KESPO Project Code Documentation

Generated on 2025-12-22 12:39:53

Python Files

File: app.py

```
from flask import Flask, render_template, request, redirect, session
import sqlite3, os
from datetime import datetime

# ================= APP =================
app = Flask(__name__)
app.secret_key = "kespo_secret_key"

BASE_DIR = os.path.dirname(os.path.abspath(__file__))
DB = os.path.join(BASE_DIR, "kespo.db")
UPLOADS = os.path.join(BASE_DIR, "static", "uploads")
os.makedirs(UPLOADS, exist_ok=True)

# ================= DB =================
def db():
    c = sqlite3.connect(DB)
    c.row_factory = sqlite3.Row
    return c

def init_db():
    c = db()
    cur = c.cursor()

    cur.execute("""
    CREATE TABLE IF NOT EXISTS farmer(
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        name TEXT,
        email TEXT UNIQUE,
        phone TEXT,
        password TEXT,
        trust_tier TEXT DEFAULT 'basic'
    )
    """)

    cur.execute("""
    CREATE TABLE IF NOT EXISTS harvest(
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        farmer_id INTEGER,
        quantity TEXT,
        expected_price TEXT,
        image TEXT,
        harvest_date TEXT,
        status TEXT DEFAULT 'pending'
    )
    """)

    cur.execute("""
    CREATE TABLE IF NOT EXISTS deal(
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        harvest_id INTEGER,
        buyer_name TEXT,
        buyer_email TEXT,
        deal_value REAL,
        commission REAL,
        farmer_net REAL,
        status TEXT DEFAULT 'initiated',
        created_at TEXT
    )
    """)

    c.commit()
    c.close()

# ================= ROOT =================
@app.route("/")
def root():
    return redirect("/login")

# ================= AUTH =================
@app.route("/login", methods=["GET","POST"])
def login():
    if request.method == "POST":
        c = db(); cur = c.cursor()
        cur.execute(
            "SELECT * FROM farmer WHERE email=? AND password=?",
            (request.form["email"], request.form["password"])
        )
        f = cur.fetchone()
        c.close()
        if f:
            session.clear()
            session["farmer_id"] = f["id"]
            session["farmer_name"] = f["name"]
            return redirect("/dashboard")
    return render_template("login.html")

@app.route("/logout")
def logout():
    session.clear()
    return redirect("/login")

```

```
# ================= FARMER =================
@app.route("/dashboard")
def dashboard():
    if "farmer_id" not in session:
        return redirect("/login")
    c = db(); cur = c.cursor()
    cur.execute("SELECT trust_tier FROM farmer WHERE id=?", (session["farmer_id"],))
    tier = cur.fetchone()["trust_tier"]
    cur.execute("SELECT COUNT(*) FROM harvest WHERE farmer_id=?", (session["farmer_id"],))
    count = cur.fetchone()[0]
    c.close()
    return render_template("dashboard.html",
        farmer_name=session["farmer_name"],
        trust_tier=tier,
        harvest_count=count
    )

@app.route("/upload", methods=["GET","POST"])
def upload():
    if "farmer_id" not in session:
        return redirect("/login")
    if request.method == "POST":
        img = request.files["image"]
        img.save(os.path.join(UPLOADS, img.filename))
        c = db(); cur = c.cursor()
        cur.execute("""
            INSERT INTO harvest(farmer_id,quantity,expected_price,image,harvest_date)
            VALUES (?,?,?,?,?)
        """, (
            session["farmer_id"],
            request.form["quantity"],
            request.form["price"],
            img.filename,
            request.form["date"]
        ))
        c.commit(); c.close()
        return redirect("/dashboard")
    return render_template("upload_harvest.html")

@app.route("/my harvests")
def my_harvests():
    if "farmer_id" not in session:
        return redirect("/login")
    c = db(); cur = c.cursor()
    cur.execute("SELECT * FROM harvest WHERE farmer_id=?", (session["farmer_id"],))
    rows = cur.fetchall()
    c.close()
    return render_template("my_harvests.html", harvests=rows)
```

```

@app.route("/edit harvest/<int:id>", methods=["GET","POST"])
def edit_harvest(id):
    if "farmer_id" not in session:
        return redirect("/login")

    c = db(); cur = c.cursor()
    cur.execute(
        "SELECT * FROM harvest WHERE id=? AND farmer_id=?",
        (id, session["farmer_id"])
    )
    h = cur.fetchone()

    if not h:
        c.close()
        return "Unauthorized", 403

    if request.method == "POST":
        cur.execute("""
            UPDATE harvest
            SET quantity=?, expected_price=?, harvest_date=?, status='pending'
            WHERE id=?
        """, (
            request.form["quantity"],
            request.form["price"],
            request.form["date"],
            id
        ))
        c.commit(); c.close()
        return redirect("/my harvests")

    c.close()
    return render_template("edit_harvest.html", harvest=h)

@app.route("/inbox")
def inbox():
    if "farmer_id" not in session:
        return redirect("/login")
    c = db(); cur = c.cursor()
    cur.execute("""
        SELECT d.*, h.quantity, h.expected_price
        FROM deal d JOIN harvest h ON d.harvest_id=h.id
        WHERE h.farmer_id=?
        ORDER BY d.created_at DESC
    """, (session["farmer_id"],))
    deals = cur.fetchall()
    c.close()
    return render_template("inbox.html", deals=deals)
```

```

# ================= BUYER =================
@app.route("/buyer")
def buyer():
    min_p = request.args.get("min_price")
    max_p = request.args.get("max_price")
    min_q = request.args.get("min_qty")

    q = """
    SELECT h.*, f.name, f.trust_tier
    FROM harvest h JOIN farmer f ON h.farmer_id=f.id
    WHERE h.status='approved'
    """
    params = []

    if min_p:
        q += " AND CAST(h.expected_price AS INTEGER)>=?"
        params.append(min_p)
    if max_p:
        q += " AND CAST(h.expected_price AS INTEGER)<=?"
        params.append(max_p)
    if min_q:
        q += " AND CAST(h.quantity AS INTEGER)>=?"
        params.append(min_q)

    c = db(); cur = c.cursor()
    cur.execute(q, params)
    rows = cur.fetchall()
    c.close()
    return render_template("buyer_dashboard.html", harvests=rows)

# ================= DEAL =================
@app.route("/initiate deal/<int:hid>", methods=["POST"])
def initiate_deal(hid):
    c = db(); cur = c.cursor()
    cur.execute("SELECT quantity, expected_price FROM harvest WHERE id=?", (hid,))
    h = cur.fetchone()
    value = float(h["quantity"]) * 1000 * float(h["expected_price"])
    commission = value * 0.02
    net = value   commission
    cur.execute("""
        INSERT INTO deal(harvest_id,buyer_name,buyer_email,
        deal_value,commission,farmer_net,status,created_at)
        VALUES (?,?,?,?,?,?, 'initiated',?)
    """, (
        hid,
        request.form["name"],
        request.form["email"],
        value, commission, net,
        datetime.now().strftime("%Y %m %d %H:%M")
    ))
    c.commit(); c.close()
    return redirect("/buyer")

@app.route("/deal/delivered/<int:id>")
def delivered(id):
    if "farmer_id" not in session:
        return redirect("/login")
    c = db(); cur = c.cursor()
    cur.execute("UPDATE deal SET status='delivered' WHERE id=?", (id,))
    c.commit(); c.close()
    return redirect("/inbox")

@app.route("/deal/complete/<int:id>")
def complete(id):
    if not session.get("admin"):
        return redirect("/admin/login")
    c = db(); cur = c.cursor()
    cur.execute("UPDATE deal SET status='completed' WHERE id=?", (id,))
    c.commit(); c.close()
    return redirect("/admin/dashboard")

# ================ ADMIN ================
@app.route("/admin/login", methods=["GET","POST"])
def admin_login():
    if request.method=="POST":
        if request.form["email"]=="admin@kespo.com" and request.form["password"]=="admin123":
            session.clear()
            session["admin"]=True
            return redirect("/admin/dashboard")
    return render_template("login.html")

@app.route("/admin/dashboard")
def admin_dashboard():
    if not session.get("admin"):
        return redirect("/admin/login")
    c = db(); cur = c.cursor()
    cur.execute("""
        SELECT h.*, f.name FROM harvest h
        JOIN farmer f ON h.farmer_id=f.id
        WHERE h.status='pending'
    """)
    harvests = cur.fetchall()
    cur.execute("SELECT * FROM deal WHERE status='delivered'")
    deals = cur.fetchall()
    c.close()
    return render_template("admin_dashboard.html", harvests=harvests, deals=deals)

@app.route("/admin/approve/<int:id>")
def approve(id):
    if not session.get("admin"):
        return redirect("/admin/login")
    c = db(); cur = c.cursor()
    cur.execute("UPDATE harvest SET status='approved' WHERE id=?", (id,))
    c.commit(); c.close()
    return redirect("/admin/dashboard")

# ================ RUN ================
if __name__=="__main__":
    init_db()
    app.run(port=5001, debug=True)

```

File: enerate_docs.py

```

```

File: generate_docs.py

```
import os
from fpdf import FPDF
from datetime import datetime
from pathlib import Path

class SimplePDF(FPDF):
    def header(self):
        self.set_font('Arial', 'B', 12)
        self.cell(0, 10, 'KESPO Project Code Documentation', 0, 1, 'C')
        self.cell(0, 10, f'Generated on {datetime.now().strftime("%Y %m %d %H:%M:%S")}', 0, 1, 'C')
        self.ln(10)

    def chapter_title(self, title):
        self.set_font('Arial', 'B', 12)
        self.set_fill_color(200, 220, 255)
        self.cell(0, 10, title, 0, 1, 'L', 1)
        self.ln(4)

def sanitize_text(text):
    """Replace or remove problematic Unicode characters"""
    if not text:
        return ""

    # Common Unicode replacements
    replacements = {
        'Rs.': 'Rs.',     # Indian Rupee symbol
        ' ': ' ',       # en dash
        '  ': '  ',      # em dash
        '"': '"',       # left double quote
        '"': '"',       # right double quote
        ''': "'",       # left single quote
        ''': "'",       # right single quote
        '...': '...',     # ellipsis
        ' ': ' ',       # horizontal bar
        ' ': ' ',       # narrow no break space
        '< ': '< ',      # leftwards arrow
        ' >': ' >',      # rightwards arrow
        '(up)': '(up)',    # upwards arrow
        '(down)': '(down)',  # downwards arrow
    }

    # Apply replacements
    for k, v in replacements.items():
        text = text.replace(k, v)

    # Remove any remaining non ASCII characters
    return ''.join(c if ord(c) < 128 and c not in '\x00 \x1f\x7f \x9f' else ' '
                 for c in text)

def get_file_content(filepath):
    try:
        with open(filepath, 'r', encoding='utf 8') as file:
            return sanitize_text(file.read())
    except Exception as e:
        try:
            with open(filepath, 'r', encoding='latin 1') as file:
                return sanitize_text(file.read())
        except Exception as e:
            print(f"Error reading {filepath}: {str(e)}")
            return f"[Error reading file: {str(e)}]"

def create_pdf_documentation(project_path, output_file):
    pdf = SimplePDF()
    pdf.set_auto_page_break(auto=True, margin=15)
    pdf.set_font('Courier', '', 8)  # Use built in Courier font

    added_sections = set()

    file_types = {
        'Python': ['.py'],
        'HTML': ['.html'],
        'CSS': ['.css'],
        'JavaScript': ['.js']
    }

    for root, _, files in os.walk(project_path):
        if any(x in root for x in ['venv', '__pycache__', '.git', '.idea', 'node_modules']):
            continue

        for file in sorted(files):
            file_path = Path(root) / file
            rel_path = file_path.relative_to(project_path)
            file_ext = file_path.suffix.lower()

            file_type = None
            for ftype, exts in file_types.items():
                if file_ext in exts:
                    file_type = ftype
                    break

            if file_type:
                try:
                    if file_type not in added_sections:
                        pdf.add_page()
                        pdf.chapter_title(f"{file_type} Files")
                        added_sections.add(file_type)

                    content = get_file_content(file_path)
                    pdf.set_font('Arial', 'B', 10)
                    pdf.cell(0, 10, f"File: {rel_path}", 0, 1)
                    pdf.set_font('Courier', '', 8)

                    lines = content.split('\n')
                    for i, line in enumerate(lines, 1):
                        try:
                            pdf.cell(0, 5, f"{i:4} | {line}", 0, 1)
                        except:
                            continue

                    pdf.ln(5)
                except Exception as e:
                    print(f"Error processing {file_path}: {str(e)}")
                    continue

    try:
        pdf.output(output_file)
        print(f"Successfully created: {output_file}")
        return True
    except Exception as e:
        print(f"Error saving PDF: {str(e)}")
        alt_output = os.path.join(os.path.expanduser('~'), 'Desktop', 'kespo_documentation.pdf')
        try:
            pdf.output(alt_output)
            print(f"PDF saved to alternative location: {alt_output}")
            return True
        except Exception as e2:
            print(f"Failed to save PDF: {str(e2)}")
            return False

if __name__ == "__main__":
    project_path = os.path.dirname(os.path.abspath(__file__))
    output_file = os.path.join(project_path, "kespo_documentation.pdf")

    print(f"Generating documentation PDF to: {output_file}")
    if create_pdf_documentation(project_path, output_file):
        print("Documentation generated successfully!")
    else:
        print("Failed to generate documentation. Please check the error messages above.")
```

KESPO Project Code Documentation

Generated on 2025-12-22 12:39:53

CSS Files

File: static/style.css

```
/* ================= RESET ================= */
* {
  box sizing: border box;
  font family:  apple system, BlinkMacSystemFont, "Segoe UI", Roboto, Arial, sans serif;
}

body {
  margin: 0;
  padding: 0;
  background: #f4f6f9;
  color: #333;
}

/* ================= GLOBAL ================= */
h2, h3 {
  margin: 16px 0;
  color: #2c3e50;
}

a {
  text decoration: none;
  color: #2e7d32;
  font weight: 500;
}

a:hover {
  text decoration: underline;
}

/* ================= LAYOUT ================= */
.container {
  max width: 1100px;
  margin: auto;
  padding: 20px;
}

/* ================= CARD ================= */
.card {
  background: #fff;
  border radius: 12px;
  padding: 16px;
  margin bottom: 18px;
  box shadow: 0 8px 20px rgba(0,0,0,0.05);
}

/* ================= FORMS ================= */
form input,
form textarea,
form button {
  width: 100%;
  padding: 10px 12px;
  margin top: 6px;
  margin bottom: 12px;
  border radius: 8px;
  border: 1px solid #ccc;
  font size: 14px;
}

form input:focus,
form textarea:focus {
  outline: none;
  border color: #2e7d32;
}

form button {
  background: #2e7d32;
  color: white;
  border: none;
  cursor: pointer;
  font size: 15px;
}

form button:hover {
  background: #256428;
}

/* ================= NAV ================= */
.nav {
  margin top: 20px;
}

.nav a {
  display: inline block;
  margin: 8px 0;
}

/* ================= IMAGES ================= */
img {
  border radius: 10px;
  max width: 100%;
}
```

```

/* ================= FILTER BAR ================= */
.filter bar {
  display: flex;
  gap: 10px;
  flex wrap: wrap;
}

.filter bar input {
  flex: 1;
}

/* ================= STATUS TAGS ================= */
.status {
  display: inline block;
  padding: 4px 10px;
  border radius: 20px;
  font size: 12px;
  font weight: bold;
}

.status.pending {
  background: #fff3cd;
  color: #856404;
}

.status.approved {
  background: #d4edda;
  color: #155724;
}

.status.delivered {
  background: #cce5ff;
  color: #004085;
}

.status.completed {
  background: #e2e3e5;
  color: #383d41;
}

/* ================= ADMIN ================= */
.admin actions a {
  display: inline block;
  margin top: 10px;
  padding: 8px 12px;
  background: #1976d2;
  color: white;
  border radius: 6px;
}

.admin actions a:hover {
  background: #155fa0;
}

/* ================= FOOT LINKS ================= */
.footer link {
  margin top: 20px;
  display: inline block;
}

/* ================= MOBILE ================= */
@media (max width: 768px) {
  .container {
    padding: 12px;
  }
}

```

KESPO Project Code Documentation

Generated on 2025-12-22 12:39:53

HTML Files

File: templates/admin_dashboard.html

```
<h2>Admin Panel</h2>

<h3>Pending Harvests</h3>
{% for h in harvests %}
<div>
{{ h.name }}   {{ h.quantity }} Tons
<a href="/admin/approve/{{ h.id }}">Approve</a>
</div>
{% endfor %}

<h3>Delivered Deals</h3>
{% for d in deals %}
<div>
{{ d.buyer_name }}   Rs.{{ d.deal_value }}
<a href="/deal/complete/{{ d.id }}">Release</a>
</div>
{% endfor %}

```

File: templates/admin_login.html

```
<!DOCTYPE html>
<html>
<head>
    <title>KESPO | Admin Login</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>
    <h2>KESPO   Admin Login</h2>

    {% with messages = get_flashed_messages(with_categories=true) %}
        {% if messages %}
            {% for category, message in messages %}
                <div class="alert alert {{ category }}">{{ message }}</div>
            {% endfor %}
        {% endif %}
    {% endwith %}

    <form method="post">
        <div>
            <label for="email">Email:</label>
            <input type="email" id="email" name="email" required>
        </div>
        <div>
            <label for="password">Password:</label>
            <input type="password" id="password" name="password" required>
        </div>
        <button type="submit">Login</button>
    </form>
</body>
</html>

```

File: templates/buyer_dashboard.html

```
<h2>Buyer Marketplace</h2>

<form method="get">
Min Price <input name="min_price">
Max Price <input name="max_price">
Min Qty <input name="min_qty">
<button type="submit">Filter</button>
</form>

{% for h in harvests %}
<div>
  <img src="/static/uploads/{{ h.image }}" width="150"><br>
  {{ h.name }} ({{ h.trust_tier }})<br>
  {{ h.quantity }} Tons @ Rs.{{ h.expected_price }}/KG
  <form method="post" action="/initiate deal/{{ h.id }}">
    <input name="name" placeholder="Your name" required>
    <input name="email" placeholder="Your email" required>
    <button type="submit">Deal</button>
  </form>
</div>
{% endfor %}

<a href="/dashboard">Back</a>

```

File: templates/dashboard.html

```
<h2>Welcome {{ farmer_name }}</h2>
<p>Trust Tier: {{ trust_tier }}</p>
<p>Total Harvests: {{ harvest_count }}</p>
<a href="/upload">Upload Harvest</a><br>
<a href="/my harvests">My Harvests</a><br>
<a href="/inbox">Inbox</a><br>
<a href="/buyer">Buyer Marketplace</a><br>
<a href="/logout">Logout</a>

```

KESPO Project Code Documentation

Generated on 2025-12-22 12:39:53

File: templates/edit_harvest.html

```
<!DOCTYPE html>
<html>
<head>
  <title>Edit Harvest</title>
</head>
<body>

<h2>Edit Harvest Details</h2>

<form method="post">
  <label>Quantity (Tons)</label><br>
  <input type="number" step="0.1" name="quantity"
         value="{{ harvest.quantity }}" required><br><br>

  <label>Expected Price (Rs. / KG)</label><br>
  <input type="number" name="price"
         value="{{ harvest.expected_price }}" required><br><br>

  <label>Harvest Date</label><br>
  <input type="date" name="date"
         value="{{ harvest.harvest_date }}" required><br><br>

  <button type="submit">Save Changes</button>
</form>

<br>
<p><b>Note:</b> Any edit will send this harvest back for admin approval.</p>

<a href="/my harvests">Cancel</a>

</body>
</html>

```

File: templates/inbox.html

```
<!DOCTYPE html>
<html>
<head>
  <title>KESPO | Farmer Inbox</title>
  <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>

<h2>Deal Inbox</h2>

```

```
{% if deals|length == 0 %}
  <p>No deals yet.</p>
{% endif %}

{% for d in deals %}
<div class="card">

  <b>Buyer Name:</b> {{ d.buyer_name }}<br>
  <b>Buyer Email:</b> {{ d.buyer_email }}<br><br>

  <b>Quantity:</b> {{ d.quantity }} Tons<br>
  <b>Price:</b> Rs. {{ d.expected_price }} / KG<br><br>

  <b>Deal Value:</b> Rs. {{ d.deal_value }}<br>
  <b>KESPO Fee (2%):</b> Rs. {{ d.commission }}<br>
  <b>You Receive:</b> Rs. {{ d.farmer_net }}<br><br>

  <b>Status:</b>
  <span class="badge">{{ d.status | upper }}</span><br><br>

  <!   FARMER ACTION   >
  {% if d.status != 'delivered' and d.status != 'completed' %}
    <a href="/deal/delivered/{{ d.id }}" class="btn">
      Mark Delivered
    </a>
  {% endif %}

</div>
{% endfor %}

<br>
<a href="/dashboard">  Back to Dashboard</a>

</body>
</html>

```

File: templates/login.html

```
<!DOCTYPE html>
<html>
<head>
    <title>KESPO | Farmer Login</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>

<h2>KESPO   Farmer Login</h2>
```

```

<form method="post">
    Email:
    <input type="email" name="email" required>

    Password:
    <input type="password" name="password" required>

    <button type="submit">Login</button>
</form>

<br>
<a href="/register">New farmer? Register</a>

</body>
</html>

```

File: templates/my_harvests.html

```
<!DOCTYPE html>
<html>
<head>
    <title>KESPO | My Harvests</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>

<h2>My Harvest Listings</h2>

<a href="/dashboard"><  Back to Dashboard</a><br><br>

{% if harvests|length == 0 %}
<p>You have not uploaded any harvests yet.</p>
{% endif %}

{% for h in harvests %}
<div class="card">
    <img src="/static/uploads/{{ h.image }}" width="200"><br><br>

    <b>Quantity:</b> {{ h.quantity }} Tons<br>
    <b>Price:</b> Rs. {{ h.expected_price }} per KG<br>
    <b>Deal Value:</b> Rs. {{ h.deal_value }}<br>
    <b>Harvest Date:</b> {{ h.harvest_date }}<br>
    <b>Status:</b> {{ h.status | upper }}<br>

    {% if h.rejection_reason %}
        <p style="color:red;">
```

```
            <b>Rejection Reason:</b> {{ h.rejection_reason }}
        </p>
    {% endif %}

    <br>
    <a href="/edit harvest/{{ h.id }}">   Edit</a> |
    <a href="/delete harvest/{{ h.id }}" onclick="return confirm('Delete this harvest permanently?')">
          Delete
    </a>
</div>
{% endfor %}

</body>
</html>

```

File: templates/register.html

```
<!DOCTYPE html>
<html>
<head>
    <title>KESPO | Farmer Registration</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>

<h2>KESPO   Farmer Registration</h2>

<form method="post">
    Name:
    <input type="text" name="name" required>

    Email:
    <input type="email" name="email" required>

    Phone:
    <input type="text" name="phone" required>

    Password:
    <input type="password" name="password" required>

    <button type="submit">Register</button>
</form>

<br>
<a href="/login">Already registered? Login</a>

</body>
```

```
</html>

```

File: templates/send_message.html

```
<!DOCTYPE html>
<html>
<head>
  <title>Contact Farmer</title>
  <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>

<h2>Send Inquiry</h2>

<form method="post">
  Your Name:
  <input name="name" required>

  Email:
  <input type="email" name="email" required>

  Message:
  <textarea name="message" required></textarea>

  <button type="submit">Send</button>
</form>

<a href="/buyer">Back</a>

</body>
</html>

```

File: templates/upload_harvest.html

```
<!DOCTYPE html>
<html>
<head>
    <title>KESPO | Upload Harvest</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>

<h2>KESPO   Upload Pomegranate Harvest</h2>

<form method="post" enctype="multipart/form data">

```

```
    Quantity (in Tons):
    <input type="number" step="0.1" name="quantity" required>

    Expected Price (Rs. per KG):
    <input type="number" name="price" required>

    Harvest Date:
    <input type="date" name="date" required>

    Harvest Image:
    <input type="file" name="image" required>

    <button type="submit">Upload Harvest</button>
</form>

<br>
<a href="/dashboard">Back to Dashboard</a>

</body>
</html>

```