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 patch_fee_ledgers_streamlit():
    st.subheader("🩺 Fix Fee Ledgers")
    if st.button("🔧 Run Ledger Patch"):
        client = MongoClient(MONGO_URI)
        db = client["class_mgmt"]
        students = db["students"]
        fee_records = db["fee_records"]
        updated = 0

        for student in students.find({}, {"_id": 0}):
            sid = student["Student ID"]
            fatherless = student.get("Fatherless", False)
            existing_months = {r["month"] for r in fee_records.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_records.insert_many(patch_docs)
                updated += 1

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

def communication_controls():
    client = MongoClient(MONGO_URI)
    db = client["class_mgmt"]
    students = db["students"]

    st.subheader("📬 Guardian Communication Center")

    student_map = {
        f"{s['Student ID']} - {s['Name']}": s
        for s in students.find({}, {"_id": 0, "Student ID": 1, "Name": 1, "Guardian Email": 1, "Mobile": 1})
    }

    selected = st.selectbox("Select Student", list(student_map.keys()))
    student = student_map[selected]
    student_id = student["Student ID"]
    email = student.get("Guardian Email", "")
    phone = student.get("Mobile", "")

    month = st.selectbox("Fee Month", SESSION_MONTHS)
    receipt_file = f"receipts/FEE2025-{student_id}-{month}.pdf"

    email_subject = f"Fee Receipt for {month}"
    email_body = st.text_area("📧 Email Message Preview", f"Dear Guardian,\n\nAttached is your official fee receipt for {month}.\n\nBest regards,\nSchool Admin")
    sms_message = st.text_area("📱 SMS Message Preview", f"Dear Guardian, fee for {month} has been received. Ref: FEE2025-{student_id}-{month}.")
    send_both = st.checkbox("📦 Send Both Email and SMS")

    col1, col2 = st.columns(2)
    with col1:
        if st.button("📧 Send Email"):
            if os.path.exists(receipt_file):
                sent = send_email_receipt(email, email_subject, email_body, receipt_file)
                st.success("✅ Email sent." if sent else "❌ Failed to send email.")
            else:
                st.warning("⚠️ Receipt file not found.")

    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 = False
        if os.path.exists(receipt_file):
            email_success = send_email_receipt(email, email_subject, email_body, receipt_file)

        if email_success and sms_success:
            st.success("✅ Both Email and SMS sent successfully.")
        elif not email_success and not sms_success:
            st.error("❌ Both dispatches failed.")
        else:
            if email_success:
                st.warning("✅ Email sent, ❌ SMS failed.")
            else:
                st.warning("✅ SMS sent, ❌ Email failed.")

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

    if uploaded_file:
        df = pd.read_csv(uploaded_file)
        st.write("🔍 Preview of Uploaded Data", df.head())

        if st.checkbox("⚠️ Clear existing MongoDB 'students' collection first"):
            if st.button("🧹 Confirm Clear and Insert"):
                client = MongoClient(MONGO_URI)
                db = client["class_mgmt"]
                students_collection = db["students"]
                students_collection.delete_many({})
                students_collection.insert_many(df.to_dict(orient="records"))
                st.success(f"✅ Imported {len(df)} students after clearing existing data.")
        elif st.button("📥 Insert Without Clearing"):
            client = MongoClient(MONGO_URI)
            db = client["class_mgmt"]
            students_collection = db["students"]
            students_collection.insert_many(df.to_dict(orient="records"))
            st.success(f"✅ Imported {len(df)} students into MongoDB.")

def admin_tools_panel():
    patch_fee_ledgers_streamlit()

    with st.expander("📬 Open Guardian Communication Center"):
        communication_controls()

    with st.expander("📁 Import Student Records from CSV"):
        import_students_csv()

```