

**Industrial**

**Practice**

**)**

**Report**

**(**

**Title: Cybersecurity Threat Database – Store and Analysis of Network Security Threats and Attack Patterns**

**Title: Cybersecurity Threat Database – Store and Analysis of Network Security Threats and Attack Patterns**

**Title: Cybersecurity Threat Database – Store and Analysis of Network Security Threats and Attack Patterns**

**Faculty**

**:**

**Ms**

**Pratibha**

**Sajwan**

Group

Roll

no.(58

to

60)

Nandini

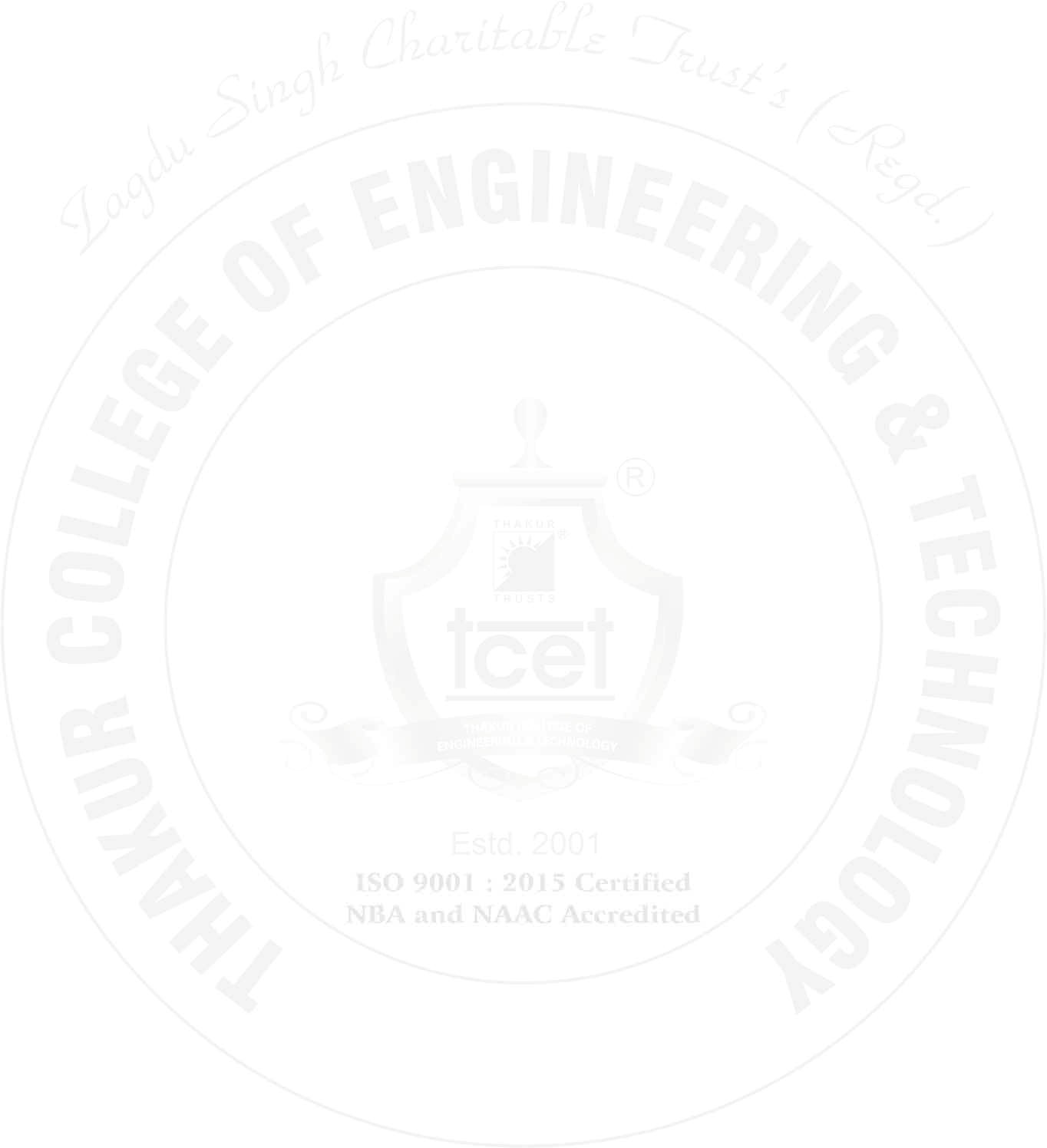
Yadav-58

Ritesh

Yadav-59

Shreekant

Yadav-60

**Title: Cybersecurity Threat Database – Store and Analysis of Network Security Threats and Attack Patterns**

# 1.Abstract

As cyber threats become more diverse and complex, the need for effective tools to manage and analyze these threats is greater than ever. This project introduces a web-based cybersecurity threat management system developed using React.js. The system allows users to input threat data, classify it by category and severity, and visualize trends using interactive bar charts. It provides an intuitive user interface with real-time data handling and supports both light and dark modes for accessibility. The application stores threat information locally using browser storage, making it a lightweight yet powerful solution for tracking and analyzing network security threats and attack patterns.

# 

# 2.Introduction

Cybersecurity has become a major concern across industries as malicious actors exploit vulnerabilities to gain unauthorized access to systems. Network administrators and cybersecurity professionals need robust tools to record, classify, and analyze incidents for threat response and proactive defense. However, many existing systems are overly complex or require extensive infrastructure.  
This project aims to fill that gap by offering a simplified yet functional system for capturing and analyzing cybersecurity threats. By providing a centralized threat tracking interface with real-time visualizations, this system empowers users to better understand the nature and frequency of attacks and make data-driven decisions.

# 

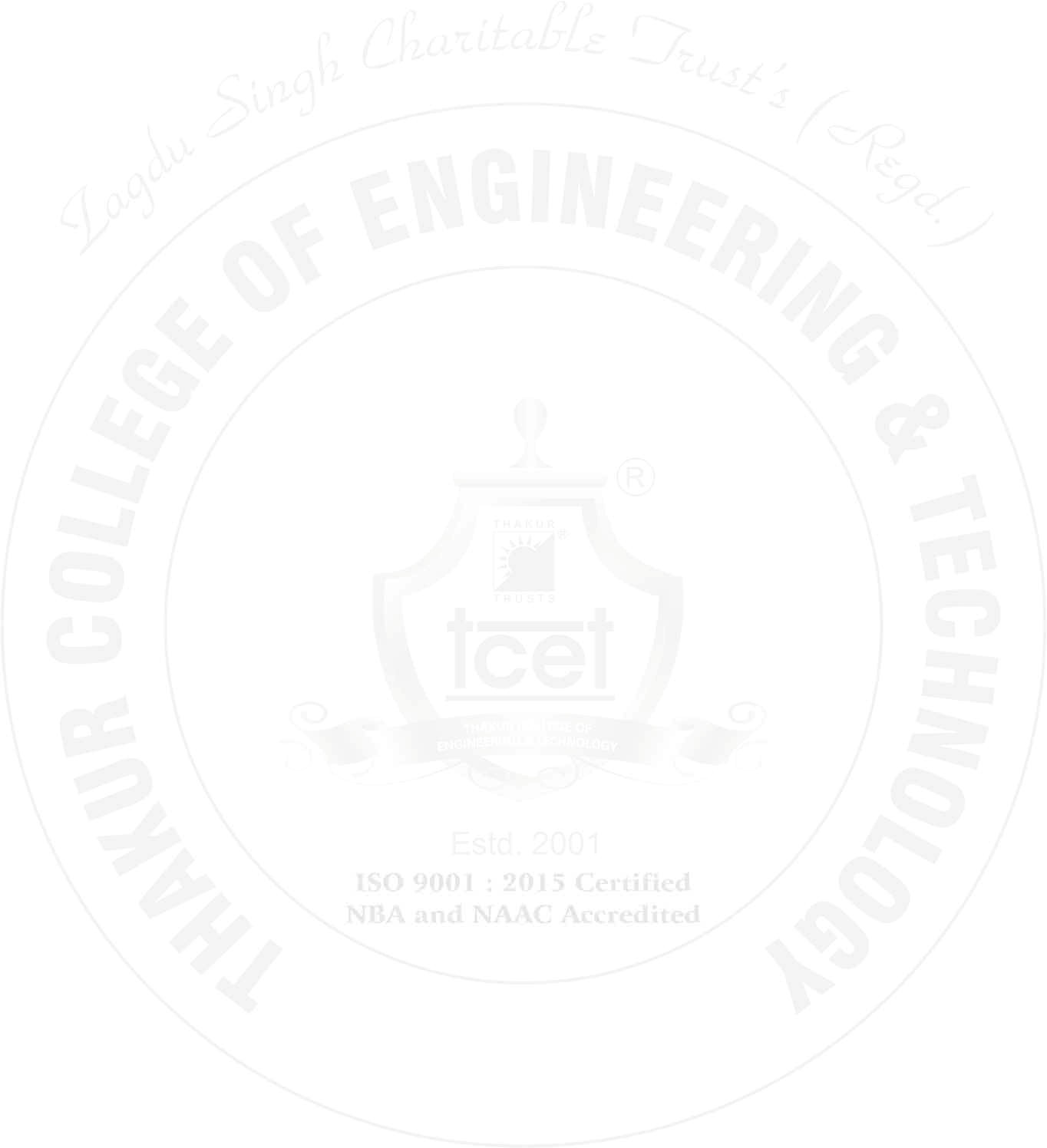
# 3.Background

Security Information and Event Management (SIEM) solutions are standard tools for threat tracking in enterprise environments. However, these tools are often expensive and complex to configure. Smaller teams and academic researchers need more accessible solutions.  
  
Typical network threats include:  
- Malware: Software designed to disrupt or gain unauthorized access.  
- Phishing: Fraudulent attempts to acquire sensitive information.  
- DDoS: Distributed Denial of Service attacks that overwhelm services.  
- Ransomware: Malware that encrypts data and demands ransom.  
- Insider Threats: Attacks from within the organization.  
  
This project provides a platform where these threats can be logged, categorized, and analyzed, all within a user-friendly frontend interface without requiring backend infrastructure.

# 4.System Description

System Architecture:  
  
User Interface (React.js)  
 |  
State Management (useState, useEffect)   
 |  
Data Storage (LocalStorage)  
 |  
Visualization (Chart.js via react-chartjs-2)  
  
Functional Modules:  
- Threat Form: Input form with validation to add threats with category and severity  
- Threat List: Dynamically displays all entered threats with delete option  
- Chart Tabs: Toggle between category-wise and weekly simulated data charts  
- Dark Mode Toggle: Switch between light and dark UI themes  
- Local Storage Sync: Stores threat data locally in the browser, persisting across sessions  
  
Technology Stack:  
- Frontend: React.js  
- Styling: CSS (Custom with media queries)  
- Charting: Chart.js with React wrapper  
- Storage: localStorage  
- Unique IDs: uuid npm package  
- UI Controls: Buttons, inputs, selects, hover effects, conditional styling  
  
User Interaction Flow:  
1. User opens app, UI loads and fetches saved threats from localStorage.  
2. User adds a new threat by entering: Threat Name, Category, Severity.  
3. Threat is validated.  
4. On submission: added to state, stored in localStorage, displayed in list, chart updated.  
5. User can: Delete threats, toggle dark mode, switch chart views.  
  
Data Visualization:  
- Threats by Category Chart: Shows how many threats belong to each category.  
- Weekly Trend Chart: Simulates random weekly data (demo purpose, expandable).

# 5. Results and Discussion

This system was tested using a variety of threat inputs across all categories and severities. Key findings and outcomes include:  
  
- Efficient Data Handling: The application smoothly handles multiple threats without lag.  
- User Experience: The interface is responsive and provides clear feedback.  
- Data Visualization: Charts help visualize hotspots in categories.  
- Persistence: Data remains after refresh due to localStorage.  
- Accessibility: Dark mode enhances usability.  
  
Limitations:  
- Weekly trend data is simulated.  
- No backend prevents multi-user syncing.  
- No authentication; limited security.

# 6. Conclusion and Future Scope The Cybersecurity Threat Database system provides a foundational tool for identifying and analyzing network security threats. It is well-suited for individual analysts, students, or small teams seeking a lightweight threat visualization platform. Future Enhancements: - Database Integration (MongoDB/Firebase) - Authentication - Real-time Data from APIs - Export Options (CSV/PDF) - Severity-Based Alerts - Timeline View

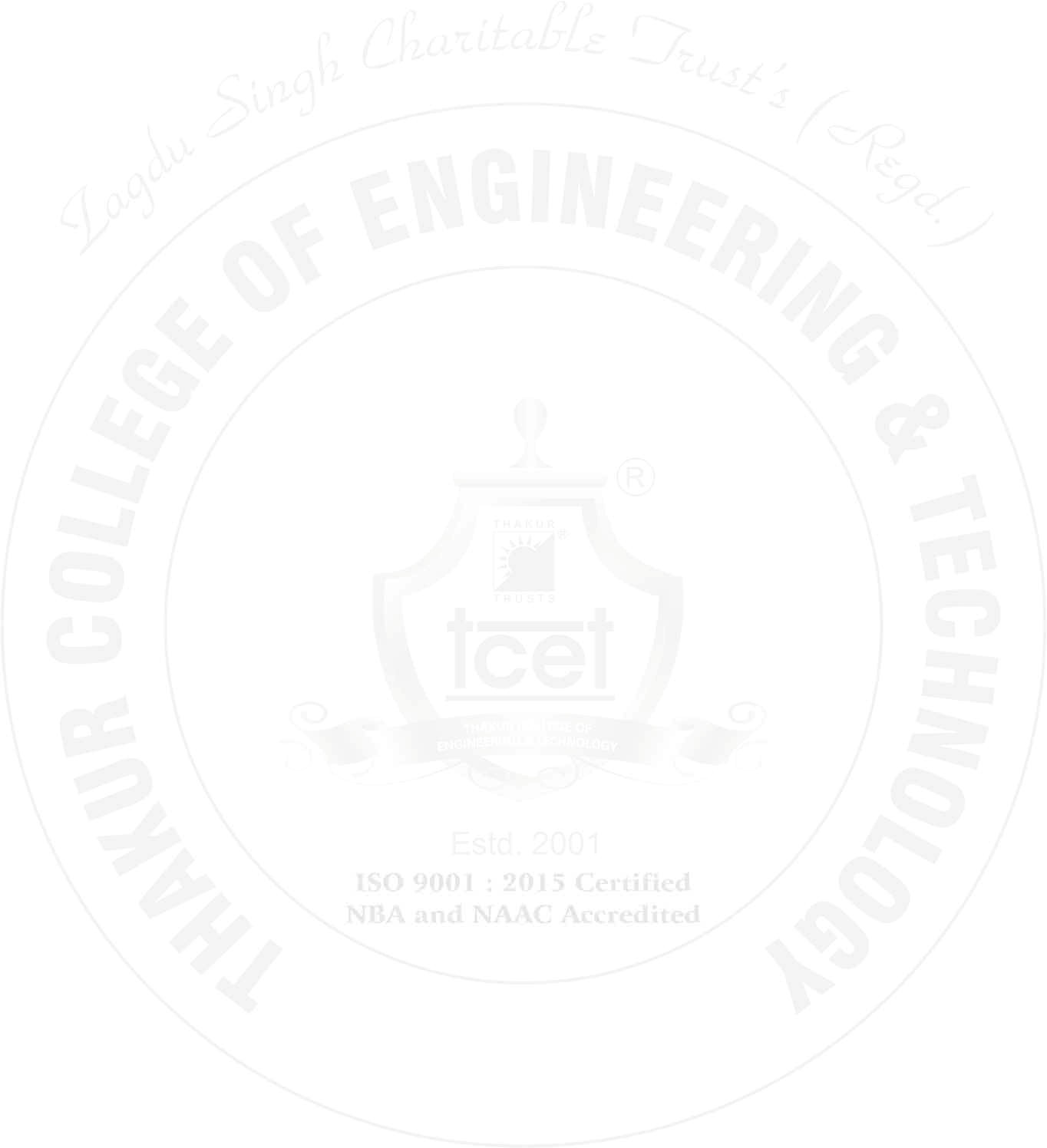
# 7. References

1. React.js Official Docs: https://react.dev/  
2. Chart.js Documentation: https://www.chartjs.org/  
3. Mozilla Developer Network: https://developer.mozilla.org/  
4. OWASP Top 10 Threats: https://owasp.org/www-project-top-ten/  
5. CISA - Cybersecurity Threats: https://www.cisa.gov/topics/cyber-threats  
6. uuid NPM Package: https://www.npmjs.com/package/uuid

# 8.Code:

**React js:**

import React, { useState, useEffect, useMemo } from "react";

import { Bar } from "react-chartjs-2";

import { v4 as uuidv4 } from "uuid";

import "chart.js/auto";

import "./App.css";

// Constants

const CATEGORIES = [

  { value: "Malware", color: "#ff4d4d" },

  { value: "Phishing", color: "#4d79ff" },

  { value: "DDoS", color: "#4dff4d" },

  { value: "Ransomware", color: "#ffcc00" },

  { value: "Insider", color: "#ff66b3" }

];

const SEVERITY\_LEVELS = ["Low", "Medium", "High", "Critical"];

const App = () => {

  const [threats, setThreats] = useState([]);

  const [newThreat, setNewThreat] = useState("");

  const [category, setCategory] = useState(CATEGORIES[0].value);

  const [severity, setSeverity] = useState(SEVERITY\_LEVELS[1]);

  const [darkMode, setDarkMode] = useState(false);

  const [error, setError] = useState(null);

  const [activeChart, setActiveChart] = useState('categories');

  // Load saved threats from localStorage

  useEffect(() => {

    const saved = localStorage.getItem('threats');

    if (saved) {

      try {

        setThreats(JSON.parse(saved));

      } catch (e) {

        console.error("Failed to parse saved threats", e);

      }

    }

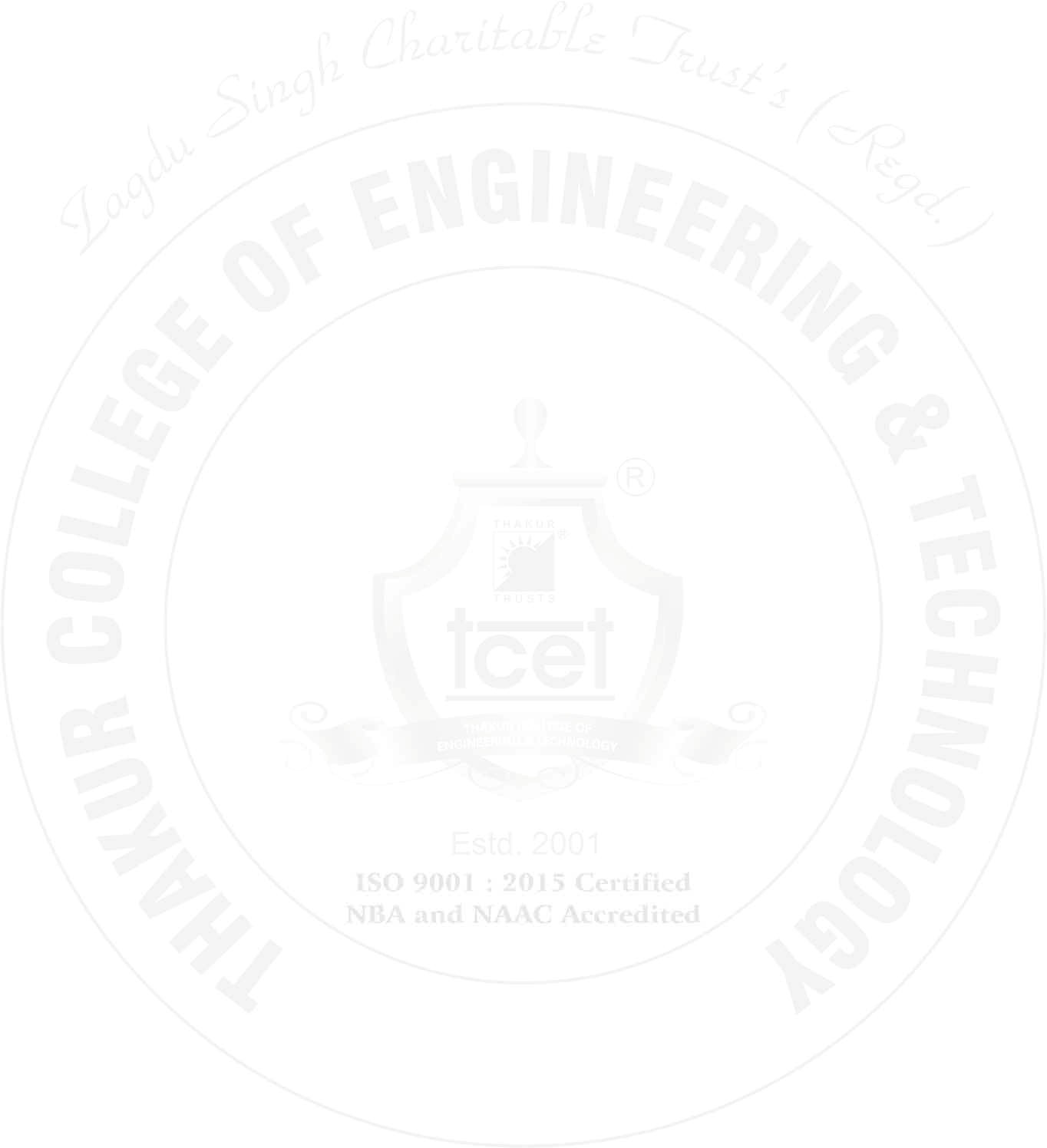
  }, []);

  // Save threats to localStorage

  useEffect(() => {

    localStorage.setItem('threats', JSON.stringify(threats));

  }, [threats]);



  const handleAddThreat = () => {

    try {

      const sanitizedThreat = newThreat.trim()

        .replace(/</g, "&lt;")

        .replace(/>/g, "&gt;");

      if (!sanitizedThreat) {

        throw new Error("Threat name cannot be empty");

      }

      if (sanitizedThreat.length > 100) {

        throw new Error("Threat name too long (max 100 chars)");

      }

      if (!CATEGORIES.some(c => c.value === category)) {

        throw new Error("Invalid threat category");

      }

      setThreats([...threats, {

        id: uuidv4(),

        name: sanitizedThreat,

        category,

        severity,

        date: new Date().toISOString()

      }]);

      setNewThreat("");

      setError(null);

    } catch (err) {

      setError(err.message);

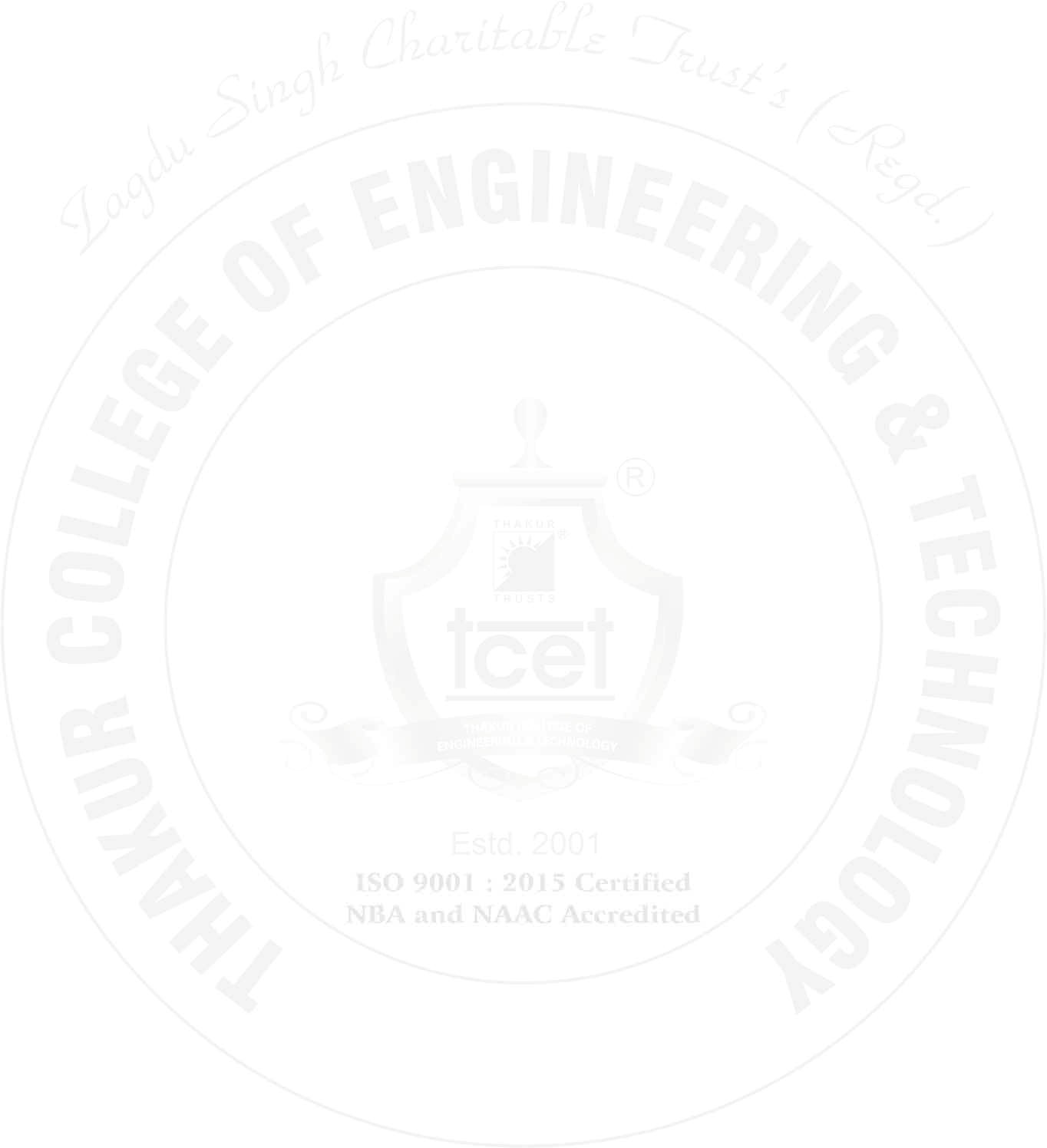
    }

  };

  const handleRemoveThreat = (idToRemove) => {

    setThreats(threats.filter(threat => threat.id !== idToRemove));

  };



  // Memoized calculations

  const threatCategories = useMemo(() =>

    threats.reduce((acc, threat) => {

      acc[threat.category] = (acc[threat.category] || 0) + 1;

      return acc;

    }, {}),

  [threats]);

  const weeklyThreatData = useMemo(() => {

    // In a real app, you would calculate this from actual dates

    return {

      labels: ["Week 1", "Week 2", "Week 3", "Week 4"],

      datasets: CATEGORIES.map(cat => ({

        label: cat.value,

        data: Array(4).fill(0).map(() =>

          Math.floor(Math.random() \* 10) // Replace with real calculation

        ),

        backgroundColor: cat.color,

        borderColor: darkMode ? '#ffffff' : '#000000',

        borderWidth: 1

      }))

    };

  }, [threats, darkMode]);

  const chartData = {

    categories: {

      labels: CATEGORIES.map(cat => cat.value),

      datasets: [{

        label: "Threats by Category",

        data: CATEGORIES.map(cat =>

          threats.filter(t => t.category === cat.value).length

        ),

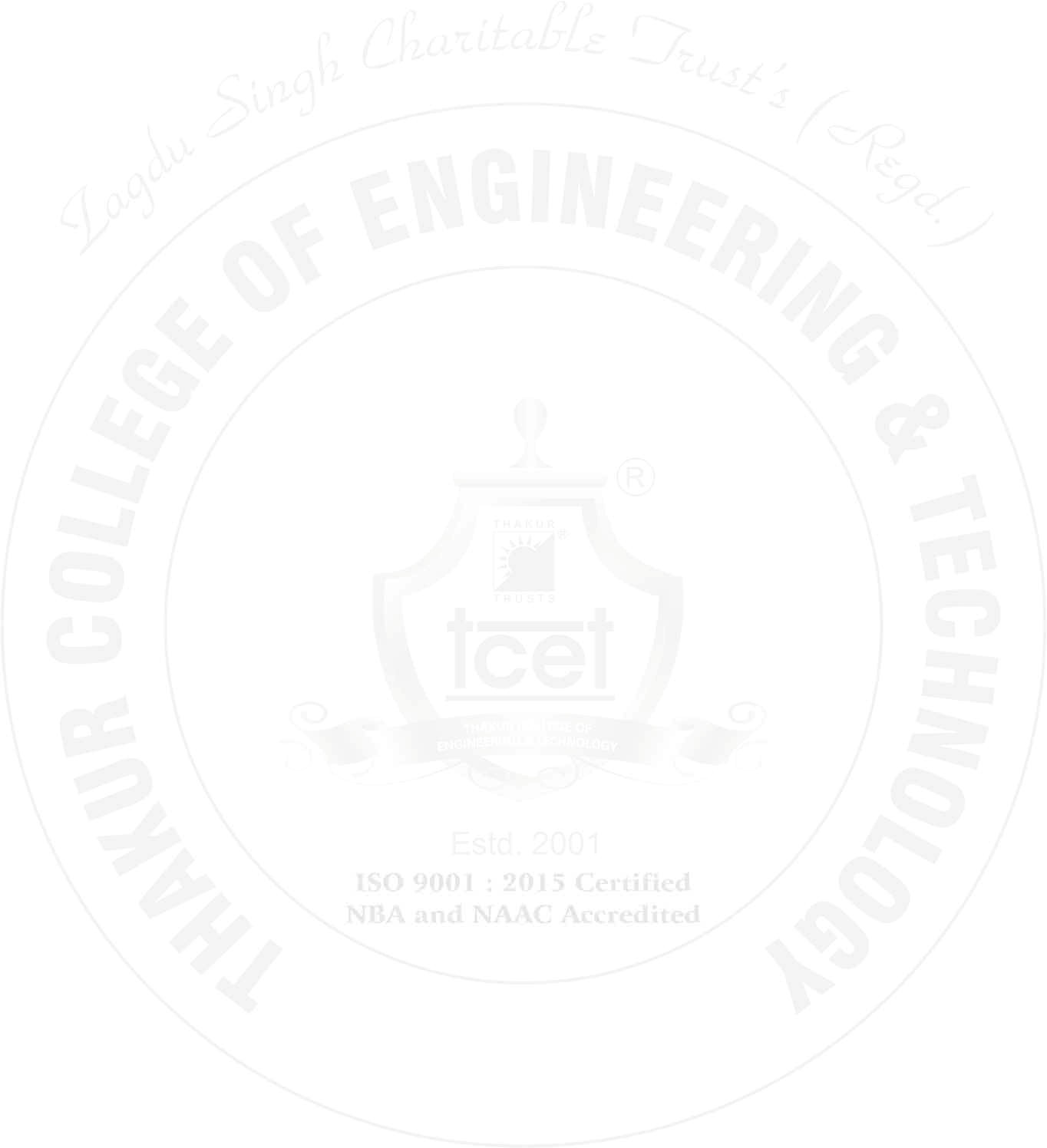
        backgroundColor: CATEGORIES.map(cat => cat.color)

      }]

    },

    weekly: weeklyThreatData

  };

  return (

    <div className={`container ${darkMode ? "dark" : ""}`}>

      <header className="navbar">

        <div className="logo"><h1 className="logo-head">NRS CYberSeCURe</h1></div>

        <nav className="links">

          <a href="#">Dashboard</a>

          <a href="#">Threats</a>

          <a href="#">Analysis</a>

          <a href="#">Training</a>

          <a href="#">Resources</a>

          <button

            onClick={() => setDarkMode(!darkMode)}

            aria-label={`Toggle ${darkMode ? "Light" : "Dark"} Mode`}

            className="toggle-mode"

          >

            {darkMode ? "☀️ Light Mode" : "🌙 Dark Mode"}

          </button>

        </nav>

      </header>

      <main className="main-content">

        <section className="hero">

          <h1>Enhance Your Cybersecurity Knowledge</h1>

          <p>Track, Analyze & Mitigate Security Threats in Real-Time</p>

          <button className="cta-button">Get Started</button>

        </section>

        <section className="threat-form">

          <input

            type="text"

            placeholder="Enter Threat Name"

            value={newThreat}

            onChange={(e) => setNewThreat(e.target.value)}

            maxLength={100}

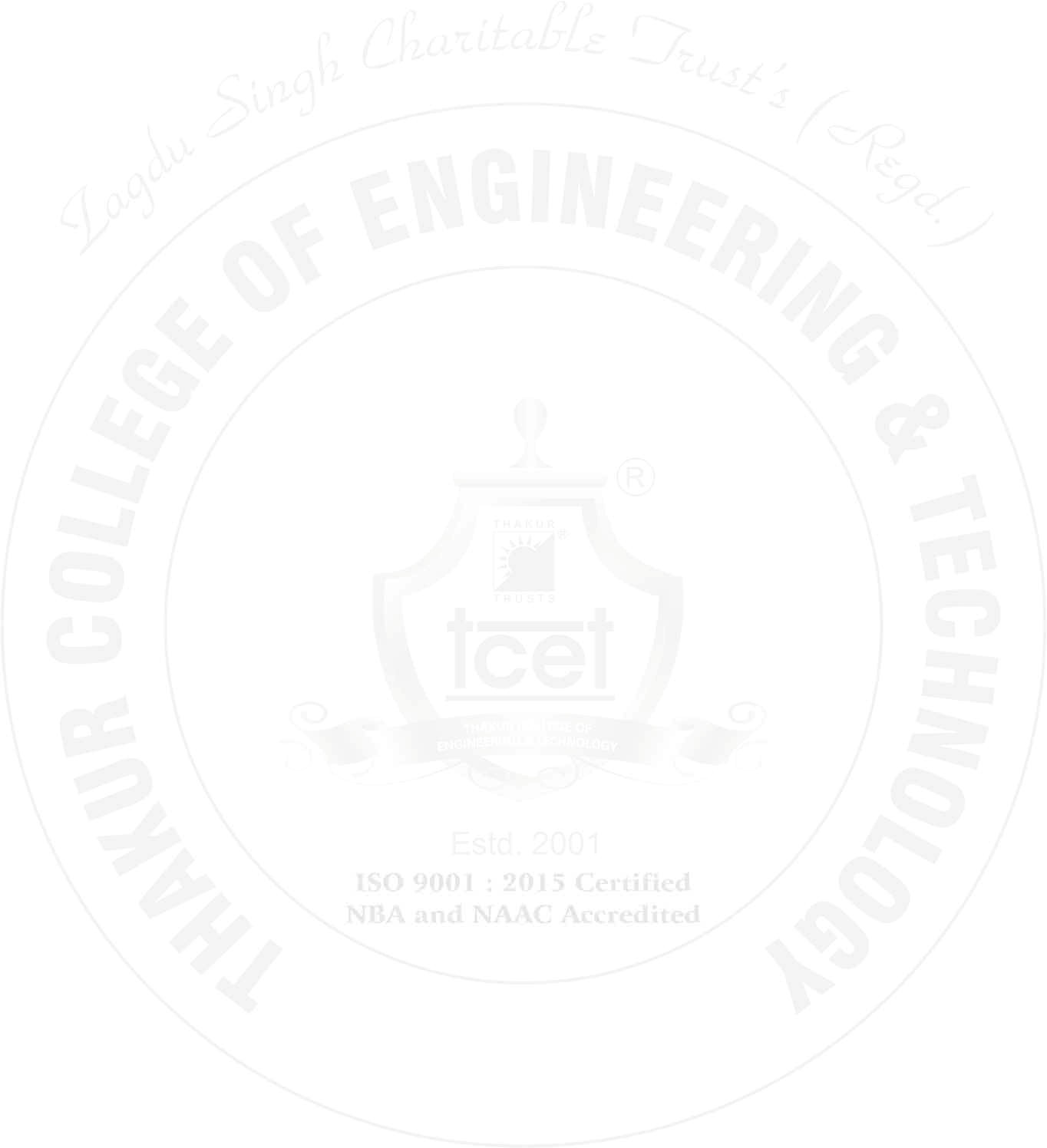
          />

          <select

            value={category}

            onChange={(e) => setCategory(e.target.value)}

            aria-label="Select threat category"

          >

            {CATEGORIES.map((cat) => (

              <option key={cat.value} value={cat.value}>{cat.value}</option>

            ))}

          </select>

          <select

            value={severity}

            onChange={(e) => setSeverity(e.target.value)}

            aria-label="Select threat severity"

          >

            {SEVERITY\_LEVELS.map(level => (

              <option key={level} value={level}>{level}</option>

            ))}

          </select>

          <button onClick={handleAddThreat}>Add Threat</button>

          {error && <div className="error-message">{error}</div>}

        </section>

        <section className="threat-list">

          <h2>Current Threats ({threats.length})</h2>

          <ul>

            {threats.map((threat) => (

              <li key={threat.id} className={`threat-item ${threat.severity.toLowerCase()}`}>

                <span className="threat-name">{threat.name}</span>

                <span className="threat-category">{threat.category}</span>

                <span className="threat-severity">{threat.severity}</span>

                <span className="threat-date">

                  {new Date(threat.date).toLocaleDateString()}

                </span>

                <button

                  className="delete-btn"

                  onClick={() => handleRemoveThreat(threat.id)}

                  aria-label={`Remove threat ${threat.name}`}

                >

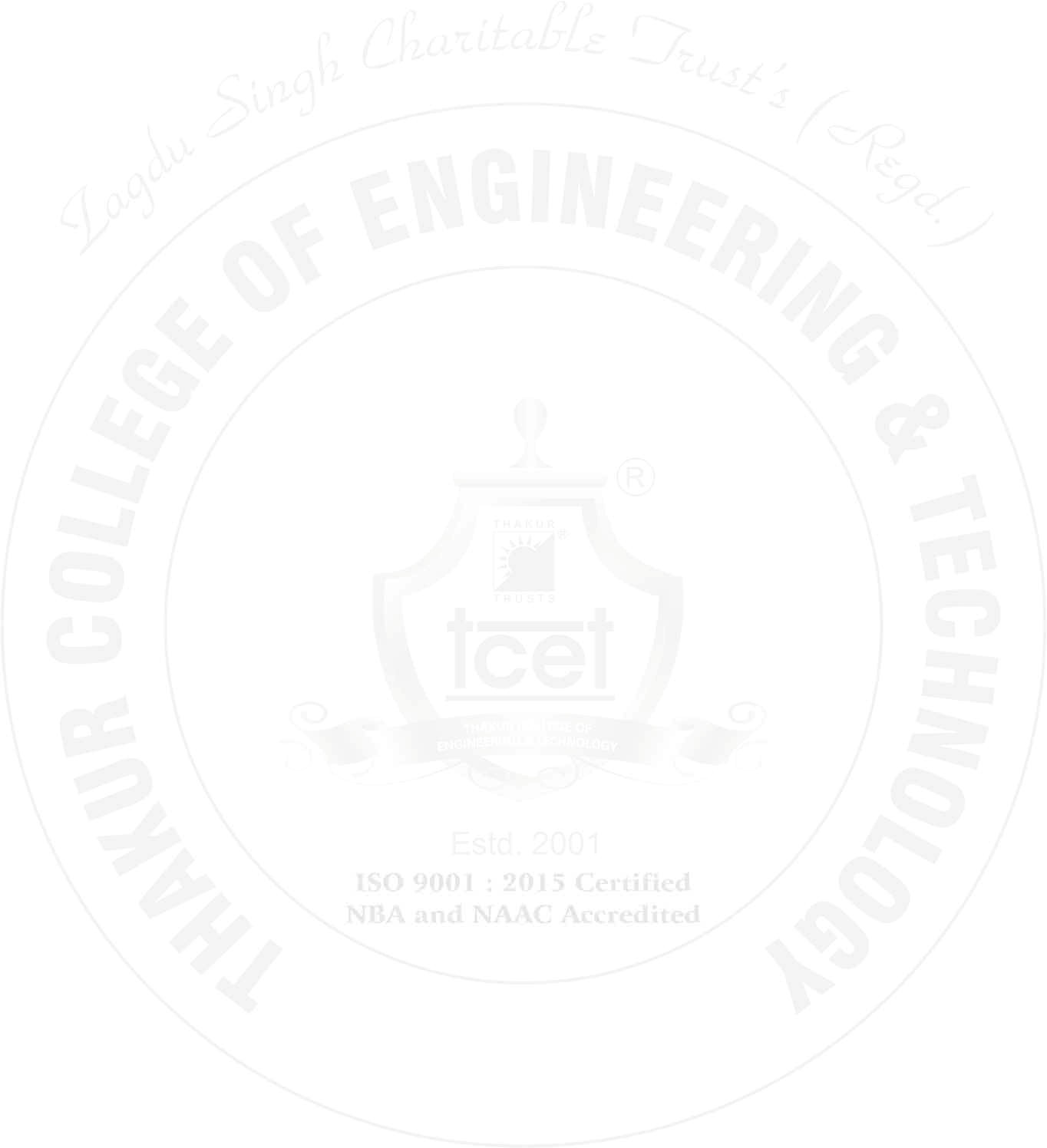
                  Delete

                </button>

              </li>

            ))}

          </ul>

        </section>

        <section className="chart-container">

          <div className="chart-tabs">

            <button

              className={activeChart === 'categories' ? 'active' : ''}

              onClick={() => setActiveChart('categories')}

            >

              By Category

            </button>

            <button

              className={activeChart === 'weekly' ? 'active' : ''}

              onClick={() => setActiveChart('weekly')}

            >

              Weekly Trend

            </button>

          </div>

          <h2>Threat Analysis</h2>

          <Bar data={chartData[activeChart]} />

        </section>

      </main>

      <footer className="footer">

        &copy; 2025 NRS Cybersecurity |

        <a href="#">Made with<span>&#10083;</span> </a> By

        <a href="#">58,59,60</a>

      </footer>

    </div>

  );

};

export default App;

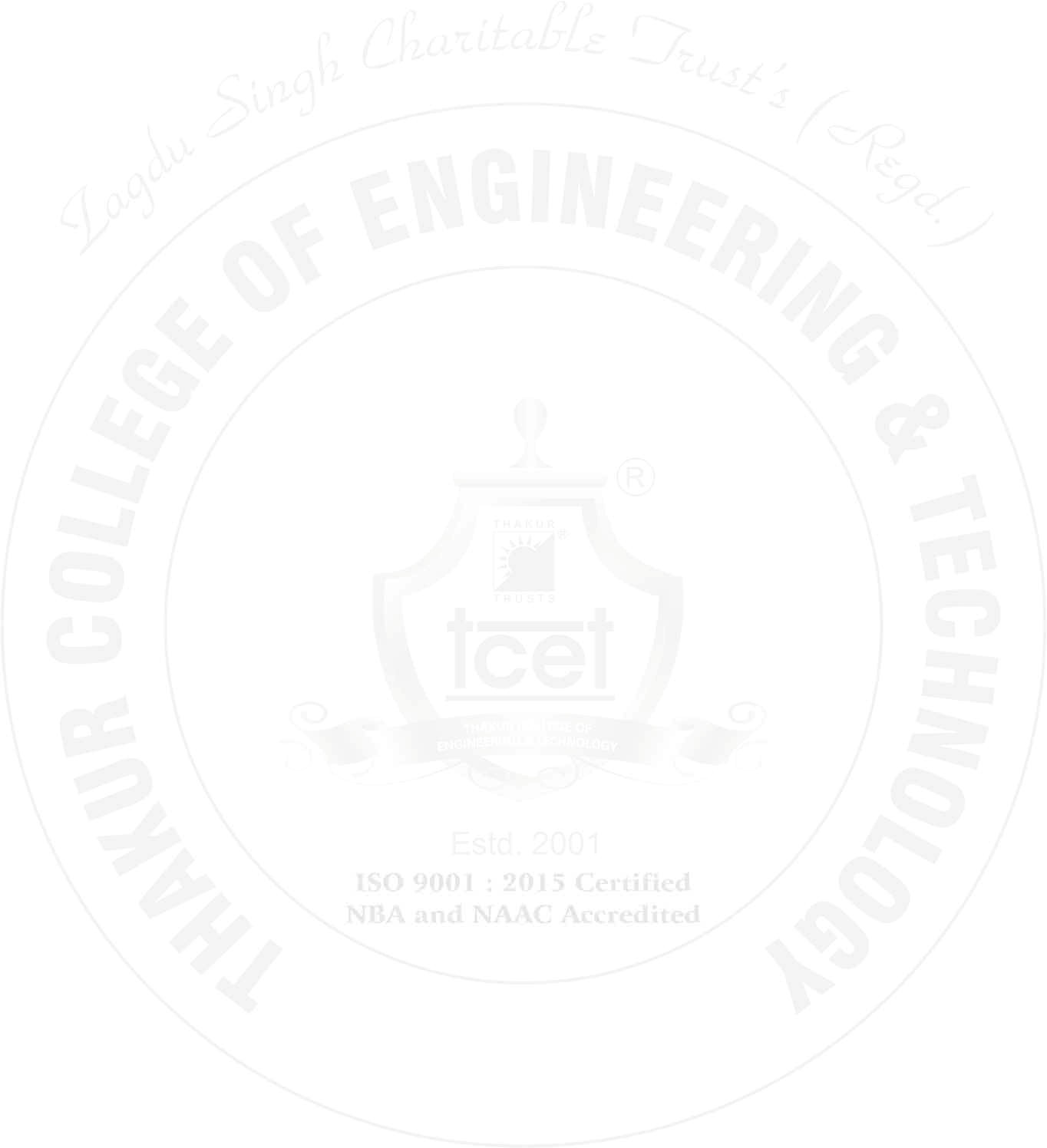
**css:**

/\* === Base Styles === \*/

\* {

  margin: 0;

  padding: 0;

  box-sizing: border-box;

}

body {

  font-family: 'Segoe UI', sans-serif;

  background-color: #f9f9f9;

  color: #222;

  transition: background-color 0.3s, color 0.3s;

}

.container {

  min-height: 100vh;

  display: flex;

  flex-direction: column;

}

/\* === Dark Mode === \*/

.container.dark {

  background-color: #111;

  color: #f0f0f0;

}

.container.dark .navbar {

  background-color: #1a1a1a;

}

.container.dark .links a:hover {

  background-color: rgba(0, 255, 255, 0.2);

  box-shadow: 0 0 10px #00ffee;

}

.container.dark .threat-item {

  background-color: #222;

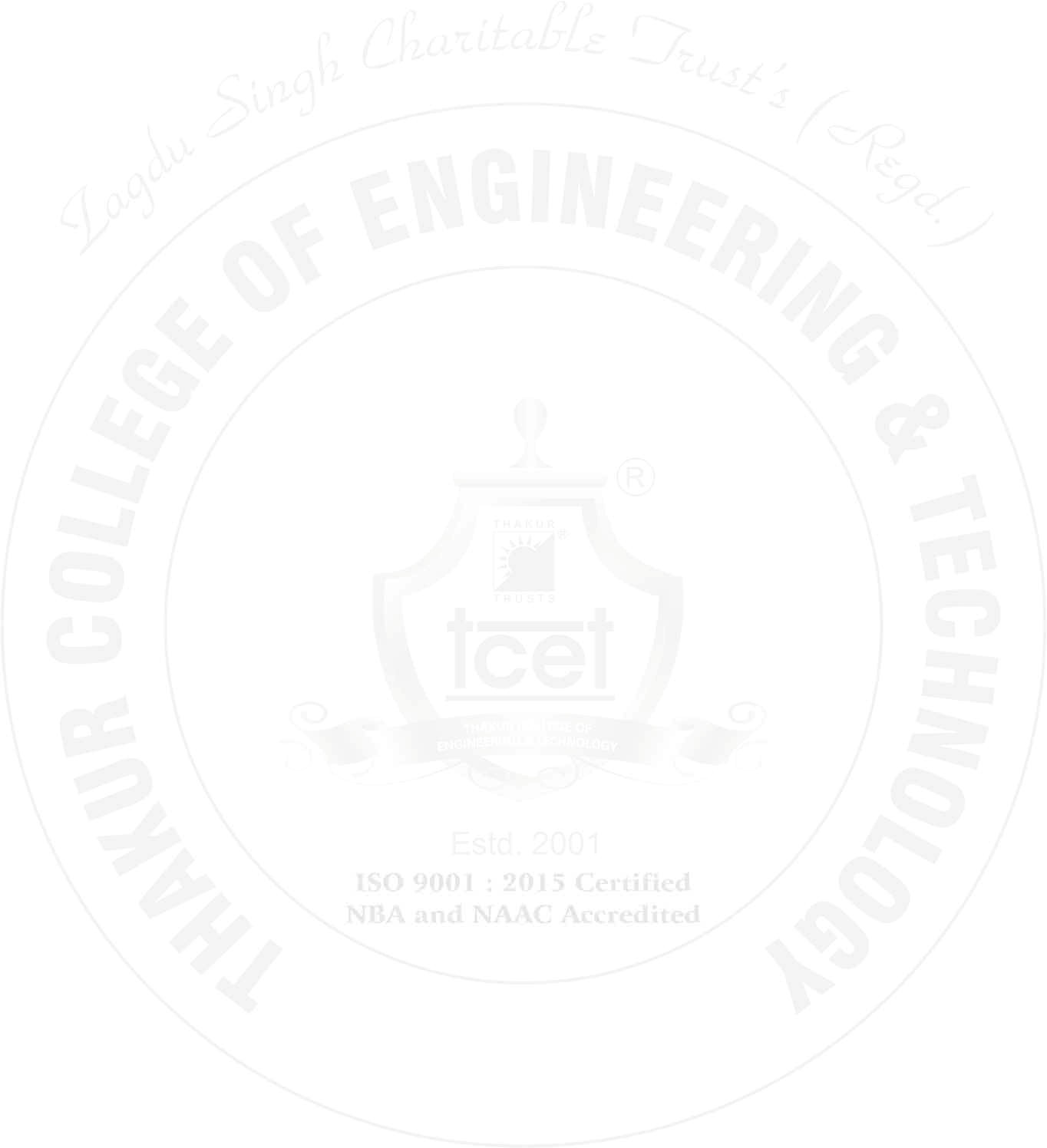
}

.container.dark .footer {

  background-color: #1a1a1a;

}

/\* === Navbar === \*/

.navbar {

  width: 100vw;

  background-color: #222;

  color: #fff;

  display: flex;

  justify-content: space-between;

  align-items: center;

  padding: 1rem 2rem;

  flex-wrap: wrap;

  position: sticky;

  top: 0;

  z-index: 1000;

}

.logo-head {

  font-size: 1.5rem;

  color: #00ffcc;

}

.links {

  display: flex;

  flex-wrap: wrap;

  align-items: center;

  gap: 1rem;

}

.links a {

  text-decoration: none;

  color: inherit;

  padding: 0.5rem 1rem;

  border-radius: 8px;

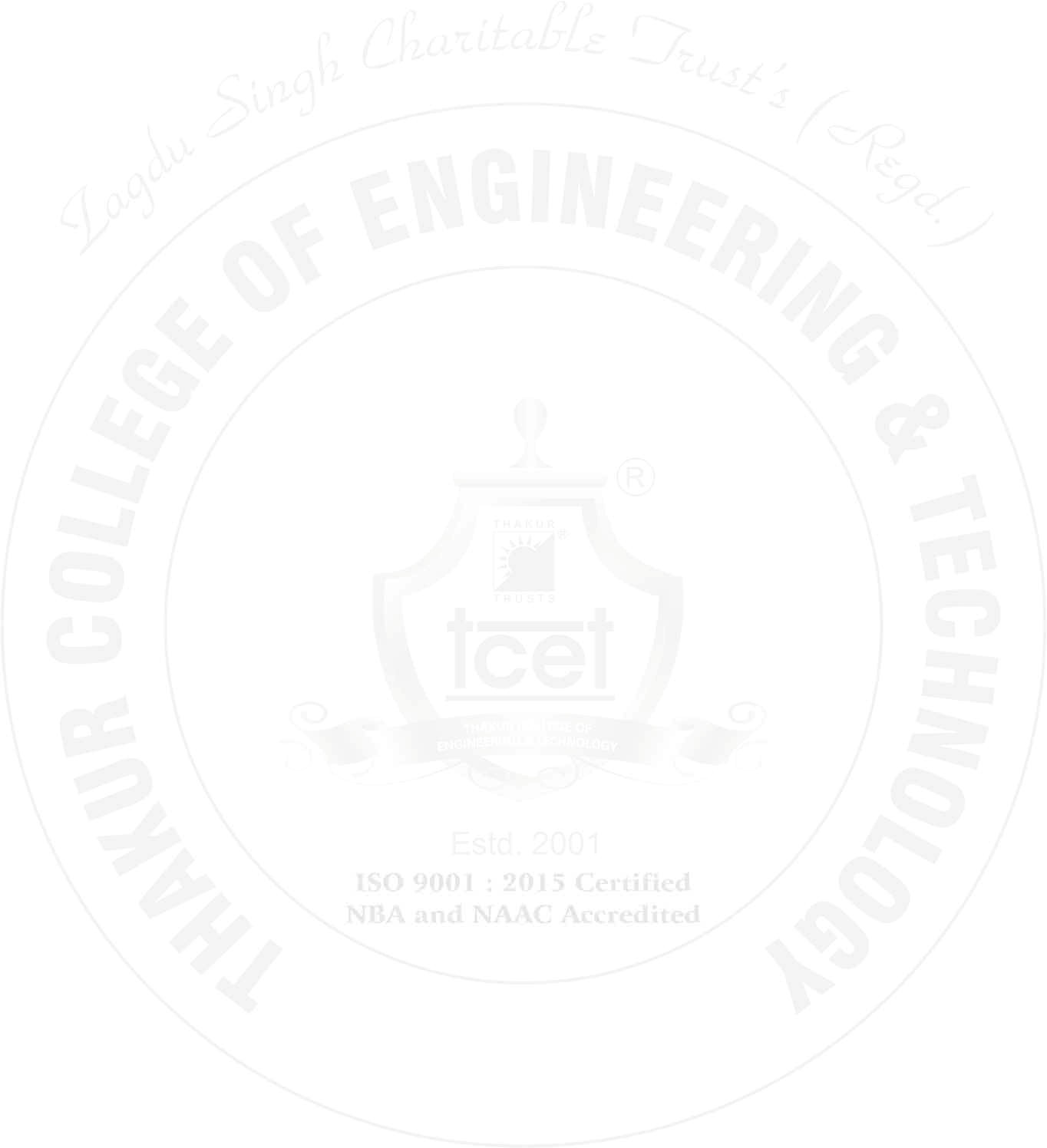
  transition: all 0.3s ease;

}

.links a:hover {

  background-color: rgba(0, 255, 255, 0.1);

  box-shadow: 0 4px 10px rgba(0, 255, 255, 0.3);

}

.toggle-mode {

  padding: 0.5rem 1rem;

  margin-left: 1rem;

  border: 1px solid #00ffcc;

  border-radius: 8px;

  background: transparent;

  color: inherit;

  cursor: pointer;

  transition: all 0.3s ease;

}

.toggle-mode:hover {

  background-color: #00ffcc;

  color: #111;

}

/\* === Hero Section === \*/

.hero {

  text-align: center;

  padding: 4rem 2rem;

  background-color: rgba(255, 255, 255, 0.8);

  backdrop-filter: blur(10px);

  margin: 2rem auto;

  border-radius: 16px;

  max-width: 1000px;

}

.hero h1 {

  font-size: 2rem;

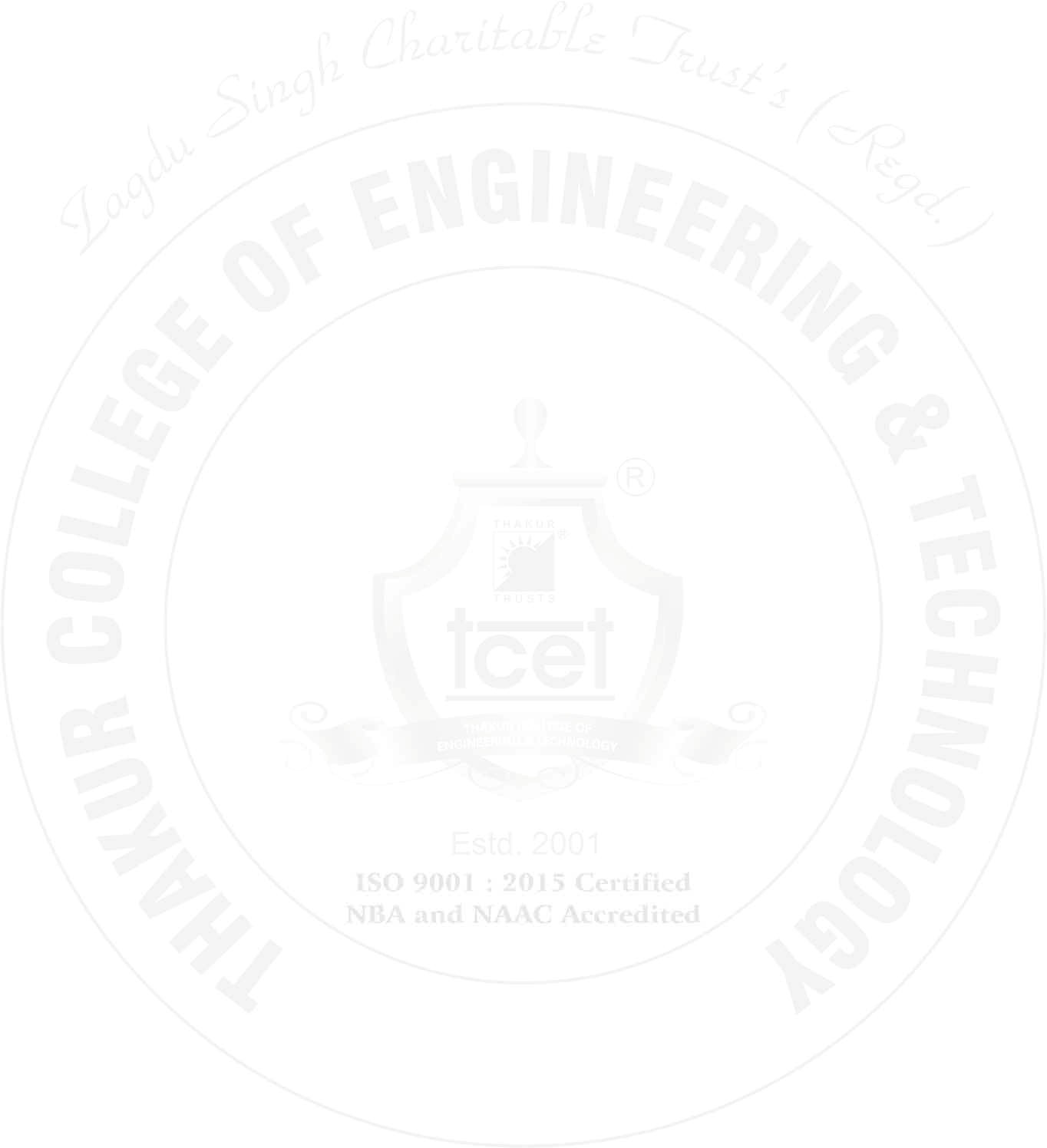
  margin-bottom: 1rem;

}

.cta-button {

  margin-top: 1rem;

  padding: 0.75rem 2rem;

  font-size: 1rem;

  background-color: #007bff;

  border: none;

  color: white;

  border-radius: 8px;

  cursor: pointer;

  transition: background-color 0.3s ease;

}

.cta-button:hover {

  background-color: #0056b3;

}

/\* === Form Section === \*/

.threat-form {

  display: flex;

  flex-wrap: wrap;

  justify-content: center;

  gap: 1rem;

  margin: 2rem auto;

  padding: 1rem;

  max-width: 900px;

}

.threat-form input,

.threat-form select,

.threat-form button {

  padding: 0.5rem 1rem;

  font-size: 1rem;

  border-radius: 8px;

  border: 1px solid #ccc;

}

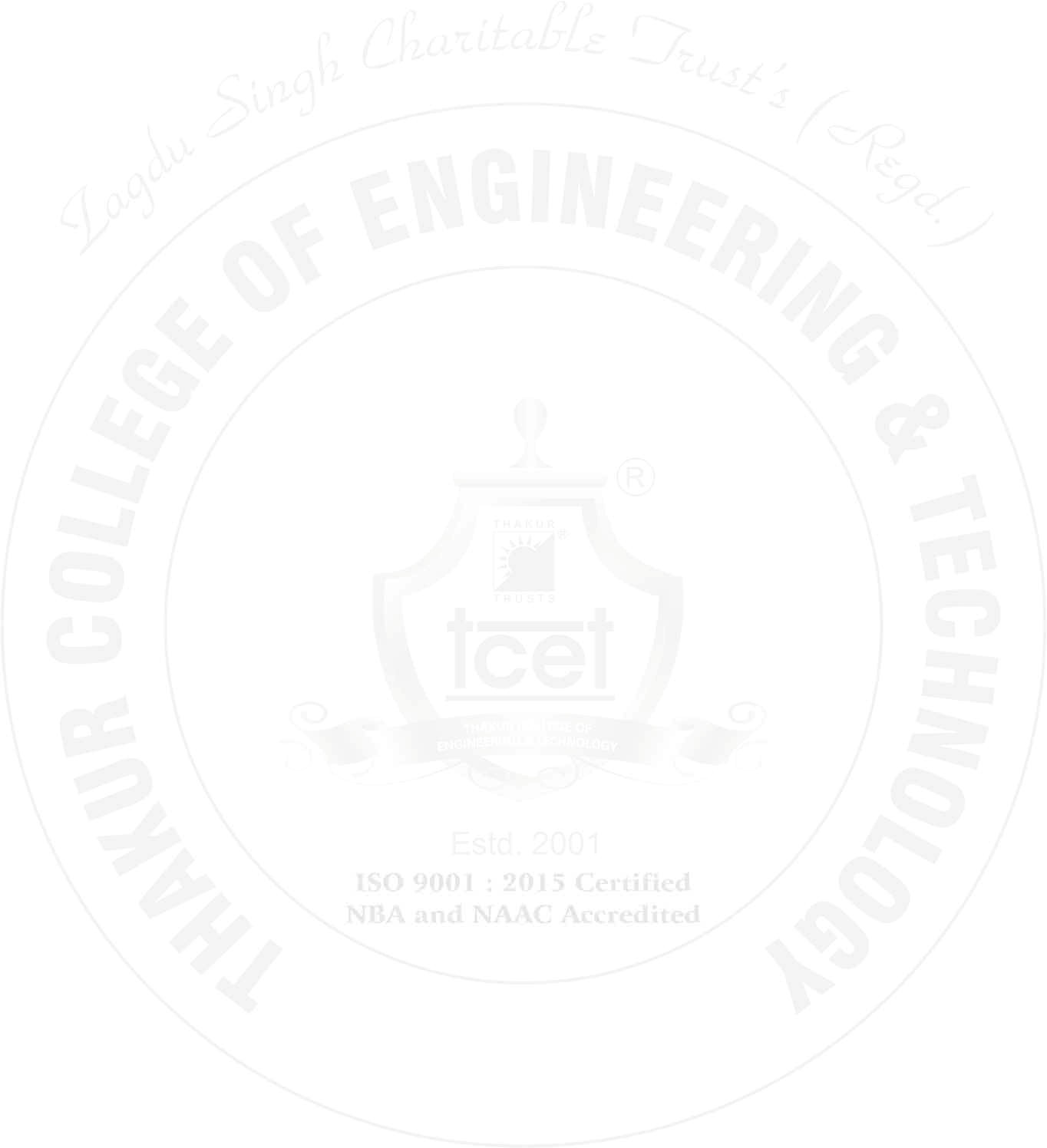
.threat-form button {

  background-color: #28a745;

  color: white;

  border: none;

}

.threat-form button:hover {

  background-color: #218838;

}

.error-message {

  color: red;

  margin-top: 1rem;

  text-align: center;

}

/\* === Threat List === \*/

.threat-list {

  max-width: 1000px;

  margin: 2rem auto;

  padding: 1rem;

}

.threat-item {

  display: flex;

  flex-wrap: wrap;

  justify-content: space-between;

  gap: 1rem;

  background-color: #f2f2f2;

  padding: 1rem;

  border-radius: 10px;

  margin-bottom: 0.5rem;

  font-size: 0.95rem;

}

.threat-name {

  font-weight: bold;

  flex: 1 1 200px;

}

.delete-btn {

  padding: 0.4rem 0.8rem;

  background-color: #dc3545;

  border: none;

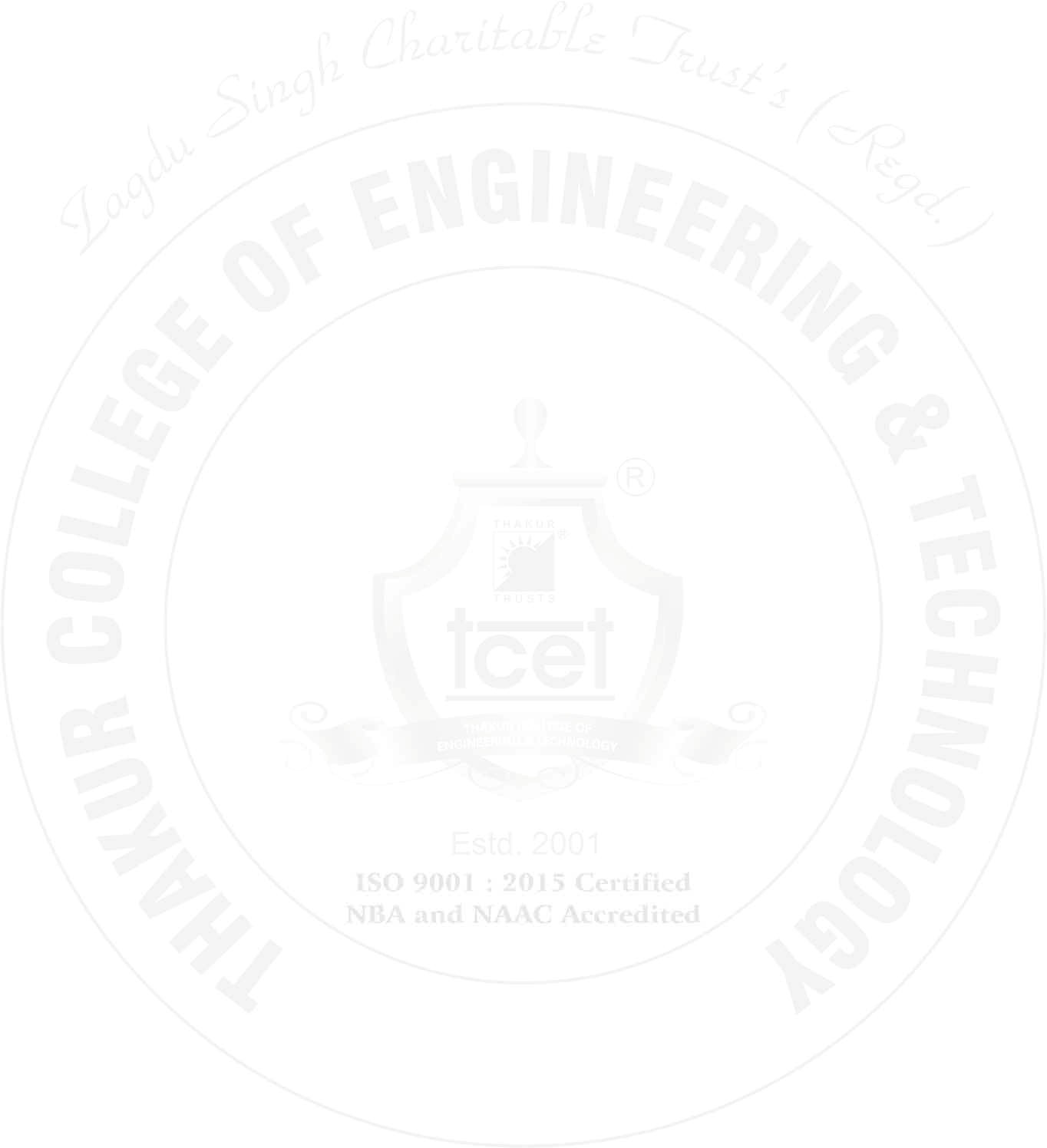
  color: white;

  border-radius: 6px;

  cursor: pointer;

}

.delete-btn:hover {

  background-color: #c82333;

}

/\* === Chart Section === \*/

.chart-container {

  max-width: 1000px;

  margin: 2rem auto;

  padding: 1rem;

}

.chart-tabs {

  display: flex;

  justify-content: center;

  gap: 1rem;

  margin-bottom: 1rem;

}

.chart-tabs button {

  padding: 0.5rem 1rem;

  border: none;

  border-radius: 8px;

  background: #eee;

  cursor: pointer;

  transition: 0.3s;

}

.chart-tabs .active {

  background-color: #00ffcc;

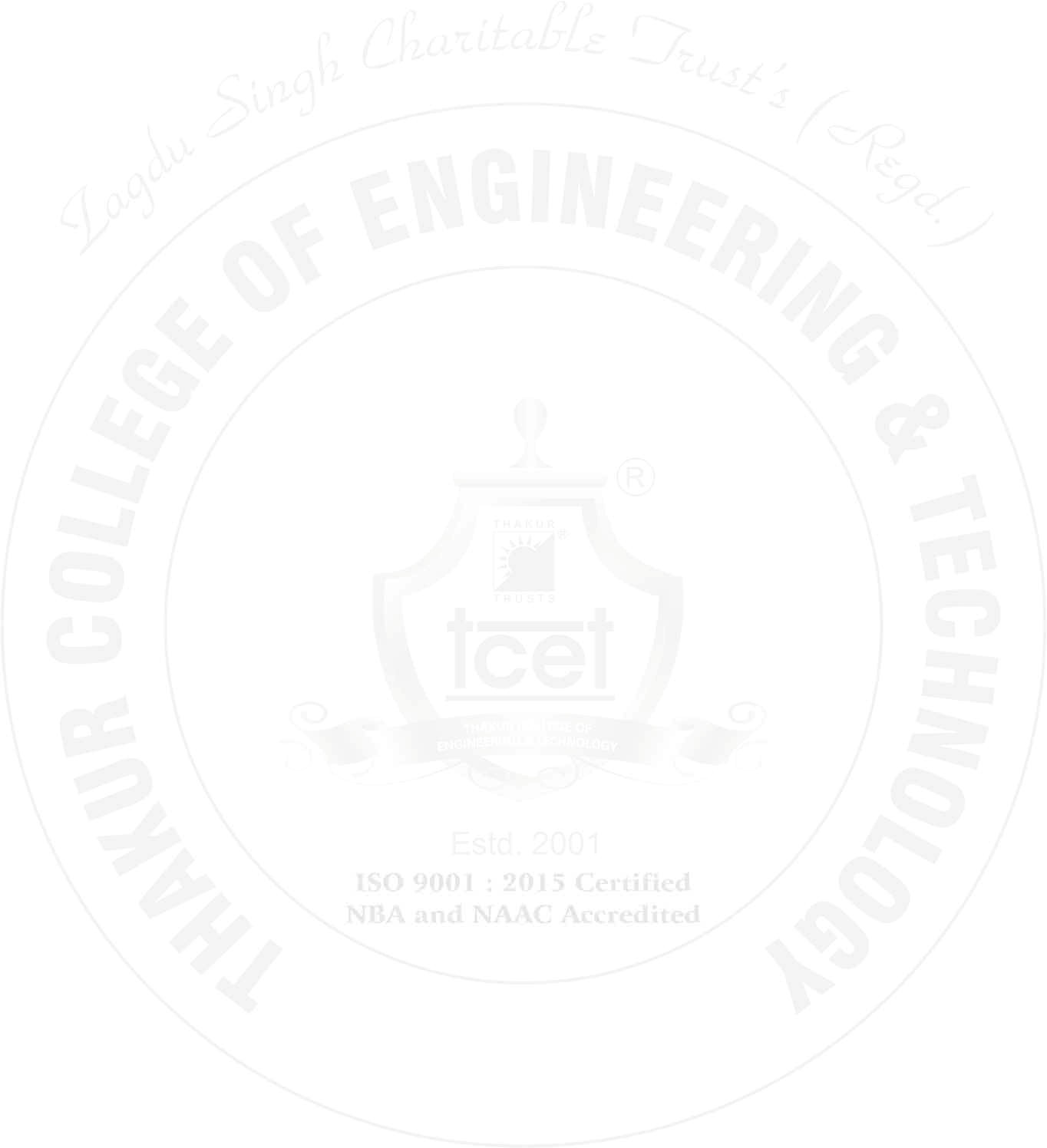
  color: #000;

  font-weight: bold;

}

/\* === Footer === \*/

.footer {

  text-align: center;

  padding: 2rem;

  background-color: #222;

  color: #fff;

  margin-top: auto;

}

.footer a {

  color: #1fd7c5;

  margin: 0 0.5rem;

  text-decoration: none;

}

.footer span{

  color: red;

}

.footer a:hover {

  text-decoration: underline;

}

/\* === Responsive === \*/

@media (max-width: 768px) {

  .threat-item {

    flex-direction: column;

    align-items: flex-start;

  }

  .links {

    flex-direction: column;

    align-items: flex-start;

  }

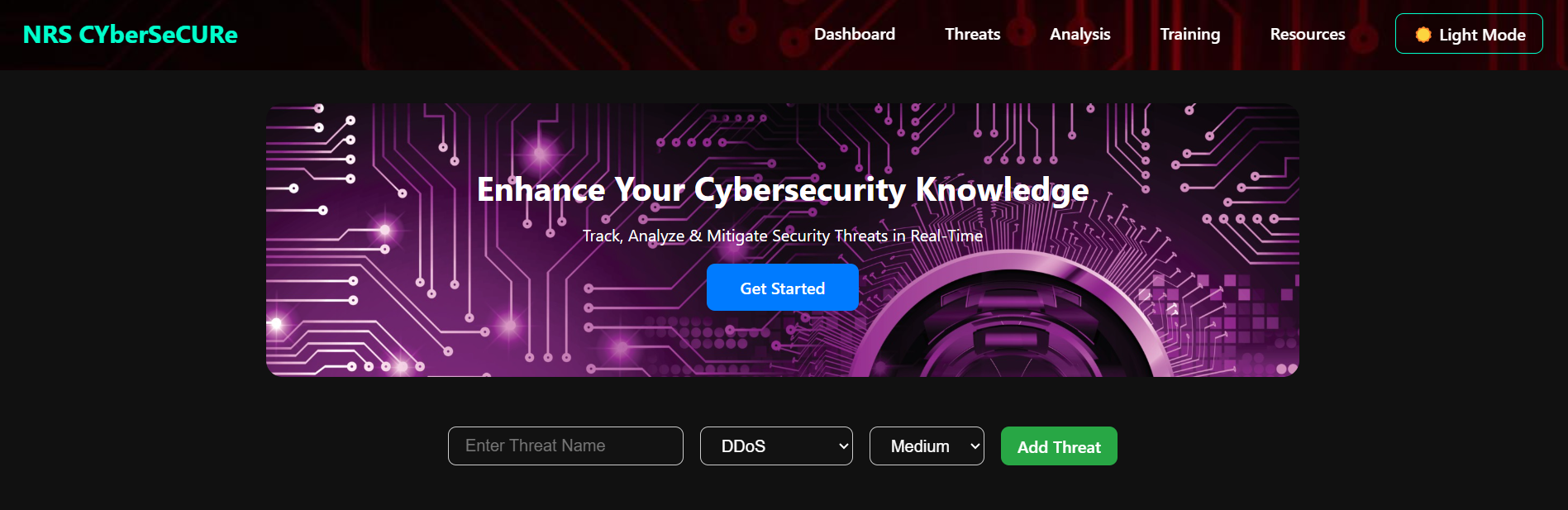
  .hero {

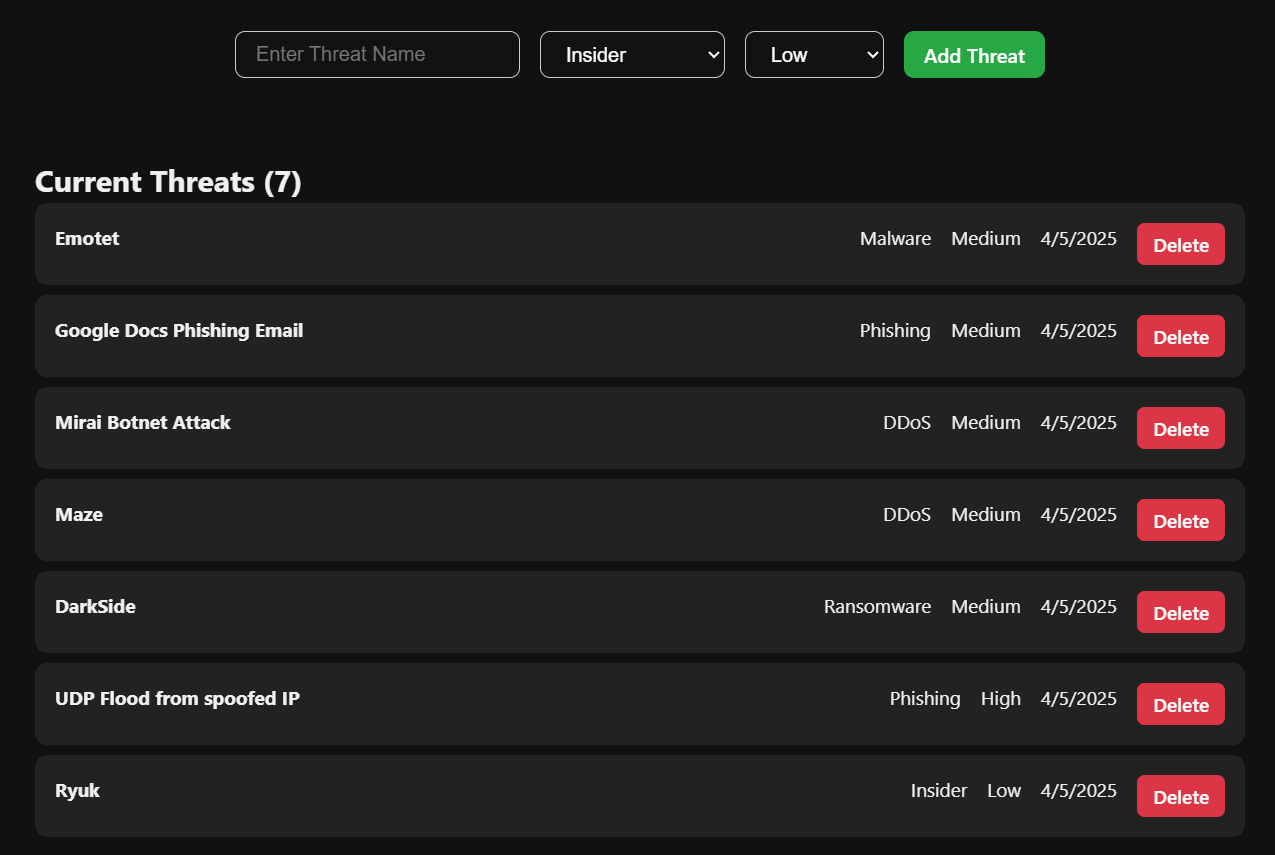
    padding: 2rem 1rem;

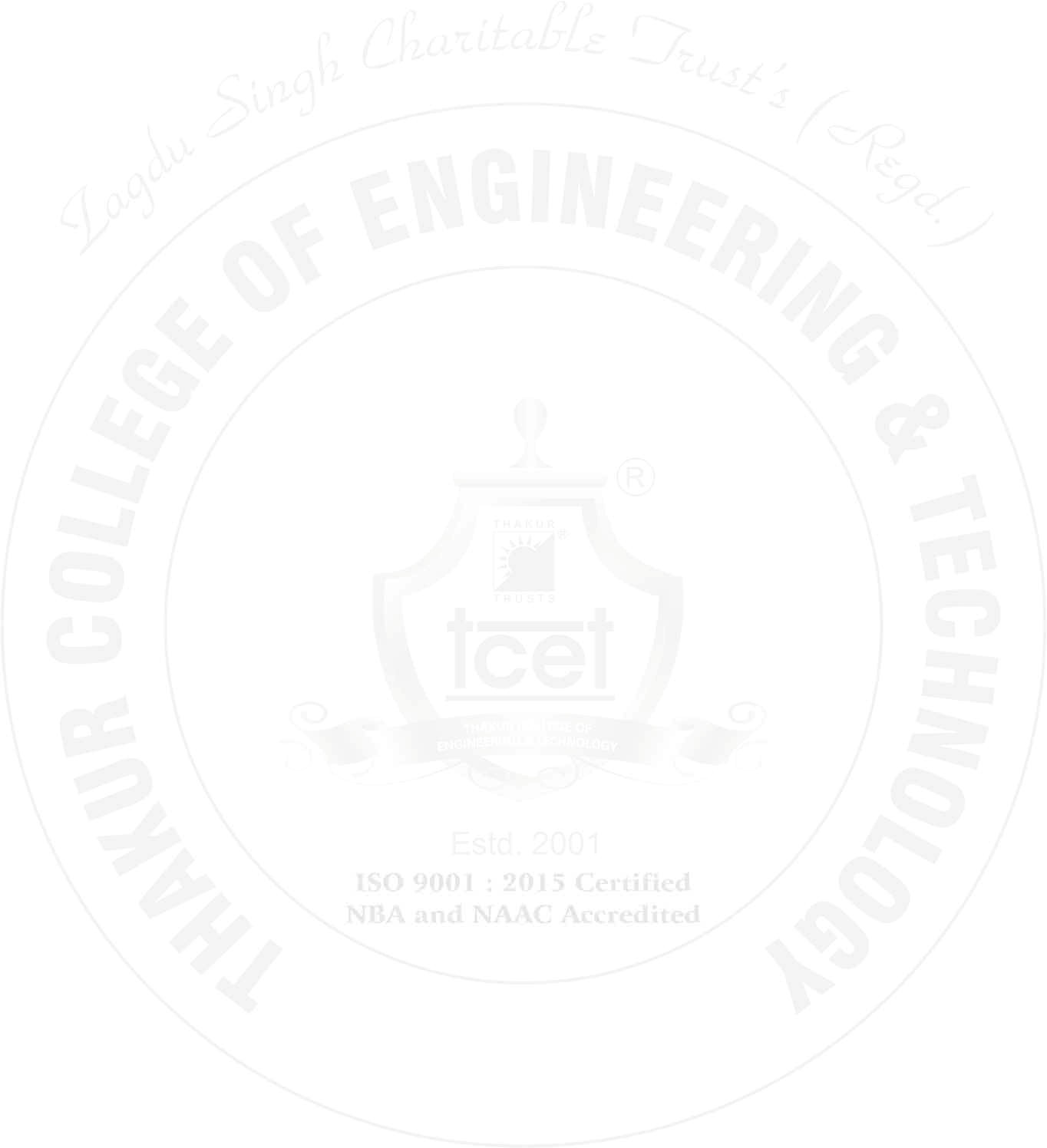
  }

}

**Output:**

****

****

****