



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:

```

1 import React, { useState } from 'react';
2 import { useNavigate } from 'react-router-dom';
3
4 export default function Login({ setUser }) {
5   const [username, setUsername] = useState('');
6   const [password, setPassword] = useState('');
7   const navigate = useNavigate();
8
9   const handleLogin = (e) => {
10     e.preventDefault();
11     if (username === 'admin' && password === '1234') {
12       setUser(username);
13       navigate('/dashboard');
14     } else {
15       alert('Invalid credentials');
16     }
17   };
18
19   return (
20     <div className="flex flex-col items-center justify-center min-h-screen bg-gray-900 text-white">
21       <h1 className="text-3xl font-bold mb-6">Login</h1>
22       <form onSubmit={handleLogin} className="bg-gray-800 p-8 rounded-lg shadow-md w-full max-w-sm">
23         <input
24           type="text"
25           placeholder="Username"
26           value={username}
27           onChange={e => setUsername(e.target.value)}
28           className="block w-full mb-4 p-3 rounded bg-gray-700 text-white"
29         />
30         <input
31           type="password"
32           placeholder="Password"
33           value={password}
34           onChange={e => setPassword(e.target.value)}
35           className="block w-full mb-4 p-3 rounded bg-gray-700 text-white"
36         />
37         <button
38           type="submit"
39           className="w-full py-2 bg-green-500 hover:bg-green-600 rounded"
40         >
41           Login
42         </button>
43       </form>
44     </div>
45   );
46 }
47

```

ii) Dashboard:

```

1 import { useTransactions } from '../context/transactionContext';
2 import { ArrowDownCircleIcon, ArrowUpCircleIcon, WalletIcon } from 'lucide-react'; // optional, use any icons
3
4 export default function Dashboard() {
5   const { transactions } = useTransactions();
6
7   const income = transactions
8     .filter((t) => t.type === 'income')
9     .reduce((sum, t) => sum + t.amount, 0);
10
11   const expense = transactions
12     .filter((t) => t.type === 'expense')
13     .reduce((sum, t) => sum + t.amount, 0);
14
15   const balance = income - expense;
16
17   return (
18     <div className="p-8 text-white">
19       <h2 className="text-3xl font-bold mb-8">Dashboard Overview</h2>
20
21       <div className="grid grid-cols-1 md:grid-cols-3 gap-6">
22         <Card
23           title="Total Income"
24           value={income}
25           color="text-green-400"
26           icon={<ArrowDownCircleIcon size={28} />}
27         />
28         <Card
29           title="Total Expense"
30           value={expense}
31           color="text-red-400"
32           icon={<ArrowUpCircleIcon size={28} />}
33         />
34         <Card
35           title="Net Balance"
36           value={balance}
37           color={balance >= 0 ? 'text-blue-400' : 'text-red-400'}
38           icon={<WalletIcon size={28} />}
39         />
40       </div>
41
42       { /* Optional: Progress bar comparing income vs expense */ }
43       <div className="mt-10">
44         <h3 className="mb-2 text-lg font-medium">Spending Ratio</h3>
45         <div className="bg-gray-700 rounded-full h-4 overflow-hidden">
46           <div
47             className="bg-green-400 h-4"
48             style={{ width: `${(income / (income + expense)) * 100 || 0}%` }}
49           ></div>
50         </div>
51         <p className="text-sm text-gray-400 mt-1">
52           {income > 0 ? `${Math.round((income / (income + expense)) * 100)}% income` : 'No income data'}
53         </p>
54       </div>
55     </div>
56   );
57 }
58
59 function Card({ title, value, color, icon }) {
60   return (
61     <div className="bg-black/20 backdrop-blur-md p-6 rounded-xl shadow-lg border border-white/10">
62       <div className="flex items-center gap-4 mb-3">
63         <div className="p-2 rounded-full bg-white/10 ${color}">{icon}</div>
64         <p className="text-lg font-sembold">{title}</p>
65       </div>
66       <h3 className="text-2xl font-bold ${color}">₹{value}</h3>
67     </div>
68   );
69 }
70

```

iii) Add Transaction:

```

1 import { useState } from 'react';
2 import { useTransactions } from '../context/TransactionContext';
3
4 export default function AddTransaction() {
5   const [desc, setDesc] = useState('');
6   const [amount, setAmount] = useState('');
7   const [type, setType] = useState('income');
8   const [category, setCategory] = useState('General');
9
10  const { addTransaction } = useTransactions();
11
12  const handleSubmit = (e) => {
13    e.preventDefault();
14    if (!desc || !amount) return;
15
16    addTransaction({
17      id: Date.now(),
18      description: desc,
19      amount: parseFloat(amount),
20      type,
21      category,
22      date: new Date().toLocaleDateString(),
23    });
24
25    setDesc('');
26    setAmount('');
27    setCategory('General');
28  };
29
30  return (
31    <div className="flex justify-center items-center min-h-[80vh] text-white">
32      <form
33        onSubmit={handleSubmit}
34        className="bg-black/20 backdrop-blur-md p-8 rounded-xl shadow-xl w-full max-w-md border border-white/10">
35        <h2 className="text-2xl font-bold mb-6 text-center">Add Transaction</h2>
36        <label className="block mb-2 text-sm font-medium">Description</label>
37        <input
38          type="text"
39          placeholder="e.g. Salary, Grocery"
40          value={desc}
41          onChange={(e) => setDesc(e.target.value)}
42          required
43          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-600">
44        </label>
45        <label className="block mb-2 text-sm font-medium">Amount</label>
46        <input
47          type="number"
48          placeholder="e.g. 1000"
49          value={amount}
50          onChange={(e) => setAmount(e.target.value)}
51          required
52          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-600">
53        </label>
54        <label className="block mb-2 text-sm font-medium">Type</label>
55        <select
56          value={type}
57          onChange={(e) => setType(e.target.value)}
58          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-600">
59          <option value="Income">Income</option>
60          <option value="Expense">Expense</option>
61        </select>
62        <label className="block mb-2 text-sm font-medium">Category</label>
63        <select
64          value={category}
65          onChange={(e) => setCategory(e.target.value)}
66          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-600">
67          <option value="General">General</option>
68          <option value="Salary">Salary</option>
69          <option value="Food">Food</option>
70          <option value="Rent">Rent</option>
71          <option value="Transport">Transport</option>
72          <option value="Entertainment">Entertainment</option>
73          <option value="Healthcare">Healthcare</option>
74          <option value="Shopping">Shopping</option>
75        </select>
76        <button
77          type="submit"
78          className="w-full py-3 bg-green-500 hover:bg-green-600 rounded-lg text-white font-medium transition">
79          Add Transaction
80        </button>
81      </form>
82    </div>
83  );
84 }

```

iv) History Page

```

1 import { useTransactions } from '../context/TransactionContext';
2 import { CalendarDaysIcon } from 'lucide-react';
3
4 export default function History() {
5   const { transactions } = useTransactions();
6
7   return (
8     <div className="p-6 max-w-4xl mx-auto text-white">
9       <h2 className="text-2xl font-bold mb-6">Transaction History</h2>
10
11       {transactions.length === 0 ? (
12         <p className="text-gray-400">No transactions yet.</p>
13       ) : (
14         <div className="space-y-4">
15           {transactions.map((tx) => (
16             <div
17               key={tx.id}
18               className="flex items-center justify-between bg-black/20 border border-white/10 backdrop-blur-md p-4 rounded-lg shadow-sm"
19             >
20               <div>
21                 <h3 className="text-lg font-medium">{tx.description}</h3>
22                 <p className="text-sm text-gray-400 flex items-center gap-1">
23                   <CalendarDaysIcon size={14} className="inline-block" /> {tx.date}
24                 </p>
25               </div>
26               <div className="text-right">
27                 <p
28                   className={`text-lg font-semibold ${
29                     tx.type === 'income' ? 'text-green-400' : 'text-red-400'
30                   }`}
31                 >
32                   ₹{tx.amount}
33                 </p>
34                 <span
35                   className={`text-xs px-2 py-1 rounded-full ${
36                     tx.type === 'income'
37                       ? 'bg-green-800 text-green-300'
38                       : 'bg-red-800 text-red-300'
39                   }`}
40                 >
41                   {tx.type}
42                 </span>
43               </div>
44             </div>
45           ))}
46         </div>
47       )}
48     </div>
49   );
50 }
51

```

v) Categories Page:

```

1 import { PieChart, Pie, Cell, Tooltip, Legend, ResponsiveContainer } from 'recharts';
2 import { useTransactions } from '../context/TransactionContext';
3
4 export default function Categories() {
5   const { transactions } = useTransactions();
6
7   const categoryTotals = transactions.reduce((acc, t) => {
8     const key = t.category || 'Uncategorized';
9     acc[key] = (acc[key] || 0) + t.amount;
10    return acc;
11  }, {});
12
13  const data = Object.entries(categoryTotals).map(([name, value]) => ({
14    name,
15    value,
16  }));
17
18  const COLORS = [
19    '#4ade80', '#f87171', '#60a5fa', '#facc15', '#a78bfa', '#fb923c', '#2dd4bf'
20  ];
21
22  return (
23    <div className="text-white px-6 py-10 max-w-3xl mx-auto">
24      <h2 className="text-3xl font-bold mb-8 text-center">Category-wise Breakdown</h2>
25
26      <div className="bg-black/20 backdrop-blur-md p-8 rounded-xl shadow-lg border border-white/10">
27        <ResponsiveContainer width="100%" height={350}>
28          <PieChart>
29            <Pie
30              data={data}
31              dataKey="value"
32              nameKey="name"
33              cx="50%"
34              cy="50%"
35              innerRadius={60}
36              outerRadius={100}
37              fill="#8884d8"
38              label={({ name, percent }) => `${name}: ${(percent * 100).toFixed(0)}%`
39            >
40              {data.map((_, index) => (
41                <Cell key={`cell-${index}`} fill={COLORS[index % COLORS.length]} />
42              ))}
43            </Pie>
44            <Tooltip />
45            <Legend />
46          </PieChart>
47        </ResponsiveContainer>
48      </div>
49    </div>
50  );
51 }
52

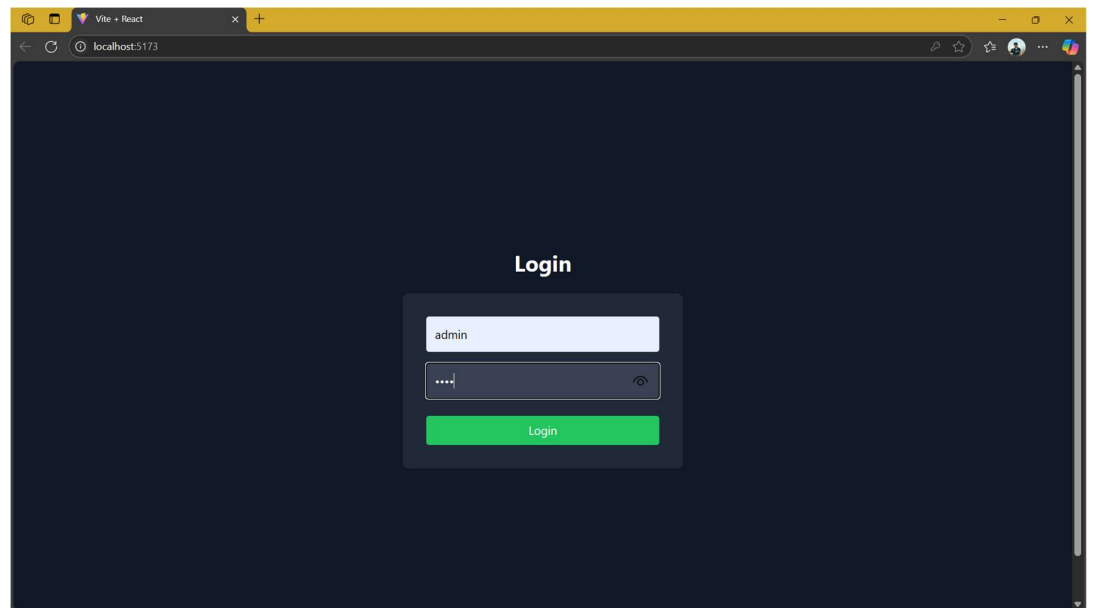
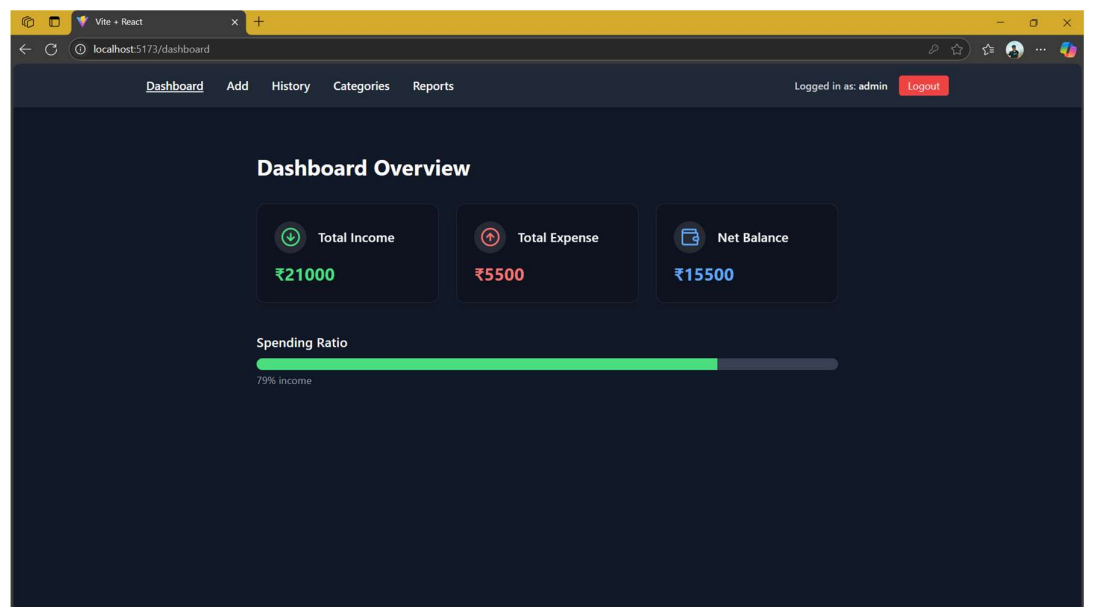
```


vi) Reports Page:

```

1 import { useTransactions } from '../context/TransactionContext';
2 import {
3   BarChart, Bar, XAxis, YAxis, Tooltip, ResponsiveContainer, Legend,
4 } from 'recharts';
5
6 export default function Reports() {
7   const { transactions } = useTransactions();
8
9   // Group by month-year and calculate income/expense totals
10  const monthlyData = transactions.reduce((acc, tx) => {
11    const [day, month, year] = tx.date.split('/');
12    const key = `${month}-${year}`;
13
14    if (!acc[key]) {
15      acc[key] = { month: key, income: 0, expense: 0 };
16    }
17
18    if (tx.type === 'income') {
19      acc[key].income += tx.amount;
20    } else {
21      acc[key].expense += tx.amount;
22    }
23
24    return acc;
25  }, {});
26
27  const chartData = Object.values(monthlyData).sort((a, b) => {
28    const [ma, ya] = a.month.split('-').map(Number);
29    const [mb, yb] = b.month.split('-').map(Number);
30    return ya === yb ? ma - mb : ya - yb;
31  });
32
33  return (
34    <div className="text-white px-6 py-10 max-w-4xl mx-auto">
35      <h2 className="text-3xl font-bold mb-8 text-center">Monthly Income vs Expense</h2>
36
37      {chartData.length === 0 ? (
38        <p className="text-gray-400 text-center">No data available.</p>
39      ) : (
40        <div className="bg-black/20 backdrop-blur-md p-6 rounded-xl border border-white/10 shadow-lg">
41          <ResponsiveContainer width="100%" height={350}>
42            <BarChart data={chartData}>
43              <XAxis dataKeys="month" stroke="#ccc" />
44              <YAxis stroke="#ccc" />
45              <Tooltip />
46              <Legend />
47              <Bar dataKey="income" fill="#4ade80" />
48              <Bar dataKey="expense" fill="#f87171" />
49            </BarChart>
50          </ResponsiveContainer>
51        </div>
52      )}
53    </div>
54  );
55 }
56

```

e) Output Screenshots:**i) Screenshot 1: Login Page (ReactJS)****ii) Screenshot 2: Dashboard**

iii) Screenshot 3: Add Transaction form

The screenshot shows a web application interface for adding a transaction. The browser address bar shows 'localhost:5173/add'. The navigation bar includes 'Dashboard', 'Add', 'History', 'Categories', and 'Reports'. The user is logged in as 'admin' with a 'Logout' button. The main content area features a form titled 'Add Transaction' with the following fields:

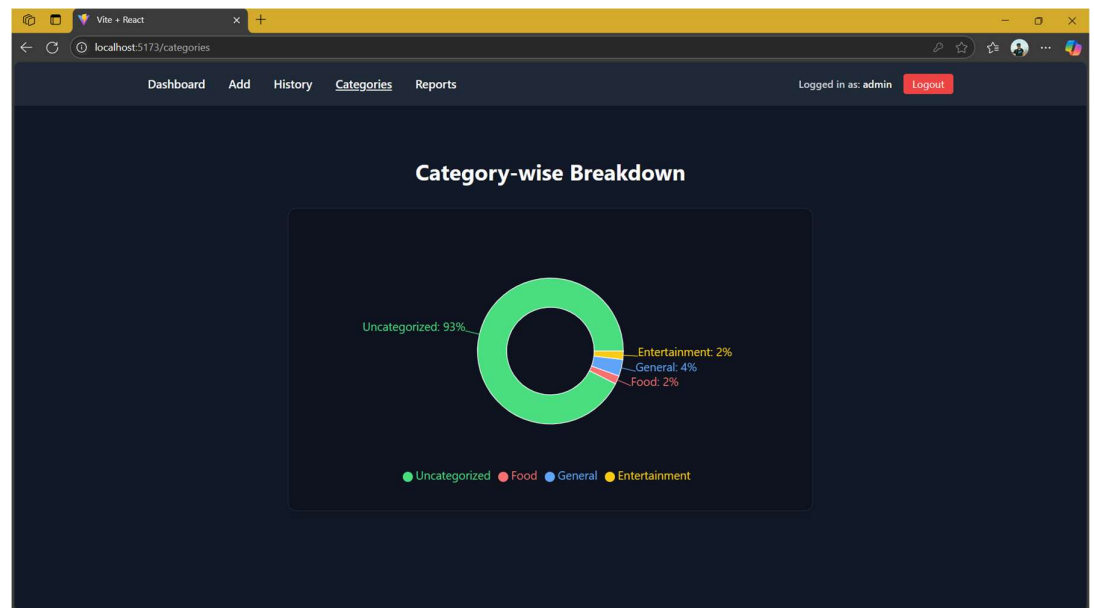
- Description: movie
- Amount: 500
- Type: Expense (dropdown menu)
- Category: Entertainment (dropdown menu)
- Add Transaction (green button)

iv) Screenshot 4: Transaction History

The screenshot shows the 'Transaction History' page. The browser address bar shows 'localhost:5173/history'. The navigation bar includes 'Dashboard', 'Add', 'History', 'Categories', and 'Reports'. The user is logged in as 'admin' with a 'Logout' button. The main content area displays a list of transactions:

Description	Date	Amount	Type
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:

```

1 import React, { useState } from 'react';
2 import { View, Text, TextInput, TouchableOpacity, StyleSheet, Alert } from 'react-native';
3 import { useTransactions } from '../context/TransactionContext';
4
5 export default function Login({ navigation }) {
6   const [username, setUsername] = useState('');
7   const [password, setPassword] = useState('');
8   const { setUser } = useTransactions();
9
10  const handleLogin = () => {
11    if (username === 'admin' && password === '1234') {
12      setUser(username);
13      navigation.replace('Dashboard'); // use replace to prevent going back
14    } else {
15      Alert.alert('Error', 'Invalid username or password');
16    }
17  };
18
19  return (
20    <View style={styles.container}>
21      <Text style={styles.title}>Login</Text>
22      <TextInput
23        placeholder="Username"
24        value={username}
25        onChangeText={setUsername}
26        style={styles.input}
27        placeholderTextColor="#ccc"
28      />
29      <TextInput
30        placeholder="Password"
31        value={password}
32        onChangeText={setPassword}
33        secureTextEntry
34        style={styles.input}
35        placeholderTextColor="#ccc"
36      />
37      <TouchableOpacity style={styles.button} onPress={handleLogin}>
38        <Text style={styles.buttonText}>Login</Text>
39      </TouchableOpacity>
40    </View>
41  );
42 }
43
44 const styles = StyleSheet.create({
45   container: {
46     backgroundColor: '#0f172a',
47     flex: 1,
48     justifyContent: 'center',
49     padding: 24,
50   },
51   title: {
52     fontSize: 24,
53     color: '#fff',
54     fontWeight: 'bold',
55     marginBottom: 30,
56     textAlign: 'center',
57   },
58   input: {
59     backgroundColor: '#1e293b',
60     color: '#fff',
61     padding: 14,
62     borderRadius: 8,
63     marginBottom: 16,
64   },
65   button: {
66     backgroundColor: '#22c55e',
67     padding: 14,
68     borderRadius: 10,
69   },
70   buttonText: {
71     textAlign: 'center',
72     color: '#fff',
73     fontWeight: '600',
74     fontSize: 16,
75   },
76 });
77

```

ii) Dashboard:

```

1 import React from 'react';
2 import { View, Text, StyleSheet, TouchableOpacity, ScrollView } from 'react-native';
3 import { useTransactions } from '../context/TransactionContext';
4 import { SafeAreaView } from 'react-native-safe-area-context';
5
6 export default function Dashboard({ navigation }) {
7   const { transactions, user, setUser } = useTransactions();
8
9   const income = transactions
10     .filter((t) => t.type === 'income')
11     .reduce((sum, t) => sum + t.amount, 0);
12
13   const expense = transactions
14     .filter((t) => t.type === 'expense')
15     .reduce((sum, t) => sum + t.amount, 0);
16
17   const balance = income - expense;
18
19   const handleLogout = () => {
20     setUser(null);
21     navigation.reset({
22       index: 0,
23       routes: [{ name: 'Login' }],
24     });
25   };
26
27   return (
28     <SafeAreaView style={styles.safeContainer}>
29       <ScrollView style={styles.container}>
30         <View style={styles.topRow}>
31           <Text style={styles.header}> Welcome, {user} </Text>
32           <TouchableOpacity onPress={handleLogout}>
33             <Text style={styles.logout}>Logout</Text>
34           </TouchableOpacity>
35         </View>
36
37         <View style={styles.card}>
38           <Text style={styles.label}>Total Income</Text>
39           <Text style={styles.amount, { color: '#22c55e'}}>{income}</Text>
40         </View>
41
42         <View style={styles.card}>
43           <Text style={styles.label}>Total Expense</Text>
44           <Text style={styles.amount, { color: '#f07171'}}>{expense}</Text>
45         </View>
46
47         <View style={styles.card}>
48           <Text style={styles.label}>Net Balance</Text>
49           <Text style={styles.amount, { color: balance >= 0 ? '#38bdf8' : '#ef4444'}}>{balance}</Text>
50         </View>
51
52         <View style={styles.navContainer}>
53           <TouchableOpacity style={styles.navButton} onPress={() => navigation.navigate('AddTransaction')}>
54             <Text style={styles.navText}>+ Add Transaction</Text>
55           </TouchableOpacity>
56           <TouchableOpacity style={styles.navButton} onPress={() => navigation.navigate('History')}>
57             <Text style={styles.navText}>View History</Text>
58           </TouchableOpacity>
59           <TouchableOpacity style={styles.navButton} onPress={() => navigation.navigate('Categories')}>
60             <Text style={styles.navText}>Category Chart</Text>
61           </TouchableOpacity>
62           <TouchableOpacity style={styles.navButton} onPress={() => navigation.navigate('Reports')}>
63             <Text style={styles.navText}>Reports</Text>
64           </TouchableOpacity>
65         </View>
66       </ScrollView>
67     </SafeAreaView>
68   );
69 }
70
71 const styles = StyleSheet.create({
72   safeContainer: {
73     flex: 1,
74     backgroundColor: '#f0f1f2',
75   },
76   container: {
77     flex: 1,
78     padding: 20,
79   },
80   topRow: {
81     flexDirection: 'row',
82     justifyContent: 'space-between',
83     alignItems: 'center',
84     marginBottom: 25,
85   },
86   header: {
87     fontSize: 20,
88     color: 'fff',
89     fontWeight: 'bold',
90   },
91   logout: {
92     color: '#f07171',
93     fontWeight: 'bold',
94     fontSize: 14,
95   },
96   card: {
97     backgroundColor: 'white',
98     padding: 20,
99     borderRadius: 12,
100    marginBottom: 15,
101    shadowColor: 'black',
102    shadowOffset: { width: 0, height: 2 },
103    shadowOpacity: 0.4,
104    shadowRadius: 4,
105    elevation: 5,
106  },
107  label: {
108    fontSize: 16,
109    color: '#333333',
110    marginBottom: 5,
111  },
112  amount: {
113    fontSize: 24,
114    fontWeight: 'bold',
115  },
116  navContainer: {
117    marginTop: 30,
118  },
119  navButton: {
120    backgroundColor: 'white',
121    padding: 14,
122    borderRadius: 8,
123    marginBottom: 12,
124  },
125  navText: {
126    color: 'fff',
127    fontSize: 16,
128    textAlign: 'center',
129  },
130 });
131

```

iii) Add Transaction:


```

1 import React, { useState } from 'react';
2 import {
3   View,
4   Text,
5   TextInput,
6   StyleSheet,
7   TouchableOpacity,
8   KeyboardAvoidingView,
9   ScrollView,
10  Platform
11 } from 'react-native';
12 import { Picker } from '@react-native/picker';
13 import { useTransactions } from '../context/TransactionContext';
14
15 export default function AddTransaction({ navigation }) {
16   const [desc, setDesc] = useState('');
17   const [amount, setAmount] = useState('');
18   const [type, setType] = useState('income');
19   const [category, setCategory] = useState('General');
20   const [addTransaction] = useTransactions();
21
22   const handleSubmit = () => {
23     if (!desc || !amount) return;
24
25     addTransaction({
26       id: Date.now(),
27       description: desc,
28       amount: parseFloat(amount),
29       type,
30       category,
31       date: new Date().toLocaleDateString(),
32     });
33
34     setDesc('');
35     setAmount('');
36     setCategory('General');
37     navigation.navigate('Dashboard');
38   };
39
40   return (
41     <KeyboardAvoidingView
42       behavior={Platform.OS === 'ios' ? 'padding' : 'underflow'}
43       style={{ flex: 1 }}
44     >
45       <ScrollView contentContainerStyle={styles.container} keyboardShouldPersistTaps="handled">
46         <Text style={styles.title}>Add Transaction</Text>
47
48         <TextInput
49           placeholder="Description"
50           placeholderTextColor="black"
51           value={desc}
52           onChangeText={setDesc}
53           style={styles.input}
54         />
55
56         <TextInput
57           placeholder="Amount"
58           placeholderTextColor="black"
59           value={amount}
60           onChangeText={setAmount}
61           keyboardType="numeric"
62           style={styles.input}
63         />
64
65         <Text style={styles.label}>Type</Text>
66         <Picker
67           selectedValue={type}
68           onValueChange={(itemValue) => setType(itemValue)}
69           style={styles.picker}
70           itemStyle={{ color: 'black' }}
71         >
72           <Picker.Item label="Income" value="income" />
73           <Picker.Item label="Expense" value="expense" />
74         </Picker>
75
76         <Text style={styles.label}>Category</Text>
77         <Picker
78           selectedValue={category}
79           onValueChange={(itemValue) => setCategory(itemValue)}
80           style={styles.picker}
81           itemStyle={{ color: 'black' }}
82         >
83           <Picker.Item label="General" value="General" />
84           <Picker.Item label="Salary" value="Salary" />
85           <Picker.Item label="Food" value="Food" />
86           <Picker.Item label="Transport" value="Transport" />
87           <Picker.Item label="Rent" value="Rent" />
88           <Picker.Item label="Shopping" value="Shopping" />
89         </Picker>
90
91         <TouchableOpacity style={styles.button} onPress={handleSubmit}>
92           <Text style={styles.buttonText}>Add Transaction</Text>
93         </TouchableOpacity>
94       </ScrollView>
95     </KeyboardAvoidingView>
96   );
97 }
98
99 const styles = StyleSheet.create({
100   container: {
101     backgroundColor: 'white',
102     padding: 20,
103     flex: 1,
104   },
105   title: {
106     fontSize: 24,
107     color: 'black',
108     marginBottom: 20,
109     fontWeight: 'bold',
110     textAlign: 'center',
111   },
112   input: {
113     backgroundColor: 'white',
114     color: 'black',
115     padding: 10,
116     borderRadius: 10,
117     margin: 10,
118   },
119   label: {
120     color: 'black',
121     fontSize: 14,
122     margin: 5,
123   },
124   picker: {
125     backgroundColor: 'white',
126     color: 'black',
127     borderRadius: 10,
128     margin: 10,
129   },
130   button: {
131     backgroundColor: 'black',
132     padding: 10,
133     borderRadius: 10,
134     margin: 10,
135   },
136   buttonText: {
137     color: 'white',
138     align: 'center',
139     font: 'bold',
140     font: '16px',
141   },
142 });
143
144

```

iv) History Page:

```

1 import React from 'react';
2 import { View, Text, FlatList, StyleSheet } from 'react-native';
3 import { useTransactions } from '../context/TransactionContext';
4
5 export default function History() {
6   const { transactions } = useTransactions();
7
8   const renderItem = ({ item }) => (
9     <View style={styles.item}>
10       <View>
11         <Text style={styles.desc}>{item.description}</Text>
12         <Text style={styles.date}>{item.date} • {item.category}</Text>
13       </View>
14       <Text style={styles.amount, item.type === 'income' ? styles.income : styles.expense}>
15         ₹{item.amount}
16       </Text>
17     </View>
18   );
19
20   return (
21     <View style={styles.container}>
22       <Text style={styles.title}>Transaction History</Text>
23       {transactions.length === 0 ? (
24         <Text style={styles.empty}>No transactions found.</Text>
25       ) : (
26         <FlatList
27           data={transactions}
28           keyExtractor={(item) => item.id.toString()}
29           renderItem={renderItem}
30           contentContainerStyle={{ paddingBottom: 20 }}
31         />
32       )}
33     </View>
34   );
35 }
36
37 const styles = StyleSheet.create({
38   container: {
39     flex: 1,
40     backgroundColor: '#0f172a',
41     padding: 20,
42   },
43   title: {
44     fontSize: 22,
45     color: 'fff',
46     fontWeight: 'bold',
47     marginBottom: 15,
48   },
49   item: {
50     backgroundColor: '#1e293b',
51     padding: 15,
52     borderRadius: 8,
53     marginBottom: 10,
54     flexDirection: 'row',
55     justifyContent: 'space-between',
56   },
57   desc: {
58     fontSize: 16,
59     color: 'fff',
60   },
61   date: {
62     fontSize: 12,
63     color: 'aaa',
64   },
65   amount: {
66     fontSize: 16,
67     fontWeight: 'bold',
68   },
69   income: {
70     color: 'lightgreen',
71   },
72   expense: {
73     color: 'salmon',
74   },
75   empty: {
76     color: 'ccc',
77     textAlign: 'center',
78     marginTop: 50,
79     fontSize: 16,
80   },
81 });
82

```

v) Categories Page:

```

1  import React from 'react';
2  import { View, Text, StyleSheet, Dimensions, ScrollView } from 'react-native';
3  import { useTransactions } from '../context/TransactionContext';
4  import { PieChart } from 'react-native-chart-kit';
5
6  const screenWidth = Dimensions.get('window').width;
7
8  export default function Categories() {
9    const { transactions } = useTransactions();
10
11    const categoryTotals = transactions.reduce((acc, tx) => {
12      const key = tx.category || 'Uncategorized';
13      acc[key] = (acc[key] || 0) + tx.amount;
14      return acc;
15    }, {});
16
17    const chartData = Object.entries(categoryTotals).map(([name, amount], index) => ({
18      name,
19      amount,
20      color: chartColors[index % chartColors.length],
21      legendFontColor: '#ccc',
22      legendFontSize: 14,
23    }));
24
25    return (
26      <ScrollView style={styles.container}>
27        <Text style={styles.title}>Income vs Expense by Category</Text>
28
29        {chartData.length === 0 ? (
30          <Text style={styles.empty}>No data available yet.</Text>
31        ) : (
32          <PieChart
33            data={chartData}
34            width={screenWidth - 32}
35            height={260}
36            accessor="amount"
37            backgroundColor="transparent"
38            paddingLeft="16"
39            absolute
40            chartConfig={{
41              backgroundColor: '#0f172a',
42              backgroundGradientFrom: '#0f172a',
43              backgroundGradientTo: '#0f172a',
44              color: () => '#fff',
45            }}
46          />
47        )}
48      </ScrollView>
49    );
50  }
51
52  const chartColors = [
53    '#22c55e', '#f87171', '#facc15', '#60a5fa', '#a78bfa', '#fb923c', '#2dd4bf'
54  ];
55
56  const styles = StyleSheet.create({
57    container: {
58      flex: 1,
59      backgroundColor: '#0f172a',
60      padding: 16,
61    },
62    title: {
63      fontSize: 20,
64      color: '#fff',
65      fontWeight: '600',
66      textAlign: 'center',
67      marginBottom: 20,
68    },
69    empty: {
70      color: '#999',
71      textAlign: 'center',
72      marginTop: 50,
73      fontSize: 16,
74    },
75  });
76

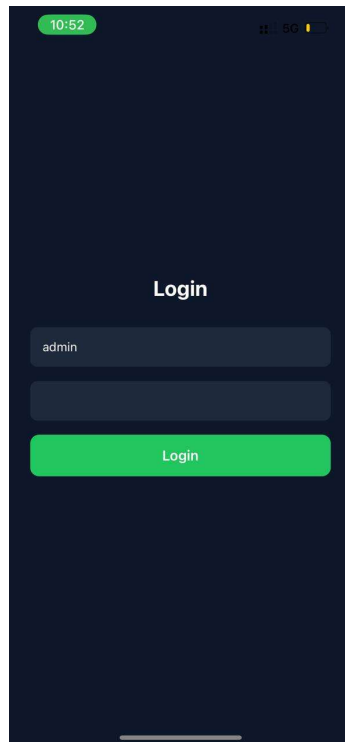
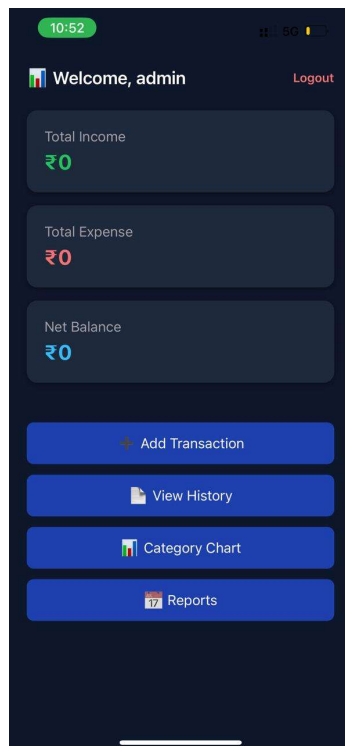
```

vi) Reports Page:

```

1 import React from 'react';
2 import { View, Text, StyleSheet, Dimensions, ScrollView } from 'react-native';
3 import { BarChart } from 'react-native-chart-kit';
4 import { useTransactions } from '../context/TransactionContext';
5
6 const screenWidth = Dimensions.get('window').width;
7
8 export default function Reports() {
9   const { transactions } = useTransactions();
10
11   const income = transactions
12     .filter((t) => t.type === 'income')
13     .reduce((sum, t) => sum + t.amount, 0);
14
15   const expense = transactions
16     .filter((t) => t.type === 'expense')
17     .reduce((sum, t) => sum + t.amount, 0);
18
19   const balance = income - expense;
20
21   // Simulate total (income - expense) per day
22   const weeklyData = [
23     { label: 'Mon', value: 500 },
24     { label: 'Tue', value: 1000 },
25     { label: 'Wed', value: 700 },
26     { label: 'Thu', value: -400 },
27     { label: 'Fri', value: 1600 },
28     { label: 'Sat', value: 300 },
29     { label: 'Sun', value: 200 },
30   ];
31
32   return (
33     <ScrollView style={styles.container}>
34       <Text style={styles.title}>Financial Report</Text>
35
36       <View style={styles.card}>
37         <Text style={styles.label}>Total Income</Text>
38         <Text style={styles.amount, { color: '#22c55e' }}>₹{income}</Text>
39       </View>
40
41       <View style={styles.card}>
42         <Text style={styles.label}>Total Expense</Text>
43         <Text style={styles.amount, { color: '#f87171' }}>₹{expense}</Text>
44       </View>
45
46       <View style={styles.card}>
47         <Text style={styles.label}>Net Balance</Text>
48         <Text style={styles.amount, { color: balance >= 0 ? '#38bdf8' : '#ef4444' }}>₹{balance}</Text>
49       </View>
50
51       <Text style={styles.chartTitle}>Net Flow (Sample Data)</Text>
52       <BarChart
53         data={
54           labels: weeklyData.map((d) => d.label),
55           datasets: [{ data: weeklyData.map((d) => d.value) }],
56         }
57         width={screenWidth - 32}
58         height={240}
59         yAxisLabel=""
60         fromZero
61         showValuesOnTopOfBars
62         chartConfig={{
63           backgroundColor: '#0f172a',
64           backgroundGradientFrom: '#0f172a',
65           backgroundGradientTo: '#0f172a',
66           decimalPlaces: 0,
67           color: (opacity = 1) => `rgba(255, 255, 255, ${opacity})`,
68           labelColor: () => 'white',
69           barPercentage: 0.6,
70         }}
71         style={styles.chart}
72       />
73     </ScrollView>
74   );
75 }
76
77 const styles = StyleSheet.create({
78   container: {
79     flex: 1,
80     backgroundColor: '#0f172a',
81     padding: 16,
82   },
83   title: {
84     fontSize: 22,
85     fontWeight: 'bold',
86     color: 'white',
87     textAlign: 'center',
88     marginBottom: 20,
89   },
90   card: {
91     backgroundColor: '#1e293b',
92     padding: 16,
93     borderRadius: 12,
94     marginBottom: 12,
95   },
96   label: {
97     fontSize: 14,
98     color: 'white',
99   },
100  amount: {
101    fontSize: 22,
102    fontWeight: 'bold',
103  },
104  chartTitle: {
105    fontSize: 16,
106    color: 'white',
107    fontWeight: 'bold',
108    marginTop: 30,
109    marginBottom: 10,
110    textAlign: 'center',
111  },
112  chart: {
113    borderRadius: 16,
114  },
115 });
116

```

e) Output:**i) Screenshot 1: Login Page****ii) Screenshot 2: Dashboard with user greeting**

iii) Screenshot 3: Add Transaction form

10:52 Add Transaction 80%

Salary

10000

Type

Income

Expense

Category

General

Salary

Food

Transport

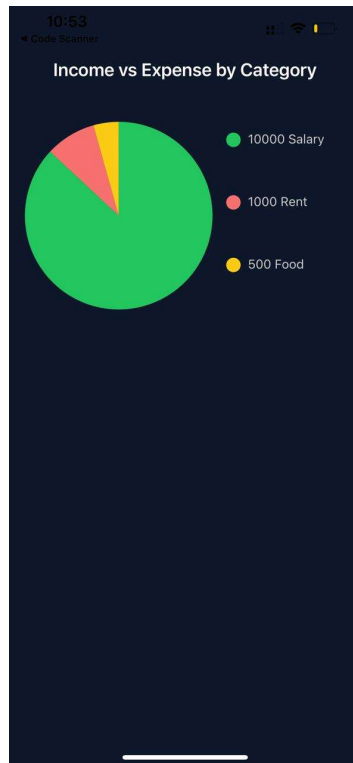
Add Transaction

iv) Screenshot 4: Transaction History

10:53 Transaction History 80%

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

v) Screenshot 5: Categories pie chart



vi) Screenshot 6: Reports bar chart



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.