MIT Art, Design and Technology University
MIT School of Computing, Pune

Department of Information Technology

Lab Manual

Practical - CPAD

Class - T.Y. (SEM-VI), SMAD

Batch - SMAD

Mustafa Bhewala- MITU22BTIT0050

A.Y. 2024 – 2025 (SEM-VI)

Assignment No 1: Project Assignment based on Unit-I and Unit-II

1) Project Title: Personal Finance Tracker - Web (React Native)

   a) Problem Statement:

      i) To develop a personal finance tracker web application using ReactJS with routing functionality that allows users to manage income and expenses, view reports, and categorize transactions efficiently.

   b) Objective:

      i) To build a responsive single-page finance tracking application using ReactJS and React Router that includes user-defined login, dashboard, transaction entry, history, reports, and category analysis features.

   c) Theory:

      i) ReactJS is an open-source JavaScript library developed by Meta for building user interfaces, especially single-page applications (SPAs). It emphasizes component-based architecture and the use of reusable UI blocks. This experiment involves multiple React concepts and supporting technologies:

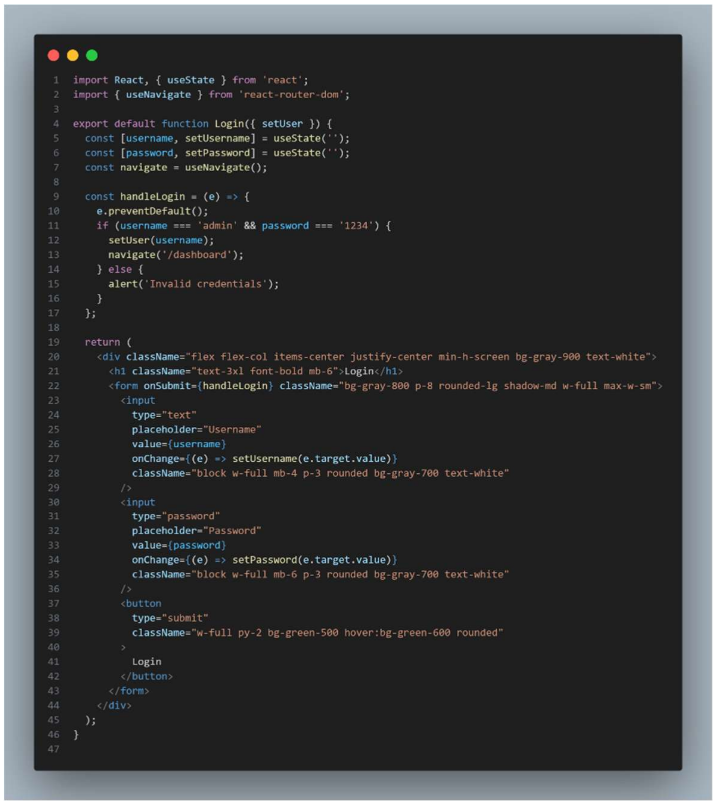
      - JSX: JavaScript XML syntax that allows HTML to be written inside JavaScript code.
      - Components: Reusable functions that return JSX. We use functional components throughout the app.
      - Hooks: React Hooks like useState, useEffect, and useContext enable state management and side effects in function components.
      - Context API: A lightweight state management tool to share global state (transactions, username) without prop drilling.
      - React Router: Provides navigation and routing capabilities between components using <Route> and <Link>.
      - Tailwind CSS: Utility-first CSS framework used for rapid and responsive UI development.

- Chart.js via react-chartjs-2: Library for rendering responsive and interactive pie and bar charts.

The app combines these technologies to demonstrate real-world development practices in a modular, scalable web application.

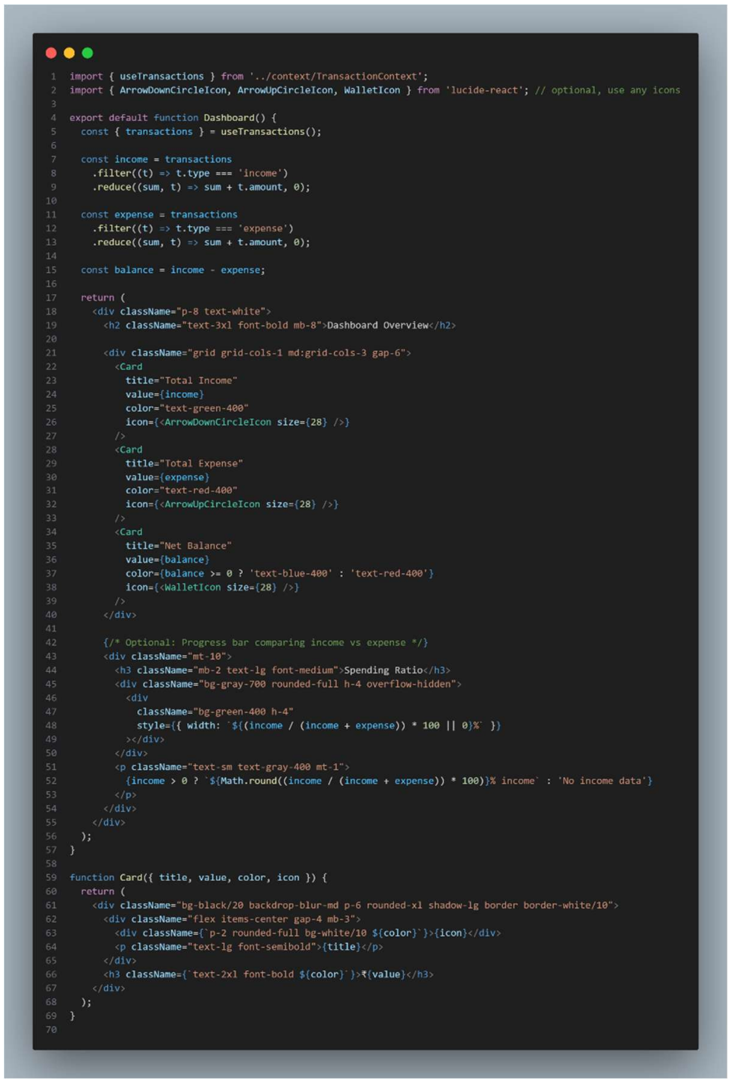
d) Code:

i) Login Page:



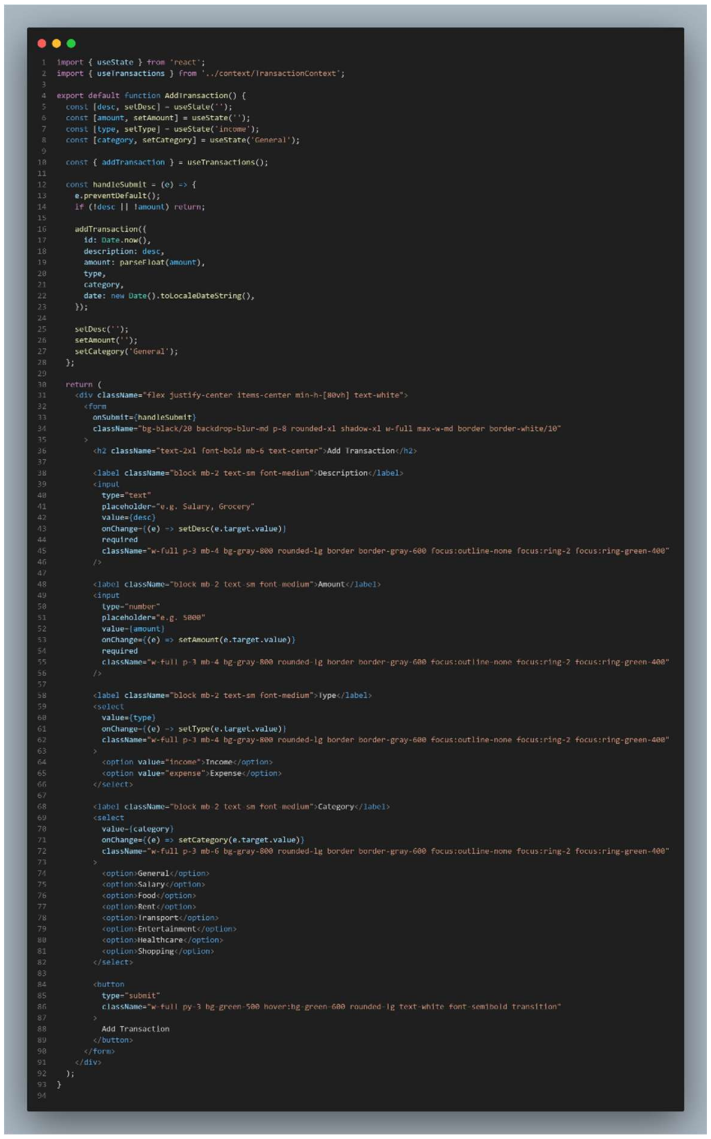
```jsx
import React, { useState } from 'react';
import { useNavigate } from 'react-router-dom';

export default function Login({ setUser }) {
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');
  const navigate = useNavigate();

  const handleLogin = (e) => {
    e.preventDefault();
    if (username === 'admin' && password === '1234') {
      setUser(username);
      navigate('/dashboard');
    } else {
      alert('Invalid credentials');
    }
  };

  return (
    <div className="flex flex-col items-center justify-center min-h-screen bg-gray-900 text-white">
      <h1 className="text-3xl font-bold mb-6">Login</h1>
      <form onSubmit={handleLogin} className="bg-gray-800 p-8 rounded-lg shadow-md w-full max-w-sm">
        <input
          type="text"
          placeholder="Username"
          value={username}
          onChange={(e) => setUsername(e.target.value)}
          className="block w-full mb-4 p-3 rounded bg-gray-700 text-white"
        />
        <input
          type="password"
          placeholder="Password"
          value={password}
          onChange={(e) => setPassword(e.target.value)}
          className="block w-full mb-6 p-3 rounded bg-gray-700 text-white"
        />
        <button
          type="submit"
          className="w-full py-2 bg-green-500 hover:bg-green-600 rounded"
        >
          Login
        </button>
      </form>
    </div>
  );
}
```

ii) Dashboard:



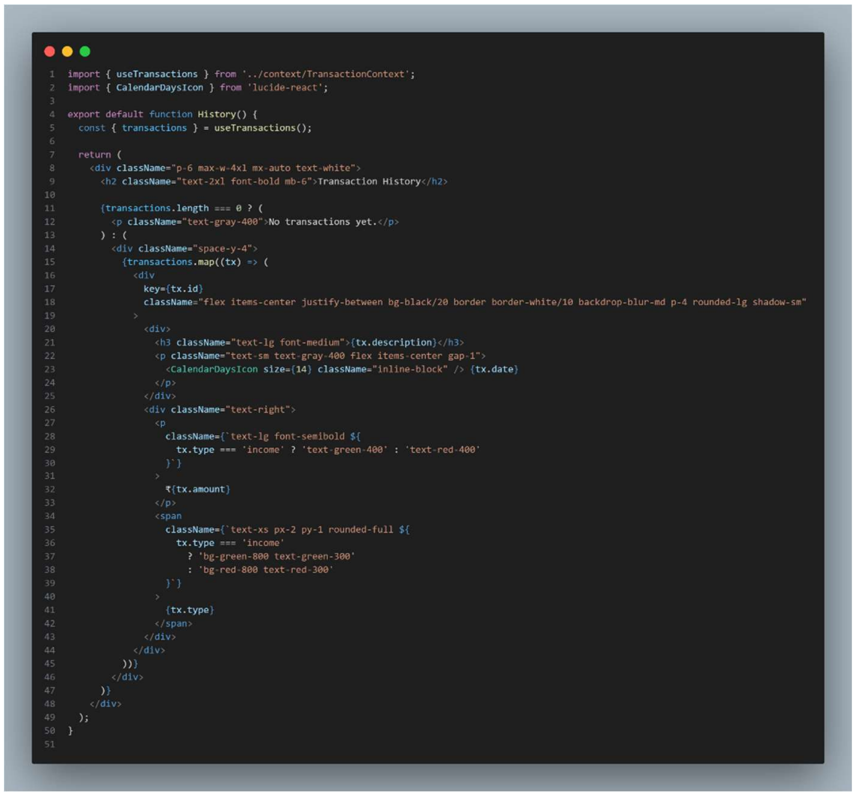
```jsx
import { useTransactions } from '../context/TransactionContext';
import { ArrowDownCircleIcon, ArrowUpCircleIcon, WalletIcon } from 'lucide-react'; // optional, use any icons

export default function Dashboard() {
  const { transactions } = useTransactions();

  const income = transactions
    .filter((t) => t.type === 'income')
    .reduce((sum, t) => sum + t.amount, 0);

  const expense = transactions
    .filter((t) => t.type === 'expense')
    .reduce((sum, t) => sum + t.amount, 0);

  const balance = income - expense;

  return (
    <div className="p-8 text-white">
      <h2 className="text-3xl font-bold mb-8">Dashboard Overview</h2>

      <div className="grid grid-cols-1 md:grid-cols-3 gap-6">
        <Card
          title="Total Income"
          value={income}
          color="text-green-400"
          icon={<ArrowDownCircleIcon size={28} />}
        />
        <Card
          title="Total Expense"
          value={expense}
          color="text-red-400"
          icon={<ArrowUpCircleIcon size={28} />}
        />
        <Card
          title="Net Balance"
          value={balance}
          color={balance >= 0 ? 'text-blue-400' : 'text-red-400'}
          icon={<WalletIcon size={28} />}
        />
      </div>

      {/* Optional: Progress bar comparing income vs expense */}
      <div className="mt-10">
        <h3 className="mb-2 text-lg font-medium">Spending Ratio</h3>
        <div className="bg-gray-700 rounded-full h-4 overflow-hidden">
          <div
            className="bg-green-400 h-4"
            style={{ width: `${(income / (income + expense)) * 100 || 0}%` }}
          ></div>
        </div>
        <p className="text-sm text-gray-400 mt-1">
          {income > 0 ? `${Math.round((income / (income + expense)) * 100)}% income` : 'No income data'}
        </p>
      </div>
    </div>
  );
}

function Card({ title, value, color, icon }) {
  return (
    <div className="bg-black/20 backdrop-blur-md p-6 rounded-xl shadow-lg border border-white/10">
      <div className="flex items-center gap-4 mb-3">
        <div className={`p-2 rounded-full bg-white/10 ${color}`}>{icon}</div>
        <p className="text-lg font-semibold">{title}</p>
      </div>
      <h3 className={`text-2xl font-bold ${color}`}>₹{value}</h3>
    </div>
  );
}
```

iii) Add Transaction:



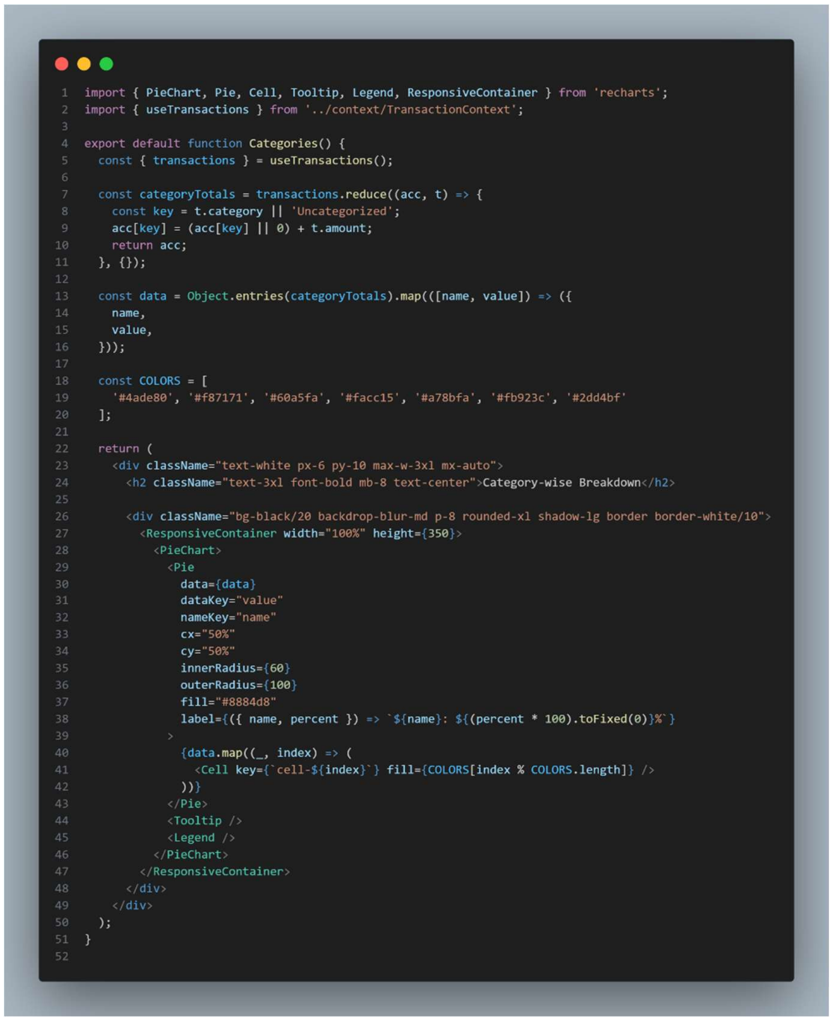
```
import { useState } from 'react';
import { useTransactions } from '../context/TransactionContext';

export default function AddTransaction() {
  const [desc, setDesc] = useState('');
  const [amount, setAmount] = useState('');
  const [type, setType] = useState('income');
  const [category, setCategory] = useState('General');

  const { addTransaction } = useTransactions();

  const handleSubmit = (e) => {
    e.preventDefault();
    if (!desc || !amount) return;

    addTransaction({
      id: Date.now(),
      description: desc,
      amount: parseFloat(amount),
      type,
      category,
      date: new Date().toLocaleDateString(),
    });

    setDesc('');
    setAmount('');
    setCategory('General');
  };

  return (
    <div className="flex justify-center items-center min-h-[80vh] text-white">
      <form
        onSubmit={handleSubmit}
        className="bg-black/20 backdrop-blur-md p-8 rounded-xl shadow-xl w-full max-w-md border border-white/10"
      >
        <h2 className="text-2xl font-bold mb-6 text-center">Add Transaction</h2>

        <label className="block mb-2 text-sm font-medium">Description</label>
        <input
          type="text"
          placeholder="e.g. Salary, Grocery"
          value={desc}
          onChange={(e) => setDesc(e.target.value)}
          required
          className="w-full p-3 mb-4 bg-gray-800 rounded-lg border border-gray-600 focus:outline-none focus:ring-2 focus:ring-green-400"
        />

        <label className="block mb-2 text-sm font-medium">Amount</label>
        <input
          type="number"
          placeholder="e.g. 5000"
          value={amount}
          onChange={(e) => setAmount(e.target.value)}
          required
          className="w-full p-3 mb-4 bg-gray-800 rounded-lg border border-gray-600 focus:outline-none focus:ring-2 focus:ring-green-400"
        />

        <label className="block mb-2 text-sm font-medium">Type</label>
        <select
          value={type}
          onChange={(e) => setType(e.target.value)}
          className="w-full p-3 mb-4 bg-gray-800 rounded-lg border border-gray-600 focus:outline-none focus:ring-2 focus:ring-green-400"
        >
          <option value="income">Income</option>
          <option value="expense">Expense</option>
        </select>

        <label className="block mb-2 text-sm font-medium">Category</label>
        <select
          value={category}
          onChange={(e) => setCategory(e.target.value)}
          className="w-full p-3 mb-6 bg-gray-800 rounded-lg border border-gray-600 focus:outline-none focus:ring-2 focus:ring-green-400"
        >
          <option>General</option>
          <option>Salary</option>
          <option>Food</option>
          <option>Rent</option>
          <option>Transport</option>
          <option>Entertainment</option>
          <option>Healthcare</option>
          <option>Shopping</option>
        </select>

        <button
          type="submit"
          className="w-full py-3 bg-green-500 hover:bg-green-600 rounded-lg text-white font-semibold transition"
        >
          Add Transaction
        </button>
      </form>
    </div>
  );
}
```

iv) History Page



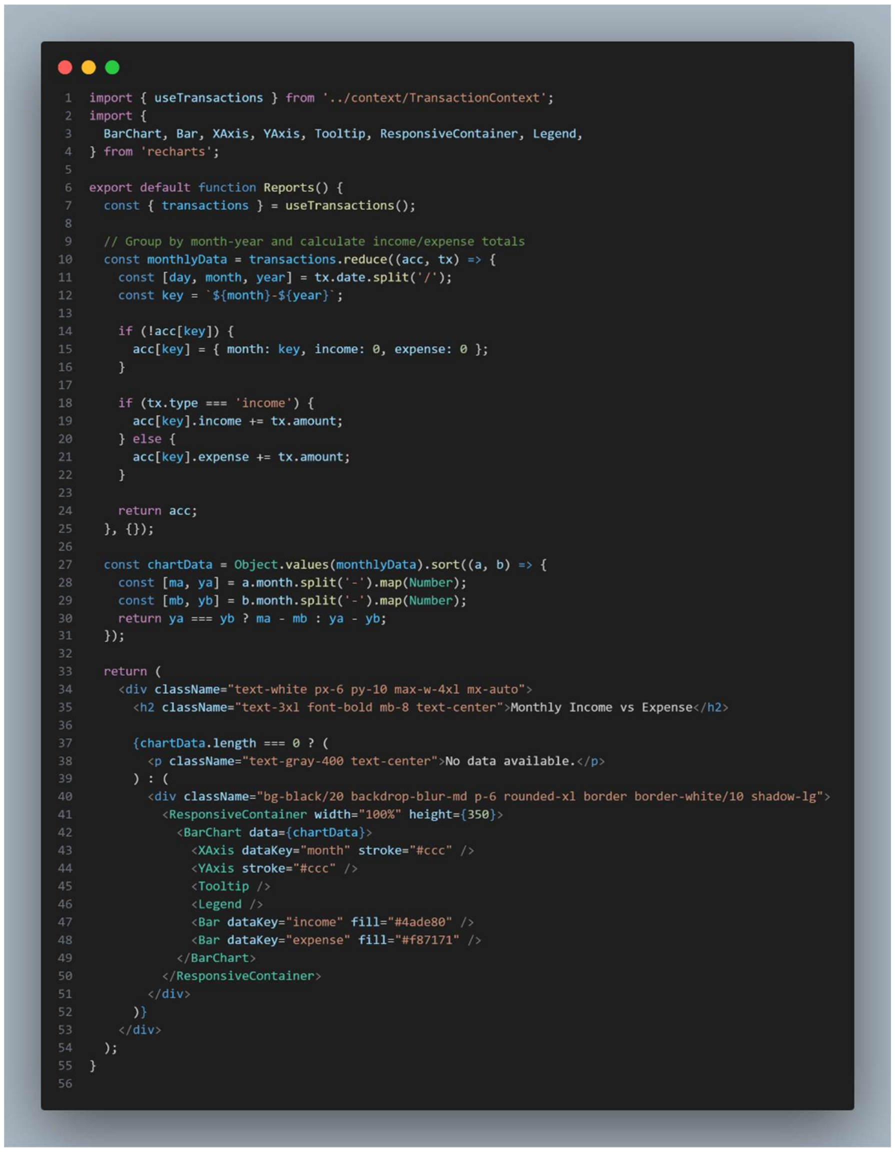
```jsx
import { useTransactions } from '../context/TransactionContext';
import { CalendarDaysIcon } from 'lucide-react';

export default function History() {
  const { transactions } = useTransactions();

  return (
    <div className="p-6 max-w-4xl mx-auto text-white">
      <h2 className="text-2xl font-bold mb-6">Transaction History</h2>

      {transactions.length === 0 ? (
        <p className="text-gray-400">No transactions yet.</p>
      ) : (
        <div className="space-y-4">
          {transactions.map((tx) => (
            <div
              key={tx.id}
              className="flex items-center justify-between bg-black/20 border border-white/10 backdrop-blur-md p-4 rounded-lg shadow-sm"
            >
              <div>
                <h3 className="text-lg font-medium">{tx.description}</h3>
                <p className="text-sm text-gray-400 flex items-center gap-1">
                  <CalendarDaysIcon size={14} className="inline-block" /> {tx.date}
                </p>
              </div>
              <div className="text-right">
                <p
                  className={`text-lg font-semibold ${
                    tx.type === 'income' ? 'text-green-400' : 'text-red-400'
                  }`}
                >
                  ₹{tx.amount}
                </p>
                <span
                  className={`text-xs px-2 py-1 rounded-full ${
                    tx.type === 'income'
                      ? 'bg-green-800 text-green-300'
                      : 'bg-red-800 text-red-300'
                  }`}
                >
                  {tx.type}
                </span>
              </div>
            </div>
          ))}
        </div>
      )}
    </div>
  );
}
```

v) Categories Page:



```jsx
import { PieChart, Pie, Cell, Tooltip, Legend, ResponsiveContainer } from 'recharts';
import { useTransactions } from '../context/TransactionContext';

export default function Categories() {
  const { transactions } = useTransactions();

  const categoryTotals = transactions.reduce((acc, t) => {
    const key = t.category || 'Uncategorized';
    acc[key] = (acc[key] || 0) + t.amount;
    return acc;
  }, {});

  const data = Object.entries(categoryTotals).map(([name, value]) => ({
    name,
    value,
  }));

  const COLORS = [
    '#4ade80', '#f87171', '#60a5fa', '#facc15', '#a78bfa', '#fb923c', '#2dd4bf'
  ];

  return (
    <div className="text-white px-6 py-10 max-w-3xl mx-auto">
      <h2 className="text-3xl font-bold mb-8 text-center">Category-wise Breakdown</h2>

      <div className="bg-black/20 backdrop-blur-md p-8 rounded-xl shadow-lg border border-white/10">
        <ResponsiveContainer width="100%" height={350}>
          <PieChart>
            <Pie
              data={data}
              dataKey="value"
              nameKey="name"
              cx="50%"
              cy="50%"
              innerRadius={60}
              outerRadius={100}
              fill="#8884d8"
              label={({ name, percent }) => `${name}: ${(percent * 100).toFixed(0)}%`}
            >
              {data.map((_, index) => (
                <Cell key={`cell-${index}`} fill={COLORS[index % COLORS.length]} />
              ))}
            </Pie>
            <Tooltip />
            <Legend />
          </PieChart>
        </ResponsiveContainer>
      </div>
    </div>
  );
}

```

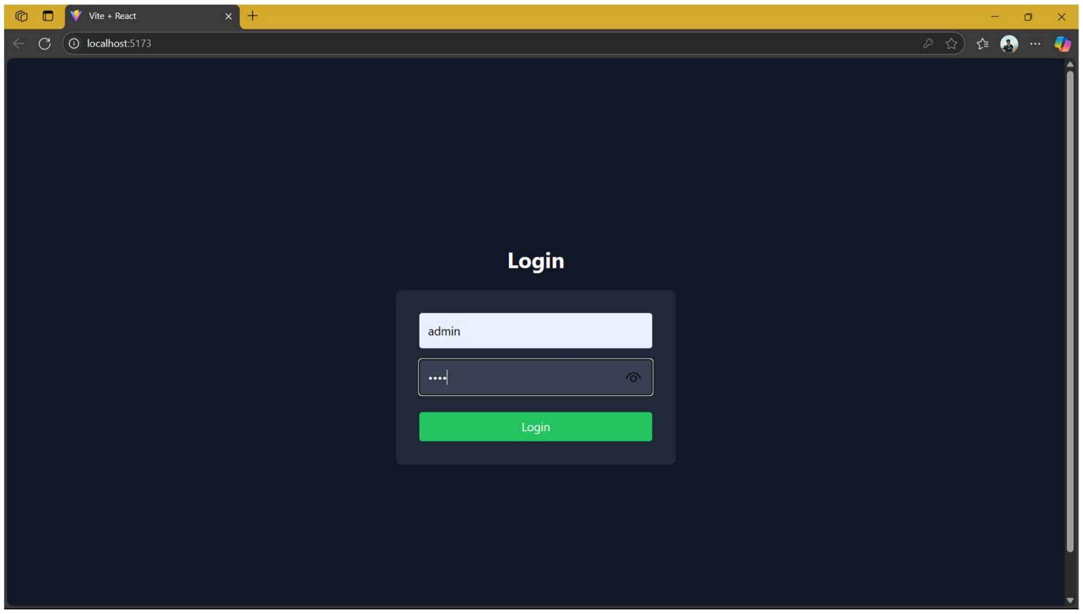
vi) Reports Page:



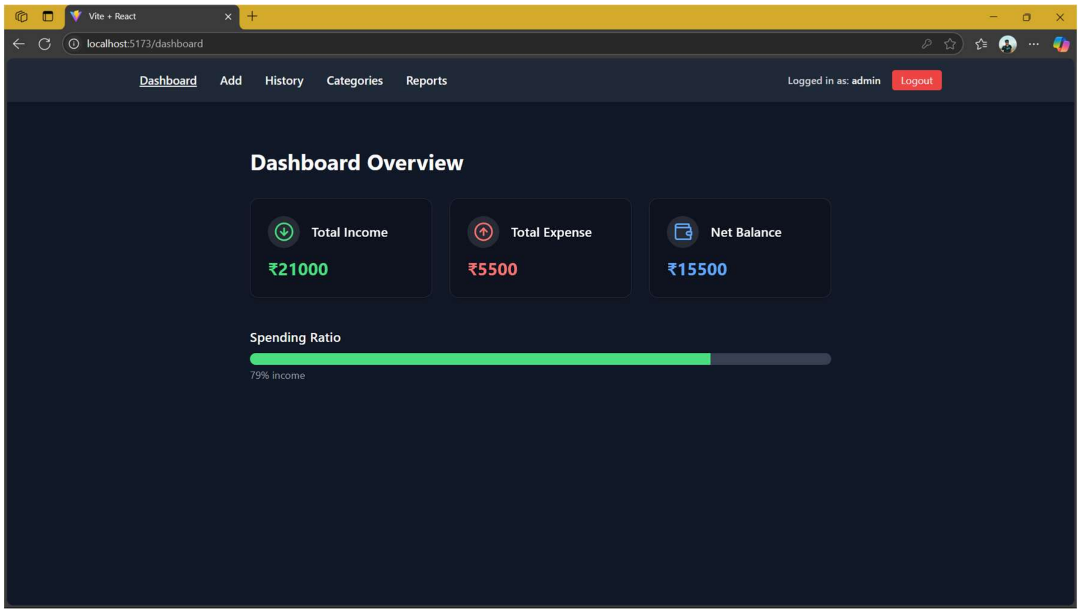
```jsx
import { useTransactions } from '../context/TransactionContext';
import {
  BarChart, Bar, XAxis, YAxis, Tooltip, ResponsiveContainer, Legend,
} from 'recharts';

export default function Reports() {
  const { transactions } = useTransactions();

  // Group by month-year and calculate income/expense totals
  const monthlyData = transactions.reduce((acc, tx) => {
    const [day, month, year] = tx.date.split('/');
    const key = `${month}-${year}`;

    if (!acc[key]) {
      acc[key] = { month: key, income: 0, expense: 0 };
    }

    if (tx.type === 'income') {
      acc[key].income += tx.amount;
    } else {
      acc[key].expense += tx.amount;
    }

    return acc;
  }, {});

  const chartData = Object.values(monthlyData).sort((a, b) => {
    const [ma, ya] = a.month.split('-').map(Number);
    const [mb, yb] = b.month.split('-').map(Number);
    return ya === yb ? ma - mb : ya - yb;
  });

  return (
    <div className="text-white px-6 py-10 max-w-4xl mx-auto">
      <h2 className="text-3xl font-bold mb-8 text-center">Monthly Income vs Expense</h2>

      {chartData.length === 0 ? (
        <p className="text-gray-400 text-center">No data available.</p>
      ) : (
        <div className="bg-black/20 backdrop-blur-md p-6 rounded-xl border border-white/10 shadow-lg">
          <ResponsiveContainer width="100%" height={350}>
            <BarChart data={chartData}>
              <XAxis dataKey="month" stroke="#ccc" />
              <YAxis stroke="#ccc" />
              <Tooltip />
              <Legend />
              <Bar dataKey="income" fill="#4ade80" />
              <Bar dataKey="expense" fill="#f87171" />
            </BarChart>
          </ResponsiveContainer>
        </div>
      )}
    </div>
  );
}

```

e) Output Screenshots:

i) Screenshot 1: Login Page (ReactJS)



ii) Screenshot 2: Dashboard



Dashboard Overview

Total Income ₹21000

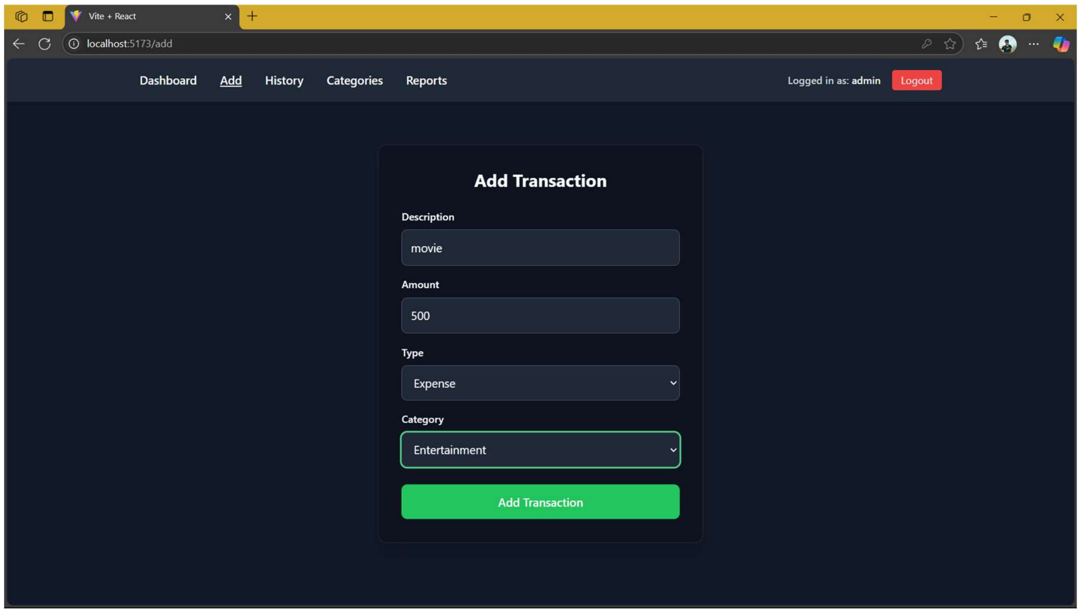
Total Expense ₹5500

Net Balance ₹15500

Spending Ratio

79% income

iii) Screenshot 3: Add Transaction form



Add Transaction

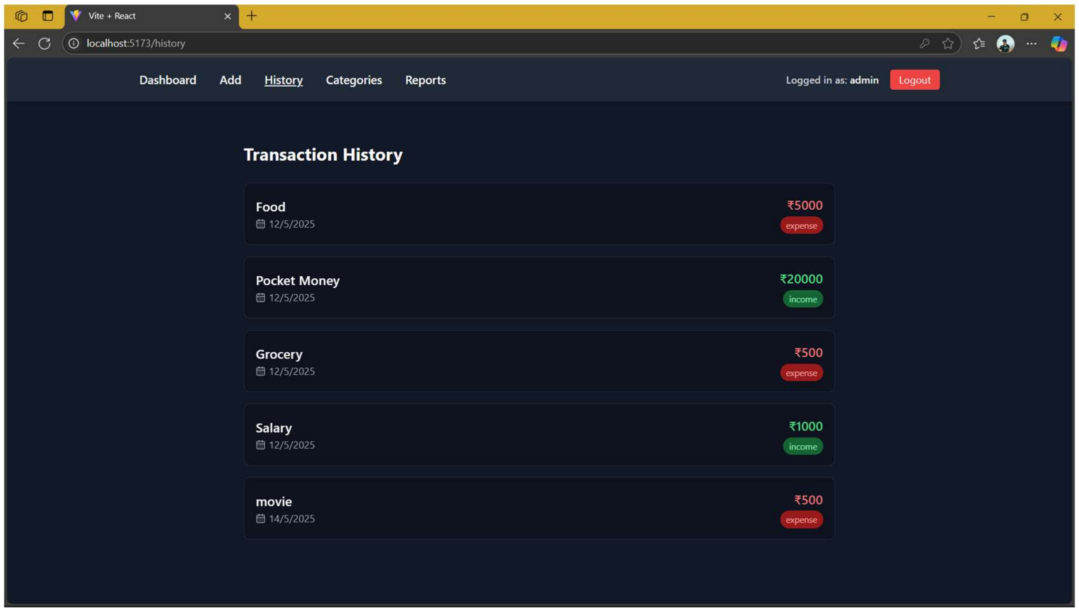
Description: movie

Amount: 500

Type: Expense

Category: Entertainment

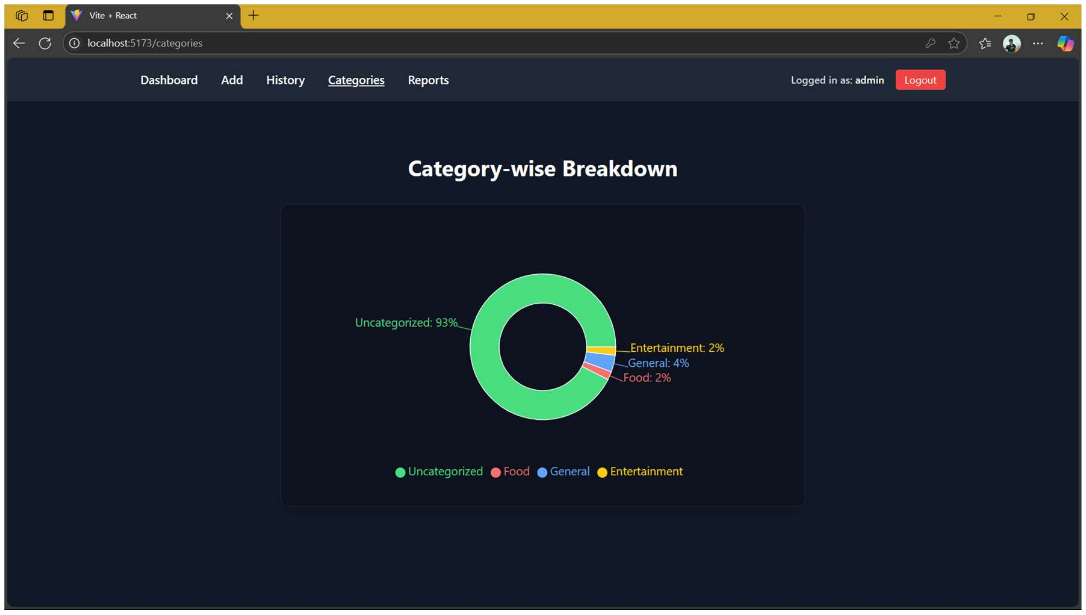
iv) Screenshot 4: Transaction History



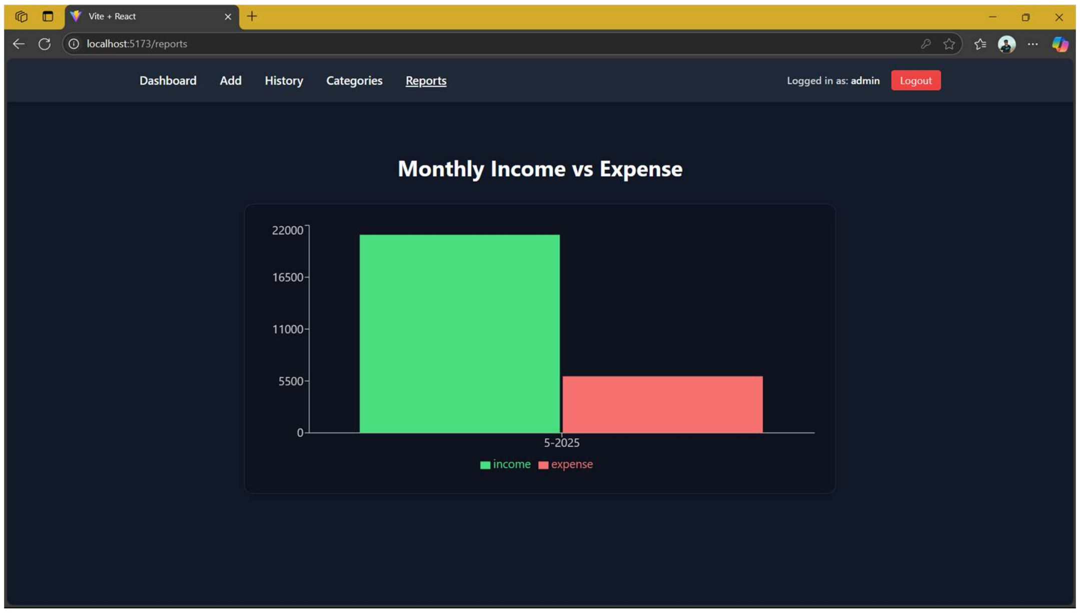
Transaction History

Food	12/5/2025	₹5000 expense
Pocket Money	12/5/2025	₹20000 income
Grocery	12/5/2025	₹500 expense
Salary	12/5/2025	₹1000 income
movie	14/5/2025	₹500 expense

v) Screenshot 5: Categories pie chart



vi) Screenshot 6: Reports bar chart



f) Conclusion:

The personal finance tracker web app successfully demonstrates the use of ReactJS for building modular, dynamic applications with routing and context-based state sharing. The system is scalable and forms a strong foundation for real-world financial tools.

Assignment No 2: Project Assignment based on Unit-III, Unit-IV and Unit-V

2) Project Title: Personal Finance Tracker - Web (React Native)

a) Problem Statement:

i) To develop a personal finance tracker mobile application using React Native that allows users to log in, manage income and expenses, view categorized reports, and interact through a modern and responsive UI.

b) Objective:

i) To build a fully functional mobile finance tracking app using React Native and navigation stacks that supports login, transaction entry, history, category breakdown with charts, and a report dashboard — all while demonstrating state management using React Context.

c) Theory:

i) React Native is an open-source framework developed by Meta for building mobile applications using JavaScript and React. It enables developers to write cross-platform apps for Android and iOS using a single codebase. This experiment leverages the following core concepts and libraries:

- React Native Components: Core components such as View, Text, TextInput, TouchableOpacity, and ScrollView are used to build native UI elements.

- Navigation: Navigation is handled using @react-navigation/native and @react-navigation/native-stack which provide stack-based routing and screen transitions.

- React Hooks: Hooks like useState, useEffect, and useContext allow functional components to manage state and share data across screens.

- React Context API: Used for global state management — it stores user login info and the transactions list accessible from any screen.

- Charts: react-native-chart-kit is used to draw visual elements like pie and bar charts for data representation.

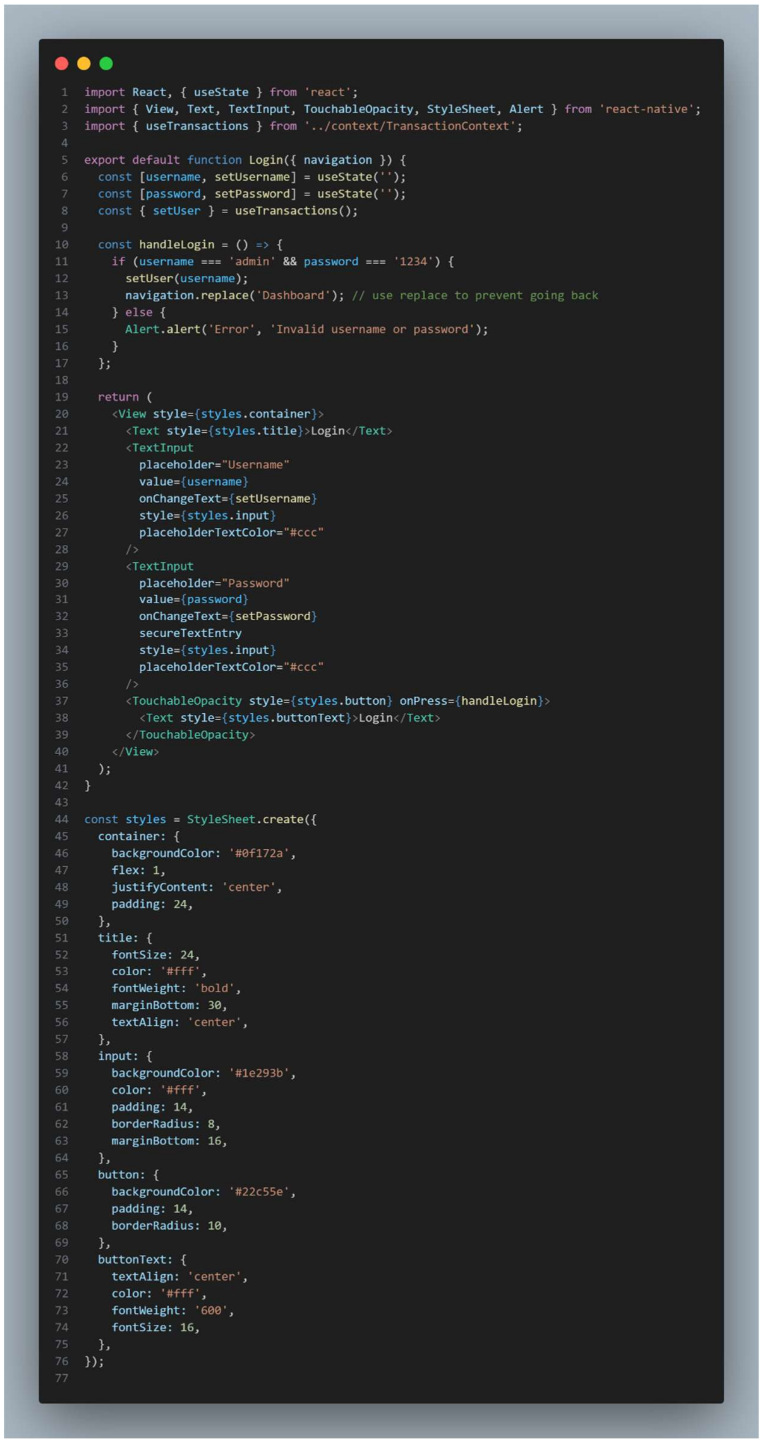
- SafeAreaView: Ensures UI doesn't overlap with the device's status bar and is properly padded.
- FlatList: Optimized component for rendering scrollable transaction history lists efficiently.

- Conditional Navigation & Reset: The app uses navigation.reset to handle redirection after login/logout to prevent returning to unauthorized screens.

This combination of UI components, navigation, data management, and visualization provides a comprehensive mobile development experience.

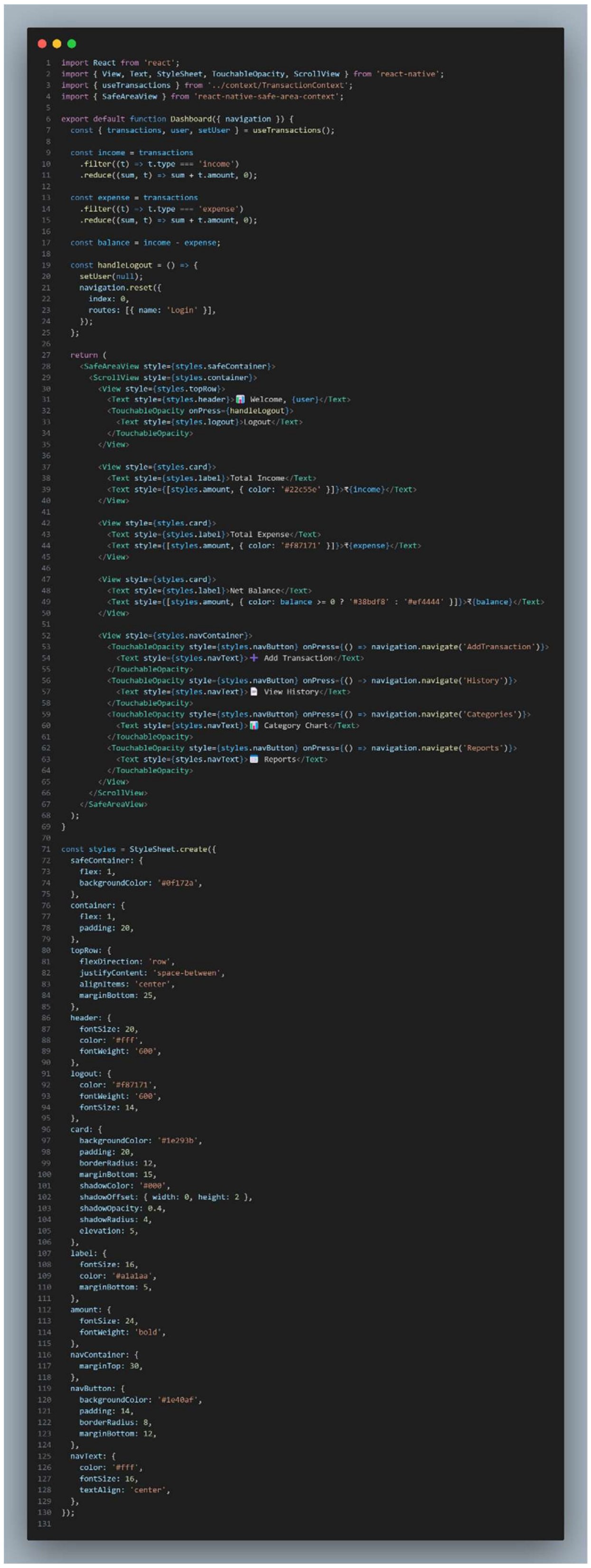
d) Code:

i) Login Page:



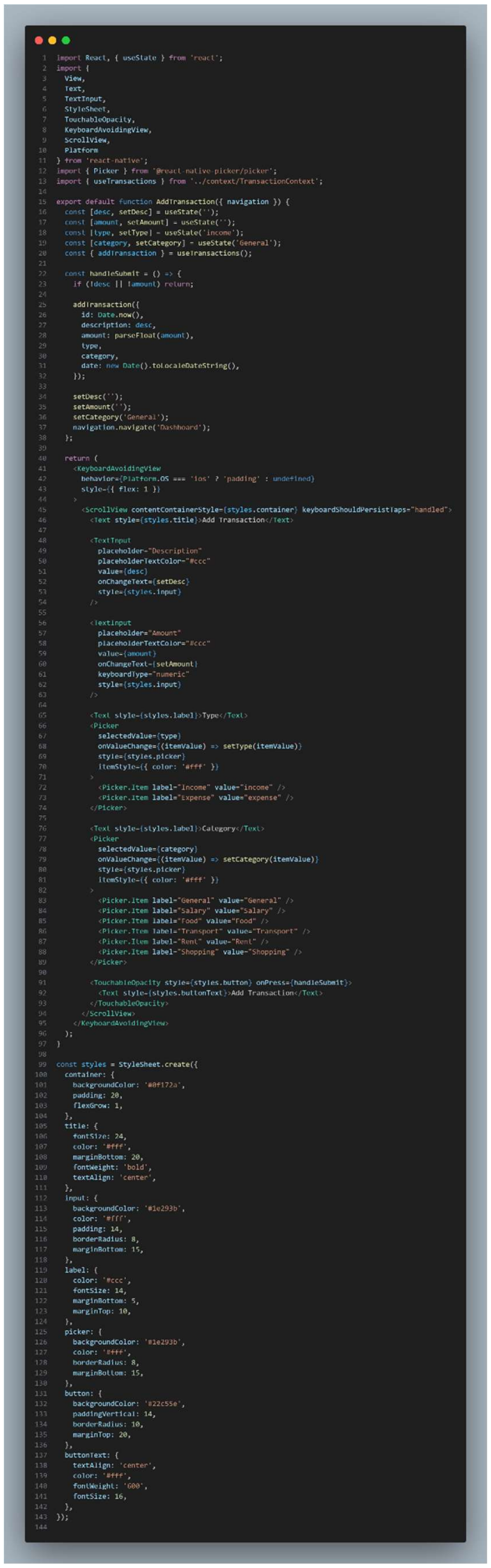
```
import React, { useState } from 'react';
import { View, Text, TextInput, TouchableOpacity, StyleSheet, Alert } from 'react-native';
import { useTransactions } from '../context/TransactionContext';

export default function Login({ navigation }) {
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');
  const { setUser } = useTransactions();

  const handleLogin = () => {
    if (username === 'admin' && password === '1234') {
      setUser(username);
      navigation.replace('Dashboard'); // use replace to prevent going back
    } else {
      Alert.alert('Error', 'Invalid username or password');
    }
  };

  return (
    <View style={styles.container}>
      <Text style={styles.title}>Login</Text>
      <TextInput
        placeholder="Username"
        value={username}
        onChangeText={setUsername}
        style={styles.input}
        placeholderTextColor="#ccc"
      />
      <TextInput
        placeholder="Password"
        value={password}
        onChangeText={setPassword}
        secureTextEntry
        style={styles.input}
        placeholderTextColor="#ccc"
      />
      <TouchableOpacity style={styles.button} onPress={handleLogin}>
        <Text style={styles.buttonText}>Login</Text>
      </TouchableOpacity>
    </View>
  );
}

const styles = StyleSheet.create({
  container: {
    backgroundColor: '#0f172a',
    flex: 1,
    justifyContent: 'center',
    padding: 24,
  },
  title: {
    fontSize: 24,
    color: '#fff',
    fontWeight: 'bold',
    marginBottom: 30,
    textAlign: 'center',
  },
  input: {
    backgroundColor: '#1e293b',
    color: '#fff',
    padding: 14,
    borderRadius: 8,
    marginBottom: 16,
  },
  button: {
    backgroundColor: '#22c55e',
    padding: 14,
    borderRadius: 10,
  },
  buttonText: {
    textAlign: 'center',
    color: '#fff',
    fontWeight: '600',
    fontSize: 16,
  },
});
```

ii) Dashboard:



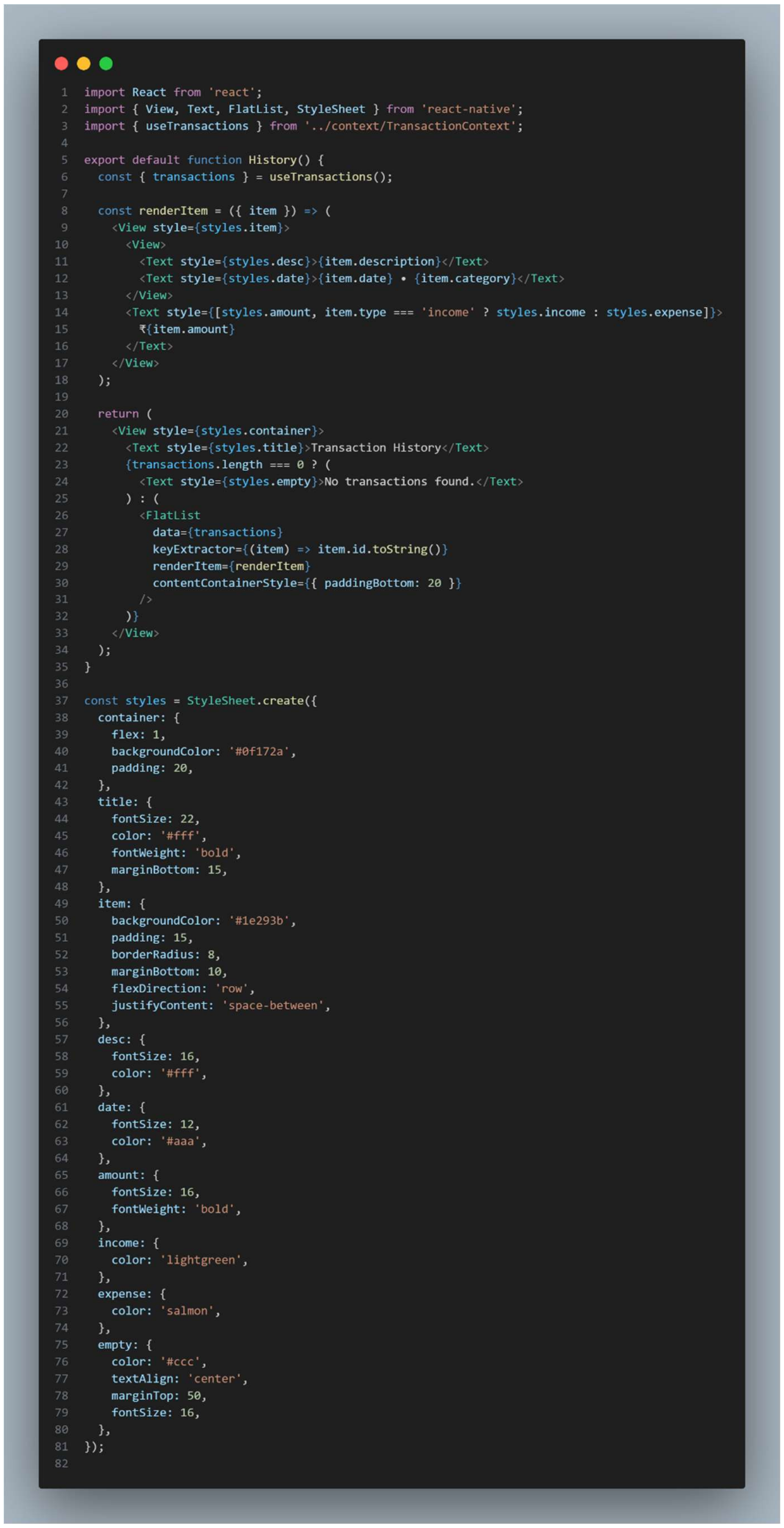
```
import React from 'react';
import { View, Text, StyleSheet, TouchableOpacity, ScrollView } from 'react-native';
import { useTransactions } from '../context/TransactionContext';
import { SafeAreaView } from 'react-native-safe-area-context';

export default function Dashboard({ navigation }) {
  const { transactions, user, setUser } = useTransactions();

  const income = transactions
    .filter((t) => t.type === 'income')
    .reduce((sum, t) => sum + t.amount, 0);

  const expense = transactions
    .filter((t) => t.type === 'expense')
    .reduce((sum, t) => sum + t.amount, 0);

  const balance = income - expense;

  const handleLogout = () => {
    setUser(null);
    navigation.reset({
      index: 0,
      routes: [{ name: 'Login' }],
    });
  };

  return (
    <SafeAreaView style={styles.safeContainer}>
      <ScrollView style={styles.container}>
        <View style={styles.topRow}>
          <Text style={styles.header}>📊 Welcome, {user}</Text>
          <TouchableOpacity onPress={handleLogout}>
            <Text style={styles.logout}>Logout</Text>
          </TouchableOpacity>
        </View>

        <View style={styles.card}>
          <Text style={styles.label}>Total Income</Text>
          <Text style={[styles.amount, { color: '#22c55e' }]}>₹{income}</Text>
        </View>

        <View style={styles.card}>
          <Text style={styles.label}>Total Expense</Text>
          <Text style={[styles.amount, { color: '#f87171' }]}>₹{expense}</Text>
        </View>

        <View style={styles.card}>
          <Text style={styles.label}>Net Balance</Text>
          <Text style={[styles.amount, { color: balance >= 0 ? '#38bdf8' : '#ef4444' }]}>₹{balance}</Text>
        </View>

        <View style={styles.navContainer}>
          <TouchableOpacity style={styles.navButton} onPress={() => navigation.navigate('AddTransaction')}>
            <Text style={styles.navText}>➕ Add Transaction</Text>
          </TouchableOpacity>
          <TouchableOpacity style={styles.navButton} onPress={() => navigation.navigate('History')}>
            <Text style={styles.navText}>📄 View History</Text>
          </TouchableOpacity>
          <TouchableOpacity style={styles.navButton} onPress={() => navigation.navigate('Categories')}>
            <Text style={styles.navText}>📊 Category Chart</Text>
          </TouchableOpacity>
          <TouchableOpacity style={styles.navButton} onPress={() => navigation.navigate('Reports')}>
            <Text style={styles.navText}>📅 Reports</Text>
          </TouchableOpacity>
        </View>
      </ScrollView>
    </SafeAreaView>
  );
}

const styles = StyleSheet.create({
  safeContainer: {
    flex: 1,
    backgroundColor: '#0f172a',
  },
  container: {
    flex: 1,
    padding: 20,
  },
  topRow: {
    flexDirection: 'row',
    justifyContent: 'space-between',
    alignItems: 'center',
    marginBottom: 25,
  },
  header: {
    fontSize: 20,
    color: '#fff',
    fontWeight: '600',
  },
  logout: {
    color: '#f87171',
    fontWeight: '600',
    fontSize: 14,
  },
  card: {
    backgroundColor: '#1e293b',
    padding: 20,
    borderRadius: 12,
    marginBottom: 15,
    shadowColor: '#000',
    shadowOffset: { width: 0, height: 2 },
    shadowOpacity: 0.4,
    shadowRadius: 4,
    elevation: 5,
  },
  label: {
    fontSize: 16,
    color: '#a1a1aa',
    marginBottom: 5,
  },
  amount: {
    fontSize: 24,
    fontWeight: 'bold',
  },
  navContainer: {
    marginTop: 30,
  },
  navButton: {
    backgroundColor: '#1e40af',
    padding: 14,
    borderRadius: 8,
    marginBottom: 12,
  },
  navText: {
    color: '#fff',
    fontSize: 16,
    textAlign: 'center',
  },
});
```

iii) Add Transaction:



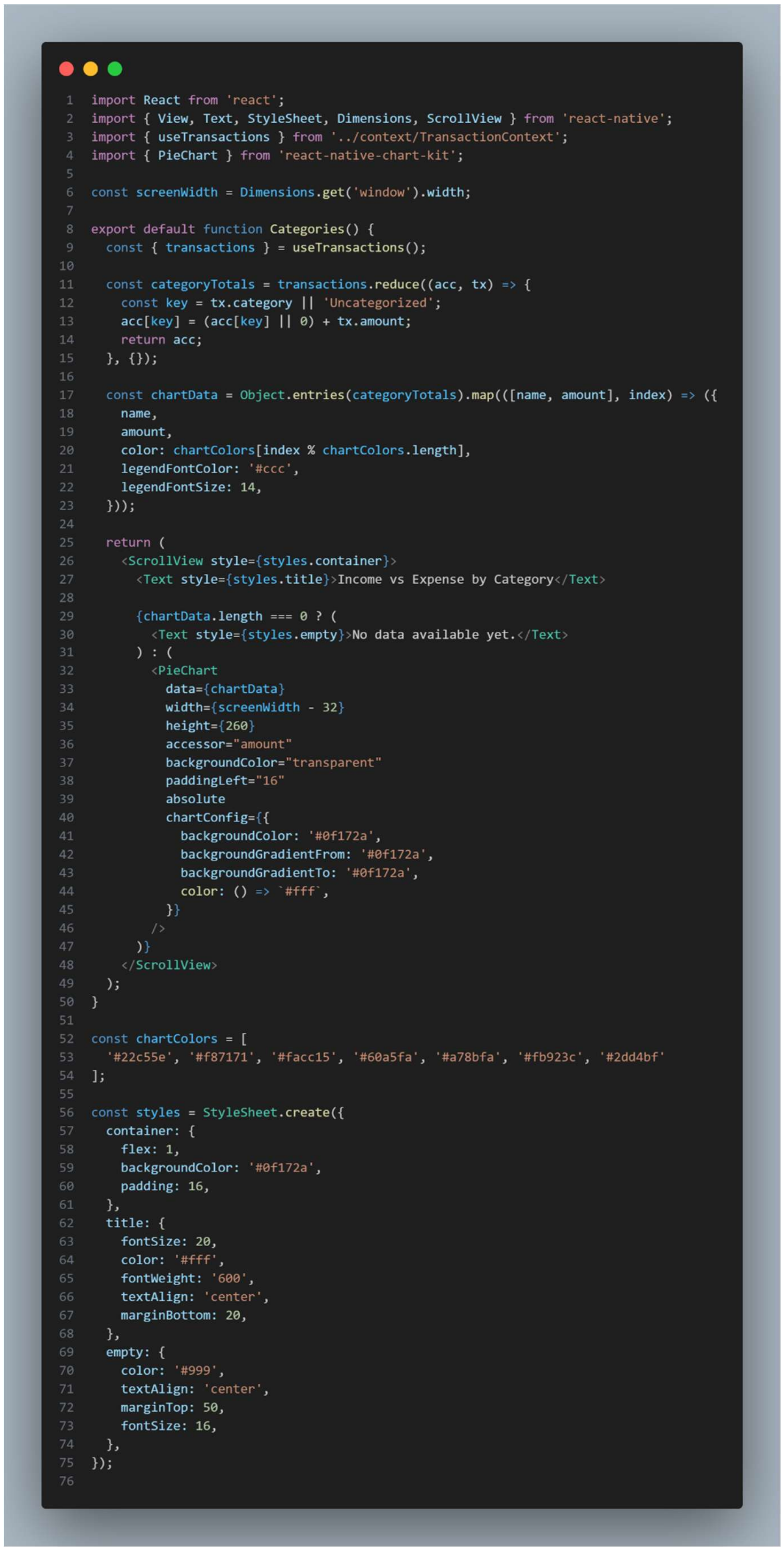
```jsx
import React, { useState } from 'react';
import {
  View,
  Text,
  TextInput,
  StyleSheet,
  TouchableOpacity,
  KeyboardAvoidingView,
  ScrollView,
  Platform
} from 'react-native';
import { Picker } from '@react-native-picker/picker';
import { useTransactions } from '../context/TransactionContext';

export default function AddTransaction({ navigation }) {
  const [desc, setDesc] = useState('');
  const [amount, setAmount] = useState('');
  const [type, setType] = useState('income');
  const [category, setCategory] = useState('General');
  const { addTransaction } = useTransactions();

  const handleSubmit = () => {
    if (!desc || !amount) return;

    addTransaction({
      id: Date.now(),
      description: desc,
      amount: parseFloat(amount),
      type,
      category,
      date: new Date().toLocaleDateString(),
    });

    setDesc('');
    setAmount('');
    setCategory('General');
    navigation.navigate('Dashboard');
  };

  return (
    <KeyboardAvoidingView
      behavior={Platform.OS === 'ios' ? 'padding' : undefined}
      style={{ flex: 1 }}
    >
      <ScrollView contentContainerStyle={styles.container} keyboardShouldPersistTaps="handled">
        <Text style={styles.title}>Add Transaction</Text>

        <TextInput
          placeholder="Description"
          placeholderTextColor="#ccc"
          value={desc}
          onChangeText={setDesc}
          style={styles.input}
        />

        <TextInput
          placeholder="Amount"
          placeholderTextColor="#ccc"
          value={amount}
          onChangeText={setAmount}
          keyboardType="numeric"
          style={styles.input}
        />

        <Text style={styles.label}>Type</Text>
        <Picker
          selectedValue={type}
          onValueChange={(itemValue) => setType(itemValue)}
          style={styles.picker}
          itemStyle={{ color: '#fff' }}
        >
          <Picker.Item label="Income" value="income" />
          <Picker.Item label="Expense" value="expense" />
        </Picker>

        <Text style={styles.label}>Category</Text>
        <Picker
          selectedValue={category}
          onValueChange={(itemValue) => setCategory(itemValue)}
          style={styles.picker}
          itemStyle={{ color: '#fff' }}
        >
          <Picker.Item label="General" value="General" />
          <Picker.Item label="Salary" value="Salary" />
          <Picker.Item label="Food" value="Food" />
          <Picker.Item label="Transport" value="Transport" />
          <Picker.Item label="Rent" value="Rent" />
          <Picker.Item label="Shopping" value="Shopping" />
        </Picker>

        <TouchableOpacity style={styles.button} onPress={handleSubmit}>
          <Text style={styles.buttonText}>Add Transaction</Text>
        </TouchableOpacity>
      </ScrollView>
    </KeyboardAvoidingView>
  );
}

const styles = StyleSheet.create({
  container: {
    backgroundColor: '#0f172a',
    padding: 20,
    flexGrow: 1,
  },
  title: {
    fontSize: 24,
    color: '#fff',
    marginBottom: 20,
    fontWeight: 'bold',
    textAlign: 'center',
  },
  input: {
    backgroundColor: '#1e293b',
    color: '#fff',
    padding: 14,
    borderRadius: 8,
    marginBottom: 15,
  },
  label: {
    color: '#ccc',
    fontSize: 14,
    marginBottom: 5,
    marginTop: 10,
  },
  picker: {
    backgroundColor: '#1e293b',
    color: '#fff',
    borderRadius: 8,
    marginBottom: 15,
  },
  button: {
    backgroundColor: '#22c55e',
    paddingVertical: 14,
    borderRadius: 10,
    marginTop: 20,
  },
  buttonText: {
    textAlign: 'center',
    color: '#fff',
    fontWeight: '600',
    fontSize: 16,
  },
});
```

iv) History Page:



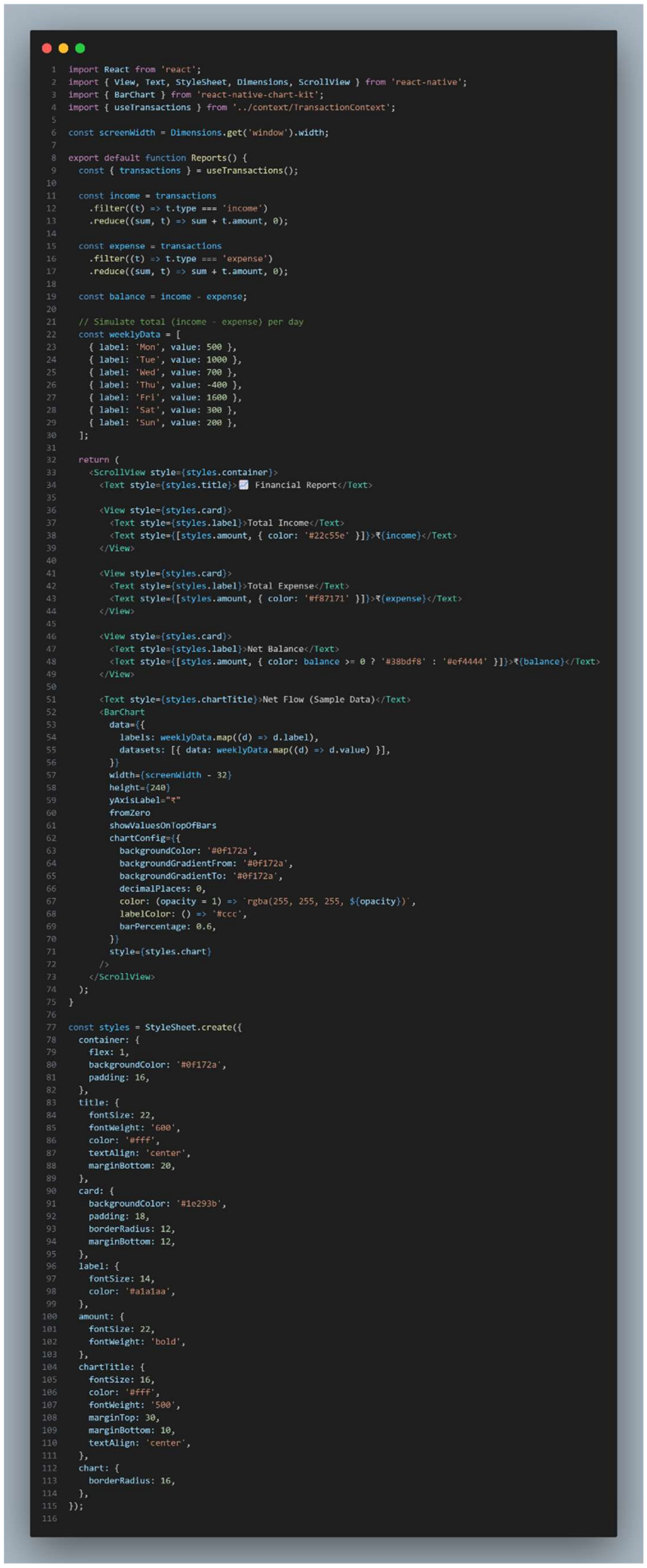
```
import React from 'react';
import { View, Text, FlatList, StyleSheet } from 'react-native';
import { useTransactions } from '../context/TransactionContext';

export default function History() {
  const { transactions } = useTransactions();

  const renderItem = ({ item }) => (
    <View style={styles.item}>
      <View>
        <Text style={styles.desc}>{item.description}</Text>
        <Text style={styles.date}>{item.date} • {item.category}</Text>
      </View>
      <Text style={[styles.amount, item.type === 'income' ? styles.income : styles.expense]}>
        ₹{item.amount}
      </Text>
    </View>
  );

  return (
    <View style={styles.container}>
      <Text style={styles.title}>Transaction History</Text>
      {transactions.length === 0 ? (
        <Text style={styles.empty}>No transactions found.</Text>
      ) : (
        <FlatList
          data={transactions}
          keyExtractor={(item) => item.id.toString()}
          renderItem={renderItem}
          contentContainerStyle={{ paddingBottom: 20 }}
        />
      )}
    </View>
  );
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: '#0f172a',
    padding: 20,
  },
  title: {
    fontSize: 22,
    color: '#fff',
    fontWeight: 'bold',
    marginBottom: 15,
  },
  item: {
    backgroundColor: '#1e293b',
    padding: 15,
    borderRadius: 8,
    marginBottom: 10,
    flexDirection: 'row',
    justifyContent: 'space-between',
  },
  desc: {
    fontSize: 16,
    color: '#fff',
  },
  date: {
    fontSize: 12,
    color: '#aaa',
  },
  amount: {
    fontSize: 16,
    fontWeight: 'bold',
  },
  income: {
    color: 'lightgreen',
  },
  expense: {
    color: 'salmon',
  },
  empty: {
    color: '#ccc',
    textAlign: 'center',
    marginTop: 50,
    fontSize: 16,
  },
});
```

v) Categories Page:



```
import React from 'react';
import { View, Text, StyleSheet, Dimensions, ScrollView } from 'react-native';
import { useTransactions } from '../context/TransactionContext';
import { PieChart } from 'react-native-chart-kit';

const screenWidth = Dimensions.get('window').width;

export default function Categories() {
  const { transactions } = useTransactions();

  const categoryTotals = transactions.reduce((acc, tx) => {
    const key = tx.category || 'Uncategorized';
    acc[key] = (acc[key] || 0) + tx.amount;
    return acc;
  }, {});

  const chartData = Object.entries(categoryTotals).map(([name, amount], index) => ({
    name,
    amount,
    color: chartColors[index % chartColors.length],
    legendFontColor: '#ccc',
    legendFontSize: 14,
  }));

  return (
    <ScrollView style={styles.container}>
      <Text style={styles.title}>Income vs Expense by Category</Text>

      {chartData.length === 0 ? (
        <Text style={styles.empty}>No data available yet.</Text>
      ) : (
        <PieChart
          data={chartData}
          width={screenWidth - 32}
          height={260}
          accessor="amount"
          backgroundColor="transparent"
          paddingLeft="16"
          absolute
          chartConfig={{
            backgroundColor: '#0f172a',
            backgroundGradientFrom: '#0f172a',
            backgroundGradientTo: '#0f172a',
            color: () => `#fff`,
          }}
        />
      )}
    </ScrollView>
  );
}

const chartColors = [
  '#22c55e', '#f87171', '#facc15', '#60a5fa', '#a78bfa', '#fb923c', '#2dd4bf'
];

const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: '#0f172a',
    padding: 16,
  },
  title: {
    fontSize: 20,
    color: '#fff',
    fontWeight: '600',
    textAlign: 'center',
    marginBottom: 20,
  },
  empty: {
    color: '#999',
    textAlign: 'center',
    marginTop: 50,
    fontSize: 16,
  },
});
```

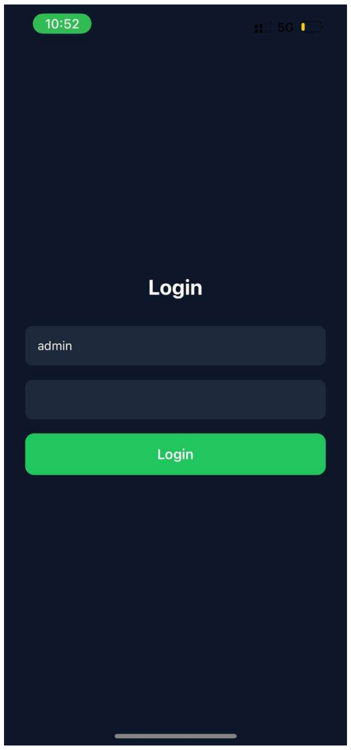
vi) Reports Page:



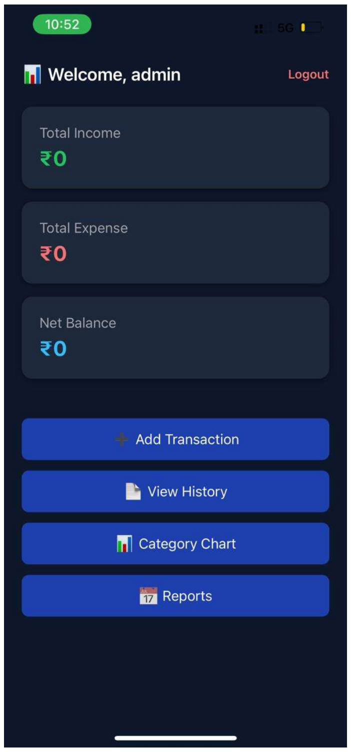
```javascript
import React from 'react';
import { View, Text, StyleSheet, Dimensions, ScrollView } from 'react-native';
import { BarChart } from 'react-native-chart-kit';
import { useTransactions } from '../context/TransactionContext';

const screenWidth = Dimensions.get('window').width;

export default function Reports() {
  const { transactions } = useTransactions();

  const income = transactions
    .filter((t) => t.type === 'income')
    .reduce((sum, t) => sum + t.amount, 0);

  const expense = transactions
    .filter((t) => t.type === 'expense')
    .reduce((sum, t) => sum + t.amount, 0);

  const balance = income - expense;

  // Simulate total (income - expense) per day
  const weeklyData = [
    { label: 'Mon', value: 500 },
    { label: 'Tue', value: 1000 },
    { label: 'Wed', value: 700 },
    { label: 'Thu', value: -400 },
    { label: 'Fri', value: 1600 },
    { label: 'Sat', value: 300 },
    { label: 'Sun', value: 200 },
  ];

  return (
    <ScrollView style={styles.container}>
      <Text style={styles.title}>📈 Financial Report</Text>

      <View style={styles.card}>
        <Text style={styles.label}>Total Income</Text>
        <Text style={[styles.amount, { color: '#22c55e' }]}>₹{income}</Text>
      </View>

      <View style={styles.card}>
        <Text style={styles.label}>Total Expense</Text>
        <Text style={[styles.amount, { color: '#f87171' }]}>₹{expense}</Text>
      </View>

      <View style={styles.card}>
        <Text style={styles.label}>Net Balance</Text>
        <Text style={[styles.amount, { color: balance >= 0 ? '#38bdf8' : '#ef4444' }]}>₹{balance}</Text>
      </View>

      <Text style={styles.chartTitle}>Net Flow (Sample Data)</Text>
      <BarChart
        data={{
          labels: weeklyData.map((d) => d.label),
          datasets: [{ data: weeklyData.map((d) => d.value) }],
        }}
        width={screenWidth - 32}
        height={240}
        yAxisLabel="₹"
        fromZero
        showValuesOnTopOfBars
        chartConfig={{
          backgroundColor: '#0f172a',
          backgroundGradientFrom: '#0f172a',
          backgroundGradientTo: '#0f172a',
          decimalPlaces: 0,
          color: (opacity = 1) => `rgba(255, 255, 255, ${opacity})`,
          labelColor: () => '#ccc',
          barPercentage: 0.6,
        }}
        style={styles.chart}
      />
    </ScrollView>
  );
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: '#0f172a',
    padding: 16,
  },
  title: {
    fontSize: 22,
    fontWeight: '600',
    color: '#fff',
    textAlign: 'center',
    marginBottom: 20,
  },
  card: {
    backgroundColor: '#1e293b',
    padding: 18,
    borderRadius: 12,
    marginBottom: 12,
  },
  label: {
    fontSize: 14,
    color: '#a1a1aa',
  },
  amount: {
    fontSize: 22,
    fontWeight: 'bold',
  },
  chartTitle: {
    fontSize: 16,
    color: '#fff',
    fontWeight: '500',
    marginTop: 30,
    marginBottom: 10,
    textAlign: 'center',
  },
  chart: {
    borderRadius: 16,
  },
});
```

e) Output:

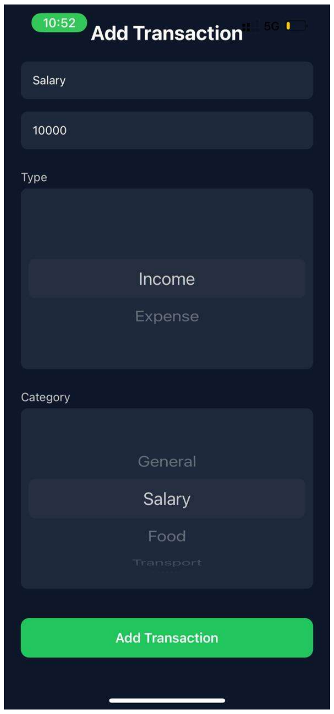
i) Screenshot 1: Login Page



ii) Screenshot 2: Dashboard with user greeting



iii) Screenshot 3: Add Transaction form

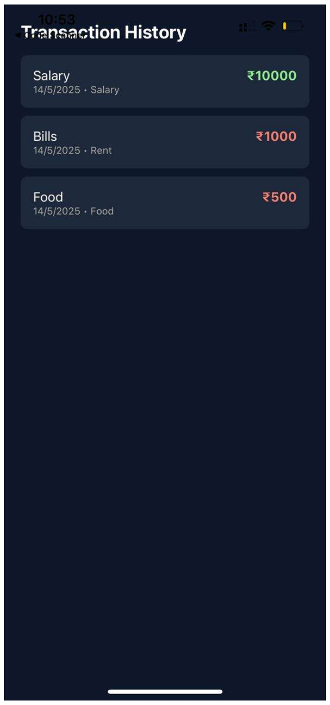


Salary

10000

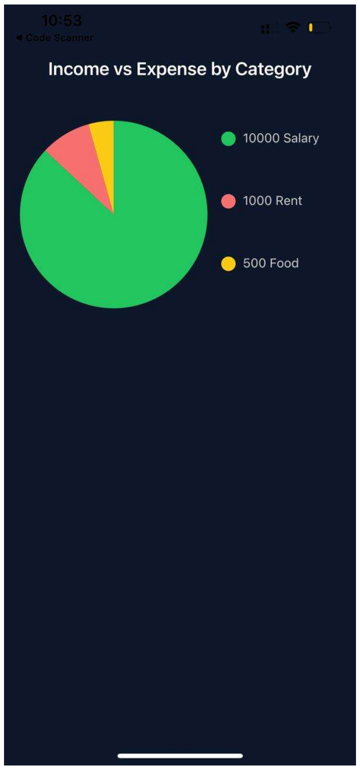
Salary

iv) Screenshot 4: Transaction History

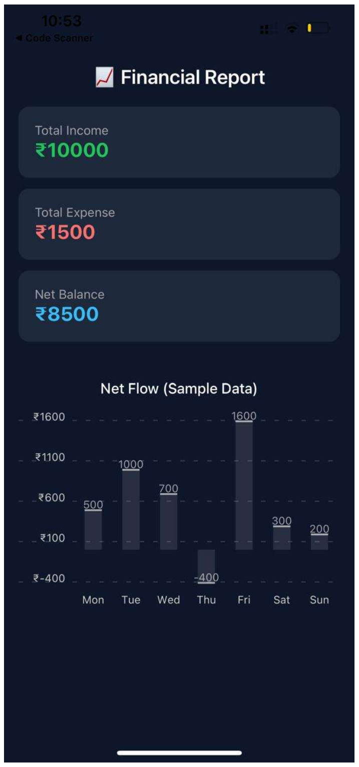


Salary 14/5/2025 • Salary	₹10000
Bills 14/5/2025 • Rent	₹1000
Food 14/5/2025 • Food	₹500

v) Screenshot 5: Categories pie chart



vi) Screenshot 6: Reports bar chart



Financial Report

Total Income
₹10000

Total Expense
₹1500

Net Balance
₹8500

f. **Conclusion:**

   a. The personal finance tracker mobile app built in React Native satisfies all the criteria for a functional mobile application, including login flow, UI interaction, navigation, and real-time data display using charts. It provides a clean and responsive experience suitable for mobile users.