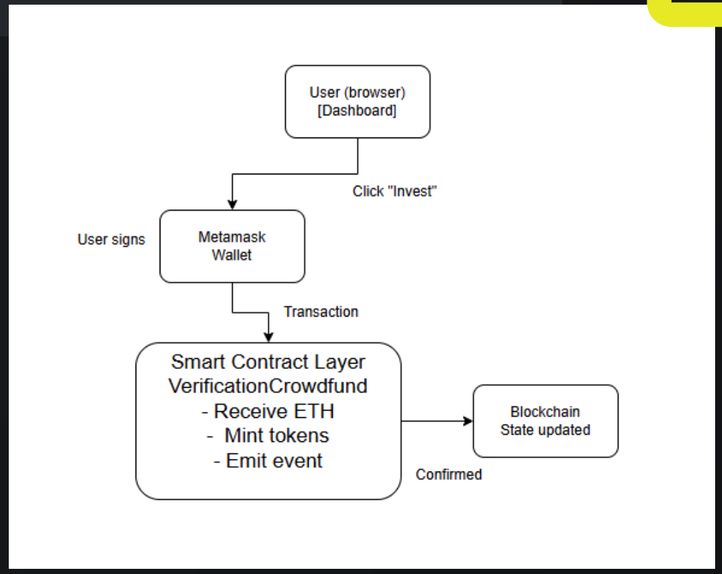
User Action (UI) →
ethers.js →
Smart Contract →
Blockchain

3.

EXAMPLE FLOW: INVESTING IN CAMPAIGN

1. Click "Invest" button
2. MetaMask popup → User signs
3. Transaction sent (0.5 ETH)
4. Smart contract mints 500 reward tokens
5. UI updates automatically (event listener)

4.



VISUAL DIAGRAM

DESCRIPTION OF SMART CONTRACT LOGIC

ROLEMANAGER — PERMISSIONS & SECURITY

The system uses role-based access control:

- ADMIN_ROLE: platform administrator
- COMPANY_ROLE: can create and submit evidence
- VERIFIER_ROLE: can review and verify evidence
- IOT_OPERATOR_ROLE: can push IoT readings (demo oracle)

This prevents unauthorized users from changing statuses or minting certificates.

CONTRACT MAP (HIGH-LEVEL)

CORE

- RoleManager — manages access control (admin, company, verifier, IoT operator)
- GEvidenceRegistry — main evidence registry: creation, lifecycle status, linking campaigns and certificates
- IoTOracleMock — demo IoT feed (push metric readings on-chain)

CONTRACT MAP (HIGH-LEVEL)

MODULES

- VerificationCrowdfund — campaign creation, donations, finalization, withdrawals/refunds
- RewardToken (ERC-20) — reward tokens minted for donors
- OffCycleCheckModule — community-triggered re-checks by staking reward tokens
- CompanyCertificateNFT (ERC-721) — NFT certificate minted after verified evidence + successful campaign

GEVIDENCEREGISTRY — EVIDENCE LIFECYCLE

EVIDENCE STATUS FLOW

- Draft → created by a company
- Submitted → company sends it for verification
- UnderReview → verifier starts review
- Verified / Rejected → verifier final decision

Key actions

Company:

- createEvidence(dataHash, uri)
- updateEvidence(id, newHash, newUri) (only while Draft)
- submitEvidence(id)

Verifier:

- moveToUnderReview(id, note)
- verifyEvidence(id, approved, note)

Registry also stores links:

- evidence ↔ crowdfunding campaign id
- evidence ↔ NFT certificate token id
- evidence ↔ off-cycle requests list

VERIFICATIONCROWDFUND + REWARDTOKEN — FUNDING VERIFICATION + INCENTIVES

CROWDFUNDING LOGIC

- createCampaign(evidenceId, title, goalWei, deadline)
- contribute(campaignId) (payable)
- increases raised amount
- mints RewardToken to donor as an incentive

After deadline:

- finalize(campaignId) marks success/failure

If successful:

- withdraw(campaignId) sends funds to treasury/beneficiary

If unsuccessful:

- refund(campaignId) returns donor contributions

+

VERIFICATIONCROWDFUND + REWARDTOKEN — FUNDING VERIFICATION + INCENTIVES

REWARDTOKEN (ERC-20)

- Minted only by the authorized “minter” contract (Crowdfund module)
- Used later for off-cycle checks (staking mechanism)

OFFCYCLECHECKMODULE — RE-CHECKS TRIGGERED BY THE COMMUNITY

requestOffCycleCheck(evidenceId, stakeAmount, reasonHash, metricsHash)

WORKFLOW

- only allowed if evidence is Verified
- requires stake ⩾ minimum threshold
- tokens are transferred into the module contract
- request id is recorded in the registry

resolveOffCycleCheck(requestId, approved, resultHash, resultUri)

WORKFLOW

- performed by authorized actor (verifier/admin)
- if approved: stake is returned
- if rejected/spam: stake goes to treasury

COMPANYCERTIFICATENFT — PROOF OF VERIFICATION AS NFT

requestOffCycleCheck(evidenceId, stakeAmount, reasonHash, metricsHash)

MINT CONDITIONS

mintCertificate(evidenceId, campaignId, tokenUri)

- evidence exists and is Verified
- certificate not issued before for that evidence
- campaign is linked to the same evidence
- campaign is finalized and successful

Goal: issue an on-chain certificate as a transferable, verifiable artifact.

Result: ERC-721 NFT minted to the company address, and the registry stores the link evidence ↔ tokenId.

CONSENSUS MECHANISMS

HANOVER

HOW THE NETWORK DECIDES WHAT'S TRUE

- Proof of Work → Requires solving complex puzzles → very secure, but energy-heavy.
- Proof of Stake → Based on ownership → efficient, faster.

QUESTION

"Would you choose more power or more efficiency?"

HANOVER

SMART CONTRACTS IN ACTION

CODE THAT KEEPS ITS PROMISES

You rent a car → Smart contract automatically unlocks it when payment is received — no human approval needed.

BENEFITS

- Automation reduces cost & error
- No middleman delays
- Instant execution

HANOVER

BEYOND CRYPTOCURRENCY

- \+ Healthcare - Secure patient data
- \+ Supply Chain - Track goods from origin to shelf
- \+ Education - Verify digital diplomas
- \+ Energy - Manage peer-to-peer energy trading

REAL-WORLD APPLICATIONS

A small coffee producer can use blockchain to prove where beans were grown — boosting consumer trust.

HANOVER

OPPORTUNITIES & OBSTACLES

+ OPPORTUNITIES

- Financial inclusion for the unbanked
- Reduced fraud and corruption
- Faster, borderless transactions

+ OBSTACLES

- Scalability limits
- High energy costs
- Unclear global laws

THE BRIGHT SIDE & THE BARRIERS

FUN FACT

More than 80% of companies exploring blockchain are still in pilot stage — the revolution is just beginning.

DEPLOYMENT & EXECUTION INSTRUCTIONS

- ✅ NODE.JS (V18+) & NPM
- ✅ METAMASK BROWSER EXTENSION
- ✅ SEPOLIA TESTNET ETH (FROM FAUCET)

① DEPLOY CORE → ② DEPLOY MODULES → ③ WIRE ROLES

- NPM INSTALL
- NPX HARDHAT COMPILE
- NPX HARDHAT RUN SCRIPTS/DEPLOY/00_DEPLOY_CORE.JS --NETWORK SEPOLIA
- NPX HARDHAT RUN SCRIPTS/DEPLOY/01_DEPLOY_MODULES.JS --NETWORK SEPOLIA
- NPX HARDHAT RUN SCRIPTS/DEPLOY/02_WIRE.JS --NETWORK SEPOLIA

SOLIDITY 0.8.28 | HARDHAT 2.28.4 | ETHERS.JS V6 | OPENZEPPELIN V5.4 | SEPOLIA / HOLESKY TESTNETS

Текст абзаца

DEPLOYMENT & EXECUTION INSTRUCTIONS

STEP	SCRIPT	CONTRACTS DEPLOYED
① CORE	00_DEPLOY_CORE.JS	ROLEMANAGER, GEVIDENCEREGISTRY, IOTORACLEMOCK
② MODULES	01_DEPLOY_MODULES.JS	VERIFICATIONCROWDFUND, REWARDTOKEN (ERC-20), COMPANYCERTIFICATENFT (ERC-721), OFFCYCLECHECKMODULE
③ WIRING	02_WIRE.JS	GRANTS ROLES (ADMIN, COMPANY, VERIFIER, IOT), SETS TREASURY & PARAMETERS

CREATE `CONFIG.JSON` WITH DEPLOYED ADDRESSES → START SERVER → OPEN IN BROWSER

NPX HTTP-SERVER → HTTP://LOCALHOST:8080

SOLIDITY 0.8.28 | HARDHAT 2.28.4 | ETHERS.JS V6 | OPENZEPPELIN V5.4 | SEPOLIA / HOLESKY TESTNETS

EXECUTION FLOW & TESTING

① "CONNECT WALLET" — METAMASK → AUTO-SWITCH TO SEPOLIA
② "BROWSE CAMPAIGNS" — DASHBOARD.HTML → VIEW ACTIVE CAMPAIGNS
③ "CONTRIBUTE ETH" — CAMPAIGN.HTML → CONFIRM IN METAMASK → EARN GEVR TOKENS
④ "TRACK INVESTMENTS" — INVESTMENTS.HTML → PORTFOLIO & REWARDS
⑤ "OFF-CYCLE CHECK" — OFFCYCLE.HTML → STAKE GEVR → REQUEST EXTRA INSPECTION
⑥ "NFT CERTIFICATE" — VERIFICATION APPROVED → ERC-721 MINTED TO COMPANY

ACTION	COMMAND
RUN ALL TESTS	NPX HARDHAT TEST
CODE COVERAGE	NPX HARDHAT COVERAGE
GAS REPORT	REPORT_GAS=TRUE NPX HARDHAT TEST
START LOCAL NODE	NPX HARDHAT NODE

"TEST SUITES:" ROLES, EVIDENCE REGISTRY, CAMPAIGN FLOW,
REWARD MINTING, NFT CERTIFICATES, OFF-CYCLE CHECKS

OBTAINING TEST ETH (LOCAL HARDHAT)

"① INSTALL METAMASK"

→ DOWNLOAD FROM HTTPS://METAMASK.IO
→ CREATE A WALLET → SAVE YOUR SEED PHRASE SECURELY

"② START LOCAL HARDHAT NODE"

→ RUN IN TERMINAL: `NPX HARDHAT NODE`
→ HARDHAT CREATES "20 TEST ACCOUNTS", EACH WITH "10,000 ETH"
→ ACCOUNTS & PRIVATE KEYS ARE PRINTED IN THE TERMINAL OUTPUT

"③ ADD HARDHAT NETWORK TO METAMASK"

→ OPEN METAMASK → SETTINGS → NETWORKS → ADD NETWORK
→ NETWORK NAME: "HARDHAT"
→ RPC URL: "HTTP://127.0.0.1:8545"
→ CHAIN ID: "31337"
→ CURRENCY SYMBOL: "ETH"

"④ IMPORT A HARDHAT ACCOUNT INTO METAMASK"

→ COPY A PRIVATE KEY FROM THE HARDHAT NODE TERMINAL OUTPUT
→ EXAMPLE: `0XAC0974BEC39A17E36BA4A6B4D238FF944BACB478CBED5EFCAE784D7BF4F2FF80`
→ METAMASK → ACCOUNT ICON → IMPORT ACCOUNT → PASTE PRIVATE KEY → IMPORT
→ BALANCE: "10,000 ETH" APPEARS INSTANTLY

"⑤ READY TO USE"

→ ACCOUNT IS FUNDED AUTOMATICALLY — NO FAUCET NEEDED
→ DEPLOY CONTRACTS AND INTERACT WITH THE DAPP IMMEDIATELY

HANOVER

FINAL THOUGHTS

BRINGING INNOVATION AND TRUST TOGETHER

Blockchain isn’t just about technology - it’s about changing how we trust.
It’s the backbone for a future where transparency, privacy, and security coexist.

Learn, explore, and build with trust.

@AliTheCreator