**Practical 2**

*AIM : To create a blockchain and implement replay attacks on blockchain.*

Code :

const SHA256 = require("crypto-js/sha256");

class Block {

  constructor(id, data, previousHash = "") {

    this.id = id;

    this.timestamp = new Date();

    this.data = data;

    this.previousHash = previousHash;

    this.hash = this.calculateHash();

  }

  calculateHash() {

    return SHA256(

      this.id + this.timestamp + this.previousHash + JSON.stringify(this.data)

    ).toString();

  }

}

class Blockchain {

  constructor() {

    this.chain = [this.createGenesisBlock()];

  }

  createGenesisBlock() {

    return new Block(0, "Genesis Block", "0");

  }

  getLatestBlock() {

    return this.chain[this.chain.length - 1];

  }

  addBlock(newBlock) {

    newBlock.previousHash = this.getLatestBlock().hash;

    newBlock.hash = newBlock.calculateHash();

    this.chain.push(newBlock);

  }

}

// Example usage:

const myBlockchain = new Blockchain();

myBlockchain.addBlock(new Block(1, { amount: 4 }));

myBlockchain.addBlock(new Block(2, { amount: 10 }));

console.log(JSON.stringify(myBlockchain, null, 4));

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Blockchain Demo</title>

    <link rel="stylesheet" href="./styles/style.css" />

  </head>

  <body>

    <div class="container">

      <h1>Blockchain Demo</h1>

      <div id="blockchain"></div>

      <button id="addBlockBtn">Add Block</button>

    </div>

    <script src="https://cdn.jsdelivr.net/npm/crypto-js@3.1.9-1/crypto-js.js"></script>

    <script src="js/blockchain.js"></script>

    <script src="js/main.js"></script>

  </body>

</html>

const express = require("express");

const app = express();

const path = require("path");

app.use(express.static(path.join(\_\_dirname, "public")));

app.get("/", (req, res) => {

  res.sendFile(path.join(\_\_dirname, "", "index.html"));

});

const PORT = process.env.PORT || 3000;

app.listen(PORT, () => {

  console.log(`Server is running on http://localhost:${PORT}`);

});

body {

  font-family: Arial, sans-serif;

  background-color: #f4f4f4;

  margin: 0;

  padding: 0;

}

.container {

  max-width: 800px;

  margin: 50px auto;

  padding: 20px;

  background-color: #fff;

  box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

  border-radius: 8px;

}

h1 {

  text-align: center;

}

#blockchain {

  margin: 20px 0;

}

.block {

  border: 1px solid #ddd;

  padding: 10px;

  margin-bottom: 10px;

  border-radius: 5px;

  background-color: #fafafa;

}

button {

  display: block;

  width: 100%;

  padding: 10px;

  border: none;

  background-color: #28a745;

  color: #fff;

  font-size: 16px;

  cursor: pointer;

  border-radius: 5px;

}

button:hover {

  background-color: #218838;

}

document.addEventListener("DOMContentLoaded", () => {

  const myBlockchain = new Blockchain();

  myBlockchain.displayBlockchain();

  document.getElementById("addBlockBtn").addEventListener("click", () => {

    const id = myBlockchain.chain.length;

    const data = { amount: Math.floor(Math.random() \* 100) };

    const newBlock = new Block(id, data);

    myBlockchain.addBlock(newBlock);

    myBlockchain.displayBlockchain();

  });

});

Output :

