# “EcoLedger”

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS

OF THE DEGREE OF

**Bachelor of Technology in Computer Science and Engineering (IoT and Cyber Security with Blockchain Technology)**

By

**STUDENT NAME:**Riya vaghasiya **SAP ID:**60019230090

**STUDENT NAME:**Prisha Furia **SAP ID:** 60019230093

**STUDENT NAME:**Tiya Shah **SAP ID:** 60019230111 **STUDENT NAME:**Rujuta Mumbarkar **SAP ID:**60019230068



**Department of Computer Science and Engineering**

**(IoT and Cyber Security with Blockchain Technology)**

SVKM's Dwarkadas Jivanlal Sanghvi College of Engineering

(Autonomous College Affiliated to the University of Mumbai)

No.U-15, J.V.P.D. Scheme, Bhaktivedanta Swami Marg, Opp.Cooper Hospital,

Vile Parle (West), Mumbai-400 056. India

**Academic Year 2025-2026**

# INTRODUCTION TO BLOCKCHAIN TECHNOLOGY

**EcoLedger**

Submitted in Partial fulfilment of the requirements of,Introduction To Blockchain Technology (DJS23BC503) (SemV) in the Department of Computer Science &

Engineering (IoT and Cyber security with Blockchain Technology)

by

|  |  |
| --- | --- |
| **STUDENT NAME:**Riya vaghasiya | **SAP ID:**60019230090 |
| **STUDENT NAME:**Prisha Furia | **SAP ID:** 60019230093 |
| **STUDENT NAME:**Tiya Shah | **SAP ID:** 60019230111 |
| **STUDENT NAME:**Rujuta Mumbarkar | **SAP ID:**60019230068 |
|  |  |
|  |  |
|  |  |
|  |  |

**Prof.** Vishakha Shelke

**Date:**

**Academic Year 2025-2026**

# Declaration

We declare that any and all sources utilized in the preparation of this report have been properly cited and referenced. The ideas, concepts, and research findings presented in this proposal are entirely our own, unless otherwise acknowledged and referenced. This report represents my genuine efforts to contribute to the field of Computer Science Engineering ((IoT and Cyber security with Blockchain Technology) and to advance scholarly knowledge in a meaningful and ethical manner

**STUDENT NAME:**Riya vaghasiya **SAP ID:**60019230090

**STUDENT NAME:**Prisha Furia **SAP ID:** 60019230093

**STUDENT NAME:**Tiya Shah **SAP ID:** 60019230111

**STUDENT NAME:**Rujuta Mumbarkar **SAP ID:**60019230068

Date:



**and Cybersecurity with Blockchain Technology)**  **Department of Computer Science and Engineering (IoT**

# Certificate

This is to certify that, topic entitled “ecoledger” has been reviewed and evaluated by undersigned members, and is submitted as partial fulfilment Introdution To Blockchain Technology(DJS23BLPC503) (Semester-V) in the Department of Computer

Science and Engineering (IoT and Cybersecurity with Blockchain Technology)

Vishakha Shelke

Prof.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Dr. Narendra Shekokar Dr. Hari Vasudevan  (Vice Principal & Head of Department) (Principal)

# Abstract

This report details the development of **EcoLedger**, a mobile application built with **Flutter** that implements a transparent and immutable system for tracking carbon emissions and credits. Utilizing a **simulated blockchain** with cryptographic hashing (SHA-256) and a custom data structure stored locally via shared\_preferences, the system ensures data integrity and verifiability, mirroring the core benefits of a real blockchain. The application features a custom, modern, eco-friendly user interface, providing users with a comprehensive **Dashboard** to monitor their net carbon balance, log new **Emission** and **Credit** transactions, and view a detailed, blockchain-verified transaction history. The project serves as a proof-of-concept for how distributed ledger technology can enhance accountability and transparency in environmental sustainability reporting.

## INTRODUCTION

Global efforts to combat climate change increasingly rely on mechanisms like **carbon credits** to incentivize emissions reduction and carbon capture. However, traditional centralized tracking systems often face issues regarding transparency, data tampering, and public trust. The **EcoLedger** application addresses these challenges by employing a **blockchain-based tracking system** for carbon emissions and credits. Built on the **Flutter** framework, EcoLedger offers a cross-platform mobile solution. Although the initial implementation uses a **simulated blockchain** for an MVP (Minimum Viable Product), it adheres to core blockchain principles, including immutability, cryptographic verification, and a ledger-like data structure. The application's design focuses on a modern, non-Material aesthetic with an eco-friendly color palette to promote a visually appealing and intuitive user experience.

### Motivation

The primary motivation for developing EcoLedger stems from the need for **enhanced accountability** and **transparency** in the burgeoning carbon economy. Current reporting standards can be opaque, leading to issues like **double-counting** of credits or **greenwashing**—where organizations inaccurately report their environmental impact.

Key motivational factors include:

1. **Ensuring Data Integrity:** Centralized databases are susceptible to modification. Blockchain's immutable nature prevents post-hoc data tampering, creating a trustworthy, verifiable record of all transactions.
2. **Promoting Trust:** By providing a publicly verifiable record (even if simulated and local in the MVP), EcoLedger builds trust among stakeholders, including consumers, regulators, and environmental organizations.
3. **Future-Proofing:** Designing the system with real blockchain principles (hashing, block structure, verification) ensures it is easily extensible for integration with established public or private blockchain networks (e.g., Ethereum, Polygon) in the future.

## OBJECTIVE

The main objective of the EcoLedger project is to **design and implement a functional prototype** of a transparent, blockchain-based carbon credit tracking system.

Specific objectives include:

1. **Develop a Flutter application** with a custom, modern UI/UX for intuitive tracking.
2. **Establish core data models** (User, Transaction, BlockchainRecord) to structure environmental data.
3. **Implement a simulated blockchain service** capable of cryptographic hashing (SHA-256) for transaction records, ensuring immutability and verification.
4. **Create user flow screens** for:
   * Viewing a real-time **Dashboard** of the carbon balance.
   * Logging new **Emission** and **Credit** activities with detailed forms.
   * Browsing all **Blockchain Transactions** with verification status.
5. **Achieve a robust and clean code structure** using dedicated service and widget layers, paving the way for future production-grade development.

## IMPLEMENTATION

The core logic of the system resides in the **BlockchainService** and **TransactionService**, which manage data persistence and the blockchain simulation. Below are excerpts demonstrating the key data model and the hash generation logic.

**1. Transaction Model (lib/models/transaction\_model.dart)**

This model defines the data structure for every emission or credit entry, including its link to the simulated blockchain via blockchainHash.

Dart

// lib/models/transaction\_model.dart

import 'package:uuid/uuid.dart';

enum TransactionType { emission, credit }

enum TransactionStatus { pending, confirmed, failed }

class Transaction {

final String id;

final String userId;

final TransactionType type;

final double amount;

final String category;

final String description;

final String blockchainHash; // Unique hash linking to the blockchain record

final int blockNumber;

final DateTime timestamp;

final TransactionStatus status;

final DateTime createdAt;

Transaction({

String? id,

required this.userId,

required this.type,

required this.amount,

required this.category,

required this.description,

this.blockchainHash = '',

this.blockNumber = 0,

required this.timestamp,

this.status = TransactionStatus.pending,

DateTime? createdAt,

}) : id = id ?? const Uuid().v4(),

createdAt = createdAt ?? DateTime.now();

// Method to create a transaction with blockchain details after submission

Transaction copyWithBlockchainDetails({

String? blockchainHash,

int? blockNumber,

TransactionStatus? status,

}) {

return Transaction(

id: id,

userId: userId,

type: type,

amount: amount,

category: category,

description: description,

blockchainHash: blockchainHash ?? this.blockchainHash,

blockNumber: blockNumber ?? this.blockNumber,

timestamp: timestamp,

status: status ?? this.status,

createdAt: createdAt,

);

}

// toJson and fromJson methods for local persistence

Map<String, dynamic> toJson() => {

'id': id,

'userId': userId,

'type': type.name,

'amount': amount,

'category': category,

'description': description,

'blockchainHash': blockchainHash,

'blockNumber': blockNumber,

'timestamp': timestamp.toIso8601String(),

'status': status.name,

'createdAt': createdAt.toIso8601String(),

};

factory Transaction.fromJson(Map<String, dynamic> json) {

return Transaction(

id: json['id'],

userId: json['userId'],

type: TransactionType.values.byName(json['type']),

amount: json['amount'],

category: json['category'],

description: json['description'],

blockchainHash: json['blockchainHash'],

blockNumber: json['blockNumber'],

timestamp: DateTime.parse(json['timestamp']),

status: TransactionStatus.values.byName(json['status']),

createdAt: DateTime.parse(json['createdAt']),

);

}

}

**2. Blockchain Service Core Function (lib/services/blockchain\_service.dart)**

This method simulates the critical action of submitting a transaction to the blockchain by generating a unique cryptographic hash and a block number.

Dart

// lib/services/blockchain\_service.dart

import 'dart:convert';

import 'package:crypto/crypto.dart';

import '../models/transaction\_model.dart';

import '../models/blockchain\_record\_model.dart';

class BlockchainService {

int \_currentBlockNumber = 1000; // Starting block number for simulation

// Helper function to generate a cryptographic hash

String \_generateSha256Hash(String data) {

var bytes = utf8.encode(data);

var digest = sha256.convert(bytes);

return digest.toString();

}

// Mock function to submit a transaction and get blockchain details

Future<BlockchainRecord> submitToBlockchain(Transaction transaction) async {

// 1. Prepare data string for hashing

final dataString = jsonEncode({

'id': transaction.id,

'userId': transaction.userId,

'type': transaction.type.name,

'amount': transaction.amount,

'category': transaction.category,

'timestamp': transaction.timestamp.toIso8601String(),

'wallet': getCurrentWalletAddress(), // Mock wallet address

});

// 2. Generate a unique, immutable transaction hash

final transactionHash = \_generateSha256Hash(dataString);

// 3. Simulate block creation and increment block number

\_currentBlockNumber++;

final blockNumber = \_currentBlockNumber;

// 4. Simulate a short delay for "mining" or confirmation

await Future.delayed(const Duration(milliseconds: 500));

// 5. Create a mock blockchain record (the 'proof')

return BlockchainRecord(

transactionHash: transactionHash,

blockNumber: blockNumber,

fromAddress: getCurrentWalletAddress(),

toAddress: 'CarbonLedgerContractAddress', // Mock smart contract address

amount: transaction.amount,

gasUsed: 21000,

timestamp: DateTime.now(),

confirmations: 12,

);

}

// Mock wallet generation

String generateWalletAddress() {

// Generates a mock Ethereum-like address for the user

return '0x' + \_generateSha256Hash('user-wallet-seed-${DateTime.now().microsecondsSinceEpoch}').substring(0, 40);

}

String getCurrentWalletAddress() {

// In a real app, this would be stored securely. Mocking a fixed one here.

return '0xCf7B2C78E35A5E3E73E73523B2A1098F4F9B3C73';

}

// Function to verify transaction hash (checking for data integrity)

Future<bool> verifyTransaction(String hash) async {

// In a real system, this would call a Web3 RPC to verify the hash on the chain.

// For the simulation, we assume any hash we generated is valid (100% confirmation)

await Future.delayed(const Duration(milliseconds: 100));

return hash.isNotEmpty;

}

}

## RESULT /OUTPUT

The EcoLedger application provides a clean, data-driven user experience focused on transparency.

**Key Output Features:**

1. **Dashboard Visualization**: The main screen displays the **Total Carbon Footprint** (Emissions), **Total Carbon Credits**, and a crucial **Net Balance** (Credits - Emissions), often presented using the fl\_chart package for a visual trend overview.
2. **Immutable Transaction Records**: When a user submits an **Emission** or **Credit** entry:
   * The BlockchainService is called, which generates a unique **SHA-256 blockchainHash** and assigns a sequential **blockNumber**.
   * The transaction's status is updated from pending to **confirmed**.
   * The transaction is permanently recorded in local storage with its blockchain details.
3. **Detailed Verification**: The **Transaction Detail Screen** allows users to tap any transaction and view the full blockchain record, including the **Transaction Hash**, **Block Number**, and **Confirmations**. This hash serves as the cryptographic proof of the transaction's contents and timestamp, effectively demonstrating the system's immutability.

**Example Console Output (Simulation of Submission):**

(Note: This is a representation of the internal application flow, not a direct screenshot.)

[LOG] Submitting new 'emission' transaction for 5.0 tons CO2.

[SIM] Generating SHA-256 hash for transaction data...

[SIM] Mock mining/confirmation delay (500ms)...

[SIM] Assigned to Block Number: 1005

[OUT] Blockchain Hash: 6a3b2e9d7c8f0a1b4e5f2d6c7b8a90d1e2f3g4h5i6j7k8l9m0n1o2p3q4r5s6t7

[LOG] Transaction ID 54a7-b2c3 confirmed and updated in local ledger.

[DASHBOARD] User Total Emissions updated to 78.5 tons CO2.

## CONCLUSION

The **EcoLedger** project successfully demonstrates the feasibility of creating a transparent, immutable **Carbon Credit Tracking System** using a blockchain-inspired architecture within a mobile application. By implementing a **simulated blockchain** with cryptographic hashing (SHA-256), the prototype achieves the core tenets of data integrity and verifiability without requiring integration with a live, resource-intensive blockchain network for the MVP stage. The system provides a modern, intuitive user interface for logging emissions and credits and offers detailed, verifiable transaction history.

The successful implementation of the data models, service layer, and simulated blockchain confirms the project's technical objectives. The foundation is now robust and ready for **Future Enhancements**, primarily the integration of real Web3 libraries (such as web3dart) to connect to an actual blockchain (e.g., Ethereum or Polygon), allowing the system to transition from a local prototype to a decentralized, globally verifiable environmental ledger. EcoLedger presents a promising path toward greater accountability in corporate and individual sustainability efforts.