

Sustainability Classifier React Project

1. SustainabilityClassifier.jsx

```
import React, { useState } from 'react';
import { base44 } from '@api/base44Client';
import { useQuery, useQueryClient } from '@tanstack/react-query';
import { Button } from '@components/ui/button';
import { Leaf, ArrowLeft, Sparkles } from 'lucide-react';
import { motion, AnimatePresence } from 'framer-motion';

import UploadZone from './UploadZone';
import ClassificationResults from './ClassificationResults';
import HistoryList from './HistoryList';

export default function SustainabilityClassifier() {
  const [selectedImage, setSelectedImage] = useState(null);
  const [imageUrl, setImageUrl] = useState(null);
  const [isAnalyzing, setIsAnalyzing] = useState(false);
  const [currentAnalysis, setCurrentAnalysis] = useState(null);

  const queryClient = useQueryClient();

  const { data: analyses = [] } = useQuery({
    queryKey: ['sustainability-analyses'],
    queryFn: () => base44.entities.SustainabilityAnalysis.list('-created_date',
20),
  });

  const analyzeImage = async (file) => {
    setIsAnalyzing(true);
    setSelectedImage(file);
    try {
      const { file_url } = await base44.integrations.Core.UploadFile({ file });
      setImageUrl(file_url);
      const analysisResult = await base44.integrations.Core.InvokeLLM({
        prompt: `Analyze this image for sustainability aspects. Provide: ...`,
        file_urls: [file_url],
        response_json_schema: { /* schema */ }
      });
      const savedAnalysis = await
base44.entities.SustainabilityAnalysis.create({
  image_url: file_url,
```

```

        ...analysisResult
      });
      setCurrentAnalysis(savedAnalysis);
      queryClient.invalidateQueries({ queryKey: ['sustainability-analyses'] });
    } catch (error) {
      console.error('Analysis error:', error);
    } finally {
      setIsAnalyzing(false);
    }
  };

  const handleReset = () => {
    setSelectedImage(null);
    setImageUrl(null);
    setCurrentAnalysis(null);
  };

  const handleSelectFromHistory = (analysis) => {
    setCurrentAnalysis(analysis);
    setImageUrl(analysis.image_url);
  };

  return (
    <div className="min-h-screen bg-gradient-to-br from-green-50 via-emerald-50
to-teal-50">
      { /* Header, Main, Footer - same as previous code */ }
    </div>
  );
}

```

2. UploadZone.jsx

```

import React, { useCallback, useState } from 'react';
import { Upload, Image as ImageIcon, Sparkles } from 'lucide-react';
import { Button } from '@/components/ui/button';
import { motion } from 'framer-motion';

export default function UploadZone({ onImageSelect, isAnalyzing }) {
  const [dragActive, setDragActive] = useState(false);
  const fileInputRef = React.useRef(null);
  const handleDrag = useCallback(...);
  const handleDrop = useCallback(...);
  const handleFileInput = (e) => { ... };

```

```

    return (
      <motion.div className={`relative border-2 border-dashed rounded-2xl p-12 $
{dragActive ? 'border-green-500 bg-green-50' : 'border-gray-300 hover:border-
green-400 bg-white'}}> ... </motion.div>
    );
  }
}

```

3. ClassificationResults.jsx

```

import React from 'react';
import { motion } from 'framer-motion';
import { Card, CardContent, CardHeader, CardTitle } from '@components/ui/card';
import { Badge } from '@components/ui/badge';
import { Progress } from '@components/ui/progress';
import { Leaf, Recycle, AlertTriangle, CheckCircle2, Lightbulb, Factory,
TreePine } from 'lucide-react';

export default function ClassificationResults({ analysis, imageUrl }) { ... }

```

4. HistoryList.jsx

```

import React from 'react';
import { motion } from 'framer-motion';
import { Card, CardContent } from '@components/ui/card';
import { Badge } from '@components/ui/badge';
import { format } from 'date-fns';
import { History, Leaf } from 'lucide-react';

export default function HistoryList({ analyses, onSelectAnalysis }) { ... }

```

5. JSON Schema: SustainabilityAnalysis

```

{
  "name": "SustainabilityAnalysis",
  "type": "object",
  "properties": {
    "image_url": { "type": "string" },
    "sustainability_score": { "type": "number" },

```

```
"categories": { "type": "array", "items": { "type": "string" } },
"materials": { "type": "array", "items": { "type": "string" } },
"environmental_impact": { "type": "string" },
"recommendations": { "type": "array", "items": { "type": "string" } },
"is_recyclable": { "type": "boolean" },
"carbon_footprint": { "type": "string" }
},
"required": ["image_url", "sustainability_score"]
}
```