|  |  |
| --- | --- |
| Practical No: | 15 |
| B. | Blockchain |
| Aim: | To create a basic blockchain application that stores transactions in a distributed ledger. |
| Program | |
| Explanation: | **Blockchain**   * A blockchain is like a **digital ledger** that stores information in **blocks**. * Each block contains:   + Transactions (bookings in your case)   + A timestamp   + A hash of the previous block * Linking blocks with hashes creates a **chain**. * Any change in a block changes its hash, breaking the chain, so data is **tamper-proof**.   This project simulates a **blockchain for hotel bookings**.   * Each **booking** (customer name, room number, check-in/check-out dates, amount, etc.) is stored as a **transaction**. * Transactions are grouped into **blocks**, and blocks are **chained together** using cryptography. * This ensures **data integrity**: once a block is added, it cannot be tampered with without breaking the chain. |
| Program: | **distributed\_hotel\_node.py**  from flask import Flask, render\_template\_string, request, redirect  import hashlib  import json  import time  import webbrowser  import os  app = Flask(\_\_name\_\_)  class Block:      def \_\_init\_\_(self, index, timestamp, data, previous\_hash):          self.index = index          self.timestamp = timestamp          self.data = data          self.previous\_hash = previous\_hash          self.hash = self.calculate\_hash()      def calculate\_hash(self):          block\_string = json.dumps({              'index': self.index,              'timestamp': self.timestamp,              'data': self.data,              'previous\_hash': self.previous\_hash          }, sort\_keys=True).encode()          return hashlib.sha256(block\_string).hexdigest()  class Blockchain:      def \_\_init\_\_(self):          self.chain = [self.create\_genesis\_block()]          self.pending\_transactions = []      def create\_genesis\_block(self):          return Block(0, time.time(), "Genesis Block", "0")      def get\_latest\_block(self):          return self.chain[-1]      def add\_transaction(self, transaction):          self.pending\_transactions.append(transaction)      def mine\_block(self):          if not self.pending\_transactions:              return None          new\_block = Block(              len(self.chain),              time.time(),              self.pending\_transactions,              self.get\_latest\_block().hash          )          self.chain.append(new\_block)          self.pending\_transactions = []          return new\_block      def is\_chain\_valid(self):          for i in range(1, len(self.chain)):              current = self.chain[i]              previous = self.chain[i - 1]              if current.hash != current.calculate\_hash():                  return False              if current.previous\_hash != previous.hash:                  return False          return True  blockchain = Blockchain()  HTML\_TEMPLATE = """  <!DOCTYPE html>  <html>  <head>      <title>Hotel Booking Blockchain</title>      <style>          body { font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif; background-color: #f0f4f8; margin: 0; padding: 20px; }          h1 { color: #1a73e8; text-align: center; margin-bottom: 20px; }          .dashboard { display: flex; justify-content: space-around; margin-bottom: 30px; flex-wrap: wrap; }          .card { background: #ffffff; padding: 20px 25px; border-radius: 10px; box-shadow: 0 4px 15px rgba(0,0,0,0.1); text-align: center; width: 180px; margin: 10px; }          .card h2 { color: #1a73e8; margin: 0 0 10px 0; }          .form-container, .mine-container { background: #ffffff; padding: 25px; border-radius: 10px; box-shadow: 0 4px 15px rgba(0,0,0,0.1); max-width: 500px; margin: 20px auto; }          label { display: block; margin-top: 15px; font-weight: bold; }          input { width: 100%; padding: 10px; margin-top: 5px; border-radius: 5px; border: 1px solid #ccc; box-sizing: border-box; }          button { margin-top: 20px; padding: 12px 25px; background-color: #1a73e8; color: white; font-size: 16px; border: none; border-radius: 6px; cursor: pointer; transition: background 0.3s ease; }          button:hover { background-color: #155ab6; }          .status { text-align: center; margin-top: 15px; font-size: 18px; }          .valid { color: green; font-weight: bold; }          .invalid { color: red; font-weight: bold; }          table { border-collapse: collapse; width: 100%; margin-top: 30px; background: #ffffff; border-radius: 8px; overflow: hidden; box-shadow: 0 4px 15px rgba(0,0,0,0.05); }          th, td { padding: 12px; text-align: left; }          th { background-color: #1a73e8; color: white; }          tr:nth-child(even) { background-color: #f9f9f9; }          ul { padding-left: 18px; margin: 0; }      </style>  </head>  <body>      <h1>Hotel Booking Blockchain</h1>      <div class="dashboard">          <div class="card">              <h2>{{ blockchain|length }}</h2>              <p>Total Blocks</p>          </div>          <div class="card">              <h2>{{ total\_bookings }}</h2>              <p>Total Confirmed Bookings</p>          </div>          <div class="card">              <h2>{{ pending\_transactions|length }}</h2>              <p>Pending Transactions</p>          </div>      </div>      <div class="form-container">          <form action="/add\_transaction" method="post">              <label>Customer Name:</label>              <input type="text" name="customer\_name" required>              <label>Room Number:</label>              <input type="text" name="room\_number" required>              <label>Check-in Date:</label>              <input type="date" name="check\_in" required>              <label>Check-out Date:</label>              <input type="date" name="check\_out" required>              <label>Amount (₹):</label>              <input type="number" name="amount" required>              <button type="submit">Add Booking</button>          </form>      </div>      <div class="mine-container">          <form action="/mine\_block" method="post">              <button type="submit">Mine Block</button>          </form>      </div>      <div class="status">          Blockchain Status:          {% if is\_valid %}              <span class="valid"> Valid</span>          {% else %}              <span class="invalid"> Compromised</span>          {% endif %}      </div>      <table>          <tr>              <th>Index</th>              <th>Timestamp</th>              <th>Transactions</th>              <th>Hash</th>              <th>Previous Hash</th>          </tr>          {% for block in blockchain %}          <tr>              <td>{{ block.index }}</td>              <td>{{ format\_time(block.timestamp) }}</td>              <td>                  {% if block.index == 0 %}                      Genesis Block                  {% else %}                      <ul>                      {% for tx in block.data %}                          <li>{{ tx.customer\_name }} | Room {{ tx.room\_number }} | {{ tx.check\_in }} → {{ tx.check\_out }} | ₹{{ tx.amount }}</li>                      {% endfor %}                      </ul>                  {% endif %}              </td>              <td>{{ block.hash }}</td>              <td>{{ block.previous\_hash }}</td>          </tr>          {% endfor %}      </table>  </body>  </html>  """  @app.route('/')  def index():      total\_bookings = sum(len(block.data) for block in blockchain.chain if block.index != 0)      return render\_template\_string(          HTML\_TEMPLATE,          blockchain=blockchain.chain,          pending\_transactions=blockchain.pending\_transactions,          total\_bookings=total\_bookings,          is\_valid=blockchain.is\_chain\_valid(),          format\_time=lambda ts: time.strftime('%Y-%m-%d %H:%M:%S', time.localtime(ts))      )  @app.route('/add\_transaction', methods=['POST'])  def add\_transaction():      transaction = {          "customer\_name": request.form.get('customer\_name'),          "room\_number": request.form.get('room\_number'),          "check\_in": request.form.get('check\_in'),          "check\_out": request.form.get('check\_out'),          "amount": request.form.get('amount')      }      blockchain.add\_transaction(transaction)      return redirect('/')  @app.route('/mine\_block', methods=['POST'])  def mine\_block():      blockchain.mine\_block()      return redirect('/')  def open\_browser():      webbrowser.open\_new("http://127.0.0.1:5000/")    if \_\_name\_\_ == '\_\_main\_\_':      app.run(host='0.0.0.0', port=int(os.environ.get('PORT', 5000)))  **procfile**  web: gunicorn distributed\_hotel\_node:app |
| Conclusion: | The above program has been executed successfully. |
| Link: | https://github.com/Meghanajoga/Blockchain.git |