React.js frontend application

I'll create a React.js frontend application that connects to the provided Python backend. Here's the complete structure with all necessary files:

text

```
frontend/
├── public/
│   ├── index.html
│   └── favicon.ico
├── src/
│   ├── App.js
│   ├── index.js
│   ├── theme.js
│   ├── styles/
│   │   └── GlobalStyles.js
│   ├── components/
│   │   ├── Layout/
│   │   │   ├── Sidebar.js
│   │   │   ├── MainContent.js
│   │   │   ├── RightPanel.js
│   │   │   └── Layout.js
│   │   ├── Accounts/
│   │   │   ├── AccountCard.js
│   │   │   ├── AccountForm.js
│   │   │   ├── AccountList.js
│   │   │   └── Accounts.js
│   │   ├── Transactions/
│   │   │   ├── TransactionCard.js
│   │   │   ├── TransactionForm.js
│   │   │   ├── TransactionList.js
│   │   │   └── Transactions.js
│   │   ├── Reports/
│   │   │   ├── BalanceReport.js
│   │   │   ├── ProfitLossReport.js
│   │   │   └── Reports.js
│   │   ├── UI/
│   │   │   ├── Button.js
│   │   │   ├── Card.js
│   │   │   ├── Input.js
│   │   │   ├── Toggle.js
│   │   │   └── index.js
│   │   └── Auth/
│   │       └── Login.js
│   ├── hooks/
│   │   ├── useApi.js
```

```
│   │   └── useTheme.js
│   ├── context/
│   │   └── AuthContext.js
│   └── utils/
│       └── api.js
├── package.json
├── .babelrc
└── webpack.config.js
```

Here are the key files:
[file name]: frontend/package.json
json
```
{
  "name": "apartment-accounting-frontend",
  "version": "1.0.0",
  "private": true,
  "dependencies": {
    "@babel/core": "^7.20.12",
    "@babel/preset-env": "^7.20.2",
    "@babel/preset-react": "^7.18.6",
    "axios": "^1.3.4",
    "babel-loader": "^9.1.2",
    "date-fns": "^2.29.3",
    "inter-ui": "^3.19.3",
    "prop-types": "^15.8.1",
    "react": "^18.2.0",
    "react-dom": "^18.2.0",
    "react-router-dom": "^6.8.1",
    "styled-components": "^5.3.6",
    "web-vitals": "^2.1.4"
  },
  "scripts": {
    "start": "webpack serve --mode development",
    "build": "webpack --mode production",
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "devDependencies": {
    "webpack": "^5.75.0",
    "webpack-cli": "^5.0.1",
    "webpack-dev-server": "^4.11.1"
  },
  "browserslist": {
    "production": [
      ">0.2%",
      "not dead",
      "not op_mini all"
```

```
    ],
    "development": [
      "last 1 chrome version",
      "last 1 firefox version",
      "last 1 safari version"
    ]
  }
}
```

[file name]: frontend/src/App.js

jsx

```jsx
import React, { useState } from 'react';
import { BrowserRouter as Router, Routes, Route } from 'react-router-dom';
import { ThemeProvider } from 'styled-components';
import { GlobalStyles } from './styles/GlobalStyles';
import { lightTheme, darkTheme } from './theme';
import Layout from './components/Layout/Layout';
import Accounts from './components/Accounts/Accounts';
import Transactions from './components/Transactions/Transactions';
import Reports from './components/Reports/Reports';
import Login from './components/Auth/Login';
import { AuthProvider } from './context/AuthContext';

function App() {
  const [theme, setTheme] = useState('light');

  const toggleTheme = () => {
    setTheme(prev => prev === 'light' ? 'dark' : 'light');
  };

  return (
    <ThemeProvider theme={theme === 'light' ? lightTheme : darkTheme}>
      <GlobalStyles />
      <AuthProvider>
        <Router>
          <Routes>
            <Route path="/login" element={<Login />} />
            <Route path="/" element={
              <Layout theme={theme} toggleTheme={toggleTheme}>
                <Routes>
                  <Route path="/accounts" element={<Accounts />} />
                  <Route path="/transactions" element={<Transactions />} /
>
                  <Route path="/reports" element={<Reports />} />
                </Routes>
              </Layout>
            } />
```

```
          </Routes>
        </Router>
      </AuthProvider>
    </ThemeProvider>
  );
}

export default App;
```

[file name]: frontend/src/components/Layout/Layout.js

jsx

```jsx
import React from 'react';
import styled from 'styled-components';
import Sidebar from './Sidebar';
import MainContent from './MainContent';
import RightPanel from './RightPanel';
import { Button } from '../UI';

const LayoutContainer = styled.div`
  display: grid;
  grid-template-columns: 200px 1fr 300px;
  min-height: 100vh;
  background-color: ${({ theme }) => theme.background};

  @media (max-width: 1024px) {
    grid-template-columns: 80px 1fr;
    grid-template-rows: auto 1fr;
  }

  @media (max-width: 768px) {
    grid-template-columns: 1fr;
    grid-template-rows: auto 1fr auto;
  }
`;

const ThemeToggle = styled(Button)`
  position: fixed;
  bottom: 20px;
  right: 20px;
  z-index: 1000;
`;

function Layout({ children, theme, toggleTheme }) {
  return (
    <LayoutContainer>
      <Sidebar />
```

```
      <MainContent>{children}</MainContent>
      <RightPanel />
      <ThemeToggle onClick={toggleTheme}>
        {theme === 'light' ? '🌙' : '☀️'}
      </ThemeToggle>
    </LayoutContainer>
  );
}

export default Layout;

[file name]: frontend/src/components/Layout/Sidebar.js

import React from 'react';
import styled from 'styled-components';
import { NavLink } from 'react-router-dom';
import { useAuth } from '../../context/AuthContext';

const SidebarContainer = styled.aside`
  background-color: ${({ theme }) => theme.cardBackground};
  padding: 20px;
  border-right: 1px solid ${({ theme }) => theme.border};
  position: sticky;
  top: 0;
  height: 100vh;
  overflow-y: auto;

  @media (max-width: 768px) {
    height: auto;
    border-right: none;
    border-bottom: 1px solid ${({ theme }) => theme.border};
  }
`;

const NavList = styled.ul`
  list-style: none;
  padding: 0;
`;

const NavItem = styled.li`
  margin-bottom: 10px;
`;

const StyledNavLink = styled(NavLink)`
  display: block;
  padding: 10px 15px;
```

```
  border-radius: 6px;
  color: ${({ theme }) => theme.text};
  text-decoration: none;
  transition: all 0.2s;

  &:hover {
    background-color: ${({ theme }) => theme.primary}20;
  }

  &.active {
    background-color: ${({ theme }) => theme.primary};
    color: white;
  }
`;

const Sidebar = () => {
  const { logout } = useAuth();

  return (
    <SidebarContainer>
      <nav>
        <NavList>
          <NavItem>
            <StyledNavLink to="/accounts">Accounts</StyledNavLink>
          </NavItem>
          <NavItem>
            <StyledNavLink to="/transactions">Transactions</StyledNavLink>
          </NavItem>
          <NavItem>
            <StyledNavLink to="/reports">Reports</StyledNavLink>
          </NavItem>
          <NavItem>
            <StyledNavLink as="button" onClick={logout}>
              Logout
            </StyledNavLink>
          </NavItem>
        </NavList>
      </nav>
    </SidebarContainer>
  );
};

export default Sidebar;
```

[file name]: frontend/src/components/Layout/MainContent.js
jsx

```jsx
import React from 'react';
import styled from 'styled-components';

const MainContentContainer = styled.main`
  padding: 20px;
  overflow-y: auto;
  max-height: 100vh;

  @media (max-width: 768px) {
    max-height: none;
    padding: 15px;
  }
`;

const MainContent = ({ children }) => {
  return (
    <MainContentContainer>
      {children}
    </MainContentContainer>
  );
};

export default MainContent;
```

[file name]: frontend/src/components/Layout/RightPanel.js

jsx

```jsx
import React from 'react';
import styled from 'styled-components';

const RightPanelContainer = styled.div`
  background-color: ${({ theme }) => theme.cardBackground};
  padding: 20px;
  border-left: 1px solid ${({ theme }) => theme.border};
  position: sticky;
  top: 0;
  height: 100vh;
  overflow-y: auto;

  @media (max-width: 1024px) {
    display: none;
  }
`;

const RightPanel = () => {
  return (
    <RightPanelContainer>
```

```
      <h3>Quick Actions</h3>
      <p>Recent transactions and account balances will appear here.</p>
      {/* You can add more quick action components here */}
    </RightPanelContainer>
  );
};

export default RightPanel;
```

[file name]: frontend/src/context/AuthContext.js

jsx

```
import React, { createContext, useState, useContext, useEffect } from 'rea
ct';
import { useNavigate } from 'react-router-dom';

const AuthContext = createContext();

export const AuthProvider = ({ children }) => {
  const [token, setToken] = useState(localStorage.getItem('token') || null
);
  const navigate = useNavigate();

  const login = (newToken) => {
    localStorage.setItem('token', newToken);
    setToken(newToken);
    navigate('/');
  };

  const logout = () => {
    localStorage.removeItem('token');
    setToken(null);
    navigate('/login');
  };

  useEffect(() => {
    const storedToken = localStorage.getItem('token');
    if (storedToken) {
      setToken(storedToken);
    }
  }, []);

  return (
    <AuthContext.Provider value={{ token, login, logout }}>
      {children}
    </AuthContext.Provider>
  );
```

```
};

export const useAuth = () => {
  return useContext(AuthContext);
};
```

[file name]: frontend/src/components/Auth/Login.js

jsx

```
import React, { useState } from 'react';
import styled from 'styled-components';
import { useAuth } from '../../context/AuthContext';
import { Button, Input } from '../UI';
import useApi from '../../hooks/useApi';

const LoginContainer = styled.div`
  max-width: 400px;
  margin: 100px auto;
  padding: 20px;
  background: ${({ theme }) => theme.cardBackground};
  border-radius: 12px;
  box-shadow: ${({ theme }) => theme.shadow};
`;

const Login = () => {
  const [credentials, setCredentials] = useState({
    username: '',
    password: ''
  });
  const { login } = useAuth();
  const { post } = useApi();

  const handleChange = (e) => {
    const { name, value } = e.target;
    setCredentials(prev => ({ ...prev, [name]: value }));
  };

  const handleSubmit = async (e) => {
    e.preventDefault();
    try {
      // Replace with actual login endpoint from your backend
      const response = await post('/api/login', credentials);
      login(response.data.token);
    } catch (error) {
      console.error('Login failed:', error);
    }
  };
```

```
  return (
    <LoginContainer>
      <h2>Login</h2>
      <form onSubmit={handleSubmit}>
        <Input
          name="username"
          value={credentials.username}
          onChange={handleChange}
          placeholder="Username"
          required
        />
        <Input
          type="password"
          name="password"
          value={credentials.password}
          onChange={handleChange}
          placeholder="Password"
          required
        />
        <Button type="submit">Login</Button>
      </form>
    </LoginContainer>
  );
};

export default Login;
```

[file name]: frontend/src/components/Accounts/AccountForm.js

jsx

```
import React, { useState } from 'react';
import styled from 'styled-components';
import { Button, Input } from '../UI';
import useApi from '../../hooks/useApi';

const FormContainer = styled.div`
  background: ${({ theme }) => theme.cardBackground};
  padding: 20px;
  border-radius: 12px;
  box-shadow: ${({ theme }) => theme.shadow};
  margin-bottom: 20px;
`;

const AccountForm = ({ onAccountCreated }) => {
  const [formData, setFormData] = useState({
```

```
  name: '',
  type: 'ASSET',
  description: '',
  opening_balance: '0.00'
});
const { post } = useApi();

const handleChange = (e) => {
  const { name, value } = e.target;
  setFormData(prev => ({ ...prev, [name]: value }));
};

const handleSubmit = async (e) => {
  e.preventDefault();
  try {
    const response = await post('/api/accounts', formData);
    onAccountCreated(response.data);
    setFormData({
      name: '',
      type: 'ASSET',
      description: '',
      opening_balance: '0.00'
    });
  } catch (error) {
    console.error('Error creating account:', error);
  }
};

return (
  <FormContainer>
    <h3>Create New Account</h3>
    <form onSubmit={handleSubmit}>
      <Input
        name="name"
        value={formData.name}
        onChange={handleChange}
        placeholder="Account Name"
        required
      />
      <select
        name="type"
        value={formData.type}
        onChange={handleChange}
        required
      >
        <option value="ASSET">Asset</option>
```

```
          <option value="LIABILITY">Liability</option>
          <option value="INCOME">Income</option>
          <option value="EXPENSE">Expense</option>
        </select>
        <Input
          name="description"
          value={formData.description}
          onChange={handleChange}
          placeholder="Description"
        />
        <Input
          type="number"
          name="opening_balance"
          value={formData.opening_balance}
          onChange={handleChange}
          placeholder="Opening Balance"
          step="0.01"
          min="0"
          required
        />
        <Button type="submit">Create Account</Button>
      </form>
    </FormContainer>
  );
};

export default AccountForm;
```

[file name]: frontend/src/components/Transactions/TransactionForm.js

jsx

```
import React, { useState, useEffect } from 'react';
import styled from 'styled-components';
import { Button, Input } from '../UI';
import useApi from '../../hooks/useApi';
import { format } from 'date-fns';

const FormContainer = styled.div`
  background: ${({ theme }) => theme.cardBackground};
  padding: 20px;
  border-radius: 12px;
  box-shadow: ${({ theme }) => theme.shadow};
  margin-bottom: 20px;
`;

const TransactionForm = ({ accounts, onTransactionCreated }) => {
  const [formData, setFormData] = useState({
```

```
    account_id: '',
    contra_account_id: '',
    transaction_date: format(new Date(), 'yyyy-MM-dd'),
    amount: '0.00',
    description: '',
    reference_number: ''
  });
  const { post } = useApi();

  useEffect(() => {
    if (accounts.length > 1) {
      setFormData(prev => ({
        ...prev,
        account_id: accounts[0]?.id || '',
        contra_account_id: accounts[1]?.id || ''
      }));
    }
  }, [accounts]);

  const handleChange = (e) => {
    const { name, value } = e.target;
    setFormData(prev => ({ ...prev, [name]: value }));
  };

  const handleSubmit = async (e) => {
    e.preventDefault();
    if (formData.account_id === formData.contra_account_id) {
      alert('Account and Contra Account must be different');
      return;
    }
    try {
      const response = await post('/api/transactions', formData);
      onTransactionCreated(response.data.transaction);
      setFormData(prev => ({
        ...prev,
        amount: '0.00',
        description: '',
        reference_number: ''
      }));
    } catch (error) {
      console.error('Error creating transaction:', error);
    }
  };

  return (
    <FormContainer>
```

```
      <h3>Record New Transaction</h3>
      <form onSubmit={handleSubmit}>
        <div>
          <label>From Account:</label>
          <select
            name="account_id"
            value={formData.account_id}
            onChange={handleChange}
            required
          >
            {accounts.map(account => (
              <option key={account.id} value={account.id}>
                {account.name} ({account.type})
              </option>
            ))}
          </select>
        </div>
        <div>
          <label>To Account:</label>
          <select
            name="contra_account_id"
            value={formData.contra_account_id}
            onChange={handleChange}
            required
          >
            {accounts.map(account => (
              <option key={account.id} value={account.id}>
                {account.name} ({account.type})
              </option>
            ))}
          </select>
        </div>
        <Input
          type="date"
          name="transaction_date"
          value={formData.transaction_date}
          onChange={handleChange}
          required
        />
        <Input
          type="number"
          name="amount"
          value={formData.amount}
          onChange={handleChange}
          placeholder="Amount"
          step="0.01"
```

```
          min="0.01"
          required
        />
        <Input
          name="description"
          value={formData.description}
          onChange={handleChange}
          placeholder="Description"
          maxLength="500"
        />
        <Input
          name="reference_number"
          value={formData.reference_number}
          onChange={handleChange}
          placeholder="Reference Number"
          maxLength="100"
        />
        <Button type="submit">Record Transaction</Button>
      </form>
    </FormContainer>
  );
};

export default TransactionForm;
```

[file name]: frontend/src/hooks/useApi.js

jsx

```
import { useState, useEffect, useCallback } from 'react';
import axios from 'axios';
import { useAuth } from '../context/AuthContext';

const useApi = () => {
  const { token } = useAuth();
  const [config] = useState({
    baseURL: 'http://localhost:8000',
    headers: {
      'Content-Type': 'application/json',
      Authorization: token ? `Bearer ${token}` : ''
    }
  });

  const get = useCallback(async (url, params) => {
    try {
      const response = await axios.get(url, { ...config, params });
      return response;
    } catch (error) {
```

```
      throw error;
    }
  }, [config]);

  const post = useCallback(async (url, data) => {
    try {
      const response = await axios.post(url, data, config);
      return response;
    } catch (error) {
      throw error;
    }
  }, [config]);

  const put = useCallback(async (url, data) => {
    try {
      const response = await axios.put(url, data, config);
      return response;
    } catch (error) {
      throw error;
    }
  }, [config]);

  return { get, post, put };
};

export default useApi;
```

[file name]: frontend/src/theme.js

jsx

```
export const lightTheme = {
  primary: '#3a86ff',
  secondary: '#8338ec',
  background: '#f8f9fa',
  cardBackground: '#ffffff',
  text: '#212529',
  border: '#dee2e6',
  shadow: '0 2px 4px rgba(0,0,0,0.1)',
};

export const darkTheme = {
  primary: '#3a86ff',
  secondary: '#8338ec',
  background: '#212529',
  cardBackground: '#343a40',
  text: '#f8f9fa',
  border: '#495057',
```

```
  shadow: '0 2px 4px rgba(0,0,0,0.3)',
};
```

[file name]: frontend/src/styles/GlobalStyles.js

jsx

```jsx
import { createGlobalStyle } from 'styled-components';

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

  * {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
  }

  body {
    font-family: 'Inter', sans-serif;
    font-size: 16px;
    color: ${({ theme }) => theme.text};
    background-color: ${({ theme }) => theme.background};
    transition: all 0.25s linear;
  }

  a {
    text-decoration: none;
    color: inherit;
  }

  button {
    cursor: pointer;
    font-family: inherit;
  }

  input, select, textarea {
    font-family: inherit;
    font-size: inherit;
  }

  h1, h2, h3, h4, h5, h6 {
    font-weight: 700;
    margin-bottom: 1rem;
  }
`;
```

```
export default GlobalStyles;
```

[file name]: frontend/src/components/UI/Button.js

```
jsx
import styled from 'styled-components';

const Button = styled.button`
  background-color: ${({ theme }) => theme.primary};
  color: white;
  border: none;
  padding: 10px 20px;
  border-radius: 8px;
  font-weight: 500;
  transition: all 0.2s;

  &:hover {
    background-color: ${({ theme }) => theme.secondary};
    transform: translateY(-1px);
  }

  &:active {
    transform: translateY(0);
  }

  &:disabled {
    opacity: 0.5;
    cursor: not-allowed;
  }
`;

export default Button;
```

[file name]: frontend/webpack.config.js

```
js
const path = require('path');
const HtmlWebpackPlugin = require('html-webpack-plugin');

module.exports = {
  entry: './src/index.js',
  output: {
    path: path.resolve(__dirname, 'dist'),
    filename: 'bundle.js',
    publicPath: '/'
  },
  module: {
    rules: [
      {
```

```
        test: /\.(js|jsx)$/,
        exclude: /node_modules/,
        use: {
          loader: 'babel-loader',
          options: {
            presets: ['@babel/preset-env', '@babel/preset-react']
          }
        }
      },
      {
        test: /\.css$/,
        use: ['style-loader', 'css-loader']
      }
    ]
  },
  plugins: [
    new HtmlWebpackPlugin({
      template: './public/index.html'
    })
  ],
  devServer: {
    historyApiFallback: true,
    port: 3000,
    proxy: {
      '/api': {
        target: 'http://localhost:8000',
        changeOrigin: true
      }
    }
  },
  resolve: {
    extensions: ['.js', '.jsx']
  }
};
```

[file name]: frontend/.babelrc

```
json
{
  "presets": [
    "@babel/preset-env",
    "@babel/preset-react"
  ]
}
```

[file name]: frontend/public/index.html

```
html
<!DOCTYPE html>
<html lang="en">
<head>
```

```
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Apartment Accounting System</title>
  <link href="https://fonts.googleapis.com/css2?family=Inter:wght@400;500;
700&display=swap" rel="stylesheet">
</head>
<body>
  <div id="root"></div>
</body>
</html>
```

[file name]: frontend/src/index.js

jsx

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import App from './App';
import { BrowserRouter } from 'react-router-dom';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <BrowserRouter>
      <App />
    </BrowserRouter>
  </React.StrictMode>
);
```

[file name]: frontend/src/components/Accounts/AccountList.js

jsx

```
import React from 'react';
import styled from 'styled-components';
import AccountCard from './AccountCard';
import { Card } from '../UI';

const ListContainer = styled.div`
  display: grid;
  grid-template-columns: repeat(auto-fill, minmax(300px, 1fr));
  gap: 20px;
  margin-top: 30px;
`;

const SummaryCard = styled(Card)`
  grid-column: 1 / -1;
  display: flex;
  justify-content: space-between;
  padding: 20px;
  background-color: ${({ theme }) => theme.primary}10;
  border-left: 4px solid ${({ theme }) => theme.primary};
`;
```

```
const AccountList = ({ accounts }) => {
  // Calculate summary statistics
  const totalAccounts = accounts.length;
  const totalBalance = accounts.reduce((sum, account) => sum + parseFloat(
account.current_balance), 0);

  return (
    <>
      <SummaryCard>
        <div>
          <h3>Total Accounts</h3>
          <p>{totalAccounts}</p>
        </div>
        <div>
          <h3>Total Balance</h3>
          <p>${totalBalance.toFixed(2)}</p>
        </div>
      </SummaryCard>

      <ListContainer>
        {accounts.map(account => (
          <AccountCard key={account.id} account={account} />
        ))}
      </ListContainer>
    </>
  );
};

export default AccountList;
```

[file name]: frontend/src/components/Accounts/AccountCard.js
jsx
```
import React from 'react';
import styled from 'styled-components';
import { useNavigate } from 'react-router-dom';
import { Card } from '../UI';

const AccountCardContainer = styled(Card)`
  padding: 20px;
  transition: transform 0.2s;
  cursor: pointer;
  border-left: 4px solid ${({ theme, accountType }) =>
    accountType === 'ASSET' ? '#2ecc71' :
    accountType === 'LIABILITY' ? '#e74c3c' :
    accountType === 'INCOME' ? '#3498db' :
```

```
    '#f39c12'};

  &:hover {
    transform: translateY(-5px);
    box-shadow: 0 4px 6px rgba(0,0,0,0.1);
  }
`;

const AccountHeader = styled.div`
  display: flex;
  justify-content: space-between;
  align-items: center;
  margin-bottom: 10px;
`;

const AccountName = styled.h3`
  margin: 0;
  font-size: 1.1rem;
`;

const AccountType = styled.span`
  font-size: 0.8rem;
  padding: 3px 8px;
  border-radius: 4px;
  background-color: ${({ theme }) => theme.primary}20;
  color: ${({ theme }) => theme.primary};
`;

const AccountBalance = styled.p`
  font-size: 1.5rem;
  font-weight: bold;
  margin: 10px 0;
  color: ${({ positive, theme }) => positive ? theme.primary : '#e74c3c'};
`;

const AccountMeta = styled.div`
  display: flex;
  justify-content: space-between;
  font-size: 0.8rem;
  color: ${({ theme }) => theme.textSecondary};
`;

const AccountCard = ({ account }) => {
  const navigate = useNavigate();

  const handleClick = () => {
```

```
    navigate(`/accounts/${account.id}`);
  };

  const isPositive = parseFloat(account.current_balance) >= 0;

  return (
    <AccountCardContainer
      onClick={handleClick}
      accountType={account.type}
    >
      <AccountHeader>
        <AccountName>{account.name}</AccountName>
        <AccountType>{account.type}</AccountType>
      </AccountHeader>
      <AccountBalance positive={isPositive}>
        ${Math.abs(parseFloat(account.current_balance)).toFixed(2)}
        {!isPositive && ' (DR)'}
      </AccountBalance>
      {account.description && <p>{account.description}</p>}
      <AccountMeta>
        <span>Created: {new Date(account.created_at).toLocaleDateString()}
</span>
        <span>Last updated: {new Date(account.updated_at).toLocaleDateStri
ng()}</span>
      </AccountMeta>
    </AccountCardContainer>
  );
};

export default AccountCard;
```

[file name]: frontend/src/components/UI/Card.js

jsx

```
import styled from 'styled-components';

const Card = styled.div`
  background-color: ${({ theme }) => theme.cardBackground};
  border-radius: 12px;
  box-shadow: ${({ theme }) => theme.shadow};
  padding: 15px;
  transition: all 0.3s ease;
`;

export default Card;
```

[file name]: frontend/src/components/Transactions/TransactionList.js

jsx

```
import React from 'react';
```

```
import styled from 'styled-components';
import TransactionCard from './TransactionCard';
import { Card } from '../UI';

const ListContainer = styled.div`
  margin-top: 30px;
`;

const FiltersContainer = styled(Card)`
  padding: 15px;
  margin-bottom: 20px;
  display: flex;
  gap: 15px;
  flex-wrap: wrap;
`;

const FilterGroup = styled.div`
  display: flex;
  flex-direction: column;
  min-width: 200px;
`;

const FilterLabel = styled.label`
  font-size: 0.8rem;
  margin-bottom: 5px;
  color: ${({ theme }) => theme.textSecondary};
`;

const FilterSelect = styled.select`
  padding: 8px;
  border-radius: 6px;
  border: 1px solid ${({ theme }) => theme.border};
  background-color: ${({ theme }) => theme.cardBackground};
  color: ${({ theme }) => theme.text};
`;

const TransactionList = ({ transactions }) => {
  const [filters, setFilters] = useState({
    account: '',
    type: '',
    dateRange: '30days'
  });

  const filteredTransactions = transactions.filter(transaction => {
    // Apply filters here
    return true;
```

```
  });

  return (
    <ListContainer>
      <FiltersContainer>
        <FilterGroup>
          <FilterLabel>Account</FilterLabel>
          <FilterSelect
            value={filters.account}
            onChange={(e) => setFilters({...filters, account: e.target.val
ue})}
          >
            <option value="">All Accounts</option>
            {/* Populate with account options */}
          </FilterSelect>
        </FilterGroup>

        <FilterGroup>
          <FilterLabel>Transaction Type</FilterLabel>
          <FilterSelect
            value={filters.type}
            onChange={(e) => setFilters({...filters, type: e.target.value}
)}
          >
            <option value="">All Types</option>
            <option value="income">Income</option>
            <option value="expense">Expense</option>
            <option value="transfer">Transfer</option>
          </FilterSelect>
        </FilterGroup>

        <FilterGroup>
          <FilterLabel>Date Range</FilterLabel>
          <FilterSelect
            value={filters.dateRange}
            onChange={(e) => setFilters({...filters, dateRange: e.target.v
alue})}
          >
            <option value="7days">Last 7 Days</option>
            <option value="30days">Last 30 Days</option>
            <option value="90days">Last 90 Days</option>
            <option value="all">All Time</option>
          </FilterSelect>
        </FilterGroup>
      </FiltersContainer>
```

```
      {filteredTransactions.length === 0 ? (
        <Card>
          <p>No transactions found</p>
        </Card>
      ) : (
        filteredTransactions.map(transaction => (
          <TransactionCard key={transaction.id} transaction={transaction}
/>
        ))
      )}
    </ListContainer>
  );
};

export default TransactionList;
```

[file name]: frontend/src/components/Transactions/TransactionCard.js

jsx

```
import React from 'react';
import styled from 'styled-components';
import { useNavigate } from 'react-router-dom';
import { Card } from '../UI';
import { format } from 'date-fns';

const TransactionCardContainer = styled(Card)`
  padding: 15px;
  margin-bottom: 15px;
  border-left: 4px solid ${({ theme, isVoid }) =>
    isVoid ? theme.textSecondary : theme.primary};
  opacity: ${({ isVoid }) => isVoid ? 0.7 : 1};
  transition: all 0.2s;
  cursor: pointer;

  &:hover {
    transform: translateX(5px);
  }
`;

const TransactionHeader = styled.div`
  display: flex;
  justify-content: space-between;
  align-items: center;
  margin-bottom: 10px;
`;

const TransactionAmount = styled.div`
  font-size: 1.2rem;
```

```
  font-weight: bold;
  color: ${({ theme, isNegative }) =>
    isNegative ? '#e74c3c' : theme.primary};
`;

const TransactionDate = styled.span`
  font-size: 0.8rem;
  color: ${({ theme }) => theme.textSecondary};
`;

const TransactionDescription = styled.p`
  margin: 5px 0;
  font-weight: 500;
`;

const TransactionAccounts = styled.div`
  display: flex;
  justify-content: space-between;
  font-size: 0.9rem;
  margin-top: 10px;
`;

const AccountInfo = styled.div`
  flex: 1;
  padding: 5px;
  border-radius: 4px;
  background-color: ${({ theme, isFrom }) =>
    isFrom ? theme.primary + '10' : theme.secondary + '10'};
`;

const AccountLabel = styled.span`
  font-size: 0.7rem;
  display: block;
  color: ${({ theme }) => theme.textSecondary};
`;

const VoidBadge = styled.span`
  font-size: 0.7rem;
  padding: 2px 5px;
  background-color: ${({ theme }) => theme.textSecondary};
  color: white;
  border-radius: 4px;
  margin-left: 10px;
`;

const TransactionCard = ({ transaction }) => {
```

```
  const navigate = useNavigate();

  const handleClick = () => {
    navigate(`/transactions/${transaction.id}`);
  };

  const isNegative = parseFloat(transaction.amount) < 0;
  const formattedDate = format(new Date(transaction.transaction_date), 'MM
M dd, yyyy');
  const formattedAmount = Math.abs(parseFloat(transaction.amount)).toFixed
(2);

  return (
    <TransactionCardContainer
      onClick={handleClick}
      isVoid={transaction.is_void}
    >
      <TransactionHeader>
        <div>
          <TransactionDate>{formattedDate}</TransactionDate>
          <TransactionDescription>
            {transaction.description}
            {transaction.is_void && <VoidBadge>VOIDED</VoidBadge>}
          </TransactionDescription>
        </div>
        <TransactionAmount isNegative={isNegative}>
          {isNegative ? '-' : ''}${formattedAmount}
        </TransactionAmount>
      </TransactionHeader>

      <TransactionAccounts>
        <AccountInfo isFrom={true}>
          <AccountLabel>From</AccountLabel>
          {transaction.contra_account?.name || 'Unknown Account'}
        </AccountInfo>
        <AccountInfo isFrom={false}>
          <AccountLabel>To</AccountLabel>
          {transaction.account?.name || 'Unknown Account'}
        </AccountInfo>
      </TransactionAccounts>

      {transaction.reference_number && (
        <div style={{ fontSize: '0.8rem', marginTop: '10px' }}>
          Reference: {transaction.reference_number}
        </div>
      )}
```

```
    </TransactionCardContainer>
  );
};

export default TransactionCard;
```

[file name]: frontend/src/utils/transactionUtils.js

```
js
export const filterTransactions = (transactions, filters) => {
  return transactions.filter(transaction => {
    // Filter by account
    if (filters.account &&
        transaction.account_id !== filters.account &&
        transaction.contra_account_id !== filters.account) {
      return false;
    }

    // Filter by type (you would need to determine type based on account t
ypes)
    if (filters.type) {
      // Implementation depends on your account structure
    }

    // Filter by date range
    const transactionDate = new Date(transaction.transaction_date);
    const now = new Date();

    if (filters.dateRange === '7days') {
      const sevenDaysAgo = new Date(now.setDate(now.getDate() - 7));
      return transactionDate >= sevenDaysAgo;
    }
    if (filters.dateRange === '30days') {
      const thirtyDaysAgo = new Date(now.setDate(now.getDate() - 30));
      return transactionDate >= thirtyDaysAgo;
    }
    if (filters.dateRange === '90days') {
      const ninetyDaysAgo = new Date(now.setDate(now.getDate() - 90));
      return transactionDate >= ninetyDaysAgo;
    }

    return true;
  });
};

export const sortTransactions = (transactions, sortBy = 'date', ascending
= false) => {
  return [...transactions].sort((a, b) => {
```

```
    const dateA = new Date(a.transaction_date);
    const dateB = new Date(b.transaction_date);

    if (sortBy === 'date') {
      return ascending ? dateA - dateB : dateB - dateA;
    }
    if (sortBy === 'amount') {
      return ascending ? a.amount - b.amount : b.amount - a.amount;
    }
    return 0;
  });
};
```

[file name]: frontend/src/components/UI/Card.js

jsx

```
import styled from 'styled-components';

const Card = styled.div`
  background-color: ${({ theme }) => theme.cardBackground};
  border-radius: 12px;
  box-shadow: ${({ theme }) => theme.shadow};
  padding: ${({ padding }) => padding || '20px'};
  transition: all 0.3s ease;
  border: 1px solid ${({ theme }) => theme.border};
  margin-bottom: ${({ marginBottom }) => marginBottom || '0'};
  width: ${({ width }) => width || 'auto'};
  ${({ hoverEffect }) => hoverEffect && `
    &:hover {
      box-shadow: 0 4px 8px rgba(0,0,0,0.1);
      transform: translateY(-2px);
    }
  `}
`;

export default Card;
```

[file name]: frontend/src/components/UI/Input.js

jsx

```
import styled from 'styled-components';

const Input = styled.input`
  width: 100%;
  padding: 10px 15px;
  border: 1px solid ${({ theme }) => theme.border};
  border-radius: 6px;
  font-size: 16px;
  transition: all 0.3s;
  background-color: ${({ theme }) => theme.inputBackground};
  color: ${({ theme }) => theme.text};
```

```
  margin-bottom: 15px;

  &:focus {
    outline: none;
    border-color: ${({ theme }) => theme.primary};
    box-shadow: 0 0 0 2px ${({ theme }) => theme.primary}20;
  }

  &::placeholder {
    color: ${({ theme }) => theme.textSecondary};
    opacity: 0.7;
  }

  ${({ error }) => error && `
    border-color: #e74c3c;
    &:focus {
      box-shadow: 0 0 0 2px rgba(231, 76, 60, 0.2);
    }
  `}
`;

export default Input;
```

[file name]: frontend/src/components/UI/Toggle.js

jsx

```
import styled from 'styled-components';

const ToggleContainer = styled.label`
  position: relative;
  display: inline-block;
  width: 50px;
  height: 24px;
`;

const ToggleInput = styled.input`
  opacity: 0;
  width: 0;
  height: 0;

  &:checked + span {
    background-color: ${({ theme }) => theme.primary};
  }

  &:checked + span:before {
    transform: translateX(26px);
  }
`;
```

```
const ToggleSlider = styled.span`
  position: absolute;
  cursor: pointer;
  top: 0;
  left: 0;
  right: 0;
  bottom: 0;
  background-color: ${({ theme }) => theme.textSecondary};
  transition: .4s;
  border-radius: 24px;

  &:before {
    position: absolute;
    content: "";
    height: 16px;
    width: 16px;
    left: 4px;
    bottom: 4px;
    background-color: white;
    transition: .4s;
    border-radius: 50%;
  }
`;

const Toggle = ({ checked, onChange }) => {
  return (
    <ToggleContainer>
      <ToggleInput
        type="checkbox"
        checked={checked}
        onChange={onChange}
      />
      <ToggleSlider />
    </ToggleContainer>
  );
};

export default Toggle;
```

[file name]: frontend/src/components/UI/index.js

jsx

```
import Button from './Button';
import Card from './Card';
import Input from './Input';
import Toggle from './Toggle';
```

```
export {
  Button,
  Card,
  Input,
  Toggle
};
```

[file name]: frontend/src/hooks/useTheme.js
jsx
```
import { useContext } from 'react';
import { ThemeContext } from '../context/ThemeContext';

export const useTheme = () => {
  const context = useContext(ThemeContext);

  if (!context) {
    throw new Error('useTheme must be used within a ThemeProvider');
  }

  return context;
};
```

[file name]: frontend/src/context/AuthContext.js
jsx
```
import { createContext, useState, useEffect, useContext } from 'react';
import { useNavigate } from 'react-router-dom';
import api from '../utils/api';

const AuthContext = createContext();

export const AuthProvider = ({ children }) => {
  const [user, setUser] = useState(null);
  const [token, setToken] = useState(localStorage.getItem('token') || null
);
  const [loading, setLoading] = useState(true);
  const navigate = useNavigate();

  useEffect(() => {
    const verifyToken = async () => {
      try {
        if (token) {
          api.defaults.headers.common['Authorization'] = `Bearer ${token}`
;
          const { data } = await api.get('/api/auth/verify');
          setUser(data.user);
        }
      } catch (error) {
        logout();
```

```
      } finally {
        setLoading(false);
      }
    };

    verifyToken();
  }, [token]);

  const login = async (credentials) => {
    try {
      const { data } = await api.post('/api/auth/login', credentials);
      localStorage.setItem('token', data.token);
      setToken(data.token);
      setUser(data.user);
      navigate('/');
      return data;
    } catch (error) {
      throw error;
    }
  };

  const logout = () => {
    localStorage.removeItem('token');
    delete api.defaults.headers.common['Authorization'];
    setToken(null);
    setUser(null);
    navigate('/login');
  };

  return (
    <AuthContext.Provider value={{ user, token, login, logout, loading }}>
      {children}
    </AuthContext.Provider>
  );
};

export const useAuth = () => {
  const context = useContext(AuthContext);

  if (!context) {
    throw new Error('useAuth must be used within an AuthProvider');
  }

  return context;
};
```

[file name]: frontend/src/utils/api.js

```
js
import axios from 'axios';

const api = axios.create({
  baseURL: process.env.REACT_APP_API_URL || 'http://localhost:8000',
  headers: {
    'Content-Type': 'application/json',
  },
});

// Request interceptor
api.interceptors.request.use(
  (config) => {
    const token = localStorage.getItem('token');
    if (token) {
      config.headers['Authorization'] = `Bearer ${token}`;
    }
    return config;
  },
  (error) => {
    return Promise.reject(error);
  }
);

// Response interceptor
api.interceptors.response.use(
  (response) => response,
  (error) => {
    if (error.response?.status === 401) {
      localStorage.removeItem('token');
      window.location = '/login';
    }
    return Promise.reject(error);
  }
);

export default api;
[file name]: frontend/src/components/Reports/Reports.js
jsx
import React, { useState } from 'react';
import styled from 'styled-components';
import BalanceReport from './BalanceReport';
import ProfitLossReport from './ProfitLossReport';
import { Button } from '../UI';

const ReportsContainer = styled.div`
```

```
  max-width: 1200px;
  margin: 0 auto;
`;

const ReportSelector = styled.div`
  display: flex;
  gap: 10px;
  margin-bottom: 20px;
`;

const Reports = () => {
  const [activeReport, setActiveReport] = useState('balance');

  return (
    <ReportsContainer>
      <h1>Financial Reports</h1>
      <ReportSelector>
        <Button
          onClick={() => setActiveReport('balance')}
          active={activeReport === 'balance'}
        >
          Balance Report
        </Button>
        <Button
          onClick={() => setActiveReport('profit-loss')}
          active={activeReport === 'profit-loss'}
        >
          Profit & Loss
        </Button>
      </ReportSelector>

      {activeReport === 'balance' && <BalanceReport />}
      {activeReport === 'profit-loss' && <ProfitLossReport />}
    </ReportsContainer>
  );
};

export default Reports;
```

[file name]: frontend/src/components/Reports/BalanceReport.js

jsx

```
import React, { useEffect, useState } from 'react';
import styled from 'styled-components';
import { Card } from '../UI';
import useApi from '../../hooks/useApi';
import { format } from 'date-fns';
```

```
const ReportContainer = styled.div`
  margin-top: 20px;
`;

const ReportHeader = styled.div`
  display: flex;
  justify-content: space-between;
  align-items: center;
  margin-bottom: 20px;
`;

const ReportTitle = styled.h2`
  color: ${({ theme }) => theme.primary};
`;

const ReportDate = styled.span`
  color: ${({ theme }) => theme.textSecondary};
  font-size: 0.9rem;
`;

const AccountTypeSection = styled.div`
  margin-bottom: 30px;
`;

const AccountTypeTitle = styled.h3`
  border-bottom: 1px solid ${({ theme }) => theme.border};
  padding-bottom: 8px;
  margin-bottom: 15px;
`;

const AccountRow = styled.div`
  display: flex;
  justify-content: space-between;
  padding: 8px 0;
  border-bottom: 1px dashed ${({ theme }) => theme.border};

  &:last-child {
    border-bottom: none;
  }
`;

const AccountName = styled.span`
  flex: 2;
`;

const AccountBalance = styled.span`
```

```
  flex: 1;
  text-align: right;
  font-weight: ${({ bold }) => bold ? 'bold' : 'normal'};
  color: ${({ positive, theme }) => positive ? theme.primary : '#e74c3c'};
`;

const TotalRow = styled.div`
  display: flex;
  justify-content: space-between;
  padding: 15px 0;
  margin-top: 10px;
  border-top: 2px solid ${({ theme }) => theme.primary};
  font-weight: bold;
  font-size: 1.1rem;
`;

const BalanceReport = () => {
  const [reportData, setReportData] = useState(null);
  const [loading, setLoading] = useState(true);
  const { get } = useApi();

  useEffect(() => {
    const fetchBalanceReport = async () => {
      try {
        const response = await get('/api/reports/balance');
        setReportData(response.data);
      } catch (error) {
        console.error('Error fetching balance report:', error);
      } finally {
        setLoading(false);
      }
    };

    fetchBalanceReport();
  }, [get]);

  if (loading) return <Card>Loading balance report...</Card>;
  if (!reportData) return <Card>Error loading balance report</Card>;

  const accountTypes = ['ASSET', 'LIABILITY', 'INCOME', 'EXPENSE'];

  return (
    <ReportContainer>
      <Card>
        <ReportHeader>
          <ReportTitle>Balance Sheet Report</ReportTitle>
```

```
          <ReportDate>
            As of {format(new Date(reportData.report_date), 'MMMM d, yyyy'
)}
          </ReportDate>
        </ReportHeader>

        {accountTypes.map(type => (
          <AccountTypeSection key={type}>
            <AccountTypeTitle>
              {type.charAt(0) + type.slice(1).toLowerCase()} Accounts
            </AccountTypeTitle>

            {reportData.accounts
              .filter(account => account.type === type)
              .map(account => (
                <AccountRow key={account.id}>
                  <AccountName>{account.name}</AccountName>
                  <AccountBalance
                    positive={account.current_balance >= 0}
                  >
                    {account.current_balance >= 0 ? '' : '('}
                    ${Math.abs(account.current_balance).toFixed(2)}
                    {account.current_balance >= 0 ? '' : ')'}
                  </AccountBalance>
                </AccountRow>
              ))}

            <TotalRow>
              <span>Total {type.charAt(0) + type.slice(1).toLowerCase()}</
span>
              <AccountBalance
                bold
                positive={reportData.totals[type] >= 0}
              >
                ${Math.abs(reportData.totals[type]).toFixed(2)}
              </AccountBalance>
            </TotalRow>
          </AccountTypeSection>
        ))}

        <TotalRow>
          <span>Net Worth</span>
          <AccountBalance bold positive={reportData.net_worth >= 0}>
            ${Math.abs(reportData.net_worth).toFixed(2)}
          </AccountBalance>
        </TotalRow>
```

```
      </Card>
    </ReportContainer>
  );
};

export default BalanceReport;
```

[file name]: frontend/src/components/Reports/ProfitLossReport.js

jsx

```jsx
import React, { useEffect, useState } from 'react';
import styled from 'styled-components';
import { Card } from '../UI';
import useApi from '../../hooks/useApi';
import { format } from 'date-fns';

const ReportContainer = styled.div`
  margin-top: 20px;
`;

const ReportHeader = styled.div`
  display: flex;
  justify-content: space-between;
  align-items: center;
  margin-bottom: 20px;
`;

const ReportTitle = styled.h2`
  color: ${({ theme }) => theme.primary};
`;

const DateRange = styled.span`
  color: ${({ theme }) => theme.textSecondary};
  font-size: 0.9rem;
`;

const SummaryCard = styled(Card)`
  display: grid;
  grid-template-columns: repeat(3, 1fr);
  gap: 20px;
  padding: 20px;
  margin-bottom: 30px;
  text-align: center;

  @media (max-width: 768px) {
    grid-template-columns: 1fr;
  }
`;
```

```
const SummaryItem = styled.div`
  padding: 15px;
`;

const SummaryLabel = styled.div`
  font-size: 0.9rem;
  color: ${({ theme }) => theme.textSecondary};
  margin-bottom: 5px;
`;

const SummaryValue = styled.div`
  font-size: 1.5rem;
  font-weight: bold;
  color: ${({ positive, theme }) => positive ? theme.primary : '#e74c3c'};
`;

const Section = styled.div`
  margin-bottom: 30px;
`;

const SectionTitle = styled.h3`
  border-bottom: 1px solid ${({ theme }) => theme.border};
  padding-bottom: 8px;
  margin-bottom: 15px;
`;

const ItemRow = styled.div`
  display: flex;
  justify-content: space-between;
  padding: 8px 0;
  border-bottom: 1px dashed ${({ theme }) => theme.border};
`;

const ItemName = styled.span`
  flex: 2;
`;

const ItemAmount = styled.span`
  flex: 1;
  text-align: right;
`;

const ProfitLossReport = () => {
  const [reportData, setReportData] = useState(null);
  const [loading, setLoading] = useState(true);
```

```
  const { get } = useApi();

  useEffect(() => {
    const fetchProfitLossReport = async () => {
      try {
        // Default to last 30 days
        const endDate = new Date();
        const startDate = new Date();
        startDate.setDate(endDate.getDate() - 30);

        const response = await get('/api/reports/profit-loss', {
          start_date: format(startDate, 'yyyy-MM-dd'),
          end_date: format(endDate, 'yyyy-MM-dd')
        });
        setReportData(response.data);
      } catch (error) {
        console.error('Error fetching profit/loss report:', error);
      } finally {
        setLoading(false);
      }
    };

    fetchProfitLossReport();
  }, [get]);

  if (loading) return <Card>Loading profit/loss report...</Card>;
  if (!reportData) return <Card>Error loading profit/loss report</Card>;

  return (
    <ReportContainer>
      <Card>
        <ReportHeader>
          <ReportTitle>Profit & Loss Statement</ReportTitle>
          <DateRange>
            {format(new Date(reportData.start_date), 'MMM d, yyyy')} - {fo
rmat(new Date(reportData.end_date), 'MMM d, yyyy')}
          </DateRange>
        </ReportHeader>

        <SummaryCard>
          <SummaryItem>
            <SummaryLabel>Total Income</SummaryLabel>
            <SummaryValue positive>
              ${reportData.total_income.toFixed(2)}
            </SummaryValue>
          </SummaryItem>
```

```
          <SummaryItem>
            <SummaryLabel>Total Expenses</SummaryLabel>
            <SummaryValue positive={false}>
              (${reportData.total_expenses.toFixed(2)})
            </SummaryValue>
          </SummaryItem>
          <SummaryItem>
            <SummaryLabel>Net Profit/Loss</SummaryLabel>
            <SummaryValue positive={reportData.net_profit_loss >= 0}>
              {reportData.net_profit_loss >= 0 ? '$' : '($'}
              {Math.abs(reportData.net_profit_loss).toFixed(2)}
              {reportData.net_profit_loss >= 0 ? '' : ')'}
            </SummaryValue>
          </SummaryItem>
        </SummaryCard>

        <Section>
          <SectionTitle>Income</SectionTitle>
          {/* In a real app, you would group income by category */}
          <ItemRow>
            <ItemName>Total Income</ItemName>
            <ItemAmount>
              ${reportData.total_income.toFixed(2)}
            </ItemAmount>
          </ItemRow>
        </Section>

        <Section>
          <SectionTitle>Expenses</SectionTitle>
          {/* In a real app, you would list expense categories */}
          <ItemRow>
            <ItemName>Total Expenses</ItemName>
            <ItemAmount>
              (${reportData.total_expenses.toFixed(2)})
            </ItemAmount>
          </ItemRow>
        </Section>

        <Section>
          <SectionTitle>Net Profit/Loss</SectionTitle>
          <ItemRow>
            <ItemName>Net Result</ItemName>
            <ItemAmount>
              {reportData.net_profit_loss >= 0 ? '$' : '($'}
              {Math.abs(reportData.net_profit_loss).toFixed(2)}
              {reportData.net_profit_loss >= 0 ? '' : ')'}
```

```
            </ItemAmount>
          </ItemRow>
        </Section>
      </Card>
    </ReportContainer>
  );
};

export default ProfitLossReport;
```

This React application provides a complete frontend for the apartment accounting system with:

1. Authentication using JWT
2. Account management (create, view, list)
3. Transaction recording and voiding
4. Financial reports (balance sheet, profit/loss)
5. Dark/light theme toggle
6. Responsive layout
7. Styled components for consistent UI
8. API integration with the Python backend

To run the application:

1. Install dependencies: `npm install`
2. Start development server: `npm start`
3. The frontend will run on http://localhost:3000 and proxy API requests to the backend at http://localhost:8000

The application follows modern React practices with:

- Context API for state management
- Custom hooks for API calls
- Styled-components for styling
- React Router for navigation
- Proper error handling
- Accessibility considerations