# **■** Project Code Export

#### **■** Frontend File List:

- .gitignore
- postcss.config.js
- public\favicon.ico
- public\index.html
- public\left.png
- public\logo192.png
- public\logo512.png
- public\manifest.json
- public\right.png
- public\robots.txt
- script.py
- src\App.css
- src\App.js
- src\App.test.js
- src\AppContent.js
- src\assets\placeholder.svg
- src\components\admin\AdminPanel.js
- src\components\common\LoadingSpinner.js
- src\components\common\TabNavigation.js
- src\components\common\Toasts.js
- src\components\form\FieldInputRow.js
- src\components\form\InputForm.js
- src\components\report\PDFPreview.js
- src\components\report\ReportOutput.js
- src\components\settings\SettingsPanel.js
- src\contexts\FormConfigContext.js
- src\data\formConfig.json
- src\hooks\useConfigImportExport.js
- src\hooks\usePDFGeneration.js
- src\hooks\useReportGeneration.js
- src\index.css
- src\index.js
- src\logo.svg
- src\reportWebVitals.js
- src\setupTests.js
- src\styles\index.css
- src\utils\constants.js
- src\utils\helpers.js
- tailwind.config.js

### ■ File: .gitignore

```
[Binary file - format]
```

\_\_\_\_\_\_

### ■ File: postcss.config.js

```
______
1: module.exports = {
 2: plugins: {
   tailwindcss: {},
4: a 5: },
    autoprefixer: {},
 6: }
```

\_\_\_\_\_\_

#### ■ File: public\favicon.ico

```
______
[Binary file - .ico format]
```

### ■ File: public\index.html

```
______
 1: <!DOCTYPE html>
 2: <html lang="en">
 3:
     <head>
       <meta charset="utf-8" />
 5:
       <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
 6:
       <meta name="viewport" content="width=device-width, initial-scale=1" />
 7:
       <meta name="theme-color" content="#000000" />
 8:
        <met.a
 9:
        name="description"
10:
         content="Web site created using create-react-app"
 11:
12:
        <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
13:
14:
         manifest.json provides metadata used when your web app is installed on a
15:
         user's mobile device or desktop. See https://developers.google.com/web/fundamentals/web-app-manifest/
16:
17:
       <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
19:
        Notice the use of %PUBLIC_URL% in the tags above.
 20:
          It will be replaced with the URL of the `public` folder during the build.
          Only files inside the `public` folder can be referenced from the HTML.
 21:
 22:
 23:
         Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC_URL%/favicon.ico" will
 24:
         work correctly both with client-side routing and a non-root public URL.
 25:
          Learn how to configure a non-root public URL by running `npm run build`
26:
 27:
       <title>React App</title>
 28:
      </head>
 29:
      <body>
 30:
        <noscript>You need to enable JavaScript to run this app./noscript>
        <div id="root"></div>
 31:
 32:
 33:
          This HTML file is a template.
 34:
          If you open it directly in the browser, you will see an empty page.
 35:
 36:
          You can add webfonts, meta tags, or analytics to this file.
 37:
          The build step will place the bundled scripts into the <body> tag.
 38:
 39:
          To begin the development, run `npm start` or `yarn start`.
         To create a production bundle, use `npm run build` or `yarn build`.
 40:
 41:
42:
     </body>
43: </html>
```

## ■ File: public\left.png

[Binary file - .png format]

## ■ File: public\logo192.png

[Binary file - .png format]

## ■ File: public\logo512.png

------

[Binary file - .png format]

\_\_\_\_\_

## ■ File: public\manifest.json

```
______
 1: {
 2:
      "short_name": "React App",
      "name": "Create React App Sample",
 3:
 4:
      "icons": [
 5:
      {
         "src": "favicon.ico",
 6:
 7:
         "sizes": "64x64 32x32 24x24 16x16",
         "type": "image/x-icon"
 8:
 9:
      },
10:
11:
         "src": "logo192.png",
12:
         "type": "image/png",
        "sizes": "192x192"
13:
14:
      },
15:
       "src": "logo512.png",
"type": "image/png",
16:
17:
         "sizes": "512x512"
18:
      }
19:
20: ],
21: "st
     "start_url": ".",
22: "display": "standalone",
23: "theme_color": "#000000",
     "background_color": "#ffffff"
24:
25: }
```

------

#### ■ File: public\right.png

[Binary file - .png format]

\_\_\_\_\_

#### ■ File: public\robots.txt

\_\_\_\_\_\_

```
# https://www.robotstxt.org/robotstxt.html
User-agent: *
Disallow:
```

\_\_\_\_\_\_

### ■ File: script.py

```
1: import os
2: from reportlab.lib.pagesizes import A4
3: from reportlab.lib.units import mm
4: from reportlab.pdfgen import canvas
5:
6: def get_code_files(directory, excluded_files=None, excluded_dirs=None):
7: """Fetch all project files except specified exclusions."""
8: if excluded_files is None:
9: excluded_files = {'package.json', 'package-lock.json'}
```

```
10:
11:
        if excluded_dirs is None:
12:
            excluded_dirs = {'node_modules', '.git', '__pycache__', 'build', '.next', 'dist'}
13:
14:
        code_files = {}
15:
16:
        for root, dirs, files in os.walk(directory):
17:
            # Skip excluded directories
18:
            dirs[:] = [d for d in dirs if d not in excluded_dirs]
19:
20:
            # Skip if current directory is an excluded directory
            if any(excluded_dir in root.split(os.sep) for excluded_dir in excluded_dirs):
21:
22:
23:
24:
            for file in files:
25:
                # Skip excluded files
                if file in excluded_files:
26:
27:
                    continue
28:
29:
                file_path = os.path.join(root, file)
30:
31:
                # Get file extension
32:
                _, ext = os.path.splitext(file)
33:
34:
                trv:
                    # Try to read as text file first
35:
                    if ext.lower() in {'.js', '.jsx', '.ts', '.tsx', '.css', '.scss', '.sass', '.less',
36:
                                      '.html', '.htm', '.json', '.md', '.txt', '.xml', '.yaml', '.yml',
37:
                                      '.config', '.gitignore', '.env', '.py', '.sh', '.bat', '.cmd',
38:
                                      '.svg', '.dockerfile', '.editorconfig', '.eslintrc', '.prettierrc'}:
39:
40:
                        with open(file_path, "r", encoding="utf-8", errors="ignore") as f:
41:
                             code_files[file_path] = f.readlines()
42:
                    else:
                        # For binary files, just note them as binary
43:
                        code_files[file_path] = [f"[Binary file - {ext} format]"]
44:
45:
46:
                except Exception as e:
47:
                    print(f"? Error reading {file_path}: {e}")
48:
                    code_files[file_path] = [f"[Error reading file: {str(e)}]"]
49:
50:
        return code_files
51:
53: def create_pdf(code_data, output_pdf="Frontend_Code_Export.pdf"):
54:
      c = canvas.Canvas(output_pdf, pagesize=A4)
55:
        width, height = A4
56:
       margin = 20 * mm
57:
       line_height = 10
58:
       y = height - margin
59:
60:
       # Title
61:
       c.setFont("Helvetica-Bold", 16)
62:
       c.drawString(margin, y, "? Project Code Export")
       y -= 2 * line_height
63:
64:
        c.setFont("Helvetica-Bold", 12)
        c.drawString(margin, y, "? Frontend File List:")
65:
       y -= 2 * line_height
66:
67:
68:
        file_paths = sorted(list(code_data.keys()))
69:
70:
        # 1. File list (original simple format)
71:
        c.setFont("Courier", 8)
72:
        for path in file_paths:
73:
            if y < margin:</pre>
74:
                c.showPage()
75:
                c.setFont("Courier", 8)
76:
                y = height - margin
77:
78:
            display_path = os.path.relpath(path)
            c.drawString(margin, y, f"- {display_path}")
79:
            y -= line_height
:08
81:
82:
        # Add page break before code content
```

```
83:
        c.showPage()
 84:
        y = height - margin
 85:
 86:
         # 2. File contents
         for file_path in file_paths:
 87:
 88:
             lines = code_data[file_path]
 89:
             print(f"? Adding: {file_path}")
 90:
 91:
             if y < margin + 3 * line_height:</pre>
 92:
                 c.showPage()
                 y = height - margin
 93:
 94:
 95:
             # File header
 96:
             rel_path = os.path.relpath(file_path)
 97:
             c.setFont("Helvetica-Bold", 12)
 98:
             c.drawString(margin, y, f"? File: {rel_path}")
 99:
             y -= line_height
100:
            # Add separator line
101:
102:
             c.setFont("Courier", 8)
             c.drawString(margin, y, "=" * 80)
103:
104:
             y -= line_height
105:
106:
             # File content
107:
             for line_num, line in enumerate(lines, 1):
                 if y < margin:</pre>
108:
109:
                     c.showPage()
                     c.setFont("Courier", 8)
110:
                     y = height - margin
111:
112:
113:
                 # Clean and truncate line
114:
                 line = line.strip("\n").encode("latin-1", "replace").decode("latin-1")
115:
                 # Add line numbers for code files
116:
                 if rel_path.endswith(('.js', '.jsx', '.ts', '.tsx', '.css', '.py', '.html', '.json')):
117:
                     display_line = f"{line_num:3d}: {line[:280]}"
118:
119:
                 else:
120:
                     display_line = line[:300]
121:
122:
                 c.drawString(margin, y, display_line)
123:
                 y -= line_height
124:
             # Add spacing between files
126:
             y -= line_height
127:
             if y > margin:
128:
                 c.setFont("Courier", 8)
129:
                 c.drawString(margin, y, "-" * 80)
130:
                 y -= 2 * line_height
131:
132:
        c.save()
133:
        print(f"? PDF successfully created: {output_pdf}")
134:
        print(f"? Total files processed: {len(code_data)}")
135:
136:
137: def main():
138:
        root_dir = os.path.dirname(os.path.abspath(__file__))
139:
140:
         # Files to exclude (including package.json as requested)
141:
         excluded_files = {
142:
             'package.json',
143:
             'package-lock.json',
144:
             'yarn.lock',
145:
             'README.md',
146:
             '.DS_Store',
             'Thumbs.db',
147:
148:
             'Desktop.ini'
149:
         }
150:
151:
         # Directories to exclude
152:
         excluded_dirs = {
153:
             'node_modules',
154:
             '.git',
155:
             '__pycache__',
```

```
156:
           'build',
157:
            'dist',
158:
            '.next',
159:
            'coverage',
            '.nyc_output',
160:
            'logs',
161:
            '*.log'
162:
163:
       }
164:
        print("? Scanning project files...")
165:
        code_files = get_code_files(root_dir, excluded_files, excluded_dirs)
166:
167:
168:
       if not code_files:
            print("? No files found to process!")
169:
170:
            return
171:
       print(f"? Found {len(code_files)} files to include in PDF")
172:
173:
        create_pdf(code_files)
174:
175:
176: if __name__ == "__main__":
      main()
```

## ■ File: src\App.css

```
______
 1: .App {
 2: text-align: center;
 3: }
 4:
 5: .App-logo {
 6: height: 40vmin;
 7: pointer-events: none;
 8: }
 9:
10: @media (prefers-reduced-motion: no-preference) {
11: .App-logo {
12:
      animation: App-logo-spin infinite 20s linear;
13:
     }
14: }
15:
16: .App-header {
17: background-color: #282c34;
18: min-height: 100vh;
19: display: flex;
20: flex-direction: column;
21: align-items: center;
22: justify-content: center;
23: font-size: calc(10px + 2vmin);
24: color: white;
25: }
26:
 27: .App-link {
28:
     color: #61dafb;
29: }
30:
31: @keyframes App-logo-spin {
 32: from {
33:
      transform: rotate(0deg);
34: }
35: to {
36:
37: }
      transform: rotate(360deg);
38: }
```

\_\_\_\_\_\_

# ■ File: src\App.js

```
1: import React from 'react';
```

```
2: import { FormConfigProvider } from './contexts/FormConfigContext';
3: import AppContent from './AppContent';
```

```
4: import Toasts from './components/common/Toasts';
6: function App() {
7:
    return (
      <FormConfigProvider>
8:
9:
        <AppContent />
10:
        <Toasts />
11:
      </FormConfigProvider>
    );
12:
13: }
14:
15: export default App;
```

-----

## ■ File: src\App.test.js

```
1: import { render, screen } from '@testing-library/react';
2: import App from './App';
3:
4: test('renders learn react link', () => {
5: render(<App />);
6: const linkElement = screen.getByText(/learn react/i);
7: expect(linkElement).toBeInTheDocument();
8: });
```

-----

## ■ File: src\AppContent.js

```
______
 1: import React, { useState } from 'react';
 2: import TabNavigation from './components/common/TabNavigation';
 3: import InputForm from './components/form/InputForm';
 4: import ReportOutput from './components/report/ReportOutput';
 5: import AdminPanel from './components/admin/AdminPanel';
 6: import SettingsPanel from './components/settings/SettingsPanel';
 7: import { useReportGeneration } from './hooks/useReportGeneration';
 8: import { AnimatePresence, motion } from 'framer-motion';
 9:
10: const AppContent = () => {
11: const [activeTab, setActiveTab] = useState('form');
12: const { reportData, generateReport } = useReportGeneration();
14: const handleGenerateReport = (formData) => {
15:
       generateReport(formData);
    };
16:
17:
18:
     const tabTransition = {
19:
     initial: { opacity: 0, y: 20 },
 20:
       animate: { opacity: 1, y: 0 },
       exit: { opacity: 0, y: -20 },
21:
 22:
       transition: { duration: 0.3 }
 23: };
 24:
 25:
      return (
       <div className="bg-gray-100 font-sans min-h-screen">
 26:
 27:
          <TabNavigation activeTab={activeTab} setActiveTab={setActiveTab} />
 28:
 29:
          <AnimatePresence mode="wait">
            {activeTab === 'form' && (
 30:
              <motion.div key="form" {...tabTransition}>
 31:
 32:
                <div className="flex flex-col md:flex-row h-screen desktop-layout">
 33:
                 <InputForm onGenerateReport={handleGenerateReport} reportData={reportData} />
 34:
                  <ReportOutput reportData={reportData} />
35:
               </div>
 36:
              </motion.div>
37:
            ) }
 38:
            {activeTab === 'admin' && (
39:
40:
             <motion.div key="admin" {...tabTransition}>
 41:
               <div className="p-6">
 42:
                 <AdminPanel />
 43:
                </div>
```

```
44:
              </motion.div>
45:
            ) }
46:
47:
            {activeTab === 'settings' && (
             <motion.div key="settings" {...tabTransition}>
48:
49:
                <div className="p-6">
50:
                  <SettingsPanel />
51:
                </div>
52:
              </motion.div>
53:
            ) }
54:
          </AnimatePresence>
55:
       </div>
56:
     );
57: };
58:
59: export default AppContent;
```

## ■ File: src\assets\placeholder.svg

\_\_\_\_\_\_

## ■ File: src\components\admin\AdminPanel.js

```
1: import React from "react";
 2: import { useFormConfig } from "../../contexts/FormConfigContext";
 4: const AdminPanel = () => {
 5: const { state, dispatch } = useFormConfig();
 6:
 7: // ? Add a new field template
 8: const addNewField = () => {
 9:
      const newField = {
 10:
         id: `field_${Date.now()}`,
         label: "New Field",
11:
         category: "",
12:
13:
        high: "High score recommendation",
        normal: "Normal score recommendation",
14:
15:
         low: "Low score recommendation",
16:
        };
17:
18:
        dispatch({ type: "ADD_FIELD", field: newField });
19:
     };
20:
21:
     // ?? Delete field by index
 22:
     const deleteField = (index) => {
 23:
      if (window.confirm("Are you sure you want to delete this field?")) {
 24:
          dispatch({ type: "DELETE_FIELD", index });
25:
     };
26:
 27:
 28:
      // ?? Update specific property in a field
 29:
      const updateField = (index, property, value) => {
       dispatch({ type: "UPDATE_FIELD", index, property, value });
30:
 31:
 32:
 33:
     return (
 34:
        <div className="max-w-6xl mx-auto px-4 py-8">
 35:
          <div className="flex justify-between items-center mb-6">
 36:
            <h2 className="text-2xl font-bold">?? Field Management</h2>
 37:
            <button
 38:
             onClick={addNewField}
39:
             className="bg-green-600 hover:bg-green-700 text-white px-5 py-2 rounded-lg font-semibold"
 40:
             ? Add New Field
41:
 42:
            </button>
43:
          </div>
44:
 45:
          {/* List all fields */}
 46:
          <div className="space-y-6">
 47:
            {state.fields.map((field, index) => (
```

```
48:
               <div
 49:
                 key={field.id}
                 className="border rounded-lg p-6 bg-white shadow-sm space-y-4"
 50:
 51:
                 <div className="flex justify-between items-center">
 52:
 53:
                   <h3 className="text-lg font-semibold">
 54:
                     Field {index + 1}: {field.label}
 55:
                   </h3>
 56:
                   <button
 57:
                     onClick={() => deleteField(index)}
 58:
                     className="text-red-600 hover:text-red-800 font-medium"
 59:
 60:
                     ?? Delete
 61:
                   </button>
 62:
                 </div>
 63:
                 <div className="grid grid-cols-1 md:grid-cols-2 gap-4">
 64:
 65:
                   {/* ID */}
 66:
                   <div>
 67:
                     <label className="block text-sm font-medium mb-1">
 68:
                       Field ID
 69:
                      </label>
 70:
                     <input
                       type="text"
 71:
 72:
                        value={field.id}
                       onChange={(e) => updateField(index, "id", e.target.value)}
 73:
 74:
                       className="w-full border border-gray-300 rounded px-3 py-2"
                     />
 75:
                   </div>
 76:
 77:
 78:
                    {/* Label */}
 79:
                   <div>
                     <label className="block text-sm font-medium mb-1">Label</label>
 80:
 81:
                      <input
                       type="text"
 82:
 83:
                        value={field.label}
                        onChange={(e) => updateField(index, "label", e.target.value)}
 84:
                       className="w-full border border-gray-300 rounded px-3 py-2"
 85:
 86:
                      />
 87:
                   </div>
 88:
                   {/* Category */}
 89:
 90:
                     <label className="block text-sm font-medium mb-1">
 91:
 92:
                       Category
 93:
                      </label>
 94:
                     <input
 95:
                       type="text"
                       value={field.category}
 96:
 97:
                       onChange={(e) =>
                         updateField(index, "category", e.target.value)
 98:
 99:
100:
                        className="w-full border border-gray-300 rounded px-3 py-2"
101:
                      />
102:
                   </div>
103:
104:
                   {/* HIGH Recommendation */}
105:
                   <div>
106:
                      <label className="block text-sm font-medium mb-1">
107:
                       High Recommendation
108:
                     </label>
109:
                     <textarea
110:
                        value={field.high}
111:
                        onChange={(e) => updateField(index, "high", e.target.value)}
                       className="w-full border border-gray-300 rounded px-3 py-2"
112:
113:
                       rows={3}
                     />
114:
115:
                   </div>
116:
117:
                    { /* NORMAL Recommendation */}
118:
                     <label className="block text-sm font-medium mb-1">
119:
120:
                       Normal Recommendation
```

```
121:
                     </label>
122:
123:
                       value={field.normal}
124:
                       onChange={(e) => updateField(index, "normal", e.target.value)}
125:
                       className="w-full border border-gray-300 rounded px-3 py-2"
126:
                     />
127:
128:
                   </div>
129:
                   { /* LOW Recommendation */}
130:
131:
                     <label className="block text-sm font-medium mb-1">
132:
133:
                       Low Recommendation
134:
                     </label>
135:
                     <textarea
136:
                       value={field.low}
                       onChange={(e) => updateField(index, "low", e.target.value)}
137:
138:
                       className="w-full border border-gray-300 rounded px-3 py-2"
139:
                       rows={3}
140:
141:
                   </div>
142:
                 </div>
143:
               </div>
144:
            ))}
145:
          </div>
145
146:
'7: );
        </div>
148: };
150: export default AdminPanel;
```

-----

## ■ File: src\components\common\LoadingSpinner.js

.-----

\_\_\_\_\_\_

## ■ File: src\components\common\TabNavigation.js

```
_______
 1: import React from "react";
 3: const TabNavigation = ({ activeTab, setActiveTab }) => {
 4: const tabs = [
      { id: "form", label: "? Form View" },
        { id: "admin", label: "?? Admin Panel" },
 6:
       { id: "settings", label: "?? Form Settings" },
 7:
 8:
     ];
 9:
10:
     return (
11:
      <div className="bg-white shadow-sm border-b sticky top-0 z-10">
         <div className="container mx-auto px-4">
13:
           <div className="flex gap-2 py-4">
14:
             {tabs.map((tab) => (
15:
               <button
                 key={tab.id}
16:
17:
                 className={`px-4 py-2 rounded-lg transition-all font-medium ${
18:
                   activeTab === tab.id
                     ? "bg-green-600 text-white"
19:
                     : "bg-gray-200 text-gray-700 hover:bg-gray-300"
20:
21:
22:
                 onClick={() => setActiveTab(tab.id)}
 23:
 24:
                 {tab.label}
25:
               </button>
 26:
             ))}
27:
           </div>
 28:
         </div>
29:
       </div>
30: );
31: };
 32:
 33: export default TabNavigation;
```

### ■ File: src\components\common\Toasts.js

```
______
 1: import { ToastContainer } from "react-toastify";
 2: import "react-toastify/dist/ReactToastify.css";
 3:
 4: const Toasts = () => {
 5: return (
      <ToastContainer
 6:
       position="top-right"
 7:
 8:
         autoClose={3000}
 9:
       hideProgressBar={false}
      newestOnTop={false}
closeOnClick
pauseOnFocusLoss
draggable
10:
11:
12:
13:
14:
       pauseOnHover
        theme="colored"
15:
16:
17: );
18: };
19:
20: export default Toasts;
```

\_\_\_\_\_\_

## ■ File: src\components\form\FieldInputRow.js

\_\_\_\_\_\_

\_\_\_\_\_\_

## ■ File: src\components\form\InputForm.js

```
______
 1: import React, { useState, useEffect } from "react";
 2: import { useFormConfig } from "../../contexts/FormConfigContext";
 3: import { usePDFGeneration } from "../../hooks/usePDFGeneration";
 4: import { isValidScore } from "../../utils/helpers";
 5: import { toast } from "react-toastify";
 7: const LOCAL_STORAGE_KEY = "genomics_form_data";
 8:
 9: const InputForm = ({ onGenerateReport, reportData }) => {
10: const { state } = useFormConfig();
 11: const { generatePDF } = usePDFGeneration();
12:
13:
      // Try restoring from localStorage
14:
     const [formData, setFormData] = useState(() => {
15:
      trv {
16:
        const stored = localStorage.getItem(LOCAL_STORAGE_KEY);
17:
         return stored ? JSON.parse(stored) : {};
18:
        } catch (e) {
19:
         return {};
      }
 20:
 21: });
 22:
 23:
      const [errors, setErrors] = useState({});
 24:
     // ? Save to localStorage on formData change
 25:
 26:
     useEffect(() => {
 27:
       localStorage.setItem(LOCAL_STORAGE_KEY, JSON.stringify(formData));
 28:
      }, [formData]);
     const handleInputChange = (fieldId, value) => {
 30:
      const updatedValue = value.replace(/\D/g, "");
 31:
 32:
        setFormData((prev) => ({
33:
          ...prev,
 34:
         [fieldId]: updatedValue,
35:
      }));
 36:
       setErrors((prev) => ({
37:
          ...prev,
38:
          [fieldId]: null,
 39:
        }));
 40:
      };
 41:
```

```
const validateForm = () => {
 42:
 43:
        const newErrors = {};
 44:
         state.fields.forEach((field) => {
 45:
           const value = formData[field.id];
 46:
           if (!isValidScore(value)) {
 47:
             newErrors[field.id] = "Score must be between 1 and 10";
 48:
 49:
         });
 50:
         setErrors(newErrors);
 51:
         return Object.keys(newErrors).length === 0;
 52:
 53:
 54:
       const handleSubmit = (e) => {
 55:
         e.preventDefault();
 56:
         if (validateForm()) {
 57:
           onGenerateReport(formData);
           toast.success("? Report generated and saved!");
 58:
 59:
         } else {
 60:
           toast.error("? Please fix errors before submitting.");
 61:
 62:
       };
 63:
 64:
       const handleDownloadPDF = () => {
 65:
         if (!reportData | reportData.length === 0) {
           toast.warning("Please generate a report first.");
 66:
 67:
 68:
 69:
         generatePDF(reportData);
 70:
 71:
 72:
       const handleClearForm = () => {
 73:
         if (window.confirm("Clear all form scores and reset saved state?")) {
 74:
           localStorage.removeItem(LOCAL_STORAGE_KEY);
 75:
           setFormData({});
 76:
           toast.info("?? Cleared saved input.");
 77:
         }
       };
 78:
 79:
 :08
       return (
 81:
         <div className="w-full md:w-1/2 bq-white p-4 md:p-8 overflow-y-auto mobile-section">
 82:
           { \{ /* \ \dots \ \text{header \& description remain the same */} \} }
 83:
 84:
           <form onSubmit={handleSubmit} className="space-y-3 md:space-y-4">
 85:
             <div className="grid grid-cols-1 gap-4">
 86:
               {state.fields.map((field, index) => (
 87:
                 <div
 88:
                   key={field.id}
 89:
                   className="flex flex-col md:flex-row md:items-center"
 90:
 91:
                   <label className="w-full md:w-48 text-sm font-semibold mb-1 md:mb-0">
 92:
                     {field.label}
 93:
                   </label>
 94:
                   <input
 95:
                     type="number"
 96:
                     min="1"
                     max="10"
 97:
 98:
                     value={formData[field.id] || ""}
 99:
                     onChange={(e) => handleInputChange(field.id, e.target.value)}
100:
                     className={`border p-2 w-full md:w-20 text-center rounded
                       ${errors[field.id] ? "border-red-500" : "border-gray-300"}`}
101:
102:
103:
                   {errors[field.id] && (
104:
                     105:
                       {errors[field.id]}
106:
                     107:
                   ) }
108:
                 </div>
109:
               ))}
110:
             </div>
111:
112:
             <div className="flex flex-col md:flex-row gap-2 mt-6">
113:
               <but.t.on
                 type="submit"
114:
```

```
115:
                className="bg-green-600 hover:bg-green-700 text-white px-6 py-3 rounded-lg font-semibold"
116:
117:
                Generate Report
118:
               </button>
119:
120:
              <button
121:
                type="button"
122:
                onClick={handleDownloadPDF}
                className="bg-blue-600 hover:bg-blue-700 text-white px-6 py-3 rounded-lg font-semibold"
123:
124:
125:
                Download PDF
126:
              </button>
127:
128:
              <button
129:
               type="button"
130:
                onClick={handleClearForm}
131:
                className="bg-gray-500 hover:bg-gray-600 text-white px-6 py-3 rounded-lg font-semibold"
132:
133:
                Clear Input
134:
              </button>
135:
            </div>
136:
          </form>
137:
       </div>
138: );
139: };
140:
141: export default InputForm;
```

### ■ File: src\components\report\PDFPreview.js

-----

\_\_\_\_\_

# ■ File: src\components\report\ReportOutput.js

```
______
 1: import React from "react";
 2: import { useFormConfig } from "../../contexts/FormConfigContext";
 3:
 4: const ReportOutput = ({ reportData }) => {
 5: const { state } = useFormConfig();
 6:
 7:
     // Assign color to score
 8:
     const getScoreColor = (score) => {
 9:
       if (score >= state.highThreshold) return "bg-red-600";
       if (score >= 4) return "bg-yellow-500";
 10:
11:
       return "bg-green-600";
     };
12:
13:
     // Style active/inactive text
14:
15: const getTextStyle = (isActive) =>
16:
      isActive ? "text-gray-900 font-semibold" : "text-gray-400";
17:
18:
     if (!reportData | | reportData.length === 0) {
19:
      return (
 20:
          <div className="w-full md:w-1/2 p-4 md:p-8 mobile-section bg-gray-50">
21:
            <div className="bg-white border border-gray-300 rounded-lg p-8 text-center text-gray-500">
 22:
             Fill in the form and generate report to view results here.
23:
            </dim>
          </div>
 25:
       );
 26:
     }
 27:
     let currentCategory = null;
 28:
 29:
30:
     return (
 31:
      <div className="w-full md:w-1/2 p-4 md:p-8 mobile-section bg-gray-50 overflow-y-auto">
 32:
          <div className="bg-white border border-gray-300 rounded-lg shadow-sm overflow-hidden">
            {reportData.map((item, index) => {
33:
 34:
             const { field, score, showHigh, showNormal, showLow } = item;
 35:
             const isNewCategory =
 36:
               field.category && field.category !== currentCategory;
```

```
37:
               const elements = [];
 38:
 39:
                // Add category header if changed
 40:
               if (isNewCategory) {
                 currentCategory = field.category;
 41:
 42:
                 elements.push(
 43:
                    <div
 44:
                     key={`category-${field.category}`}
 45:
                     className="bg-gray-200 px-4 py-2 font-bold text-sm text-gray-700 border-1-4 border-gray-400
 46:
 47:
                      {currentCategory}
 48:
                   </div>
 49:
                 );
                }
 50:
 51:
 52:
                // Add field row
 53:
               elements.push(
 54:
                  <div
 55:
                   key={field.id}
 56:
                   className="flex flex-col md:flex-row items-stretch border-b border-gray-200"
 57:
 58:
                    {/* Field Label */}
 59:
                    <div className="w-full md:w-48 px-3 py-3 text-center md:text-right bg-gray-100 md:bg-white">
 60:
                      <div className="text-xs font-bold text-gray-700 uppercase leading-tight">
                        {field.label}
 61:
                      </div>
 62:
 63:
                    </div>
 64:
                    {/* Score */}
 65:
 66:
                    <div className="w-full md:w-16 flex justify-center items-center py-3">
 67:
 68:
                        className={`w-10 h-10 ${getScoreColor(
 69:
 70:
                        )} text-white font-bold text-lg flex items-center justify-center rounded-full`}
 71:
 72:
                        {score}
 73:
                      </div>
 74:
                    </div>
 75:
                    {/* Recommendations */}
 76:
 77:
                    <div className="flex-1 px-3 py-3 flex flex-col md:flex-row">
 78:
                      {/* High */}
 79:
                      <div className="w-full md:w-1/3 md:pr-2 mb-3 md:mb-0">
 80:
                        <div
 81:
                          className={`text-xs font-bold mb-1 ${getTextStyle(
 82:
                            showHigh
 83:
                          ) } ` }
 84:
 85:
                          HIGH
 86:
                        </div>
 87:
                        <div
                          className={`text-xs leading-tight ${getTextStyle(
 88:
 89:
                            showHigh
 90:
                          )}`}
 91:
                          {field.high.split("\n").map((line, i, arr) => (}
 92:
 93:
                            <React.Fragment key={i}>
 94:
                              {line}
 95:
                              {i < arr.length - 1 && <br />}
 96:
                            </React.Fragment>
 97:
                          ))}
 98:
                        </div>
 99:
                      </div>
100:
                      {/* Normal */}
101:
102:
                      <div className="w-full md:w-1/3 md:px-2 mb-3 md:mb-0">
103:
104:
                          className={`text-xs font-bold mb-1 ${getTextStyle(
                            showNormal
105:
106:
                          ) } ` }
107:
108:
                          NORMAL
109:
                        </div>
```

```
110:
                        <div
111:
                          className={`text-xs leading-tight ${getTextStyle(
112:
                            showNormal
113:
                          ) } ` }
114:
                          {field.normal?.split("\n").map((line, i, arr) => (}
115:
116:
                            <React.Fragment key={i}>
117:
                              {line}
118:
                              {i < arr.length - 1 && <br />}
119:
                            </React.Fragment>
120:
                          ))}
121:
                        </div>
122:
                      </div>
123:
124:
                      {/* Low */}
125:
                      <div className="w-full md:w-1/3 md:pl-2">
126:
                        <div
127:
                          className={`text-xs font-bold mb-1 ${getTextStyle(
128:
                            showLow
129:
                          ) } ` }
130:
                         LOW
131:
132:
                        </div>
133:
                        <div
134:
                          className={`text-xs leading-tight ${getTextStyle(showLow)}`}
135:
136:
                          {field.low.split("\n").map((line, i, arr) => (}
137:
                            <React.Fragment key={i}>
138:
                             {line}
139:
                              {i < arr.length - 1 && <br />}
140:
                            </React.Fragment>
141:
                          ))}
142:
                        </div>
                     </div>
143:
144:
                   </div>
145:
                 </div>
               );
146:
147:
148:
               return elements;
149:
             })}
150:
          </div>
151:
        </div>
152: );
153: };
154:
155: export default ReportOutput;
```

-----

#### ■ File: src\components\settings\SettingsPanel.js

```
______
 1: import React, { useState } from "react";
 2: import { useFormConfig } from "../../contexts/FormConfigContext";
 4: const SettingsPanel = () => {
 5: const { state, dispatch } = useFormConfig();
 6:
 7:
     // Internal editable state
 8:
     const [settings, setSettings] = useState({
 9:
       title: state.title,
10:
       quote: state.quote,
11:
       description: state.description,
12:
       headerColor: state.headerColor,
13:
       highThreshold: state.highThreshold,
      colors: {
14:
15:
        low: state.colors.low,
        medium: state.colors.medium,
16:
17:
         high: state.colors.high,
      },
18:
19:
     });
 20:
 21:
     // Change handlers
 22:
      const handleChange = (key, value) => {
```

```
23:
       setSettings((prev) => ({
24:
         ...prev,
25:
          [key]: value,
26:
       }));
     };
27:
28:
29:
     const handleColorChange = (level, value) => {
30:
      setSettings((prev) => ({
31:
         ...prev,
         colors: {
32:
33:
            ...prev.colors,
34:
           [level]: value,
35:
          },
36:
       }));
37:
     };
38:
39:
     // Apply changes to context
40:
     const applySettings = () => {
       dispatch({ type: "UPDATE_SETTINGS", settings });
41:
42:
       alert("? Settings applied successfully!");
43:
     };
44:
45:
     // Reset to initial/default config
46:
     const resetSettings = () => {
47:
      if (
48:
         window.confirm("Are you sure you want to reset all settings to default?")
49:
         window.location.reload(); // simplest way to reload JSON state
50:
51:
       }
52:
     };
53:
54:
     return (
       <div className="max-w-4xl mx-auto px-4 py-8">
55:
          <h2 className="text-2xl font-bold mb-6">?? Form Customization</h2>
57:
58:
         <div className="grid grid-cols-1 md:grid-cols-2 gap-6">
           {/* Left Section */}
59:
60:
            <div className="bg-white rounded-lg shadow-sm p-6 space-y-4">
61:
              <h3 className="text-lg font-semibold">? Header Info</h3>
62:
63:
              <div>
64:
               <label className="block text-sm font-medium mb-1">Main Title</label>
65:
               <input
66:
                  type="text"
67:
                  value={settings.title}
68:
                  onChange={(e) => handleChange("title", e.target.value)}
69:
                 className="w-full border rounded px-3 py-2"
70:
                />
              </div>
71:
72:
73:
              <div>
74:
               <label className="block text-sm font-medium mb-1">Quote</label>
75:
               <textarea
76:
                 rows={2}
77:
                  value={settings.quote}
78:
                  onChange={(e) => handleChange("quote", e.target.value)}
79:
                  className="w-full border rounded px-3 py-2"
                />
80:
81:
              </div>
82:
83:
84:
                <label className="block text-sm font-medium mb-1">
85:
                 Description
86:
                </label>
87:
                <textarea
88:
                 rows={4}
                 value={settings.description}
89:
90:
                  onChange={(e) => handleChange("description", e.target.value)}
91:
                  className="w-full border rounded px-3 py-2"
               />
92:
93:
              </div>
94:
95:
              <div>
```

```
96:
                <label className="block text-sm font-medium mb-1">
 97:
                  Header Background Color
 98:
                 </label>
 99:
                 <input
                  type="color"
100:
101:
                  value={settings.headerColor}
102:
                  onChange={(e) => handleChange("headerColor", e.target.value)}
103:
                  className="h-10 w-full border rounded"
104:
               </div>
105:
             </div>
106:
107:
108:
             {/* Right Section */}
109:
             <div className="bg-white rounded-lg shadow-sm p-6 space-y-4">
110:
               <h3 className="text-lg font-semibold">? Score Logic</h3>
111:
112:
              <div>
113:
                <label className="block text-sm font-medium mb-1">
114:
                  High Score Threshold (?)
                 </label>
116:
                 <input
117:
                   type="number"
118:
                  min="1"
119:
                  max="10"
120:
                  value={settings.highThreshold}
121:
                  onChange={(e) =>
122:
                    handleChange("highThreshold", parseInt(e.target.value))
123:
124:
                   className="w-full border rounded px-3 py-2"
125:
                 />
126:
               </div>
127:
               <h3 className="text-lg font-semibold mt-6">? Score Colors</h3>
128:
129:
130:
               <div className="space-v-2">
131:
                 <div>
132:
                   <label className="block text-sm font-medium mb-1">High</label>
133:
                   <input
134:
                     type="color"
135:
                     value={settings.colors.high}
136:
                     onChange={(e) => handleColorChange("high", e.target.value)}
137:
                     className="h-10 w-full border rounded"
138:
                   />
                 </div>
139:
140:
141:
                 <div>
142:
                  <label className="block text-sm font-medium mb-1">Medium</label>
143:
                   <input
                     type="color"
144:
145:
                     value={settings.colors.medium}
                     onChange={(e) => handleColorChange("medium", e.target.value)}
146:
                    className="h-10 w-full border rounded"
147:
148:
                   />
149:
                </div>
150:
151:
                 <div>
                   <label className="block text-sm font-medium mb-1">Low</label>
152:
153:
                   <input
154:
                     type="color"
155:
                     value={settings.colors.low}
                     onChange={(e) => handleColorChange("low", e.target.value)}
156:
157:
                     className="h-10 w-full border rounded"
158:
                   />
159:
                 </div>
160:
               </div>
161:
             </div>
162:
           </div>
163:
          {/* Buttons */}
164:
165:
           <div className="mt-6 flex gap-4">
166:
167:
               onClick={applySettings}
168:
               className="bg-green-600 hover:bg-green-700 text-white px-6 py-3 rounded-lg font-semibold"
```

```
169:
170:
             ? Apply Settings
171:
            </but.ton>
172:
            <button
173:
            onClick={resetSettings}
174:
             className="bg-gray-600 hover:bg-gray-700 text-white px-6 py-3 rounded-lg font-semibold"
175:
176:
             ?? Reset to Default
177:
            </button>
178:
          </div>
179:
       </div>
180: );
181: };
182:
183: export default SettingsPanel;
```

## ■ File: src\contexts\FormConfigContext.js

```
______
 1: import React, { createContext, useContext, useReducer } from "react";
 2: import initialConfig from "../data/formConfig.json";
 3:
 4: // --- Reducer Function ---
 5: const formConfigReducer = (state, action) => {
 6: switch (action.type) {
      case "UPDATE_FIELD":
 7:
 8:
        return {
 9:
           ...state,
10:
           fields: state.fields.map((field, i) =>
11:
             i === action.index
               ? { ...field, [action.property]: action.value }
12:
13:
               : field
14:
           ),
         };
15:
16:
        case "ADD_FIELD":
17:
18:
        return {
19:
           ...state,
20:
           fields: [...state.fields, action.field],
21:
22:
23:
      case "DELETE_FIELD":
        return {
24:
 25:
           ...state,
           fields: state.fields.filter((_, i) => i !== action.index),
26:
27:
28:
       case "UPDATE_SETTINGS":
 29:
30:
         return {
31:
           ...state,
32:
            ...action.settings,
         };
33:
34:
35:
       case "IMPORT_CONFIG":
        return {
36:
37:
          ...state,
           ...action.config,
38:
39:
40:
41:
      default:
42:
         return state;
43:
     }
 44: };
 46: // --- Context Creation ---
47: const FormConfigContext = createContext(null);
 49: // --- Provider ---
 50: export const FormConfigProvider = ({ children }) => {
 51: const [state, dispatch] = useReducer(formConfigReducer, initialConfig);
 52:
53:
     return (
```

```
54:
       <FormConfigContext.Provider value={{ state, dispatch }}>
55:
         {children}
56:
       </FormConfigContext.Provider>
57:
     );
58: };
59:
60: // --- Custom Hook ---
61: export const useFormConfig = () => {
     const context = useContext(FormConfigContext);
    if (!context) {
63:
64:
       throw new Error("useFormConfig must be used within a FormConfigProvider");
65: }
66:
     return context;
67: };
```

## ■ File: src\data\formConfig.json

```
______
      "title": "GENOMICS & DIET",
 2:
      "quote": "\"YOU ARE WHAT YOU EAT\" - Victor Lindlahr",
 3:
 4:
      "description": "A right diet is the one that makes you feel happy, keeps you healthy, does not make you f
 5:
     "headerColor": "#16a34a",
 6:
     "colors": {
 7:
        "low": "#16a34a",
 8:
        "medium": "#f59e0b",
        "high": "#dc2626"
 9:
 10:
     },
11:
      "highThreshold": 6,
 12:
      "categories": ["MACRONUTRIENTS", "MEAL PATTERN", "FOOD SENSITIVITIES"],
13:
      "fields": [
14:
          "id": "carb",
 15:
          "label": "Carbohydrate Sensitivity",
16:
17:
          "category": "MACRONUTRIENTS",
          "high": "Maintain carb intake <45%\nFor obesity & IR control",
18:
          "normal": "Maintain carb intake <50%\nBalanced recommendation",
19:
 20:
          "low": "Maintain carb intake <60%\nFor obesity & IR control"
 21:
        },
 22:
        {
          "id": "fat",
23:
          "label": "Fat Sensitivity",
 24:
 25:
          "category": "MACRONUTRIENTS",
 26:
          "high": "Fat intake not to exceed\n15% of total calories",
 27:
          "normal": "Fat intake should be\n20% of total calories",
 28:
          "low": "Fat intake advised up to\n25% of total calories"
 29:
 30:
 31:
          "id": "protein",
          "label": "Protein Requirement",
32:
33:
          "category": "MACRONUTRIENTS",
 34:
          "high": "Protein supplements\nneeded along with dietary\nsource",
 35:
          "normal": "Maintain adequate protein\nthrough balanced diet and\noccasional supplements",
 36:
          "low": "Protein supplements not\nneeded, intake through\ndiet is enough"
37:
        }
38:
     ]
39: }
```

\_\_\_\_\_

## ■ File: src\hooks\useConfigImportExport.js

```
1: import { useCallback } from "react";
2: import { useFormConfig } from "../contexts/FormConfigContext";
3:
4: export const useConfigImportExport = () => {
5: const { state, dispatch } = useFormConfig();
6:
7: const exportConfig = useCallback(() => {
8: const dataStr = JSON.stringify(state, null, 2);
9: const blob = new Blob([dataStr], { type: "application/json" });
10: const url = URL.createObjectURL(blob);
```

```
11:
12:
       const link = document.createElement("a");
13:
      link.href = url;
       link.download = "genomics-form-config.json";
14:
15:
       link.click();
16:
      URL.revokeObjectURL(url);
17:
    }, [state]);
18:
19:
     const importConfig = useCallback(
      (file) => {
20:
         const reader = new FileReader();
21:
22:
         reader.onload = (e) => {
23:
           try {
24:
             const parsed = JSON.parse(e.target.result);
25:
             dispatch({ type: "IMPORT_CONFIG", config: parsed });
26:
             alert("Configuration imported successfully.");
27:
            } catch (err) {
28:
             alert("Error importing config: " + err.message);
           }
29:
         };
30:
31:
         reader.readAsText(file);
32:
        },
33:
       [dispatch]
34:
    );
35:
36:
    return { exportConfig, importConfig };
37: };
```

## ■ File: src\hooks\usePDFGeneration.js

```
______
 1: import { useCallback } from "react";
 2: import { jsPDF } from "jspdf";
 3: import { useFormConfig } from "../contexts/FormConfigContext";
 4: import { hexToRgb } from "../utils/helpers";
 5:
 6: export const usePDFGeneration = () => {
 7: const { state } = useFormConfig();
 8:
 9:
     // Base64 logo (optional)
10: const leftLogoUrl = "/left.png";
11: const rightLogoUrl = "/right.png";
12:
13:
     const generatePDF = useCallback(
14:
       (reportData) => {
15:
         if (!reportData | | reportData.length === 0) {
16:
           alert("No report data found.");
17:
           return;
18:
         }
19:
 20:
         const doc = new jsPDF();
 21:
         const pageHeight = doc.internal.pageSize.height;
 22:
         const pageWidth = doc.internal.pageSize.width;
 23:
          const margin = 10;
 24:
         let y = 20;
 25:
         // -----
 26:
 27:
         // Header section
         // -----
 28:
 29:
         doc.setFillColor(...hexToRqb(state.headerColor));
 30:
         doc.rect(margin, y, pageWidth - margin * 2, 20, "F");
         doc.setTextColor(255, 255, 255);
 31:
 32:
         doc.setFontSize(16);
         doc.setFont(undefined, "bold");
33:
 34:
         doc.text(state.title, pageWidth / 2, y + 13, { align: "center" });
35:
         y += 30;
 36:
         // Quote
37:
38:
         doc.setTextColor(0, 0, 0);
 39:
         doc.setFontSize(11);
         doc.setFont(undefined, "bold");
 40:
 41:
         doc.text(state.quote, pageWidth / 2, y, { align: "center" });
```

```
y += 10;
 42:
 43:
 44:
           // Description
 45:
           doc.setFont(undefined, "normal");
 46:
          doc.setFontSize(10);
 47:
          const descLines = doc.splitTextToSize(
 48:
            state.description,
 49:
            pageWidth - 2 * margin
 50:
 51:
           doc.text(descLines, margin, y);
 52:
          y += descLines.length * 5 + 5;
 53:
 54:
           let currentCategory = null;
 55:
 56:
          reportData.forEach((item, index) => {
 57:
             const { field, score, showHigh, showNormal, showLow } = item;
 58:
 59:
             // Insert page break if needed
             const estimatedFieldHeight = 40; // Rough estimate
 60:
 61:
             if (y + estimatedFieldHeight > pageHeight - 20) {
 62:
              doc.addPage();
              y = 20;
 63:
 64:
 65:
             // Render category title if needed
 66:
            if (field.category && field.category !== currentCategory) {
 67:
 68:
              currentCategory = field.category;
 69:
               doc.setFontSize(12);
              doc.setFont(undefined, "bold");
 70:
 71:
              doc.setTextColor(50, 50, 50);
 72:
              doc.text(currentCategory, margin, y);
              y += 8;
 73:
             }
 74:
 75:
             // Field Label
 76:
 77:
             doc.setFontSize(10);
             doc.setFont(undefined, "bold");
 78:
 79:
             doc.setTextColor(0, 0, 0);
 :08
             doc.text(field.label, margin, y);
 81:
            y += 6;
 82:
 83:
             // Score Circle
 84:
             const circleX = margin + 5;
 85:
             doc.setDrawColor(0);
 86:
             const rgb =
 87:
              score >= state.highThreshold
 88:
                ? hexToRgb(state.colors.high)
 89:
                 : score >= 4
 90:
                 ? hexToRgb(state.colors.medium)
 91:
                 : hexToRgb(state.colors.low);
 92:
 93:
             doc.setFillColor(...rgb);
             doc.circle(circleX, y + 5, 4, "FD");
 94:
             doc.setTextColor(255, 255, 255);
 95:
 96:
             doc.setFontSize(8);
             doc.text(String(score), circleX, y + 6, { align: "center" });
 97:
 98:
 99:
            y += 12;
100:
             // Render matching text
101:
102:
             const renderTextBlock = (label, text, active) => {
103:
              if (!text) return;
104:
               doc.setFontSize(9);
105:
               doc.setFont(undefined, "bold");
106:
              doc.setTextColor(
107:
                active ? 0 : 180,
                active ? 0 : 180,
108:
109:
                active ? 0 : 180
              );
110:
111:
              doc.text(`${label}:`, margin, y);
112:
              y += 5;
113:
114:
               doc.setFont(undefined, "normal");
```

```
115:
              const lines = doc.splitTextToSize(text, pageWidth - 2 * margin);
116:
              lines.forEach((line) => {
117:
                if (y + 6 > pageHeight - 15) {
                  doc.addPage();
118:
119:
                  y = 20;
120:
121:
                doc.text(line, margin, y);
122:
                y += 5;
123:
              });
124:
              y += 2;
            };
125:
126:
            renderTextBlock("HIGH", field.high, showHigh);
127:
            renderTextBlock("NORMAL", field.normal, showNormal);
128:
129:
            renderTextBlock("LOW", field.low, showLow);
130:
            y += 4;
131:
132:
          });
133:
134:
          // Footer (optional logos)
135:
          const addLogos = () => {
136:
           try {
              doc.addImage(leftLogoUrl, "PNG", 10, pageHeight - 20, 30, 10);
137:
138:
              doc.addImage(
139:
                rightLogoUrl,
                "PNG",
140:
141:
                pageWidth - 40,
                pageHeight - 20,
142:
143:
                30,
144:
               10
145:
              );
146:
            } catch (e) {
147:
              console.warn("Logo failed to load. Skipping...");
148:
          };
149:
150:
          addLogos();
151:
152:
          // Save file
153:
         doc.save("genomics-diet-report.pdf");
154:
        },
155:
        [state]
156:
     );
157:
158:
     return { generatePDF };
159: };
```

# ■ File: src\hooks\useReportGeneration.js

```
1: import { useState, useCallback } from 'react';
2: import { useFormConfig } from '../contexts/FormConfigContext';
3: import { isValidScore } from '.../utils/helpers';
4:
5: /**
6: * Hook: useReportGeneration
7: * Transforms form input into structured report data based on score thresholds
8: */
9: export const useReportGeneration = () => {
10: const { state } = useFormConfig(); // Get field config and settings from context
11: const [reportData, setReportData] = useState([]);
12:
13:
     /**
      * generateReport
14:
     * @param {Object} formData - { fieldId: score }
15:
16:
     const generateReport = useCallback((formData) => {
17:
18:
      const processedData = [];
19:
20:
       // Loop through all fields from config
21:
      state.fields.forEach((field) => {
22:
       const rawValue = formData[field.id];
23:
         const score = parseInt(rawValue);
```

```
24:
25:
         if (!isValidScore(score)) {
26:
          return; // Skip invalid scores
27:
28:
29:
         // Determine logic: high / normal / low
30:
         const isHigh = score >= state.highThreshold;
31:
         const isNormal = score >= 4 && score < state.highThreshold;</pre>
32:
         const isLow = score < 4;</pre>
33:
34:
        processedData.push({
         35:
36:
          showHigh: isHigh,
37:
38:
          showNormal: isNormal,
39:
          showLow: isLow,
40:
        });
41:
       });
42:
43:
       // Update state
44:
       setReportData(processedData);
45:
46:
       // Return for immediate use
47:
       return processedData;
48:
    }, [state.fields, state.highThreshold]);
49:
50:
     return { reportData, generateReport };
51: };
```

#### ■ File: src\index.css

```
______
 1: body {
 2: margin: 0;
 3:
     font-family: -apple-system, BlinkMacSystemFont, "Segoe UI", "Roboto", "Oxygen",
       "Ubuntu", "Cantarell", "Fira Sans", "Droid Sans", "Helvetica Neue",
 4:
      sans-serif;
 6: -webkit-font-smoothing: antialiased;
 7:
     -moz-osx-font-smoothing: grayscale;
 8: }
 9:
10: code {
11: font-family: source-code-pro, Menlo, Monaco, Consolas, "Courier New",
12:
      monospace;
13: }
14: @tailwind base;
15: @tailwind components;
16: @tailwind utilities;
```

------

#### ■ File: src\index.js

```
______
 1: import React from 'react';
 2: import ReactDOM from 'react-dom/client';
 3: import './index.css';
 4: import App from './App';
 5: import reportWebVitals from './reportWebVitals';
 6:
 7: const root = ReactDOM.createRoot(document.getElementById('root'));
 8: root.render(
 9: <React.StrictMode>
10:
       <App />
    </React.StrictMode>
11:
12: );
13:
14: // If you want to start measuring performance in your app, pass a function
15: // to log results (for example: reportWebVitals(console.log))
16: // or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
17: reportWebVitals();
```

------

### ■ File: src\logo.svg

\_\_\_\_\_\_

<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 841.9 595.3"><g fill="#61DAFB"><path d="M666.3 296.5c0-32.5"</pre>

\_\_\_\_\_

### ■ File: src\reportWebVitals.js

```
______
 1: const reportWebVitals = onPerfEntry => {
 2: if (onPerfEntry && onPerfEntry instanceof Function) {
     import('web-vitals').then(({ getCLS, getFID, getFCP, getLCP, getTTFB }) => {
 4:
      getCLS(onPerfEntry);
       getFID(onPerfEntry);
 5:
       getFCP(onPerfEntry);
getLCP(onPerfEntry);
 6:
 7:
 8:
       getTTFB(onPerfEntry);
 9:
      });
    }
10:
11: };
13: export default reportWebVitals;
```

------

### ■ File: src\setupTests.js

```
1: // jest-dom adds custom jest matchers for asserting on DOM nodes.
2: // allows you to do things like:
3: // expect(element).toHaveTextContent(/react/i)
4: // learn more: https://github.com/testing-library/jest-dom
5: import '@testing-library/jest-dom';
```

\_\_\_\_\_\_

### ■ File: src\styles\index.css

------

-----

## ■ File: src\utils\constants.js

\_\_\_\_\_\_

-----

#### ■ File: src\utils\helpers.js

```
______
 1: // Convert HEX color to RGB array for jsPDF
 2: export function hexToRgb(hex) {
 3: const result = /^{\#?([a-f\d]{2})([a-f\d]{2})([a-f\d]{2}); i.exec(hex);
 4:
     return result
 5:
      ? [
         parseInt(result[1], 16),
 7:
         parseInt(result[2], 16),
 8:
          parseInt(result[3], 16)
 9:
10:
       : [0, 0, 0];
11: }
12:
13: // Simple validation helper
14: export const isValidScore = (value) => {
15: const num = parseInt(value);
16: return !isNaN(num) && num >= 1 && num <= 10;</pre>
17: };
```

■ File: tailwind.config.js

```
1: /** @type {import('tailwindcss').Config} */
2: module.exports = {
3: content: [
4: "./src/**/*.{js,jsx,ts,tsx}",
5: ],
```

------

```
6: theme: {
7: extend: {},
8: },
9: plugins: [],
10: }
```