🔐 BLOCKCHAIN-EVIDENCE (EVID-DGC)

CONTRIBUTOR GUIDE & DOCUMENTATION

WELCOME CONTRIBUTORS! 👋

Thank you for your interest in contributing to **Blockchain-Evidence (Evid-DGC)**, a decentralized evidence management system using blockchain technology. This comprehensive guide will help you get started and understand how to contribute effectively.

Project: Blockchain-Evidence - Secure Evidence Management System

GitHub: https://github.com/Gooichand/blockchain-evidence

Live Demo: https://blockchain-evidence.onrender.com

Maintainer: Gopichand Dandimeni

TABLE OF CONTENTS

1. GETTING STARTED & PREREQUISITES

Before you start contributing, ensure you have:

✅ **Essential Requirements:**

- GitHub account (create at github.com if you don't have one)
- Basic knowledge of JavaScript/Node.js
- MetaMask wallet (for blockchain testing)/ Regularly email log-in
- Git installed on your machine
- 30-45 minutes to complete setup
- Internet connection

✅ **Nice to Have:**

- Solidity basics (for smart contract contributions)
- React experience (for frontend work)
- Docker knowledge (optional)
- Command line/terminal experience

Clone the Repository

```
# Step 1: Fork the repository on GitHub
# Visit: https://github.com/Gooichand/blockchain-evidence
# Click: Fork button (top right)
```

```
# Step 2: Clone YOUR fork to your machine
git clone https://github.com/YOUR-USERNAME/blockchain-evidence.git

# Step 3: Navigate to project directory
cd blockchain-evidence

# Step 4: Add upstream remote to stay updated
git remote add upstream https://github.com/Gooichand/blockchain-evidence.git
```

2. SYSTEM REQUIREMENTS

Development Environment

Requirement	Version	Notes
Node.js	v16.0.0 or higher	Use nvm for version management
npm or yarn	Latest stable	npm comes with Node.js
Git	v2.30+	Download from git-scm.com
Operating System	Windows, macOS, Linux	Any OS is supported
RAM	4GB minimum, 8GB recommended	For blockchain testing
Disk Space	2GB available	For node_modules and blockchain data

Browser Requirements

- Chrome/Chromium (latest)
- Firefox (latest)
- Edge (latest)
- MetaMask extension installed

Optional Tools

- Ganache CLI (for local blockchain)
- Hardhat (smart contract development)
- VS Code (recommended IDE)

3. INSTALLATION & SETUP (STEP-BY-STEP)

⚙ Step 1: Install Dependencies

Open your terminal/command prompt in the project directory:

```
npm install
```

OR if you prefer yarn:

```
yarn install
```

Wait for completion - This downloads all required packages (may take 2-3 minutes).

🔧 Step 2: Environment Configuration

Create a new file named `.env.local` in the project root directory:

```
REACT_APP_BLOCKCHAIN_RPC=http://127.0.0.1:8545
REACT_APP_CONTRACT_ADDRESS=0x[your-contract-address]
REACT_APP_NETWORK_ID=1337
REACT_APP_API_URL=http://localhost:5000
```

Note: Replace `[your-contract-address]` with actual address after deployment.

🔗 Step 3: Start Local Blockchain (Ganache)

Open a NEW terminal window and run:

```
# Install Ganache globally (one-time)
npm install -g ganache-cli

# Start Ganache (in a separate terminal)
ganache-cli --deterministic
```

Expected Output:

```
Ganache CLI v7.x.x
Starting...
Available Accounts:
(0) 0x...
(1) 0x...
... (8 more accounts)

Listening on 127.0.0.1:8545
```

Keep this terminal running while developing.

📦 Step 4: Deploy Smart Contracts

In your original terminal:

```
npm run deploy
```

Expected Output:

```
✅ Contract deployed at: 0x[contract-address]
Network: Ganache (localhost:8545)
```

Copy the contract address and update `.env.local`.

🚀 Step 5: Start Development Server

```
npm start
```

Expected Output:

```
✅ Compiled successfully!

You can now view blockchain-evidence in the browser.

Local:  http://localhost:3000
```

Open http://localhost:3000 in your browser.

🦊 Step 6: Connect MetaMask

1. **Open MetaMask extension** (top right in browser)
2. **Click Network dropdown** (currently shows "Ethereum Mainnet")
3. **Select "Add Network"** (or "Custom RPC")
4. **Fill in:**
5. **Click Save**
6. **Import Test Account:**
7. **You're ready!** You now have test ETH to transact

4. PROJECT STRUCTURE OVERVIEW

```
blockchain-evidence/
│
├── frontend/                    (React UI)
│   ├── src/
│   │   ├── components/          (Reusable React components)
│   │   │   ├── Header.js
│   │   │   ├── FileUpload.js
│   │   │   └── Dashboard.js
│   │   ├── pages/               (Full page components)
│   │   │   ├── HomePage.js
│   │   │   ├── EvidencePage.js
│   │   │   └── AdminPanel.js
│   │   ├── styles/              (CSS/SCSS files)
│   │   ├── App.js               (Main app component)
│   │   └── index.js
│   └── public/
│       └── index.html           (HTML entry point)
│
├── contracts/                   (Smart Contracts)
│   ├── Evidence.sol             ⚠️ DO NOT MODIFY
│   ├── ChainOfCustody.sol       ⚠️ DO NOT MODIFY
│   └── AccessControl.sol        ⚠️ DO NOT MODIFY
│
├── scripts/
│   └── deploy.js                (Deployment script)
│
├── tests/                       (Test files)
│   ├── unit/
│   └── integration/
│
├── .env.local                   (Your local config)
├── package.json                 (Dependencies)
├── README.md                    (Project info)
└── LICENSE                      (MIT License)
```

Key Directories

- **src/components** - Add new UI components here
- **src/pages** - Add new page layouts here
- **contracts** - Smart contracts (experts only)
- **tests** - Add tests for your code

5. CONTRIBUTING GUIDELINES & STANDARDS

Code Quality Standards

For JavaScript/React Code:

✅ **DO:**

- Use ES6+ syntax (arrow functions, destructuring, etc.)
- Write meaningful variable names (`userData` not `ud`)
- Use functional components over class components
- Keep functions small and focused
- Add comments for complex logic
- Use const/let, avoid var
- Format code with Prettier (auto-format on save)

❌ **DON'T:**

- Leave console.log statements in production code
- Use var for variable declaration
- Create deeply nested components
- Write inline styles (use CSS modules)
- Hardcode values (use environment variables)
- Mix arrow functions and regular functions

For Smart Contracts (Solidity):

✅ **DO:**

- Optimize for gas efficiency
- Add clear function documentation (JSDocs)
- Use SafeMath library
- Implement access control
- Write clear event logs

❌ **DON'T:**

- Modify existing contract logic
- Use floating point arithmetic
- Hardcode addresses or values
- Ignore security warnings
- Write untested code

For Testing:

✅ **REQUIRED:**

- Minimum 80% code coverage for new features
- Unit tests for functions
- Integration tests for workflows
- Test edge cases
- Test error conditions

Git Commit Standards

Use clear, descriptive commit messages:

```
Good:  "feat: Add file encryption before upload"
Bad:   "fix stuff"

Good:  "fix: Resolve MetaMask connection timeout"
Bad:   "update"

Good:  "docs: Add API endpoint documentation"
Bad:   "changed readme"
```

Commit Message Format:

```
<type>: <short description>

<optional body explaining why>
```

Types: feat (feature), fix (bug), docs (documentation), style (formatting), refactor (code change), test (tests)

6. WHAT YOU CAN CONTRIBUTE ☑

Areas Open for Contribution

Feature	Details	Difficulty
UI Improvements	Better design, responsive layout, accessibility	Easy
Bug Fixes	Fix issues marked with "bug" label	Easy-Medium
Documentation	README updates, guides, API docs	Easy
Tests	Unit tests, integration tests	Medium
New Features	Features requested in GitHub issues	Medium-Hard
Security	Report vulnerabilities responsibly	Hard
Performance	Optimize code, reduce load time	Hard

Specific Issues to Work On

Look for GitHub Issues labeled:

- `good-first-issue` (perfect for beginners)
- `help-wanted` (community contributions needed)
- `documentation` (write guides)
- `feature-request` (implement new features)

Feature Ideas You Can Implement

1. **User Authentication** - JWT or blockchain-based login
2. **File Encryption** - Encrypt files before blockchain storage
3. **IPFS Integration** - Store large files off-chain
4. **User Dashboard** - Display upload/download history
5. **Advanced Search** - Filter evidence by metadata
6. **Mobile Responsive** - Better mobile experience
7. **API Documentation** - Swagger/OpenAPI specs
8. **CI/CD Pipeline** - GitHub Actions automation
9. **Multi-language Support** - Add language translations
10. **Email Notifications** - Alert users on activity

7. WHAT YOU CANNOT MODIFY ❌

STRICTLY FORBIDDEN - DO NOT TOUCH

These files/functions are locked because they affect core blockchain functionality:

```
❌ contracts/Evidence.sol
   → Core smart contract logic
   → Chain of custody tracking
   → Blockchain immutability

❌ contracts/ChainOfCustody.sol
   → Evidence integrity verification
   → Legal compliance functions

❌ contracts/AccessControl.sol
   → Role-based access control
```

```
   → User permission system

❌ scripts/deploy.js
   → Contract deployment logic
   → Network configuration

❌ Database Schema
   → Evidence data structure
   → Blockchain data format
```

What Else You CANNOT Do

Action	Reason	Consequence
Hardcode Private Keys	Critical security risk	Account compromise
Modify Evidence Storage	Breaks blockchain integrity	Legal non-compliance
Remove Security Features	Violates legal requirements	Evidence inadmissibility
Delete Test Cases	Reduces code quality	More bugs in production
Add Dangerous Dependencies	May introduce vulnerabilities	Security breach
Commit Directly to Main	Bypasses code review	Immediate revert
Ignore Code Review Comments	Quality assurance failure	PR rejection
Use Deprecated Code	Creates maintenance issues	Technical debt
Hardcode Environment Variables	Exposes secrets	Account compromise
Modify User Permissions System	Breaks access control	Unauthorized access

Protected Code Sections

```
// ❌ These functions cannot be modified:
- lockEvidence()
- createChainOfCustody()
- verifyIntegrity()
- transferCustody()
- auditLog()

// ✅ You CAN modify:
- UI components
- Display functions
- API endpoints
- Frontend validation
- Error messages
```

8. GIT WORKFLOW & PULL REQUESTS

Complete Workflow

Step 1: Update Your Fork

```
git fetch upstream
git rebase upstream/main
```

Step 2: Create Feature Branch

```
git checkout -b feature/your-feature-name

# Examples:
git checkout -b feature/add-user-dashboard
git checkout -b fix/metamask-connection-bug
git checkout -b docs/api-documentation
```

Step 3: Make Changes

- Edit files
- Test your changes locally
- Commit with clear messages

```
git add .
git commit -m "feat: Add user dashboard with activity history"
```

Step 4: Push to Your Fork

```
git push origin feature/your-feature-name
```

Step 5: Create Pull Request

1. Go to https://github.com/Gooichand/blockchain-evidence
2. Click "New Pull Request"
3. Select your branch
4. Fill in PR template:

```
## Description
What does this PR do?
Fix #123 (if fixing an issue)

## Type of Change
- [x] Bug fix
- [ ] New feature
- [ ] Documentation
- [ ] Performance

## How to Test
1. Clone the branch
2. Run `npm install`
3. Run `npm start`
4. Open browser and...

## Screenshots
[Add if UI change]

## Checklist
- [x] Code follows style guidelines
- [x] Tests added/updated
- [x] No breaking changes
- [x] Documentation updated
```

Step 6: Code Review & Updates

- Reviewers will comment on code
- Make requested changes
- Commit and push updates
- Respond to comments

Step 7: Approval & Merge

- Need 2 maintainer approvals
- All CI checks must pass
- Maintainer merges your PR
- You're now a contributor! 🎉

Branch Naming Convention

```
feature/short-description      (New features)
fix/short-description          (Bug fixes)
docs/short-description         (Documentation)
refactor/short-description     (Code cleanup)
test/short-description         (Test additions)
```

9. COMMON ISSUES & SOLUTIONS

❌ MetaMask Not Connecting to Ganache

Problem: MetaMask shows "No network"

Solution:

```
1. Ensure Ganache is running (separate terminal)
2. In MetaMask → Settings → Networks
3. Add/Edit Localhost:
   - Name: Ganache Local
   - RPC URL: http://127.0.0.1:8545
   - Chain ID: 1337
   - Currency: ETH
4. Click Save
5. Reload page (Ctrl+R)
```

❌ npm install Fails

Problem: Dependencies won't install

Solution:

```
# Clear cache and reinstall
rm -rf node_modules package-lock.json
npm cache clean --force
npm install
```

❌ Port 3000 Already in Use

Problem: "Port 3000 is already in use"

Solution:

```
# Use different port
npm start -- --port 3001
# Then open http://localhost:3001
```

❌ Smart Contract Deploy Fails

Problem: "Contract deployment failed"

Solution:

```
# 1. Verify Ganache is running
# 2. Check .env.local has correct RPC URL
# 3. Clear build folder
rm -rf build/

# 4. Redeploy contracts
npm run deploy

# 5. Copy address to .env.local
```

❌ Git Merge Conflicts

Problem: "Merge conflict in file.js"

Solution:

```
# 1. Open conflicted file
# 2. Find <<<<<<, ======, >>>>>> markers
# 3. Keep desired code, remove markers
# 4. Commit fix
git add .
git commit -m "resolve: Merge conflict in file.js"
git push origin your-branch
```

❌ Tests Failing

Problem: "Test failed: Expected X but got Y"

Solution:

```
# Run tests with verbose output
npm test -- --verbose

# Run specific test file
npm test filename.test.js

# Debug test
node --inspect-brk node_modules/.bin/jest --runInBand
```

10. RESOURCES & SUPPORT

Official Project Links

- **GitHub Repository:** https://github.com/Gooichand/blockchain-evidence
- **Live Demo:** https://blockchain-evidence.onrender.com
- **Personal Portfolio:** https://gooichand.github.io/

Communication Channels

For Code Issues:

- GitHub Issues tab (click "New Issue")
- Provide error message, steps to reproduce, expected behavior

For Questions:

- GitHub Discussions (click "Discussions" tab)
- Ask clearly, provide context

For Urgent/Private:

- Email: dandimenigopichand6@gmail.com
- LinkedIn: https://www.linkedin.com/in/gopichand-d-269709287/
- WhatsApp: +91 9390606393

BONUS: CODE OF CONDUCT

We're building an inclusive community. Please follow these guidelines:

✅ **DO:**

- Be respectful and kind
- Include others in decisions
- Ask questions (no stupid questions!)
- Help struggling contributors
- Give credit to others
- Celebrate success together

❌ **DON'T:**

- Use offensive language
- Harass or discriminate against others
- Steal credit for work
- Ignore feedback or be dismissive
- Rush through code reviews
- Post irrelevant content

Reporting Issues: Email dandimenigopichand6@gmail.com with details.

RECOGNITION & REWARDS

Every contributor is valued and will be:

✅ Added to `CONTRIBUTORS.md`

✅ Mentioned in release notes

✅ Credited in documentation

✅ Listed on project website

✅ Invited to future project discussions

YOU'RE READY! 🚀

Congratulations! You now have everything needed to contribute to Blockchain-Evidence.

Quick Start Checklist:

- Fork the repository
- Clone to your machine
- Install dependencies (`npm install`)
- Set up `.env.local`
- Start Ganache
- Deploy contracts (`npm run deploy`)
- Start dev server (`npm start`)
- Connect MetaMask
- Pick an issue labeled "good-first-issue"
- Create your feature branch
- Submit your first PR!

Questions?

Open a GitHub Discussion or reach out to maintainers.

Thank you for contributing to blockchain-evidence! 🙏

Together, we're building the future of secure evidence management.

Document Version: 1.0

Last Updated: December 31, 2025

Maintainer: Gopichand Dandimeni

License: MIT