Name – Pratham Raval

Reg No – 23BCI0147

Course Code - BCSE203E

Course Title - Web Programming

Slot - L37+L38+ L53+L54

Assessment 5- Mini Project

1. Given Problem Statement

The problem addressed by this system is the inefficiency and potential for errors in manual cash handling during physical Monopoly board games. Traditional Monopoly gameplay involves:

- Physical cash transactions between players

- Manual calculation of rents, taxes, and fines

- Paper-based tracking of property ownership

- Time-consuming bank operations that slow down gameplay

- Human errors in calculations that can disrupt game fairness

2. Problem Description

The Monopoly Bank Management System solves these challenges by:

- **Digitizing all financial transactions** to eliminate physical cash handling

- **Automating calculations** for rents, taxes, and fines to prevent errors

- **Tracking property ownership** digitally with visual indicators

- **Managing player balances** in real-time with intuitive interfaces

- **Providing transaction history** through local Storage persistence

- **Streamlining complex operations** like property management and rent payments

3. Modules Details and Key Features

3.1 Core Modules

Player Management Module

- Add/Remove Players: Create new player profiles with unique IDs

- Balance Tracking: Real-time display of player funds (₹ currency)

Property Management Module

- Property Database: Complete Indian-themed property list (28 properties)
- Group Organization: Color-coded by property groups (Brown, Light Blue, etc.)
- Ownership Tracking: Clear display of property owners (Bank or player)
- Buy/Sell Functionality: Integrated with player balances
- Rent Management: Automatic rent calculation and payment processing

Banking Operations Module

- Player-to-Player Transactions: Secure money transfers between players
- Bank Payments: Handling taxes, fines, and fees
- Bulk Payments: Pay all players simultaneously
- Transaction Verification: Confirmation dialogs for all financial operations

Game Management Module

- Data Persistence: localStorage for saving game state
- Reset Functionality: Secure game reset with passkey protection
- Rule Reference: Detailed game rules and card effects
- Responsive Design: Works on desktop and mobile devices using media queries

3.2 Special Features

Advanced Property System

- Complete property list with Indian city names instead of traditional Monopoly locations
- Color-coded groups with visual indicators
- Rent prices based on property value
- Buy/Sell functionality with automatic price calculation (sell at 50% of purchase)

Transaction System

- Multiple transaction types:
  - Player-to-player transfers
  - Bank payments (taxes, fines)

  - Property purchases
  - Rent payments
- Reason tracking for transactions (optional notes)
- Balance validation to prevent overdrafts

Security Feature

- Passkey-protected reset (default: "0000")
- localStorage encryption for game data
- Confirmation dialogs for critical actions

User Experience

- Modern UI with gradient buttons and card-based design
- Responsive layout that works on all devices
- Animated interactions (hover effects, transitions)
- Custom scrollbars and smooth scrolling
- Font Awesome icons for intuitive navigation

3.3 Tech Stack

Frontend

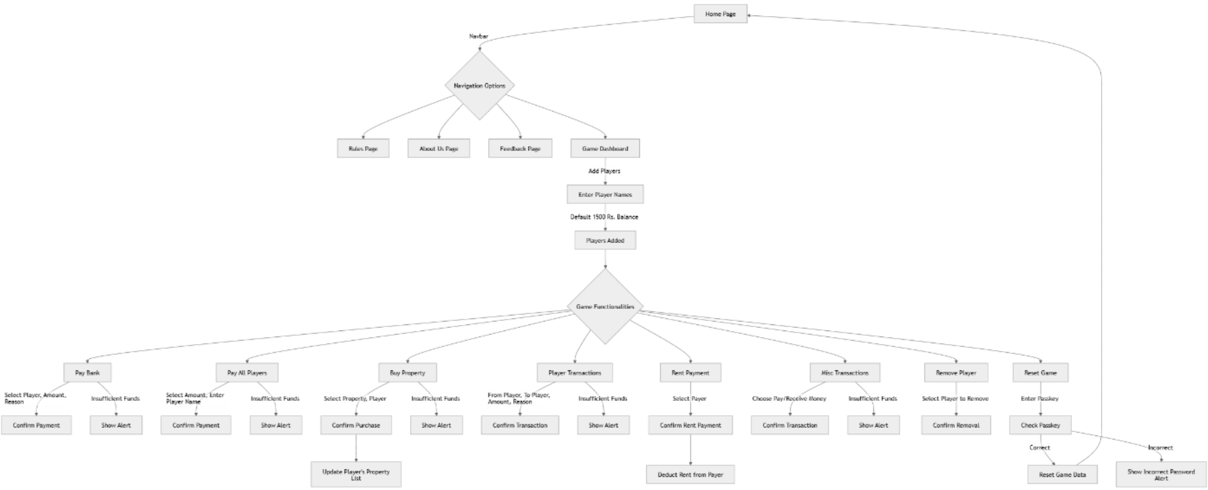
- React.js for main application structure using Routing and Hooks
- HTML/CSS for game interface
- JavaScript for game logic
- localStorage for data persistence
- Responsive design using media queries

Game Engine

- Object-oriented design (Player and Property classes)
- State management through localStorage
- Event-driven architecture for user interactions
- Modular code organization for maintainability

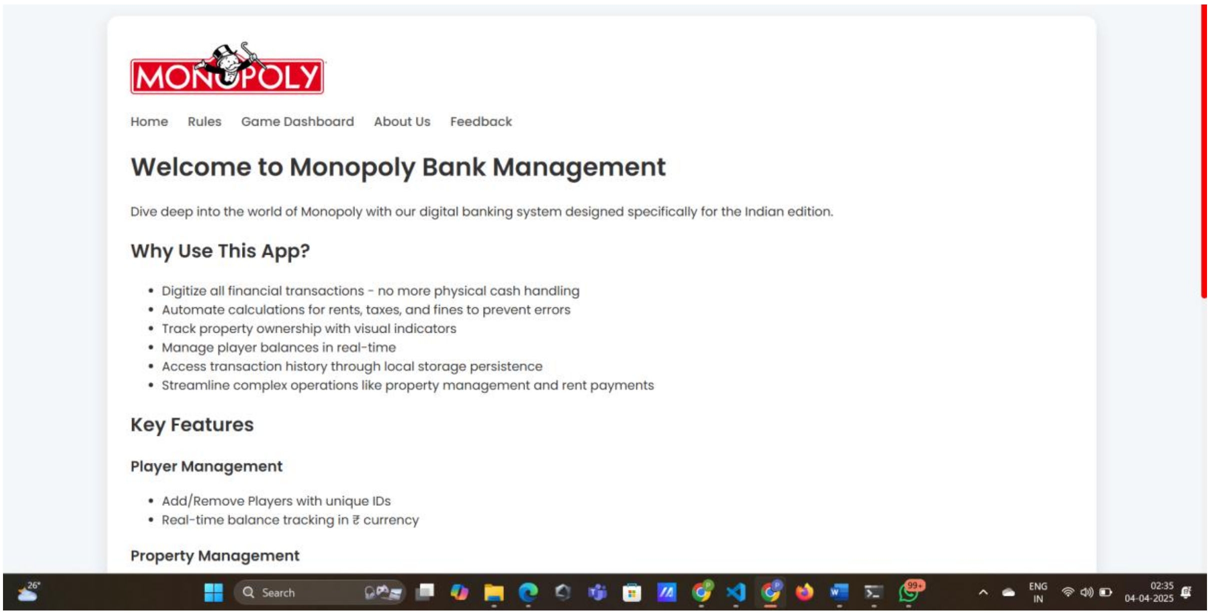
This system effectively solves the problems of manual Monopoly banking by providing a digital, error-free alternative that enhances gameplay speed and fairness while maintaining all the strategic elements of the classic game.

4. Design Layout



5. User Interface

Home Page:



Welcome to Monopoly Bank Management

Dive deep into the world of Monopoly with our digital banking system designed specifically for the Indian edition.

Why Use This App?

- Digitize all financial transactions - no more physical cash handling
- Automate calculations for rents, taxes, and fines to prevent errors
- Track property ownership with visual indicators
- Manage player balances in real-time
- Access transaction history through local storage persistence
- Streamline complex operations like property management and rent payments

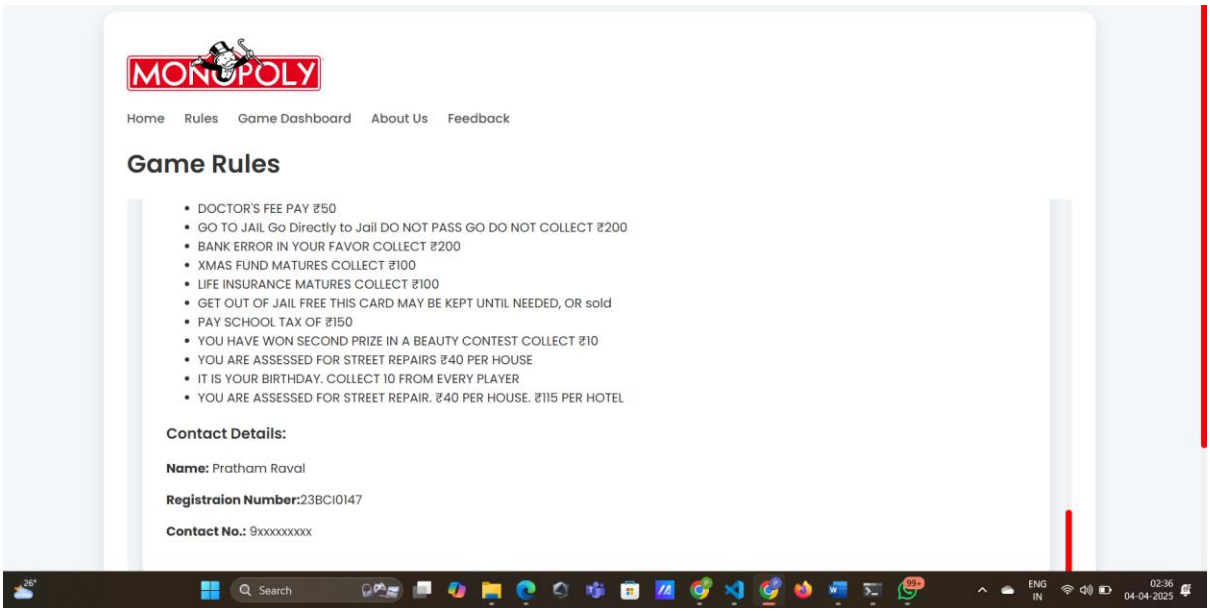
Key Features

Player Management

- Add/Remove Players with unique IDs
- Real-time balance tracking in ₹ currency

Property Management

Rules Page:



Game Rules

- DOCTOR'S FEE PAY ₹50
- GO TO JAIL Go Directly to Jail DO NOT PASS GO DO NOT COLLECT ₹200
- BANK ERROR IN YOUR FAVOR COLLECT ₹200
- XMAS FUND MATURES COLLECT ₹100
- LIFE INSURANCE MATURES COLLECT ₹100
- GET OUT OF JAIL FREE THIS CARD MAY BE KEPT UNTIL NEEDED, OR sold
- PAY SCHOOL TAX OF ₹150
- YOU HAVE WON SECOND PRIZE IN A BEAUTY CONTEST COLLECT ₹10
- YOU ARE ASSESSED FOR STREET REPAIRS ₹40 PER HOUSE
- IT IS YOUR BIRTHDAY. COLLECT 10 FROM EVERY PLAYER
- YOU ARE ASSESSED FOR STREET REPAIR. ₹40 PER HOUSE. ₹115 PER HOTEL

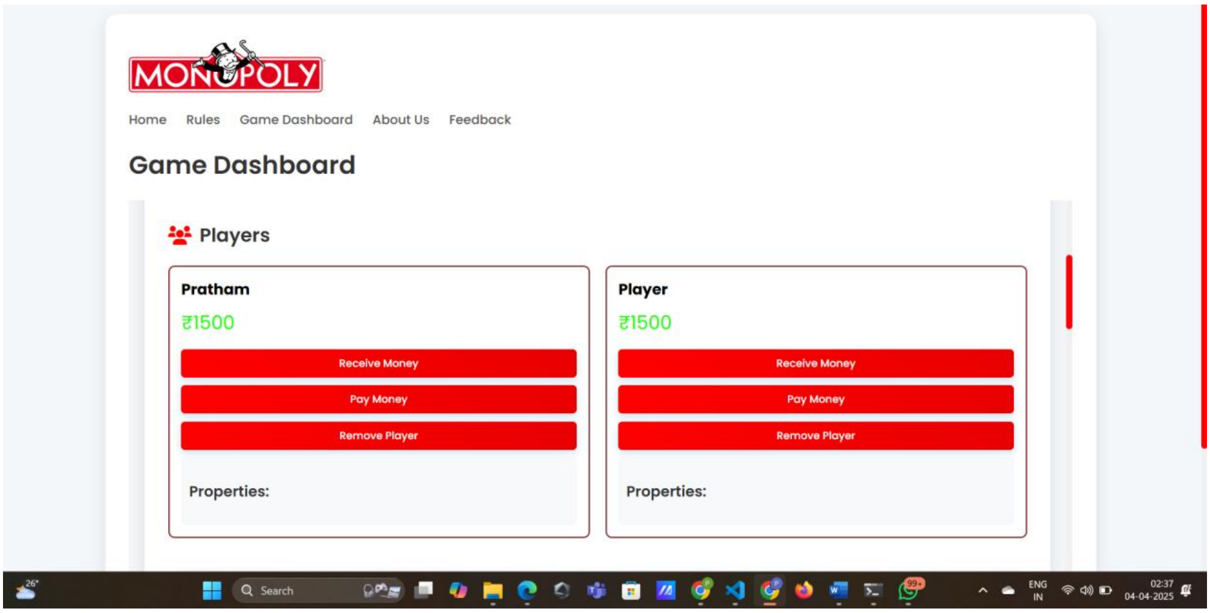
Contact Details:

Name: Pratham Raval

Registraion Number:23BCI0147

Contact No.: 9xxxxxxxxx

Game Dashboard Page:



Game Dashboard

Players

Pratham

₹1500

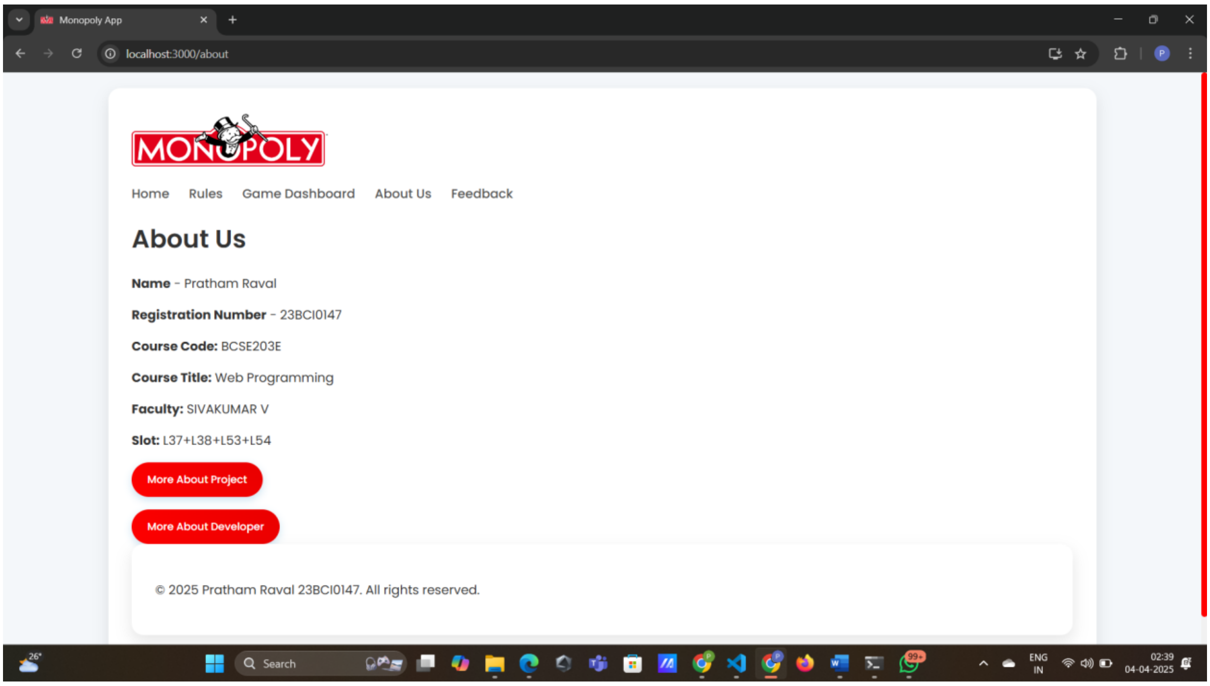
Properties:

Player

₹1500

Properties:

About Us page:



About Us

Name - Pratham Raval

Registration Number - 23BCI0147

Course Code: BCSE203E

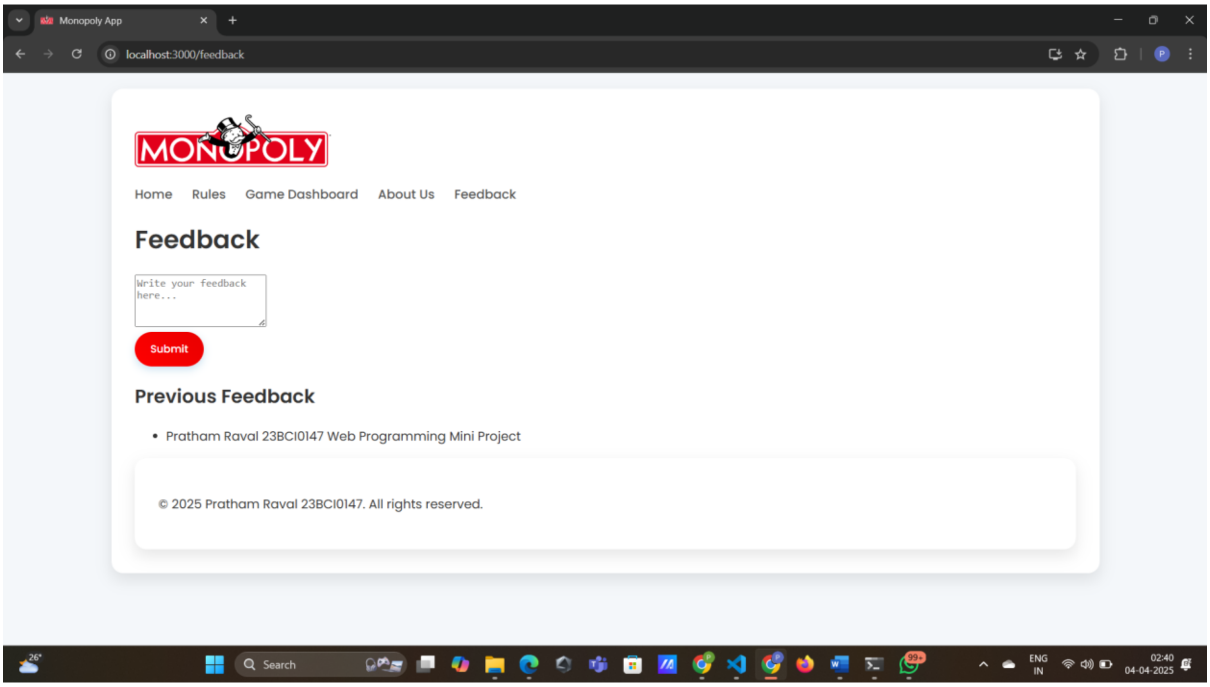
Course Title: Web Programming

Faculty: SIVAKUMAR V

Slot: L37+L38+L53+L54

© 2025 Pratham Raval 23BCI0147. All rights reserved.

Feedback Page:



Feedback

Previous Feedback

- Pratham Raval 23BCI0147 Web Programming Mini Project

© 2025 Pratham Raval 23BCI0147. All rights reserved.

6. Source Code

6.1 monopoly/src/App.js

```
import React from "react";
import { BrowserRouter as Router, Routes, Route, Link } from "react-router-dom";
import Home from "./pages/Home";
import Rules from "./pages/Rules";
import Bank from "./pages/Bank";
import About from "./pages/About";
import Feedback from "./pages/Feedback";
import "./style.css";
import logo from "./Logo.png";
import Footer from "./pages/Footer";

function App() {
  return (
    <Router>
      <div className="container min-h-screen flex flex-col">
    <nav className="navbar">
    <img src={logo} alt="Monopoly Logo" className="logo-img" />
     <ul>
      <li><Link to="/">Home</Link></li>
      <li><Link to="/rules">Rules</Link></li>
      <li><Link to="/bank">Game Dashboard</Link></li>
      <li><Link to="/about">About Us</Link></li>
      <li><Link to="/feedback">Feedback</Link></li>
     </ul>
    </nav>

    <div className="flex-grow">
     <Routes>
      <Route path="/" element={<Home />} />
      <Route path="/rules" element={<Rules />} />
      <Route path="/bank" element={<Bank />} />
      <Route path="/about" element={<About />} />
      <Route path="/feedback" element={<Feedback />} />
     </Routes>
    </div>

    <Footer />
   </div>
  </Router>
 );
}

export default App;
```

6.2 /monopoly/src/style.css

```
@import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@400;500;600;700&display=swap');

.container {
  max-width: 1200px;
```

```
   margin: 0 auto;

   background-color: white;

   padding: 30px;

   border-radius: 15px;

   box-shadow: 0 8px 20px rgba(0,0,0,0.1);

}

.logo {

   width: 200px;

   height: auto;

   display: flex;


  }


header {

   display: flex;

   justify-content: space-between;

   align-items: center;

   margin-bottom: 30px;

   padding-bottom: 20px;

   border-bottom: 2px solid #f0f0f0;

}


.logo {

   display: flex;

   align-items: center;

   gap: 15px;

}


.logo-img{

   width :250px;

}


.logo h1 {

   margin: 0;

   font-size: 1.8rem;
```

```
    background: linear-gradient(90deg, #ff0000, #e90000);
    -webkit-background-clip: text;
    -webkit-text-fill-color: transparent;
}

nav ul {
    display: flex;
    list-style: none;
    gap: 25px;
    margin: 0;
    padding: 0;
}

nav a {
    text-decoration: none;
    color: #555;
    font-weight: 500;
    padding: 5px 0;
    position: relative;
    transition: color 0.3s;
}

nav a:hover, nav a.active {
    color: #ff1b1b;
}

nav a:after {
    content: '';
    position: absolute;
    width: 0;
    height: 2px;
    bottom: 0;
    left: 0;
    background-color: #000000;
    transition: width 0.3s;
```

```
}

nav a:hover:after, nav a.active:after {
    width: 100%;
}

.page-title {
    text-align: center;
    margin-bottom: 30px;
    position: relative;
    padding-bottom: 15px;
}

.page-title:after {
    content: '';
    position: absolute;
    width: 100px;
    height: 3px;
    bottom: 0;
    left: 50%;
    transform: translateX(-50%);
    background: linear-gradient(90deg, #ff0000, #e90000);
}

.input-group {
    display: flex;
    gap: 10px;
    margin-bottom: 30px;
    box-shadow: 0 4px 12px rgba(0,0,0,0.05);
    border-radius: 30px;
    padding: 5px;
    background-color: #f9f9f9;
}

.input-group input {
    border-radius: 25px;
    border: none;
    background-color: transparent;
    padding: 12px 20px;
    flex-grow: 1;
}

.action-buttons {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(150px, 1fr));
    gap: 15px;
    margin-bottom: 40px;
}

.action-button {
    display: flex;
    flex-direction: column;
    align-items: center;
    gap: 10px;
    padding: 15px;
    border-radius: 15px;
    background-color: #f5f9ff;
    transition: all 0.3s ease;
}

.action-button:hover {
    background-color: #e1eeff;
}

.action-button i {
    font-size: 24px;
    color: #ff0000;
}

.main-content {
    margin-bottom: 40px;
}

.section-title {
    margin-bottom: 20px;
    display: flex;
    align-items: center;
    gap: 10px;
    font-size: 1.5rem;
}

.section-title i {
    color: #ff0000;
}

.players-grid {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(280px, 1fr));
    gap: 20px;
    margin-bottom: 40px;
}

.player-card {
    position: relative;
    overflow: hidden;
}

.player-card .player-info {
    display: flex;
    align-items: center;
    gap: 15px;
    margin-bottom: 15px;
}

.player-card .player-avatar {
    width: 50px;
    height: 50px;
    border-radius: 50%;
    background-color: #ffe1e1;
    display: flex;
    align-items: center;
    justify-content: center;
    font-size: 20px;
    font-weight: bold;
    color: #0072bb;
}

.player-card .player-name {
    font-size: 1.2rem;
    font-weight: bold;
    margin: 0;
}

.player-card .player-balance {
    font-size: 1.8rem;
    font-weight: bold;
    color: #0072bb;
    margin: 10px 0;
}

.player-card .player-properties {
    display: flex;
    flex-wrap: wrap;
    gap: 8px;
    margin-top: 15px;
}

.player-card .property-chip {
    padding: 5px 10px;
    border-radius: 15px;
    font-size: 0.8rem;
    color: white;
    background-color: #0072bb;
}

.player-actions {
    display: flex;
    gap: 10px;
    margin-top: 15px;
}

.player-actions button {
    padding: 8px 15px;
    font-size: 0.9rem;
}

.properties-section {
    background-color: #f5f9ff;
    border-radius: 15px;
    padding: 20px;
    margin-top: 40px;
}

.property-groups {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));
    gap: 20px;
}

.property-group {
    border-radius: 10px;
    padding: 15px;
    box-shadow: 0 4px 8px rgba(0,0,0,0.05);
    background-color: white;
}
```

```
;

.property-group-header {
    padding: 8px;
    border-radius: 8px;
    margin-bottom: 10px;
    text-align: center;
    font-weight: bold;
    color: white;
}

.property-item {
    padding: 8px;
    margin: 5px 0;
    border-radius: 8px;
    background-color: #f9f9f9;
    display: flex;
    justify-content: space-between;
    align-items: center;
}

.property-name {
    font-size: 0.9rem;
}

.property-price {
    font-weight: bold;
    color: #0072bb;
}

.transaction-modal, .property-modal {
    background-color: white;
    padding: 25px;
    border-radius: 15px;
    box-shadow: 0 10px 30px rgba(0,0,0,0.2);
    width: 400px;
    max-width: 90%;
}

.modal-title {
    text-align: center;
    margin-bottom: 20px;
    color: #0072bb;
}

.modal-input {
    margin-bottom: 20px;
}

.modal-actions {
    display: flex;
    justify-content: flex-end;
    gap: 15px;
}

@media (max-width: 768px) {
    header {
        flex-direction: column;
        gap: 20px;
    }

    .logo {
        justify-content: center;
    }

    nav ul {
        justify-content: center;
        flex-wrap: wrap;
    }
}
/* General Styles */
```

```
body {
    background-color: #f4f7fa;
    font-family: 'Poppins', sans-serif;
    margin: 0;
    padding: 20px;
    color: #333;
}

h1, h2, h3 {
    font-weight: 600;
}

/* Button Styles */
button {
    border-radius: 25px;
    background: linear-gradient(45deg, #ff0000, #e90000);
    color: white;
    padding: 12px 20px;
    border: none;
    cursor: pointer;
    font-family: 'Poppins', sans-serif;
    font-weight: 500;
    display: flex;
    align-items: center;
    justify-content: center;
    gap: 8px;
    transition: all 0.3s ease;
    box-shadow: 0 4px 10px rgba(0, 114, 187, 0.2);
}

button:hover {
    background: linear-gradient(45deg, #ff0101, #890000);
    transform: translateY(-2px);
    box-shadow: 0 6px 15px rgba(187, 0, 0, 0.25);
}
```

```
button:active {
    transform: translateY(1px);
}

/* Card Styles */
.player-card, .property-card {
    background-color: #ffffff;
    border-radius: 15px;
    box-shadow: 0 5px 15px rgba(0, 0, 0, 0.05);
    padding: 20px;
    transition: all 0.3s ease;
    border-left: 4px solid #0072bb;
}

.player-card:hover, .property-card:hover {
    transform: translateY(-5px);
    box-shadow: 0 8px 25px rgba(0, 0, 0, 0.1);
}

/* Input and Select Styles */
input[type="text"], input[type="number"], select {
    border: 1px solid #e1e5eb;
    border-radius: 25px;
    padding: 12px 20px;
    width: 100%;
    font-family: 'Poppins', sans-serif;
    transition: all 0.3s ease;
    box-sizing: border-box;
    background-color: #f8fafc;
}

input[type="text"]:focus, input[type="number"]:focus, select:focus {
    border-color: #0072bb;
    outline: none;
    box-shadow: 0 0 0 3px rgba(0, 114, 187, 0.1);
    background-color: #fff;
}

/* Color Schemes for Property Groups */
.brown { background-color: #955436; }
.light-blue { background-color: #aae0fa; }
.pink { background-color: #d93a96; }
.orange { background-color: #f7941d; }
.red { background-color: #ed1b24; }
.yellow { background-color: #fef200; }
.green { background-color: #1fb25a; }
.dark-blue { background-color: #0072bb; }
.railroad { background-color: #221e1f; }
.utility { background-color: #6d6e71; }

/* Animations */
@keyframes fadeIn {
    from { opacity: 0; transform: translateY(10px); }
    to { opacity: 1; transform: translateY(0); }
}

@keyframes pulse {
    0% { transform: scale(1); }
    50% { transform: scale(1.05); }
    100% { transform: scale(1); }
}

.fade-in {
    animation: fadeIn 0.5s ease forwards;
}

/* Responsive Design */
@media (max-width: 768px) {
    .container {
        padding: 15px;
    }

    .page-title {
        font-size: 1.5rem;
    }

    .action-buttons {
        grid-template-columns: 1fr 1fr;
    }

    .players-grid {
        grid-template-columns: 1fr;
    }
}

/* Custom Scrollbar */
::-webkit-scrollbar {
    width: 8px;
    height: 8px;
}

::-webkit-scrollbar-track {
    background: #f1f1f1;
    border-radius: 10px;
}

::-webkit-scrollbar-thumb {
    background: #ff0000;
    border-radius: 10px;
}

::-webkit-scrollbar-thumb:hover {
    background: #ff0000;
}
```

```
html{
   scroll-behavior: smooth;
}
```

6.3 /monopoly/src/pages/Home.js

```
function Home() {
  return (
    <div className="page">
      <h1>Welcome to Monopoly Bank Management</h1>
      <p className="intro">Dive deep into the world of Monopoly with our digital banking system designed specifically for the Indian edition.</p>

      <section className="overview">
        <h2>Why Use This App?</h2>
        <ul>
          <li>Digitize all financial transactions - no more physical cash handling</li>
          <li>Automate calculations for rents, taxes, and fines to prevent errors</li>
          <li>Track property ownership with visual indicators</li>
          <li>Manage player balances in real-time</li>
          <li>Access transaction history through local storage persistence</li>
          <li>Streamline complex operations like property management and rent payments</li>
        </ul>
      </section>

      <section className="modules">
        <h2>Key Features</h2>

        <div className="module">
          <h3>Player Management</h3>
          <ul>
            <li>Add/Remove Players with unique IDs</li>
            <li>Real-time balance tracking in ₹ currency</li>
          </ul>
        </div>
        <div className="module">
          <h3>Property Management</h3>
          <ul>
            <li>Complete Indian-themed property database (28 properties)</li>
            <li>Color-coded property groups</li>
            <li>Clear ownership tracking</li>
            <li>Integrated buy/sell functionality</li>
            <li>Automatic rent calculation</li>
          </ul>
        </div>

        <div className="module">
          <h3>Banking Operations</h3>
          <ul>
            <li>Player-to-player transactions</li>
            <li>Bank payments for taxes, fines, and fees</li>
            <li>Bulk payment options</li>
            <li>Transaction verification</li>
          </ul>
        </div>

        <div className="module">
          <h3>Game Management</h3>
          <ul>
            <li>Data persistence with localStorage</li>
            <li>Secure game reset functionality</li>
            <li>Detailed rule reference</li>
            <li>Responsive design for desktop and mobile</li>
          </ul>
        </div>
      </section>

      <p className="footer-note">This web app is designed to manage the bank while playing Monopoly live on a physical
board.</p>
```

```
    </div>
  );
}

export default Home;
```

6.4 /monopoly/src/pages/Rules.js

```
function Rules() {
    return (
      <div className="page">
        <h1>Game Rules</h1>
        <iframe
          src="/rules.html"
          width="100%"
          height="500px"
          title="Rules"
          style={{ border: "none" }}
        ></iframe>
      </div>
    );
  }

  export default Rules;
```

6.5 /monopoly/src/pages/Bank.js

```
import { useState } from "react";

function Bank() {
  return (
    <div className="page">
      <h1>Game Dashboard</h1>
      <iframe
```

```
        src="/bank.html"
        width="100%"
        height="500px"
        title="Games"
        style={{ border: "none" }}
      ></iframe>
    </div>
  );
}

export default Bank;
```

6.6 /monopoly/src/pages/About.js

```
import { useState } from "react";

function About() {
  const [showDetails, setShowDetails] = useState(false);
  const [showDetails1, setShowDetails1] = useState(false);

  return (
    <div className="page">
      <h1>About Us</h1>
      <p><strong>Name</strong> - Pratham Raval </p>
      <p><strong>Registration Number</strong> - 23BCI0147 </p>
      <p><strong>Course Code:</strong> BCSE203E</p>
        <p><strong>Course Title:</strong> Web Programming</p>
        <p><strong>Faculty:</strong> SIVAKUMAR V</p>
        <p><strong>Slot:</strong> L37+L38+L53+L54</p>
      <p></p>
      <button onClick={() => setShowDetails(!showDetails)}>
        {showDetails ? "Hide Project Details" : "More About Project"}
      </button>
      <p></p>
      <button onClick={() => setShowDetails1(!showDetails1)}>
        {showDetails1 ? "Hide Developer Details" : "More About Developer"}
      </button>

      {showDetails && (
        <>
        <h2>About the Project</h2>
        <p>Welcome to the <strong>Monopoly Bank Management System</strong>, a modern solution designed to digitalize the banking experience in the classic Monopoly game. Managing cash transactions manually can be slow and error-prone, so we created an intuitive and efficient system to handle all your banking needs seamlessly.
          Our web app, built with <strong>HTML, CSS, JavaScript, and React.js</strong>, automates the Monopoly banking process, ensuring smooth transactions, accurate calculations, and a hassle-free gaming experience. Whether you're transferring money, collecting rent, or handling fines, our system makes every financial move effortless.
          With a user-friendly interface and all the essential banking features, our goal is to enhance your Monopoly sessions—making them faster, fairer, and more enjoyable. Say goodbye to misplaced cash and math mistakes; let our digital banker take care of everything while you focus on strategy and fun!
          Enjoy a smarter, faster, and more engaging Monopoly experience with our digital banking solution!
        </p>
        </>
      )}
      {showDetails1 && (
        <>
        <h2>About the Developer</h2>
        <p>
        Meet the mind behind the magic! <strong>Pratham Raval 23BCI0147</strong>. A passionate front-end developer who transforms lines of code into stunning, interactive, and futuristic web experiences. From crafting pixel-perfect designs to ensuring seamless user interactions, every project is an art piece in the making.
        Armed with <strong>HTML, CSS, JavaScript, and React.js</strong>, they breathe life into ideas, making websites not just functional but visually mesmerizing. Whether it's building sleek UI components, optimizing performance, or adding those subtle animations that make a website feel alive—this developer does it all.
        Fueled by caffeine and creativity, they are always exploring the latest trends in web development, pushing boundaries, and making the web a more beautiful place. When not coding, you’ll find them geeking out over futuristic UI concepts, testing new design trends, or contributing to open-source projects.
        If you love innovation, speed, and an eye for detail—this is the developer you’d want on your team!
        </p>
        </>
      )}
    </div>
  );
}
```

```
export default About;
```

6.7 /monopoly/src/pages/Feedback.js

```
import { useState, useEffect } from "react";

function Feedback() {
  const [feedback, setFeedback] = useState("");
  const [submittedFeedback, setSubmittedFeedback] = useState([]);

  useEffect(() => {
    const savedFeedback = JSON.parse(localStorage.getItem("feedback")) || [];
    setSubmittedFeedback(savedFeedback);
  }, []);

  const handleSubmit = () => {
    if (feedback.trim() === "") return;

    const newFeedback = [...submittedFeedback, feedback];
    setSubmittedFeedback(newFeedback);
    localStorage.setItem("feedback", JSON.stringify(newFeedback));
    setFeedback("");
  };

  return (
    <div className="flex justify-center items-center min-h-screen bg-gray-100 p-4">
      <div className="bg-white p-6 rounded-lg shadow-lg w-full max-w-md">
        <h1 className="text-2xl font-bold mb-4 text-center">Feedback</h1>
        <textarea
          className="w-full p-3 border border-gray-300 rounded-lg focus:ring focus:ring-blue-300"
          value={feedback}
          onChange={(e) => setFeedback(e.target.value)}
          placeholder="Write your feedback here..."
          rows="4"
        ></textarea>
        <button
          className="w-full bg-blue-500 text-white p-2 rounded-lg mt-3 hover:bg-blue-600 transition"
          onClick={handleSubmit}
        >
          Submit
        </button>

        <h2 className="text-xl font-semibold mt-6 mb-2">Previous Feedback</h2>
        <ul className="max-h-40 overflow-y-auto border-t pt-2">
          {submittedFeedback.length === 0 ? (
            <p className="text-gray-500 text-sm">No feedback submitted yet.</p>
          ) : (
            submittedFeedback.map((fb, index) => (
              <li
                key={index}
                className="p-2 bg-gray-100 rounded-lg mt-2 shadow-sm text-gray-700"
              >
                {fb}
              </li>
            ))
          )}
        </ul>
      </div>
    </div>
  );
}

export default Feedback;
```

6.8 /monopoly/src/pages/Footer.js

```
function Footer() {
   return (
     <footer className="bg-gray-800 text-white py-4 mt-10 text-center">
       <div className="container mx-auto px-4">
         <p className="text-sm">&copy; {new Date().getFullYear()} Pratham Raval 23BCI0147. All rights reserved.</p>
       </div>
     </footer>
   );
 }

 export default Footer;
```

6.9 /monopoly/public/rules.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Monopoly Bank - Rules</title>
   <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css">
   <link rel="stylesheet" href="./style.css">
</head>
<body>
   <div class="container">

       <h1 class="page-title">Monopoly Game - Rules</h1>

       <div class="rules-content">
           <h2>About the Event:</h2>
           <p>Monopoly is a classic board game that revolves around real estate trading, where players aim to accumulate wealth by buying, renting, and selling properties. The objective is to bankrupt your opponents by managing your assets wisely while avoiding bankruptcy yourself.</p>

           <h2>Event Description:</h2>
           <ul>
```

```
      <li>The event will be played individually.</li>
      <li>This event has time efficiency of 10 minutes for one board.</li>
      <li>Players will buy and trade properties and collect rent from opponents if they arrive at their property, this way one who will be the richest among all will be the winner.</li>
      <li>Players roll six-sided dice at the beginning of their turn, and the total value of the dice is the number of spaces they can move.</li>
    </ul>

    <h2>Rules & Regulations</h2>
    <ul>
      <li><strong>Objective:</strong> The goal is to bankrupt opponents by acquiring and developing properties.</li>
      <li><strong>Setup:</strong> Each player starts with a set amount of money. Properties are shuffled, and players take turns rolling dice to determine initial positions.</li>
      <li><strong>Gameplay:</strong> Players move around the board based on dice rolls. If a player lands on an unowned property, they can buy it. Monopolies are formed by owning all properties of a color group. Houses and hotels can be built on owned properties to increase rent.</li>
      <li><strong>Income:</strong> Players collect rent when opponents land on their properties. Rent increases with the number of houses or hotels.</li>
      <li><strong>Chance and Community Chest:</strong> Players draw cards from these decks, leading to various outcomes like receiving money or advancing to specific spaces.</li>
      <li><strong>Jail:</strong> Players can end up in jail by landing on the "Go to Jail" space, rolling doubles three times, or drawing certain cards. They can pay to get out or attempt to roll doubles on their turn.</li>
      <li><strong>Bankruptcy:</strong> A player declares bankruptcy when they can't pay debts. Remaining assets are sold, and the player is out of the game.</li>
      <li><strong>Ending the Game:</strong> The last player remaining after others go bankrupt or the player with the maximum revenue wins.</li>
    </ul>

    <h2>Jail Rules</h2>
    <p>When you start your turn in Jail, you have 3 options:</p>
    <ul>
      <li>Pay ₹50, then roll and move as normal.</li>
      <li>Use a Get Out Of Jail Free card, then roll and move as normal.</li>
      <li>Try to roll a double. If you do, you’re free! You will use the roll to move, but will not get to roll again (unlike after a normal double). If you fail to roll a double after 3 turns in Jail, you must pay ₹50 and use your roll to move.</li>
    </ul>

    <h2>Chance Cards</h2>
    <ul>
      <li>You are been ELECTED CHAIRMAN PAY EACH PLAYER ₹50</li>
      <li>YOUR BUILDING AND LOAN MATURED COLLECT ₹150</li>

      <li>THIS CARD MAY BE KEPT UNTIL NEEDED, GET OUT OF JAIL FREE</li>

      <li>BANK PAYS YOU DIVIDEND OF ₹50</li>

      <li>PAY POOR TAX OF ₹15</li>

      <li>TAKE A RIDE ON THE READING IF YOU PASS GO COLLECT ₹200</li>

      <li>ADVANCE TO GO (COLLECT ₹200)</li>

      <li>ADVANCE TO ST. CHARLES PLACE IF YOU PASS GO, COLLECT ₹200</li>

      <li>GO BACK 3 SPACES</li>

      <li>TAKE A WALK ON THE BOARDWALK ADVANCE TOKEN TO BOARDWALK</li>

      <li>Advance token to the nearest Railway and pay owner Twice the Rental to which he/she is otherwise entitled. If railroad is unowned, you may buy it from the bank.</li>

      <li>Advance token to nearest Utility. If unowned, you may buy it from the Bank. If owned, throw dice and pay owner a total ten times amount thrown.</li>

      <li>Make general repairs on all your property. For each house pay ₹25. For each hotel pay ₹100.</li>

      <li>Take a trip to Reading Railroad. If you pass Go, collect ₹200.</li>

      <li>Speeding fine ₹15.</li>

      <li>Bank pays you dividend of ₹50.</li>

      <li>Go Back 3 Spaces.</li>

      <li>Get Out of Jail Free.</li>

    </ul>


    <h2>Community Chest Cards</h2>

    <ul>

      <li>ADVANCE TO GO (COLLECT ₹200)</li>

      <li>FROM SALE OF STOCK YOU GET ₹45</li>

      <li>YOU INHERIT ₹100</li>

      <li>PAY HOSPITAL ₹100</li>

      <li>Grand Opera Opening COLLECT ₹50 FROM EVERY PLAYER</li>

      <li>INCOME TAX REFUND COLLECT ₹20</li>

      <li>RECEIVE FOR SERVICES ₹25</li>

      <li>DOCTOR'S FEE PAY ₹50</li>

      <li>GO TO JAIL Go Directly to Jail DO NOT PASS GO DO NOT COLLECT ₹200</li>

      <li>BANK ERROR IN YOUR FAVOR COLLECT ₹200</li>

      <li>XMAS FUND MATURES COLLECT ₹100</li>

      <li>LIFE INSURANCE MATURES COLLECT ₹100</li>

      <li>GET OUT OF JAIL FREE THIS CARD MAY BE KEPT UNTIL NEEDED, OR sold</li>
```

```
            <li>PAY SCHOOL TAX OF ₹150</li>
            <li>YOU HAVE WON SECOND PRIZE IN A BEAUTY CONTEST COLLECT ₹10</li>
            <li>YOU ARE ASSESSED FOR STREET REPAIRS ₹40 PER HOUSE</li>
            <li>IT IS YOUR BIRTHDAY. COLLECT 10 FROM EVERY PLAYER</li>
            <li>YOU ARE ASSESSED FOR STREET REPAIR. ₹40 PER HOUSE. ₹115 PER HOTEL</li>
          </ul>
        </div>

        <div class="contact-details">
          <h3>Contact Details:</h3>
          <p><strong>Name:</strong> Pratham Raval</p>
          <p><strong>Registraion Number:</strong>23BCI0147</p>
          <p><strong>Contact No.:</strong> 9xxxxxxxxx</p>
        </div>
    </div>

    <script src="./script.js"></script>
</body>
</html>
```

6.10 /monopoly/public/bank.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Monopoly Bank</title>
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css">
    <link rel="stylesheet" href="./style.css">
</head>

<body>
    <div class="container">
```

```
<h1 class="page-title">Monopoly Bank</h1>

<div class="input-group">
    <input type="text" id="newPlayerName" placeholder="Enter player name">
    <button onclick="addPlayer()"><i class="fas fa-user-plus"></i> Add Player</button>
</div>

<div class="action-buttons">
    <div class="action-button" onclick="resetDataWithPasskey()">
        <i class="fas fa-undo  "></i>
        <span>Reset Game</span>
    </div>
    <div class="action-button" onclick="payBank()">
        <i class="fas fa-money-bill-wave"></i>
        <span>Pay Bank</span>
    </div>
    <div class="action-button" onclick="payAllPlayers()">
        <i class="fas fa-money-bill-wave"></i>
        <span>Pay all players</span>
    </div>
    <div class="action-button" onclick="showPropertyModal()">
        <i class="fas fa-home"></i>
        <span>Buy Property</span>
    </div>
</div>

<div class="main-content">
    <h2 class="section-title"><i class="fas fa-users"></i> Players</h2>
    <div id="playersContainer" class="players-grid">
        <div class="player-card">

        </div>
    </div>
</div>
```

```
        <div id="properties" class="properties-section">
            <h2 class="section-title"><i class="fas fa-building"></i> Properties</h2>
            <div id="propertiesContainer" class="property-groups">
                <div class="property-group">

                </div>
            </div>
        </div>
    </div>

    <div id="transactionModal" style="display: none; position: fixed; top: 0; left: 0; width: 100%; height: 100%; background-color: rgba(0,0,0,0.5); justify-content: center; align-items: center;">
        <div class="transaction-modal">
            <h2 class="modal-title">Transaction</h2>
            <div class="modal-input">
                <input type="number" id="transactionAmount" placeholder="Enter amount">
            </div>
            <div class="modal-actions">
                <button onclick="cancelTransaction()"><i class="fas fa-times"></i> Cancel</button>
                <button onclick="confirmTransaction()"><i class="fas fa-check"></i> Confirm</button>
            </div>
        </div>
    </div>

    <div id="propertyModal" style="display: none; position: fixed; top: 0; left: 0; width: 100%; height: 100%; background-color: rgba(0,0,0,0.5); justify-content: center; align-items: center;">
        <div class="property-modal">
            <h2 class="modal-title">Buy Property</h2>
            <div class="modal-input">
                <select id="propertySelect">
                    <!-- Properties will be added here -->
                </select>
            </div>
            <div class="modal-input">
                <select id="playerSelect">
                    <!-- Players will be added here -->
                </select>
            </div>
            <div class="modal-actions">
                <button onclick="cancelPropertyPurchase()"><i class="fas fa-times"></i> Cancel</button>
                <button onclick="confirmPropertyPurchase()"><i class="fas fa-check"></i> Buy Property</button>
            </div>
        </div>
    </div>

    <script src="./script.js"></script>
    </body>
    </html>
```

6.11 /monopoly/public/script.js

```
class Player {
    constructor(name) {
        this.name = name;
        this.balance = 1500;
        this.id = Date.now().toString();
        this.properties = [];
    }
}

class Property {
    constructor(name, price, group,rent, color) {
        this.name = name;
        this.price = price;
        this.group = group;
        this.rent = rent;
        this.color = color;
        this.owner = null;
    }
}

let players = [];
let properties = [];
let currentTransaction = null;

function loadData() {
    const storedPlayers = localStorage.getItem('monopolyPlayers');
    const storedProperties = localStorage.getItem('monopolyProperties');

    if (storedPlayers) {
        players = JSON.parse(storedPlayers);
    }
    updatePlayersDisplay();
    updatePropertyDisplay();
}

function saveData() {
    localStorage.setItem('monopolyPlayers', JSON.stringify(players));
    localStorage.setItem('monopolyProperties', JSON.stringify(properties));
}

function addPlayer() {
    const nameInput = document.getElementById('newPlayerName');
    const name = nameInput.value.trim();

    if (name) {
        const player = new Player(name);
        players.push(player);
        nameInput.value = '';
        updatePlayersDisplay();
        updatePropertyDisplay();
        saveData();
    }
}
```

```
function updatePlayersDisplay() {
    const container = document.getElementById('playersContainer');
    container.innerHTML = '';

    players.forEach(player => {
        const playerCard = document.createElement('div');
        playerCard.style.cssText = 'background-color: #fff; border: 2px solid rgb(122, 26, 26); border-radius: 8px; padding: 15px;';

        let propertiesHtml = `<div style="margin-top: 10px; padding: 10px; background-color: #f8f9fa; border-radius: 5px;">
            <h3>Properties:</h3><ul style="list-style: none; padding: 0;">`;

        player.properties.forEach(prop => {
            propertiesHtml += `
                <li style="padding: 8px; margin: 5px 0; background-color: #fff; border: 1px solid #ddd; border-radius: 4px; display: flex; justify-content: space-between; align-items: center;">
                    <span>
                        <span style="width: 20px; height: 20px; border-radius: 50%; display: inline-block; margin-right: 10px; background-color: ${prop.color};"></span>
                        ${prop.name}
                    </span>
                    <button onclick="sellProperty('${player.id}', '${prop.name}')" style="background-color:rgb(122, 26, 26); color: white; border: none; padding: 8px 15px; border-radius: 5px; cursor: pointer;">Sell</button>
                </li>
            `;
        });
        propertiesHtml += '</ul></div>';

        playerCard.innerHTML = `
            <div style="font-size: 1.2em; font-weight: bold; color:rgb(0, 0, 0); margin-bottom: 10px;">${player.name}</div>
            <div style="font-size: 1.5em; color:rgb(21, 255, 0); margin-bottom: 15px;"> &#8377;${player.balance}</div>
            <div style="display: flex; flex-direction: column; gap: 10px;">
                <button onclick="showTransactionModal('receive', '${player.id}')" style="background-color:rgb(122, 26, 26); color: white; border: none; padding: 8px 15px; border-radius: 5px; cursor: pointer;">Receive Money</button>
                <button onclick="showTransactionModal('pay', '${player.id}')" style="background-color:rgb(122, 26, 26); color: white; border: none; padding: 8px 15px; border-radius: 5px; cursor: pointer;">Pay Money</button>
                <button onclick="removePlayer('${player.id}')" style="background-color:rgb(122, 26, 26); color: white; border: none; padding: 8px 15px; border-radius: 5px; cursor: pointer;">Remove Player</button>
            </div>
            ${propertiesHtml}
        `;
        container.appendChild(playerCard);
    });
}

function updatePropertyDisplay() {
    const container = document.getElementById('propertiesContainer');
    container.innerHTML = '';

    const groupedProperties = properties.reduce((acc, prop) => {
        if (!acc[prop.group]) {
            acc[prop.group] = [];
        }
        acc[prop.group].push(prop);
        return acc;
    }, {});

    Object.entries(groupedProperties).forEach(([group, props]) => {
        const groupDiv = document.createElement('div');
        groupDiv.style.marginBottom = '20px';

        let groupHtml = `<div style="font-weight: bold; color:rgb(122, 26, 26); margin-bottom: 10px;">${group}</div>`;
        props.forEach(prop => {
            groupHtml += `
                <div style="background-color: white; border: 1px solid #ddd; border-radius: 8px; padding: 15px; margin-bottom: 10px;">
                    <span style="width: 20px; height: 20px; border-radius: 50%; display: inline-block; margin-right: 10px; background-color: ${prop.color};"></span>
                    <strong>${prop.name}</strong><br>
                    Price: &#8377; ${prop.price}<br>
                    Rent: &#8377; ${prop.rent}<br>
                    Owner: ${prop.owner ? prop.owner : 'Bank'}
                </div>
            `;
        });

        groupDiv.innerHTML = groupHtml;
        container.appendChild(groupDiv);
    });
}

function showPropertyModal() {
    const modal = document.getElementById('propertyModal');
    modal.style.display = 'flex';

}

function cancelPropertyPurchase() {
    const modal = document.getElementById('propertyModal');
    modal.style.display = 'none';
}

function showTransactionModal(type, playerId) {
    const modal = document.getElementById('transactionModal');
    modal.style.display = 'flex';
    currentTransaction = { type, playerId };
}

function cancelTransaction() {
    const modal = document.getElementById('transactionModal');
    modal.style.display = 'none';
}

function confirmPropertyPurchase() {
    const propertyName = document.getElementById('propertySelect').value;
    const playerId = document.getElementById('playerSelect').value;
    const property = properties.find(p => p.name === propertyName);
    const player = players.find(p => p.id === playerId);

    if (!property || !player) {
        alert('Please select a valid property and player');
        return;
    }

    if (player.balance < property.price) {
        alert('Insufficient funds to purchase this property!');
        return;
    }

    player.balance -= property.price;
    property.owner = player.name;
    player.properties.push(property);

    updatePlayersDisplay();
    updatePropertyDisplay();
    cancelPropertyPurchase();
    saveData(); // Save data to LocalStorage
}

function sellProperty(playerId, propertyName) {
    const player = players.find(p => p.id === playerId);
    const property = properties.find(p => p.name === propertyName);

    if (!player || !property) {
        alert('Invalid player or property');
        return;
    }

    const sellPrice = Math.floor(property.price / 2);
    player.balance += sellPrice;
    property.owner = null;
    player.properties = player.properties.filter(p => p.name !== propertyName);
```

```
  updatePlayersDisplay();
  updatePropertyDisplay();
  saveData(); // Save data to LocalStorage
}

function showPropertyModal() {
const modal = document.getElementById('propertyModal');
const propertySelect = document.getElementById('propertySelect');
const playerSelect = document.getElementById('playerSelect');

propertySelect.innerHTML = '';
playerSelect.innerHTML = '';

const availableProperties = properties.filter(p => !p.owner);
if (availableProperties.length === 0) {
alert('No properties available for purchase!');
return;
}

availableProperties.forEach(prop => {
const option = document.createElement('option');
option.value = prop.name;
option.textContent = `${prop.name} - Rs. ${prop.price}`;
propertySelect.appendChild(option);
});

if (players.length === 0) {
alert('Add players before buying properties!');
return;
}

players.forEach(player => {
const option = document.createElement('option');
option.value = player.id;
option.textContent = `${player.name} - Rs. ${player.balance}`;
playerSelect.appendChild(option);
});

modal.style.display = 'flex';
}

function initializeProperties() {
// Brown properties
properties.push(new Property("Bhubaneshwar", 60, "Brown", 4, "#955436"));
properties.push(new Property("Guwahati", 60, "Brown", 2, "#955436"));

// Light Blue properties
properties.push(new Property("Goa", 100, "Dark Blue", 6, "#0072bb"));
properties.push(new Property("Agra", 100, "Dark Blue", 6, "#0072bb"));
properties.push(new Property("Vadodara", 120, "Dark Blue", 8, "#0072bb"));

// Pink properties
properties.push(new Property("Ludhiana", 140, "Pink", 10, "#d93a96"));
properties.push(new Property("Patna", 140, "Pink", 10, "#d93a96"));
properties.push(new Property("Bhopal", 160, "Pink", 12, "#d93a96"));

// Orange properties
properties.push(new Property("Indore", 180, "Orange", 14, "#f7921c"));
properties.push(new Property("Nagpur", 180, "Orange", 14, "#f7921c"));
properties.push(new Property("Kochi", 200, "Orange", 16, "#f7921c"));

// Red properties
properties.push(new Property("Lucknow", 220, "Red", 18, "#ed1b24"));
properties.push(new Property("Chandigarh", 220, "Red", 18, "#ed1b24"));
properties.push(new Property("Jaipur", 240, "Red", 20, "#ed1b24"));

// Yellow properties
properties.push(new Property("Pune", 260, "Yellow", 22, "#fef200"));
properties.push(new Property("Hyderabad", 260, "Yellow", 22, "#fef200"));
properties.push(new Property("Ahmedbad", 280, "Yellow", 24, "#fef200"));

// Green properties
properties.push(new Property("Chennai", 300, "Green", 26, "#1fb25a"));
properties.push(new Property("Kolkata", 300, "Green", 26, "#1fb25a"));
properties.push(new Property("Bengaluru", 320, "Green", 28, "#1fb25a"));

// Dark Blue properties
properties.push(new Property("Delhi", 350, "Light Blue", 35, "#aae0fa"));
properties.push(new Property("Mumbai", 400, "Light Blue", 50, "#aae0fa"));

// Railroads
properties.push(new Property("Chennai Central Railway Station", 200, "Railroad", 25, "#000000"));
properties.push(new Property("New Delhi Railway Station", 200, "Railroad", 25, "#000000"));
properties.push(new Property("Howrah Railway Station", 200, "Railroad", 25, "#000000"));
properties.push(new Property("Chhatrapati Shivaji Terminus", 200, "Railroad", 25, "#000000"));

// Utilities
properties.push(new Property("Electric Company", 150, "Utility", "0" , "#999999"));
properties.push(new Property("Water Works", 150, "Utility", "0", "#999999"));
}

// Add transaction confirmation functions
function confirmTransaction() {
const amount = parseInt(document.getElementById('transactionAmount').value);
if (!amount || isNaN(amount)) {
alert('Please enter a valid amount');
return;
}

const player = players.find(p => p.id === currentTransaction.playerId);
if (!player) {
alert('Invalid player');
return;
}
```

```
if (currentTransaction.type === 'receive') {
player.balance += amount;
} else {
if (player.balance < amount) {
   alert('Insufficient funds!');
   return;
}
player.balance -= amount;
}

updatePlayersDisplay();
cancelTransaction();
}

function payAllPlayers() {
const amount = parseInt(prompt('Enter amount to pay to each player:'));
if (!amount || isNaN(amount)) return;

const playerName = prompt('Enter paying player name:');
const payingPlayer = players.find(p => p.name.toLowerCase() === playerName.toLowerCase());

if (payingPlayer) {
const totalAmount = amount * (players.length - 1);
if (payingPlayer.balance >= totalAmount) {
   players.forEach(player => {
      if (player !== payingPlayer) {
         player.balance += amount;
      }
   });
   payingPlayer.balance -= totalAmount;
   updatePlayersDisplay();
   alert(`${payingPlayer.name} paid Rs;${amount} to each player`);
} else {
   alert('Insufficient funds!');
}
} else {
alert('Player not found!');
}
}

function showPlayerTransactionModal() {
    const modal = document.createElement('div');
    modal.id = 'playerTransactionModal';
    modal.style.cssText = 'position: fixed; top: 0; left: 0; width: 100%; height: 100%; background-color: rgba(0,0,0,0.5); display: flex; justify-content: center; align-items: center; z-index: 1000;';

    const modalContent = document.createElement('div');
    modalContent.className = 'transaction-modal';
    modalContent.style.cssText = 'background-color: white; padding: 25px; border-radius: 15px; box-shadow: 0 10px 30px rgba(0,0,0,0.2); width: 400px; max-width: 90%;';

    modalContent.innerHTML = `
        <h2 class="modal-title" style="text-align: center; margin-bottom: 20px; color:rgb(255, 2, 2);">Player Transaction</h2>

        <div class="modal-input" style="margin-bottom: 20px;">
            <label for="fromPlayerSelect" style="display: block; margin-bottom: 8px; font-weight: 500;">From Player:</label>
            <select id="fromPlayerSelect" style="width: 100%; padding: 10px; border-radius: 8px; border: 1px solid #ddd;">
                ${players.map(player => `<option value="${player.id}">${player.name} - Rs. ${player.balance}</option>`).join('')}
            </select>
        </div>

        <div class="modal-input" style="margin-bottom: 20px;">
            <label for="toPlayerSelect" style="display: block; margin-bottom: 8px; font-weight: 500;">To Player:</label>
            <select id="toPlayerSelect" style="width: 100%; padding: 10px; border-radius: 8px; border: 1px solid #ddd;">
                ${players.map(player => `<option value="${player.id}">${player.name} - Rs. ${player.balance}</option>`).join('')}
            </select>
        </div>

        <div class="modal-input" style="margin-bottom: 20px;">
```

```
            <label for="playerTransactionAmount" style="display: block; margin-bottom: 8px; font-weight: 500;">Amount:</label>

            <input type="number" id="playerTransactionAmount" min="1" placeholder="Enter amount" style="width: 100%; padding: 10px; border-radius: 8px; border: 1px solid #ddd;">

        </div>



        <div class="modal-input" style="margin-bottom: 20px;">

            <label for="transactionReason" style="display: block; margin-bottom: 8px; font-weight: 500;">Reason (optional):</label>

            <input type="text" id="transactionReason" placeholder="Rent, Trade, etc." style="width: 100%; padding: 10px; border-radius: 8px; border: 1px solid #ddd;">

        </div>



        <div class="modal-actions" style="display: flex; justify-content: flex-end; gap: 15px;">

            <button onclick="cancelPlayerTransaction()" style="padding: 10px 15px;"><i class="fas fa-times"></i> Cancel</button>

            <button onclick="confirmPlayerTransaction()" style="padding: 10px 15px;"><i class="fas fa-check"></i> Transfer</button>

        </div>

    `;



    modal.appendChild(modalContent);

    document.body.appendChild(modal);



    if (players.length > 1) {

        document.getElementById('toPlayerSelect').selectedIndex = 1;

    }

}



function cancelPlayerTransaction() {

    const modal = document.getElementById('playerTransactionModal');

    if (modal) {

        document.body.removeChild(modal);

    }

}



function confirmPlayerTransaction() {

    const fromPlayerId = document.getElementById('fromPlayerSelect').value;
    const toPlayerId = document.getElementById('toPlayerSelect').value;
    const amount = parseInt(document.getElementById('playerTransactionAmount').value);
    const reason = document.getElementById('transactionReason').value;

    if (fromPlayerId === toPlayerId) {
        alert('Cannot transfer money to the same player');
        return;
    }

    if (!amount || isNaN(amount) || amount <= 0) {
        alert('Please enter a valid amount');
        return;
    }

    const fromPlayer = players.find(p => p.id === fromPlayerId);
    const toPlayer = players.find(p => p.id === toPlayerId);

    if (!fromPlayer || !toPlayer) {
        alert('Invalid player selection');
        return;
    }

    if (fromPlayer.balance < amount) {
        alert(`${fromPlayer.name} does not have enough money for this transaction`);
        return;
    }

    fromPlayer.balance -= amount;
    toPlayer.balance += amount;

    const reasonText = reason ? ` for ${reason}` : '';
    alert(`${fromPlayer.name} paid Rs. ${amount} to ${toPlayer.name}${reasonText}`);

    updatePlayersDisplay();
    cancelPlayerTransaction();
```

```
}


// Add function to pay bank
function payBank() {
  const modal = document.createElement('div');
  modal.id = 'payBankModal';
  modal.style.cssText = 'position: fixed; top: 0; left: 0; width: 100%; height: 100%; background-color: rgba(0,0,0,0.5);
display: flex; justify-content: center; align-items: center; z-index: 1000;';


  const modalContent = document.createElement('div');
  modalContent.className = 'transaction-modal';
  modalContent.style.cssText = 'background-color: white; padding: 25px; border-radius: 15px; box-shadow: 0 10px 30px
rgba(0,0,0,0.2); width: 400px; max-width: 90%;';


  modalContent.innerHTML = `
    <h2 class="modal-title" style="text-align: center; margin-bottom: 20px; color:rgb(255, 0, 0);">Pay to Bank</h2>


    <div class="modal-input" style="margin-bottom: 20px;">
      <label for="playerSelectBank" style="display: block; margin-bottom: 8px; font-weight: 500;">Player:</label>
      <select id="playerSelectBank" style="width: 100%; padding: 10px; border-radius: 8px; border: 1px solid #ddd;">
        ${players.map(player => `<option value="${player.id}">${player.name} -
&#8377;${player.balance}</option>`).join('')}
      </select>
    </div>


    <div class="modal-input" style="margin-bottom: 20px;">
      <label for="bankPaymentAmount" style="display: block; margin-bottom: 8px; font-weight: 500;">Amount:</label>
      <input type="number" id="bankPaymentAmount" min="1" placeholder="Enter amount" style="width: 100%;
padding: 10px; border-radius: 8px; border: 1px solid #ddd;">
    </div>


    <div class="modal-input" style="margin-bottom: 20px;">
      <label for="bankPaymentReason" style="display: block; margin-bottom: 8px; font-weight: 500;">Reason:</label>
      <select id="bankPaymentReason" style="width: 100%; padding: 10px; border-radius: 8px; border: 1px solid #ddd;">
        <option value="Tax">Income Tax</option>
        <option value="Chance">Chance Card</option>
        <option value="Community">Community Chest</option>
```

```
                <option value="Repairs">Property Repairs</option>
                <option value="Luxury">Luxury Tax</option>
                <option value="Other">Other</option>
            </select>
        </div>

        <div class="modal-actions" style="display: flex; justify-content: flex-end; gap: 15px;">
            <button onclick="cancelBankPayment()" style="padding: 10px 15px;"><i class="fas fa-times"></i> Cancel</button>
            <button onclick="confirmBankPayment()" style="padding: 10px 15px;"><i class="fas fa-check"></i> Pay</button>
        </div>
    `;

    modal.appendChild(modalContent);
    document.body.appendChild(modal);
}

// Function to cancel bank payment
function cancelBankPayment() {
    const modal = document.getElementById('payBankModal');
    if (modal) {
        document.body.removeChild(modal);
    }
}

// Function to confirm bank payment
function confirmBankPayment() {
    const playerId = document.getElementById('playerSelectBank').value;
    const amount = parseInt(document.getElementById('bankPaymentAmount').value);
    const reason = document.getElementById('bankPaymentReason').value;

    if (!amount || isNaN(amount) || amount <= 0) {
        alert('Please enter a valid amount');
        return;
    }
    const player = players.find(p => p.id === playerId);

    if (!player) {
        alert('Invalid player selection');
        return;
    }

    // Check if player has enough funds
    if (player.balance < amount) {
        alert(`${player.name} does not have enough money`);
        return;
    }
    player.balance -= amount;

    alert(`${player.name} paid Rs. ${amount} to the bank for ${reason}`);

    updatePlayersDisplay();
    cancelBankPayment();
}

// Function to handle property rent payments
function payRent(propertyName) {
    const property = properties.find(p => p.name === propertyName);

    if (!property || !property.owner) {
        alert('This property is not owned by any player');
        return;
    }

    const owner = players.find(p => p.name === property.owner);

    if (!owner) {
        alert('Property owner not found in the game');
        return;
    }
    const modal = document.createElement('div');
    modal.id = 'rentPaymentModal';
    modal.style.cssText = 'position: fixed; top: 0; left: 0; width: 100%; height: 100%; background-color: rgba(0,0,0,0.5); display: flex; justify-content: center; align-items: center; z-index: 1000;';

    const modalContent = document.createElement('div');
    modalContent.className = 'transaction-modal';
    modalContent.style.cssText = 'background-color: white; padding: 25px; border-radius: 15px; box-shadow: 0 10px 30px rgba(0,0,0,0.2); width: 400px; max-width: 90%;';

    const potentialPayers = players.filter(p => p.name !== owner.name);

    modalContent.innerHTML = `
        <h2 class="modal-title" style="text-align: center; margin-bottom: 20px; color:rgb(255, 0, 0);">Pay Rent for ${property.name}</h2>

        <div style="margin-bottom: 15px; padding: 10px; background-color: ${property.color}; color: white; border-radius: 8px; text-align: center;">
            <p style="margin: 0;">Property Owner: ${owner.name}</p>
            <p style="margin: 5px 0 0 0; font-size: 1.2em; font-weight: bold;">Rent: &#8377;${property.rent}</p>
        </div>

        <div class="modal-input" style="margin-bottom: 20px;">
            <label for="rentPayerSelect" style="display: block; margin-bottom: 8px; font-weight: 500;">Paying Player:</label>
            <select id="rentPayerSelect" style="width: 100%; padding: 10px; border-radius: 8px; border: 1px solid #ddd;">
                ${potentialPayers.map(player => `<option value="${player.id}">${player.name} - &#8377;${player.balance}</option>`).join('')}
            </select>
        </div>

        <div class="modal-actions" style="display: flex; justify-content: flex-end; gap: 15px;">
            <button onclick="cancelRentPayment()" style="padding: 10px 15px;"><i class="fas fa-times"></i> Cancel</button>
            <button onclick="confirmRentPayment('${propertyName}')" style="padding: 10px 15px;"><i class="fas fa-check"></i> Pay Rent</button>
        </div>
    `;

    modal.appendChild(modalContent);
    document.body.appendChild(modal);
}

// Function to cancel rent payment
function cancelRentPayment() {
    const modal = document.getElementById('rentPaymentModal');
    if (modal) {
        document.body.removeChild(modal);
    }
}

// Function to confirm rent payment
function confirmRentPayment(propertyName) {
    const property = properties.find(p => p.name === propertyName);
    const payerId = document.getElementById('rentPayerSelect').value;
    const payer = players.find(p => p.id === payerId);
    const owner = players.find(p => p.name === property.owner);

    if (!payer || !owner || !property) {
        alert('Invalid transaction details');
        return;
    }
    if (payer.balance < property.rent) {
        alert(`${payer.name} does not have enough money to pay rent`);
        return;
    }

    payer.balance -= property.rent;
    owner.balance += property.rent;

    alert(`${payer.name} paid Rs. ${property.rent} rent to ${owner.name} for ${property.name}`);

    updatePlayersDisplay();
    cancelRentPayment();
}
```

```
// Update the property display to include a "Pay Rent" button
function updatePropertyDisplay() {
    const container = document.getElementById('propertiesContainer');
    container.innerHTML = '';

    const groupedProperties = properties.reduce((acc, prop) => {
        if (!acc[prop.group]) {
            acc[prop.group] = [];
        }
        acc[prop.group].push(prop);
        return acc;
    }, {});

    Object.entries(groupedProperties).forEach(([group, props]) => {
        const groupDiv = document.createElement('div');
        groupDiv.className = 'property-group';
        groupDiv.style.cssText = 'border-radius: 10px; padding: 15px; box-shadow: 0 4px 8px rgba(0,0,0,0.05); background-color: white; margin-bottom: 20px;';

        const groupColor = props[0].color;

        let groupHtml = `
            <div class="property-group-header" style="padding: 8px; border-radius: 8px; margin-bottom: 15px; text-align: center; font-weight: bold; color: white; background-color: ${groupColor};">
                ${group}
            </div>
        `;

        props.forEach(prop => {
            let ownerInfo = 'Bank';
            let rentButton = '';

            if (prop.owner) {
                ownerInfo = prop.owner;
                rentButton = `<button onclick="payRent('${prop.name}')" style="padding: 5px 10px; font-size: 0.9rem; margin-top: 5px;"><i class="fas fa-hand-holding-usd"></i> Pay Rent</button>`;
            }

            groupHtml += `
                <div class="property-item" style="padding: 10px; margin: 8px 0; border-radius: 8px; background-color: #f9f9f9; border-left: 3px solid ${prop.color};">
                    <div style="margin-bottom: 5px;">
                        <strong>${prop.name}</strong>
                        <div style="display: flex; justify-content: space-between; margin-top: 5px;">
                            <span>Price: &#8377;${prop.price}</span>
                            <span>Rent: &#8377;${prop.rent}</span>
                        </div>
                        <div style="margin-top: 8px;">
                            <span>Owner: <strong>${ownerInfo}</strong></span>
                        </div>
                    </div>
                    ${rentButton}
                </div>
            `;
        });

        groupDiv.innerHTML = groupHtml;
        container.appendChild(groupDiv);
    });
}

// Update the action buttons in the HTML to include the new transaction button
function updateActionButtons() {
    const actionButtons = document.querySelector('.action-buttons');

    // Check if action buttons exist and the transaction button doesn't already exist
    if (actionButtons && !document.getElementById('playerTransactionBtn')) {
        const transactionButton = document.createElement('div');
        transactionButton.className = 'action-button';
        transactionButton.id = 'playerTransactionBtn';
        transactionButton.onclick = showPlayerTransactionModal;
        transactionButton.innerHTML = `
            <i class="fas fa-exchange-alt"></i>
            <span>Player Transaction</span>
        `;

        actionButtons.appendChild(transactionButton);
    }
}

// Update existing function to include our new button
document.addEventListener('DOMContentLoaded', () => {
    loadData();
    initializeProperties();
    updatePropertyDisplay();
    updateActionButtons();
});

// Modify existing functions to maintain compatibility
function removePlayer(playerId) {
    players = players.filter(p => p.id !== playerId);
    updatePlayersDisplay();
    saveData(); // Save data to LocalStorage
}

// Add this function to reset local data
const PRESET_PASSKEY = "0000";

// Function to show the passkey modal
function showPasskeyModal() {
    const modal = document.createElement('div');
    modal.id = 'passkeyModal';
    modal.style.cssText = 'position: fixed; top: 0; left: 0; width: 100%; height: 100%; background-color: rgba(0,0,0,0.5); display: flex; justify-content: center; align-items: center; z-index: 1000;';
    const modalContent = document.createElement('div');
    modalContent.className = 'passkey-modal';
    modalContent.style.cssText = 'background-color: white; padding: 25px; border-radius: 15px; box-shadow: 0 10px 30px rgba(0,0,0,0.2); width: 400px; max-width: 90%;';

    modalContent.innerHTML = `
        <h2 class="modal-title" style="text-align: center; margin-bottom: 20px; color:rgb(255, 0, 0);">Reset Data</h2>
        <div class="modal-input" style="margin-bottom: 20px;">
            <label for="passkeyInput" style="display: block; margin-bottom: 8px; font-weight: 500;">Enter Passkey:</label>
            <input type="password" id="passkeyInput" placeholder="Enter passkey" style="width: 100%; padding: 10px; border-radius: 8px; border: 1px solid #ddd;">
        </div>
        <div class="modal-actions" style="display: flex; justify-content: flex-end; gap: 15px;">
            <button onclick="cancelPasskeyModal()" style="padding: 10px 15px;"><i class="fas fa-times"></i> Cancel</button>
            <button onclick="confirmPasskey()" style="padding: 10px 15px;"><i class="fas fa-check"></i> Confirm</button>
        </div>
    `;

    modal.appendChild(modalContent);
    document.body.appendChild(modal);
}

// Function to cancel the passkey modal
function cancelPasskeyModal() {
    const modal = document.getElementById('passkeyModal');
    if (modal) {
        document.body.removeChild(modal);
    }
}

// Function to confirm the passkey and reset data
function confirmPasskey() {
    const passkeyInput = document.getElementById('passkeyInput').value;

    if (passkeyInput === PRESET_PASSKEY) {
        resetLocalData();
        cancelPasskeyModal();
    } else {
        alert('Incorrect passkey! Data was not reset.');
    }
}

// Function to reset local data
function resetLocalData() {

    localStorage.removeItem('monopolyPlayers');
    localStorage.removeItem('monopolyProperties');

    players = [];
    properties = [];

    initializeProperties();

    updatePlayersDisplay();
    updatePropertyDisplay();

    alert('Local data has been reset successfully!');
}

function resetDataWithPasskey() {
    showPasskeyModal();
}
```

6.12 /monopoly/public/style.css

```
@import url('https://fonts.googleapis.com/css2?family=Poppins:wght@400;500;600;700&display=swap');

.container {
    max-width: 1200px;
    margin: 0 auto;
    background-color: white;
    padding: 30px;
    border-radius: 15px;
    box-shadow: 0 8px 20px rgba(0,0,0,0.1);
}

header {
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin-bottom: 30px;
    padding-bottom: 20px;
    border-bottom: 2px solid #f0f0f0;
}

.logo {
    display: flex;
    align-items: center;
    gap: 15px;
}

.logo-img{
    width :250px;
}

.logo h1 {
    margin: 0;
    font-size: 1.8rem;
    background: linear-gradient(90deg, #ff0000, #e90000);
    -webkit-background-clip: text;
    -webkit-text-fill-color: transparent;
}

nav ul {
    display: flex;
    list-style: none;
    gap: 25px;
    margin: 0;
    padding: 0;
}

nav a {
    text-decoration: none;
    color: #555;
    font-weight: 500;
    padding: 5px 0;
    position: relative;
    transition: color 0.3s;
}

nav a:hover, nav a.active {
    color: #ff1b1b;
}

nav a:after {
    content: '';
    position: absolute;
    width: 0;
    height: 2px;
    bottom: 0;
    left: 0;
    background-color: #000000;
    transition: width 0.3s;
}

nav a:hover:after, nav a.active:after {
    width: 100%;
}
```

```
.page-title {
   text-align: center;
   margin-bottom: 30px;
   position: relative;
   padding-bottom: 15px;
}

.page-title:after {
   content: '';
   position: absolute;
   width: 100px;
   height: 3px;
   bottom: 0;
   left: 50%;
   transform: translateX(-50%);
   background: linear-gradient(90deg, #ff0000, #e90000);
}

.input-group {
   display: flex;
   gap: 10px;
   margin-bottom: 30px;
   box-shadow: 0 4px 12px rgba(0,0,0,0.05);
   border-radius: 30px;
   padding: 5px;
   background-color: #f9f9f9;
}

.input-group input {
   border-radius: 25px;
   border: none;
   background-color: transparent;
   padding: 12px 20px;
   flex-grow: 1;
}
```

```
.action-buttons {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(150px, 1fr));
    gap: 15px;
    margin-bottom: 40px;
}

.action-button {
    display: flex;
    flex-direction: column;
    align-items: center;
    gap: 10px;
    padding: 15px;
    border-radius: 15px;
    background-color: #f5f9ff;
    transition: all 0.3s ease;
}

.action-button:hover {
    background-color: #e1eeff;
}

.action-button i {
    font-size: 24px;
    color: #ff0000;
}

.main-content {
    margin-bottom: 40px;
}

.section-title {
    margin-bottom: 20px;
    display: flex;
    align-items: center;

    gap: 10px;

    font-size: 1.5rem;

}


.section-title i {

    color: #ff0000;

}


.players-grid {

    display: grid;

    grid-template-columns: repeat(auto-fit, minmax(280px, 1fr));

    gap: 20px;

    margin-bottom: 40px;

}


.player-card {

    position: relative;

    overflow: hidden;

}


.player-card .player-info {

    display: flex;

    align-items: center;

    gap: 15px;

    margin-bottom: 15px;

}


.player-card .player-avatar {

    width: 50px;

    height: 50px;

    border-radius: 50%;

    background-color: #ffe1e1;

    display: flex;

    align-items: center;
    justify-content: center;
    font-size: 20px;
    font-weight: bold;
    color: #0072bb;
}

.player-card .player-name {
    font-size: 1.2rem;
    font-weight: bold;
    margin: 0;
}

.player-card .player-balance {
    font-size: 1.8rem;
    font-weight: bold;
    color: #0072bb;
    margin: 10px 0;
}

.player-card .player-properties {
    display: flex;
    flex-wrap: wrap;
    gap: 8px;
    margin-top: 15px;
}

.player-card .property-chip {
    padding: 5px 10px;
    border-radius: 15px;
    font-size: 0.8rem;
    color: white;
    background-color: #0072bb;
}

.player-actions {
    display: flex;
    gap: 10px;
    margin-top: 15px;
}

.player-actions button {
    padding: 8px 15px;
    font-size: 0.9rem;
}

.properties-section {
    background-color: #f5f9ff;
    border-radius: 15px;
    padding: 20px;
    margin-top: 40px;
}

.property-groups {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));
    gap: 20px;
}

.property-group {
    border-radius: 10px;
    padding: 15px;
    box-shadow: 0 4px 8px rgba(0,0,0,0.05);
    background-color: white;
}

.property-group-header {
    padding: 8px;
    border-radius: 8px;
    margin-bottom: 10px;
    text-align: center;
```

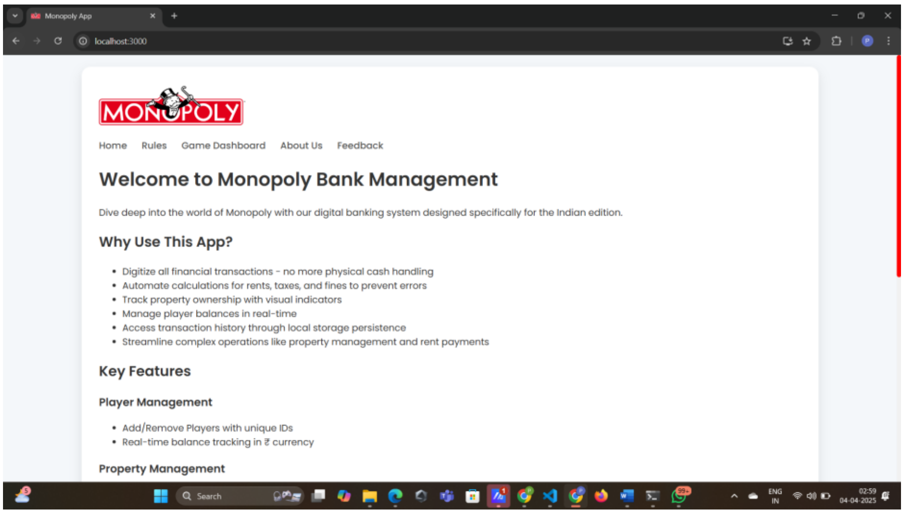
```
    font-weight: bold;
    color: white;
}

.property-item {
    padding: 8px;
    margin: 5px 0;
    border-radius: 8px;
    background-color: #f9f9f9;
    display: flex;
    justify-content: space-between;
    align-items: center;
}

.property-name {
    font-size: 0.9rem;
}

.property-price {
    font-weight: bold;
    color: #0072bb;
}

.transaction-modal, .property-modal {
    background-color: white;
    padding: 25px;
    border-radius: 15px;
    box-shadow: 0 10px 30px rgba(0,0,0,0.2);
    width: 400px;
    max-width: 90%;
}

.modal-title {
    text-align: center;
    margin-bottom: 20px;
    color: #0072bb;
}

.modal-input {
    margin-bottom: 20px;
}

.modal-actions {
    display: flex;
    justify-content: flex-end;
    gap: 15px;
}

@media (max-width: 768px) {
    header {
        flex-direction: column;
        gap: 20px;
    }

    .logo {
        justify-content: center;
    }

    nav ul {
        justify-content: center;
        flex-wrap: wrap;
    }
}
body {
    background-color: #f4f7fa;
    font-family: 'Poppins', sans-serif;
    margin: 0;
    padding: 20px;
    color: #333;
}
```

```
h1, h2, h3 {
    font-weight: 600;
}

button {
    border-radius: 25px;
    background: linear-gradient(45deg, #ff0000, #e90000);
    color: white;
    padding: 12px 20px;
    border: none;
    cursor: pointer;
    font-family: 'Poppins', sans-serif;
    font-weight: 500;
    display: flex;
    align-items: center;
    justify-content: center;
    gap: 8px;
    transition: all 0.3s ease;
    box-shadow: 0 4px 10px rgba(0, 114, 187, 0.2);
}

button:hover {
    background: linear-gradient(45deg, #ff0101, #890000);
    transform: translateY(-2px);
    box-shadow: 0 6px 15px rgba(187, 0, 0, 0.25);
}

button:active {
    transform: translateY(1px);
}

.player-card, .property-card {
    background-color: #ffffff;
    border-radius: 15px;
    box-shadow: 0 5px 15px rgba(0, 0, 0, 0.05);
    padding: 20px;
    transition: all 0.3s ease;
    border-left: 4px solid #0072bb;
}

.player-card:hover, .property-card:hover {
    transform: translateY(-5px);
    box-shadow: 0 8px 25px rgba(0, 0, 0, 0.1);
}

input[type="text"], input[type="number"], select {
    border: 1px solid #e1e5eb;
    border-radius: 25px;
    padding: 12px 20px;
    width: 100%;
    font-family: 'Poppins', sans-serif;
    transition: all 0.3s ease;
    box-sizing: border-box;
    background-color: #f8fafc;
}

input[type="text"]:focus, input[type="number"]:focus, select:focus {
    border-color: #0072bb;
    outline: none;
    box-shadow: 0 0 0 3px rgba(0, 114, 187, 0.1);
    background-color: #fff;
}

/* Color Schemes for Property Groups */
.brown { background-color: #955436; }
.light-blue { background-color: #aae0fa; }
.pink { background-color: #d93a96; }
.orange { background-color: #f7941d; }
.red { background-color: #ed1b24; }
```

```css
.yellow { background-color: #fef200; }
.green { background-color: #1fb25a; }
.dark-blue { background-color: #0072bb; }
.railroad { background-color: #221e1f; }
.utility { background-color: #6d6e71; }

/* Animations */
@keyframes fadeIn {
    from { opacity: 0; transform: translateY(10px); }
    to { opacity: 1; transform: translateY(0); }
}

@keyframes pulse {
    0% { transform: scale(1); }
    50% { transform: scale(1.05); }
    100% { transform: scale(1); }
}

.fade-in {
    animation: fadeIn 0.5s ease forwards;
}

/* Media Query to make it responsive */
@media (max-width: 768px) {
    .container {
        padding: 15px;
    }

    .page-title {
        font-size: 1.5rem;
    }

    .action-buttons {
        grid-template-columns: 1fr 1fr;
    }
    .players-grid {
        grid-template-columns: 1fr;
    }
}

/* Custom Scrollbar */
::-webkit-scrollbar {
    width: 8px;
    height: 8px;
}

::-webkit-scrollbar-track {
    background: #f1f1f1;
    border-radius: 10px;
}

::-webkit-scrollbar-thumb {
    background: #ff0000;
    border-radius: 10px;
}

::-webkit-scrollbar-thumb:hover {
    background: #ff0000;
}

html{
    scroll-behavior: smooth;
}
```

7. Input Output

7.1 Home



Welcome to Monopoly Bank Management

Dive deep into the world of Monopoly with our digital banking system designed specifically for the Indian edition.

Why Use This App?

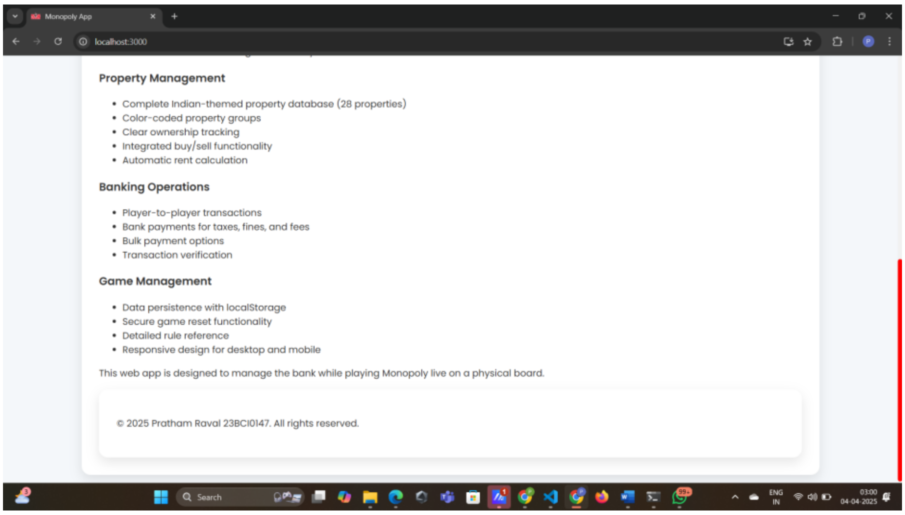
- Digitize all financial transactions - no more physical cash handling
- Automate calculations for rents, taxes, and fines to prevent errors
- Track property ownership with visual indicators
- Manage player balances in real-time
- Access transaction history through local storage persistence
- Streamline complex operations like property management and rent payments

Key Features

Player Management

- Add/Remove Players with unique IDs
- Real-time balance tracking in ₹ currency

Property Management



Property Management

- Complete Indian-themed property database (28 properties)
- Color-coded property groups
- Clear ownership tracking
- Integrated buy/sell functionality
- Automatic rent calculation

Banking Operations

- Player-to-player transactions
- Bank payments for taxes, fines, and fees
- Bulk payment options
- Transaction verification

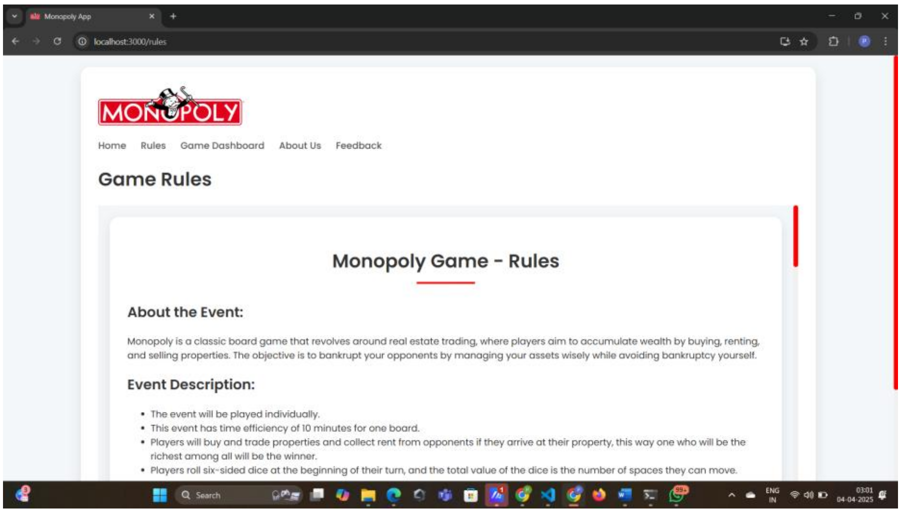
Game Management

- Data persistence with localStorage
- Secure game reset functionality
- Detailed rule reference
- Responsive design for desktop and mobile

This web app is designed to manage the bank while playing Monopoly live on a physical board.

© 2025 Pratham Raval 23BCI0147. All rights reserved.

7.2 Rules



Game Rules

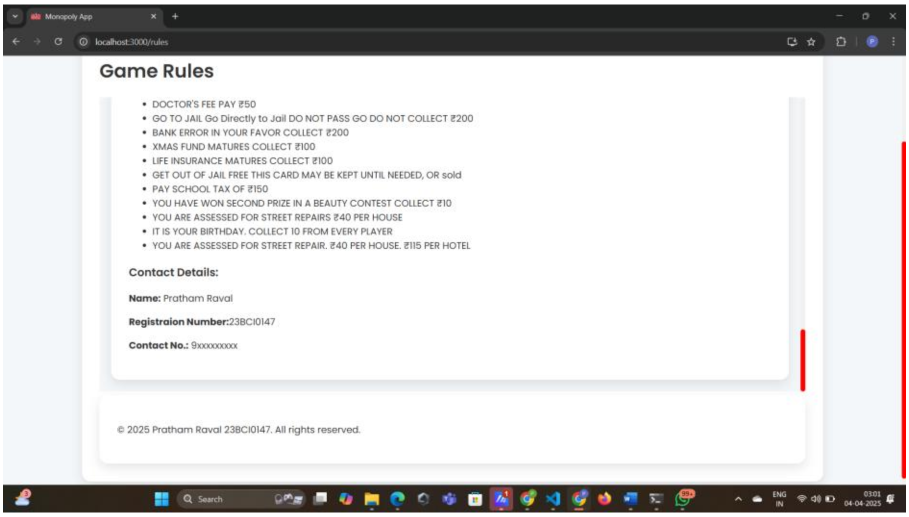
Monopoly Game - Rules

About the Event:

Monopoly is a classic board game that revolves around real estate trading, where players aim to accumulate wealth by buying, renting, and selling properties. The objective is to bankrupt your opponents by managing your assets wisely while avoiding bankruptcy yourself.

Event Description:

- The event will be played individually.
- This event has time efficiency of 10 minutes for one board.
- Players will buy and trade properties and collect rent from opponents if they arrive at their property, this way one who will be the richest among all will be the winner.
- Players roll six-sided dice at the beginning of their turn, and the total value of the dice is the number of spaces they can move.



Game Rules

- DOCTOR'S FEE PAY ₹50
- GO TO JAIL Go Directly to Jail DO NOT PASS GO DO NOT COLLECT ₹200
- BANK ERROR IN YOUR FAVOR COLLECT ₹200
- XMAS FUND MATURES COLLECT ₹100
- LIFE INSURANCE MATURES COLLECT ₹100
- GET OUT OF JAIL FREE THIS CARD MAY BE KEPT UNTIL NEEDED, OR sold
- PAY SCHOOL TAX OF ₹150
- YOU HAVE WON SECOND PRIZE IN A BEAUTY CONTEST COLLECT ₹10
- YOU ARE ASSESSED FOR STREET REPAIRS ₹40 PER HOUSE
- IT IS YOUR BIRTHDAY. COLLECT 10 FROM EVERY PLAYER
- YOU ARE ASSESSED FOR STREET REPAIR. ₹40 PER HOUSE. ₹115 PER HOTEL

Contact Details:

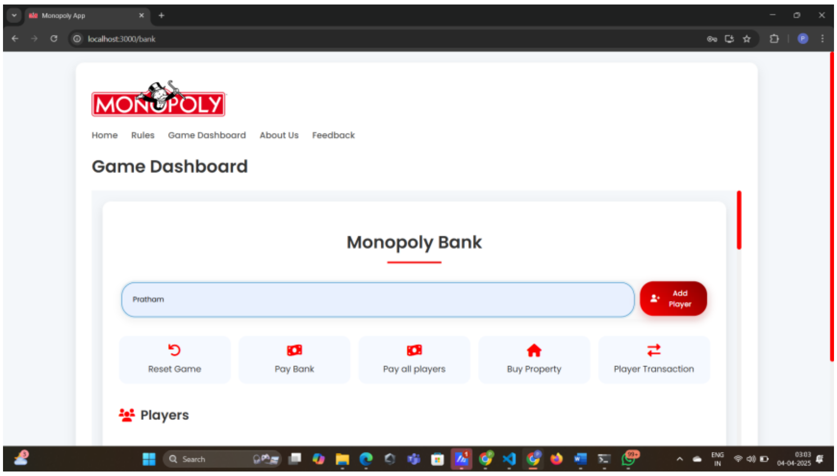
Name: Pratham Raval

Registraion Number:23BCI0147

Contact No.: 9xxxxxxxxx

© 2025 Pratham Raval 23BCI0147. All rights reserved.

7.3 Game Dashboard

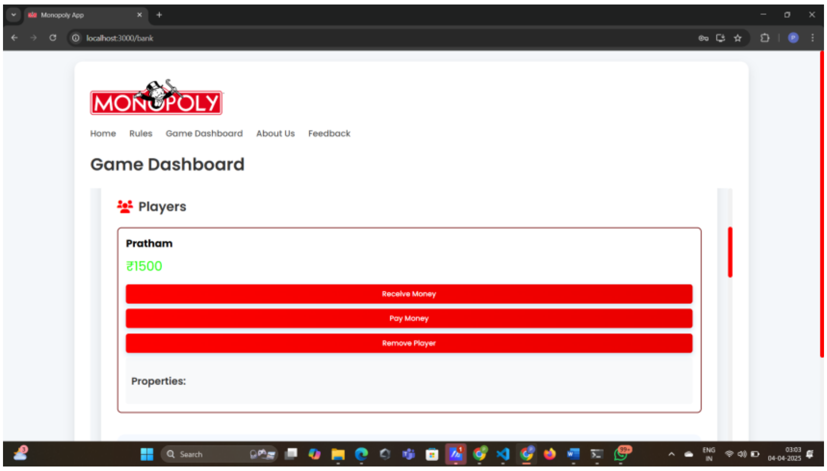


Game Dashboard

Monopoly Bank

Pratham

Players



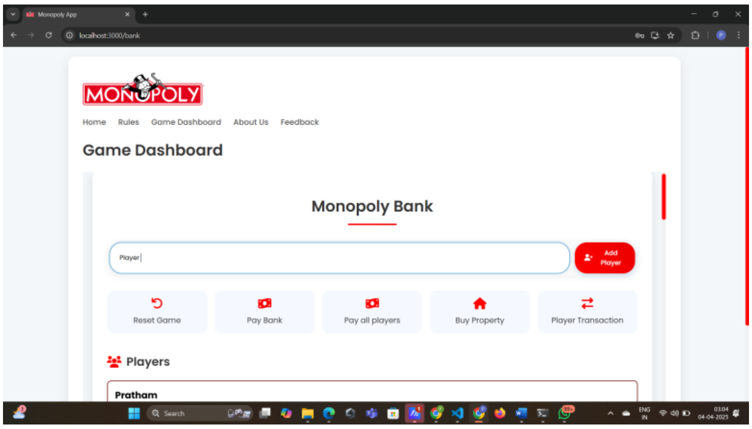
Game Dashboard

Players

Pratham

₹1500

Properties:



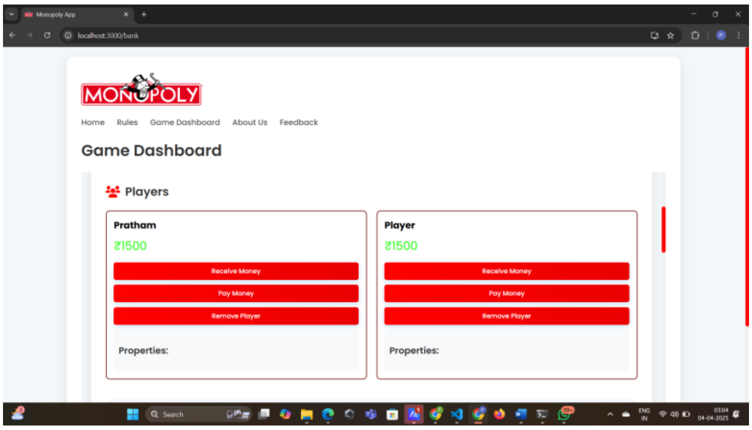
Game Dashboard

Monopoly Bank

Player|

Players

Pratham



Game Dashboard

Players

Pratham

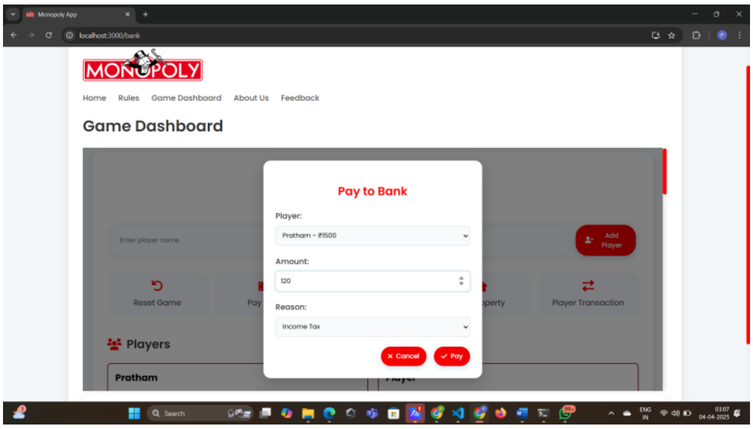
₹1500

Properties:

Player

₹1500

Properties:



Game Dashboard

Pay to Bank

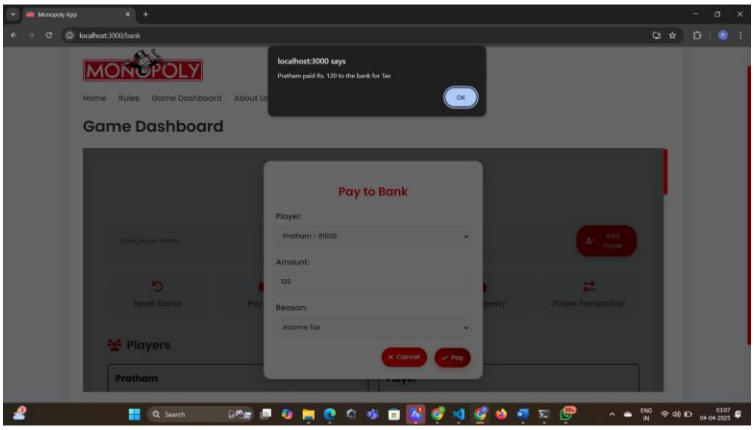
Player: Pratham - ₹1500

Amount: 120

Reason: Income Tax

Players

Pratham



localhost:3000 says

Pratham paid Rs. 120 to the bank for Tax

Game Dashboard

Pay to Bank

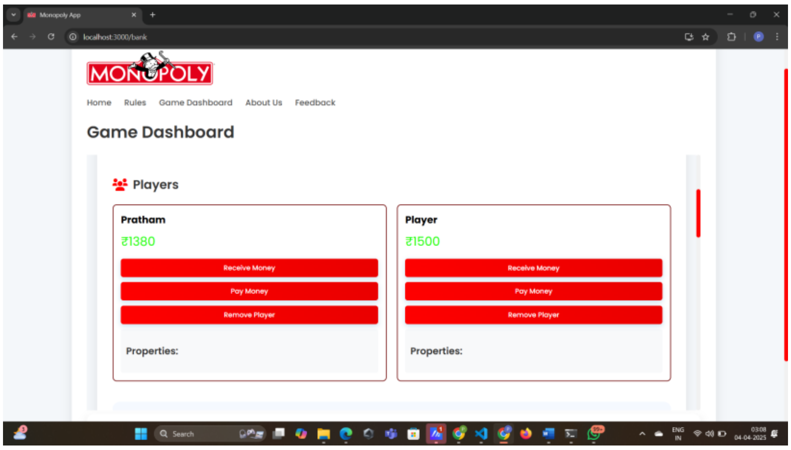
Player: Pratham - ₹1500

Amount: 120

Reason: Income Tax

Players

Pratham


Monopoly App

localhost:3000/bank

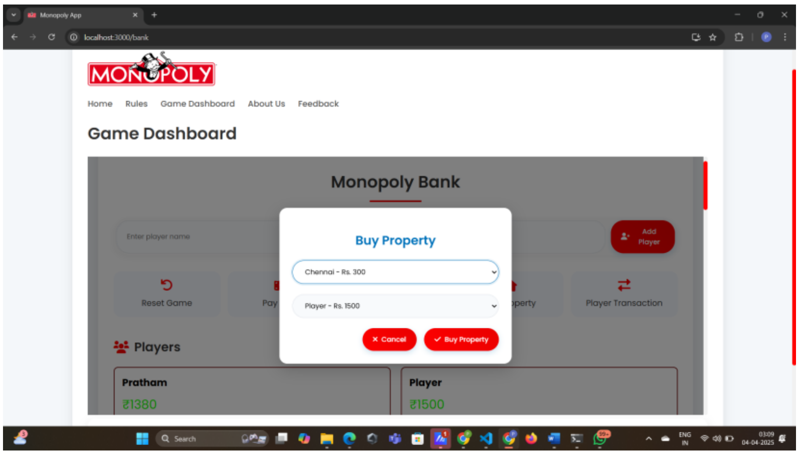
Home Rules Game Dashboard About Us Feedback

Game Dashboard

Players

Pratham	Player
₹1380	₹1500
Receive Money Pay Money Remove Player	Receive Money Pay Money Remove Player
Properties:	Properties:

03:08
04-04-2025


Monopoly App

localhost:3000/bank

Home Rules Game Dashboard About Us Feedback

Game Dashboard

Monopoly Bank

Enter player name

Add Player

Reset Game

Pay

perty

Player Transaction

Buy Property

Chennai - Rs. 300

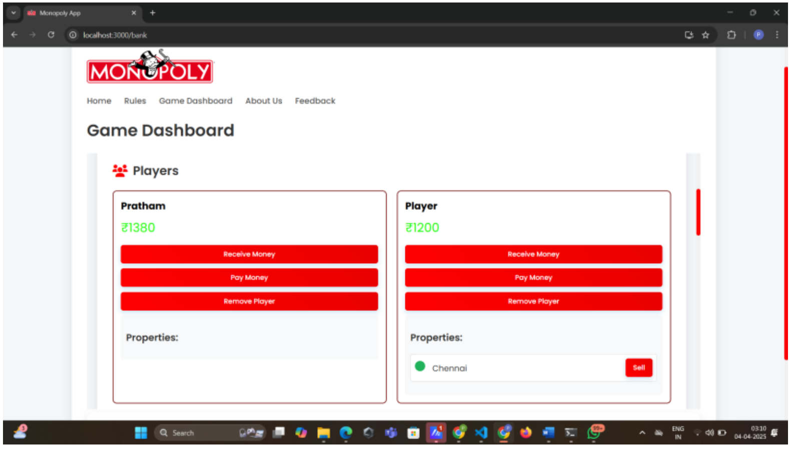
Player - Rs. 1500

✕ Cancel ✓ Buy Property

Players

Pratham	Player
₹1380	₹1500

03:09
04-04-2025


Monopoly App

localhost:3000/bank

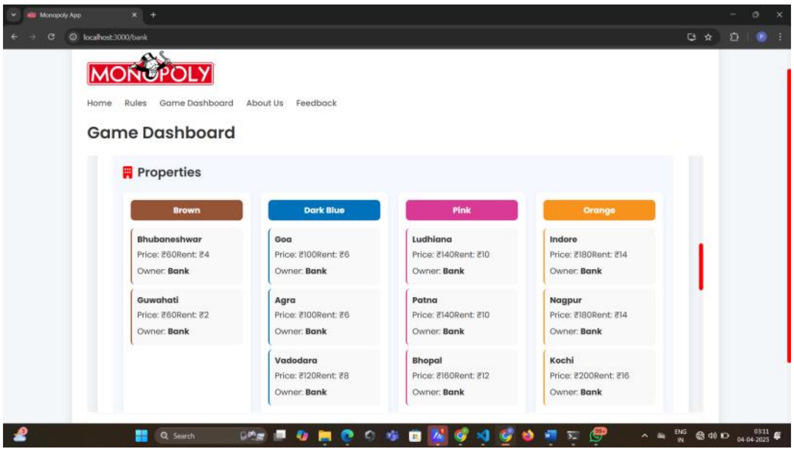
Home Rules Game Dashboard About Us Feedback

Game Dashboard

Players

Pratham	Player
₹1380	₹1200
Receive Money Pay Money Remove Player	Receive Money Pay Money Remove Player
Properties:	Properties: ● Chennai Sell

03:10
04-04-2025


Monopoly App

localhost:3000/bank

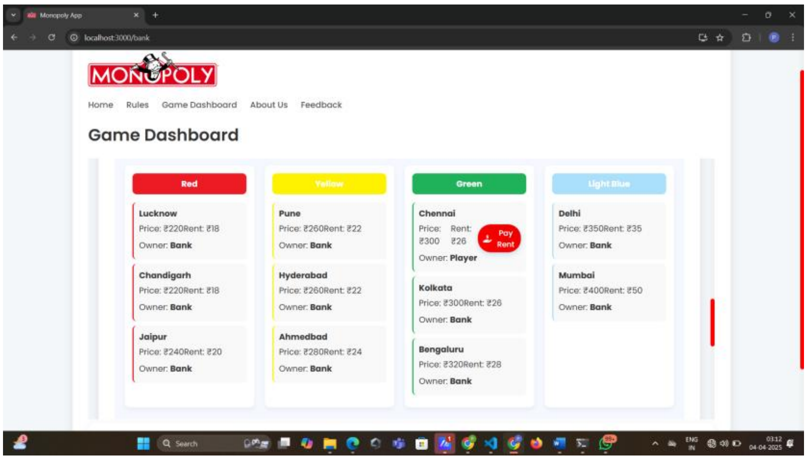
Home Rules Game Dashboard About Us Feedback

Game Dashboard

Properties

Brown	Dark Blue	Pink	Orange
Bhubaneshwar Price: ₹60Rent: ₹4 Owner: **Bank**	Goa Price: ₹100Rent: ₹6 Owner: **Bank**	Ludhiana Price: ₹140Rent: ₹10 Owner: **Bank**	Indore Price: ₹180Rent: ₹14 Owner: **Bank**
Guwahati Price: ₹60Rent: ₹2 Owner: **Bank**	Agra Price: ₹100Rent: ₹6 Owner: **Bank**	Patna Price: ₹140Rent: ₹10 Owner: **Bank**	Nagpur Price: ₹180Rent: ₹14 Owner: **Bank**
	Vadodara Price: ₹120Rent: ₹8 Owner: **Bank**	Bhopal Price: ₹160Rent: ₹12 Owner: **Bank**	Kochi Price: ₹200Rent: ₹16 Owner: **Bank**

03:11
04-04-2025



Game Dashboard

Lucknow
Price: ₹220Rent: ₹18
Owner: **Bank**

Chandigarh
Price: ₹220Rent: ₹18
Owner: **Bank**

Jaipur
Price: ₹240Rent: ₹20
Owner: **Bank**

Pune
Price: ₹260Rent: ₹22
Owner: **Bank**

Hyderabad
Price: ₹260Rent: ₹22
Owner: **Bank**

Ahmedbad
Price: ₹280Rent: ₹24
Owner: **Bank**

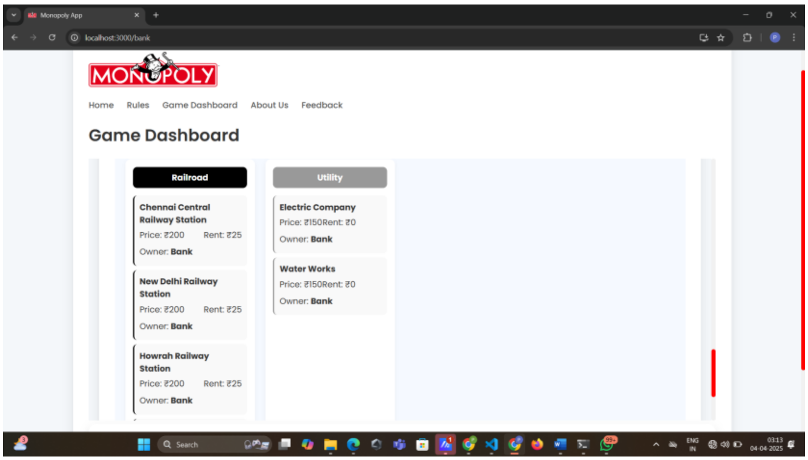
Chennai
Price: ₹300 Rent: ₹26
Owner: **Player**

Kolkata
Price: ₹300Rent: ₹26
Owner: **Bank**

Bengaluru
Price: ₹320Rent: ₹28
Owner: **Bank**

Delhi
Price: ₹350Rent: ₹35
Owner: **Bank**

Mumbai
Price: ₹400Rent: ₹50
Owner: **Bank**



Game Dashboard

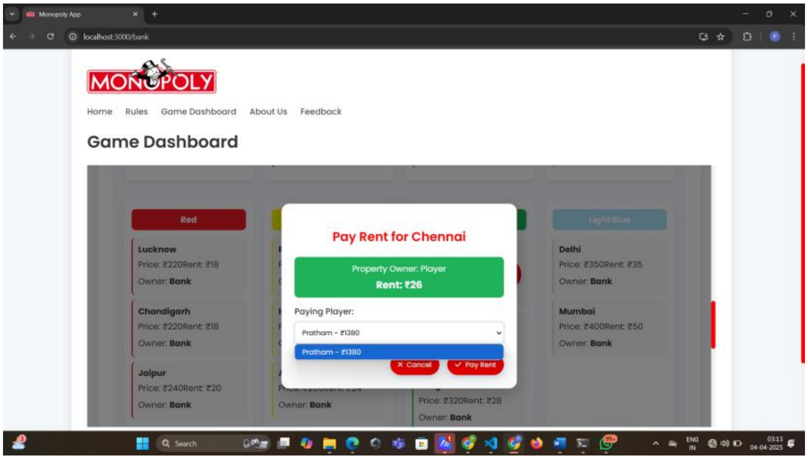
Chennai Central Railway Station
Price: ₹200 Rent: ₹25
Owner: **Bank**

New Delhi Railway Station
Price: ₹200 Rent: ₹25
Owner: **Bank**

Howrah Railway Station
Price: ₹200 Rent: ₹25
Owner: **Bank**

Electric Company
Price: ₹150Rent: ₹0
Owner: **Bank**

Water Works
Price: ₹150Rent: ₹0
Owner: **Bank**



Game Dashboard

Pay Rent for Chennai

Property Owner: Player
Rent: ₹26

Paying Player:
Pratham - ₹1380
Pratham - ₹1380

Lucknow
Price: ₹220Rent: ₹18
Owner: **Bank**

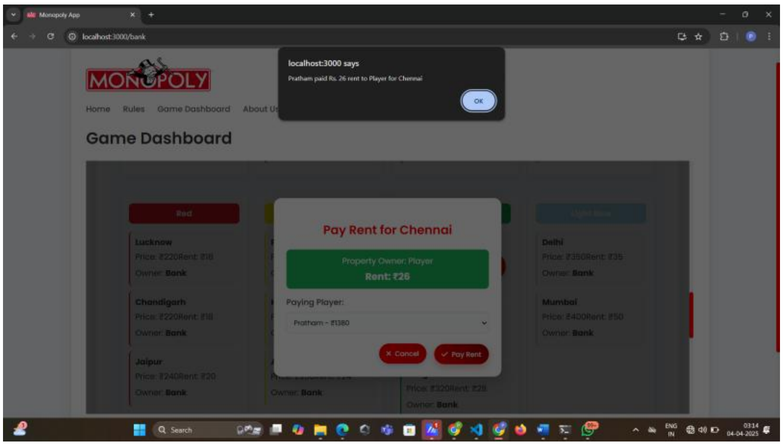
Chandigarh
Price: ₹220Rent: ₹18
Owner: **Bank**

Jaipur
Price: ₹240Rent: ₹20
Owner: **Bank**

Price: ₹320Rent: ₹28
Owner: **Bank**

Delhi
Price: ₹350Rent: ₹35
Owner: **Bank**

Mumbai
Price: ₹400Rent: ₹50
Owner: **Bank**



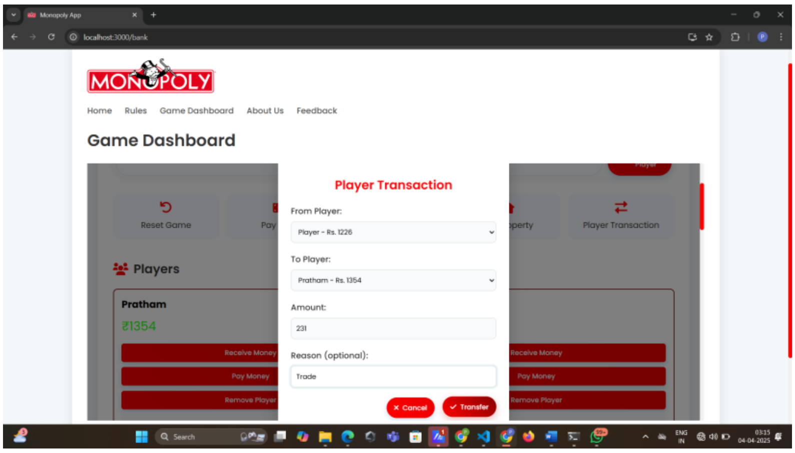
localhost:3000 says
Pratham paid Rs. 26 rent to Player for Chennai

Game Dashboard

Pay Rent for Chennai

Property Owner: Player
Rent: ₹26

Paying Player:
Pratham - ₹1380



Game Dashboard

Player Transaction

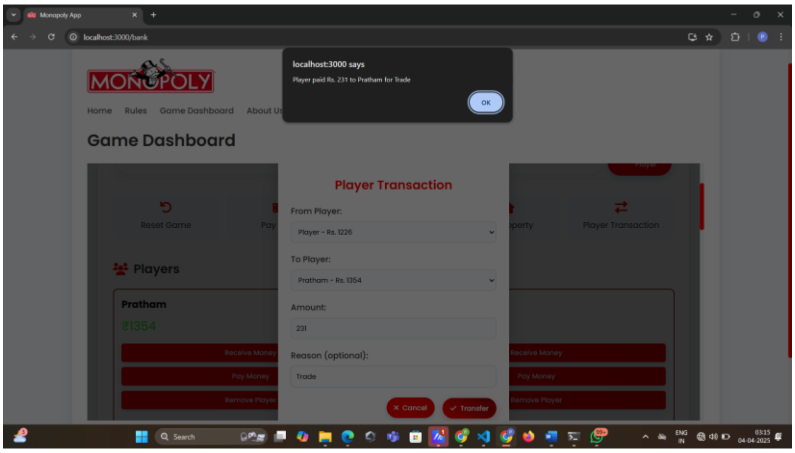
From Player: Player - Rs. 1226

To Player: Pratham - Rs. 1354

Amount: 231

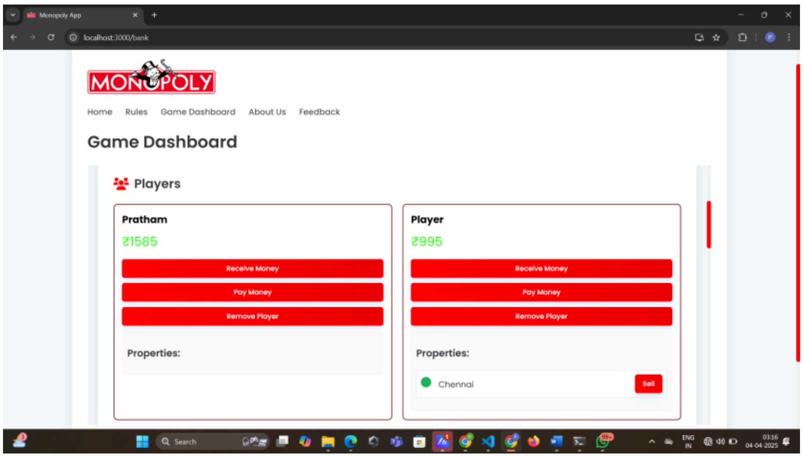
Reason (optional): Trade

Pratham ₹1354



localhost:3000 says

Player paid Rs. 231 to Pratham for Trade



Game Dashboard

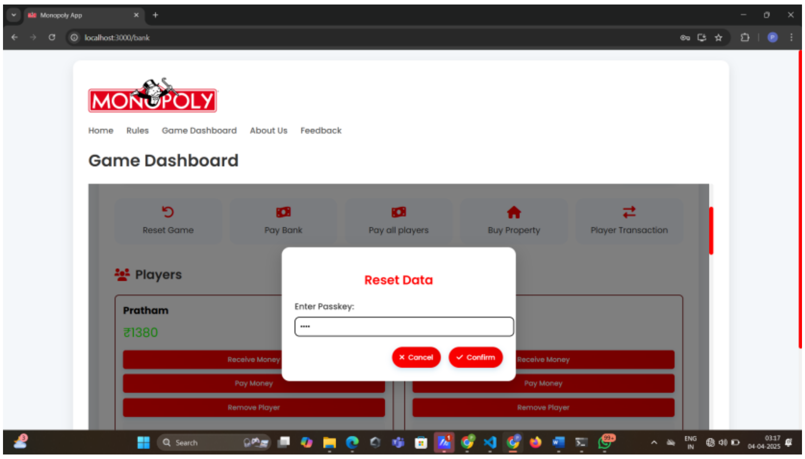
Pratham ₹1585

Properties:

Player ₹995

Properties:

● Chennai

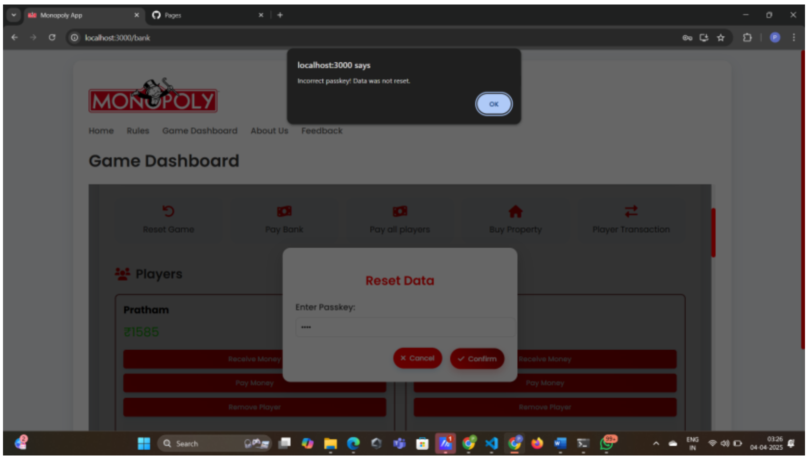


Game Dashboard

Reset Data

Enter Passkey: ••••

Pratham ₹1380



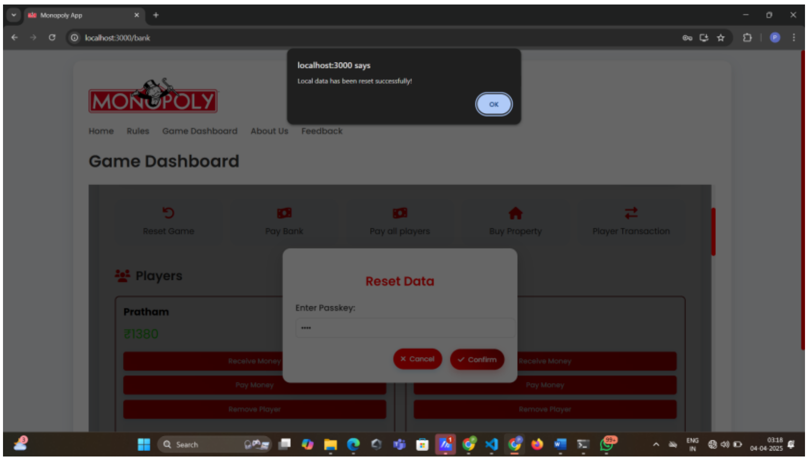
localhost:3000 says

Incorrect passkey! Data was not reset.

Game Dashboard

Reset Data

Enter Passkey: ••••



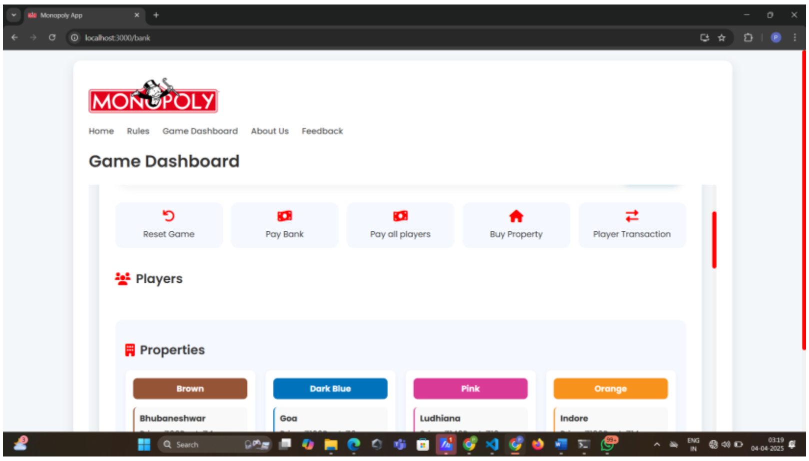
localhost:3000 says

Local data has been reset successfully!

Game Dashboard

Reset Data

Enter Passkey: ••••

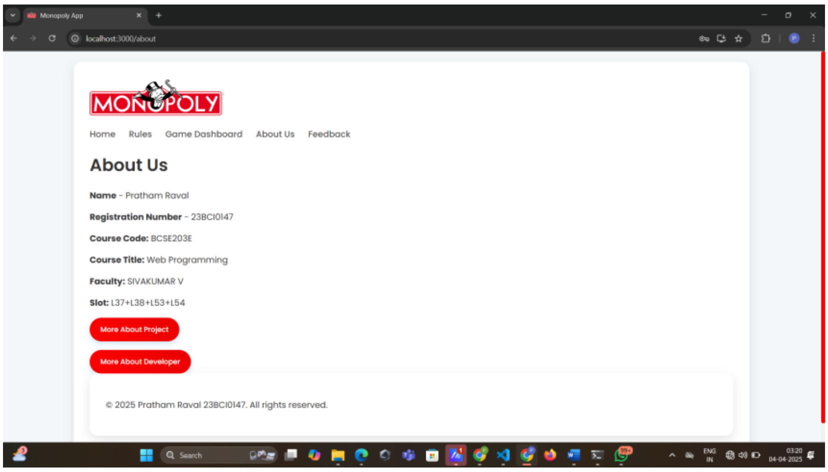


Game Dashboard

Players

Properties

7.4 About Us page



About Us

Name - Pratham Raval

Registration Number - 23BCI0147

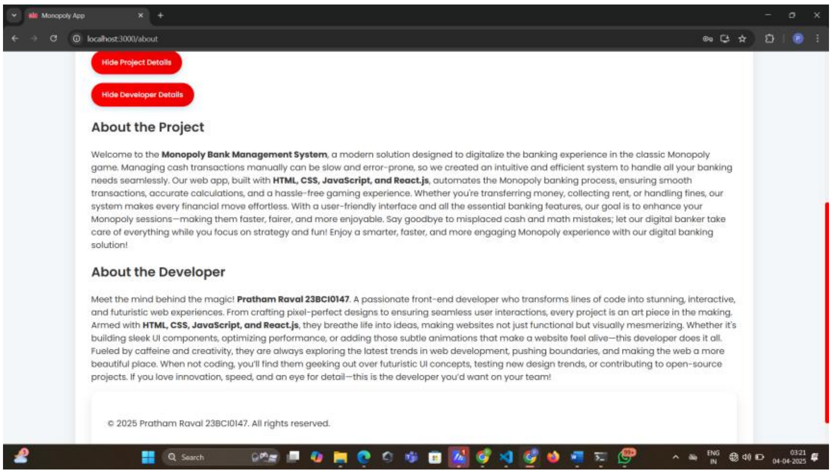
Course Code: BCSE203E

Course Title: Web Programming

Faculty: SIVAKUMAR V

Slot: L37+L38+L53+L54

© 2025 Pratham Raval 23BCI0147. All rights reserved.



About the Project

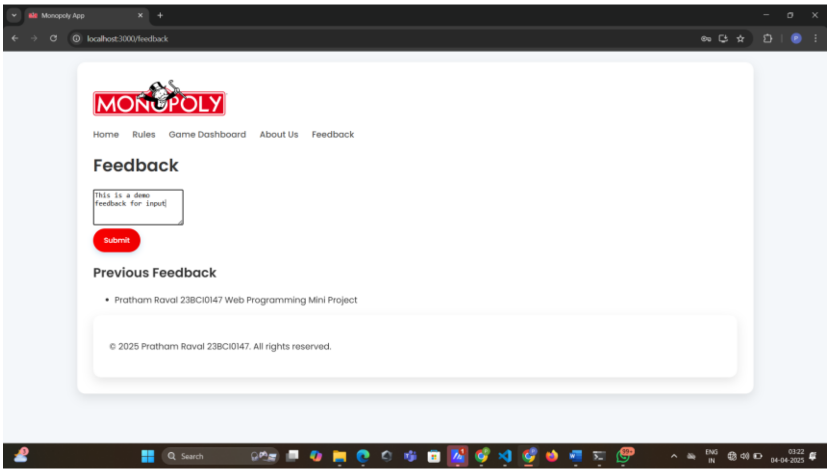
Welcome to the **Monopoly Bank Management System**, a modern solution designed to digitalize the banking experience in the classic Monopoly game. Managing cash transactions manually can be slow and error-prone, so we created an intuitive and efficient system to handle all your banking needs seamlessly. Our web app, built with **HTML, CSS, JavaScript, and React.js**, automates the Monopoly banking process, ensuring smooth transactions, accurate calculations, and a hassle-free gaming experience. Whether you're transferring money, collecting rent, or handling fines, our system makes every financial move effortless. With a user-friendly interface and all the essential banking features, our goal is to enhance your Monopoly sessions—making them faster, fairer, and more enjoyable. Say goodbye to misplaced cash and math mistakes; let our digital banker take care of everything while you focus on strategy and fun! Enjoy a smarter, faster, and more engaging Monopoly experience with our digital banking solution!

About the Developer

Meet the mind behind the magic! **Pratham Raval 23BCI0147**. A passionate front-end developer who transforms lines of code into stunning, interactive, and futuristic web experiences. From crafting pixel-perfect designs to ensuring seamless user interactions, every project is an art piece in the making. Armed with **HTML, CSS, JavaScript, and React.js**, they breathe life into ideas, making websites not just functional but visually mesmerizing. Whether it's building sleek UI components, optimizing performance, or adding those subtle animations that make a website feel alive—this developer does it all. Fueled by caffeine and creativity, they are always exploring the latest trends in web development, pushing boundaries, and making the web a more beautiful place. When not coding, you'll find them geeking out over futuristic UI concepts, testing new design trends, or contributing to open-source projects. If you love innovation, speed, and an eye for detail—this is the developer you'd want on your team!

© 2025 Pratham Raval 23BCI0147. All rights reserved.

7.5 Feedback Page



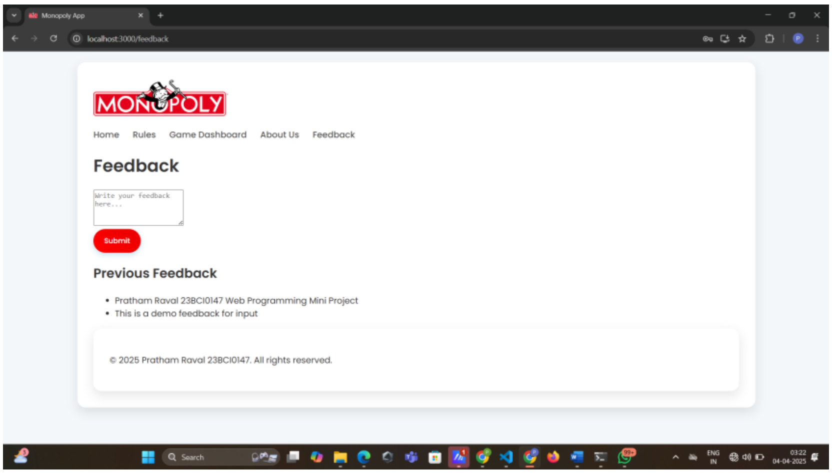
Feedback

This is a demo feedback for input

Previous Feedback

- Pratham Raval 23BCI0147 Web Programming Mini Project

© 2025 Pratham Raval 23BCI0147. All rights reserved.



Feedback

Write your feedback here...

Previous Feedback

- Pratham Raval 23BCI0147 Web Programming Mini Project
- This is a demo feedback for input

© 2025 Pratham Raval 23BCI0147. All rights reserved.

8. Portfolio and Hyperlinks

- Complete Project – Monopoly Bank management System using React js, html, css and javascript
- https://github.com/PrathamR015/Monopoly-React

- Complete Project – Monopoly (html, css and javascript version) (Without React)
- https://github.com/PrathamR015/Monopoly
- Hosted Website for this version. The website works on all devices as media queries are used to make it responsive.
- https://prathamr015.github.io/Monopoly/