views\admin_tools.py

```python
import streamlit as st
import pandas as pd
import os
from pymongo import MongoClient
from config import MONGO_URI
from communication import send_email_receipt, send_sms
from fee_calculator import generate_fee_record

SESSION_MONTHS = [
    "April", "May", "June", "July", "August", "September",
    "October", "November", "December", "January", "February", "March"
]

def get_db_collections():
    client = MongoClient(MONGO_URI)
    db = client["class_mgmt"]
    return db["students"], db["fee_records"]

def patch_fee_ledgers_streamlit():
    st.subheader("🩺 Fix Fee Ledgers")
    if st.button("🔧 Run Ledger Patch"):
        students_col, fee_col = get_db_collections()
        updated = 0

        for student in students_col.find({}, {"_id": 0}):
            sid = student.get("Student ID")
            fatherless = student.get("Fatherless", False)
            existing_months = {r["month"] for r in fee_col.find({"student_id": sid})}
            missing_months = [m for m in SESSION_MONTHS if m not in existing_months]

            if missing_months:
                full_ledger = generate_fee_record(sid, fatherless)
                patch_docs = [r for r in full_ledger if r["month"] in missing_months]
                fee_col.insert_many(patch_docs)
                updated += 1

        if updated:
            st.success(f"✅ Patched fee records for {updated} students.")
        else:
            st.info("✅ All fee ledgers already complete.")
```

```
41
42  def communication_controls():
43      students_col, _ = get_db_collections()
44
45      st.subheader("📬 Guardian Communication Center")
46      student_map = {
47          f"{s['Student ID']} - {s['Name']}": s
48          for s in students_col.find({}, {"_id": 0, "Student ID": 1, "Name":
    1, "Guardian Email": 1, "Mobile": 1})
49      }
50
51      if not student_map:
52          st.error("⚠️ No student records found.")
53          return
54
55      selected = st.selectbox("🎓 Select Student", ["-- Select --"] +
    list(student_map.keys()))
56
57      if selected == "-- Select --":
58          st.info("👈 Please select a student to continue.")
59          return
60
61      student = student_map.get(selected)
62      if not student:
63          st.error("❌ Selected student not found.")
64          return
65
66      student_id = student.get("Student ID", "")
67      email = student.get("Guardian Email", "")
68      phone = student.get("Mobile", "")
69
70      month = st.selectbox("🗓️ Select Month", SESSION_MONTHS)
71      receipt_file = os.path.join("receipts", f"FEE2025-{student_id}-
    {month}.pdf")
72
73      email_subject = f"Fee Receipt for {month}"
74      email_body = st.text_area("📧 Email Message", f"Dear
    Guardian,\n\nAttached is your official fee receipt for {month}.\n\nBest
    regards,\nSchool Admin")
75      sms_message = st.text_area("📱 SMS Message", f"Dear Guardian, fee for
    {month} has been received. Ref: FEE2025-{student_id}-{month}.")
76      send_both = st.checkbox("📦 Send Both Email and SMS")
77
78      col1, col2 = st.columns(2)
79      with col1:
80          if st.button("📧 Send Email"):
```

```
            if os.path.exists(receipt_file):
                success = send_email_receipt(email, email_subject, email_body, receipt_file)
                st.success("✅ Email sent." if success else "❌ Email failed.")
            else:
                st.warning(f"⚠️ Receipt not found: {receipt_file}")

    with col2:
        if st.button("📱 Send SMS"):
            success = send_sms(phone, sms_message)
            st.success("✅ SMS sent." if success else "❌ SMS failed.")

    if send_both and st.button("🚀 Send Both"):
        sms_success = send_sms(phone, sms_message)
        email_success = os.path.exists(receipt_file) and send_email_receipt(email, email_subject, email_body, receipt_file)

        if sms_success and email_success:
            st.success("✅ Both Email and SMS sent successfully.")
        elif sms_success:
            st.warning("✅ SMS sent, ❌ Email failed.")
        elif email_success:
            st.warning("✅ Email sent, ❌ SMS failed.")
        else:
            st.error("❌ Both dispatches failed.")

def import_students_csv():
    st.subheader("📁 Import Students from CSV")
    uploaded_file = st.file_uploader("Upload Student CSV", type=["csv"])

    if not uploaded_file:
        return

    try:
        df = pd.read_csv(uploaded_file)
        st.write("🔍 Preview of Uploaded Data", df.head())
    except Exception as e:
        st.error(f"❌ Error reading CSV: {e}")
        return

    students_col, _ = get_db_collections()

    clear_first = st.checkbox("⚠️ Clear existing 'students' collection before insert")

```

```
123     if st.button("📥 Import Students"):
124         if clear_first:
125             students_col.delete_many({})
126         students_col.insert_many(df.to_dict(orient="records"))
127         st.success(f"✅ Imported {len(df)} students.")
128 
129 def admin_tools_panel():
130     st.title("🛠️ Admin Tools Dashboard")
131 
132     patch_fee_ledgers_streamlit()
133 
134     with st.expander("📬 Guardian Communication Center"):
135         communication_controls()
136 
137     with st.expander("📁 Import Student Records from CSV"):
138         import_students_csv()
139 
```