views\fee_view.py

```python
import streamlit as st
import logging
from database import students_collection
from fee_calculator import generate_fee_record

# MongoDB collection for fee records
fee_collection = students_collection.database["fee_records"]

# Configure audit logging
logging.basicConfig(filename="fee_updates.log", level=logging.INFO)

def create_fee_ledger(student):
    """Generate fee ledger and insert records."""
    student_id = student["Student ID"]
    fatherless = student.get("Fatherless", False)
    ledger = generate_fee_record(student_id, fatherless)
    fee_collection.insert_many(ledger)

def fetch_fee_records(student_id):
    """Retrieve fee records for a student."""
    return list(fee_collection.find({"student_id": student_id}, {"_id": 0}))

def update_payment_status(student_name, student_id, selected_months, unpaid_months):
    """Update paid status and log each update."""
    for record in unpaid_months:
        label = f"{record['month']} {record['year']}"
        if label in selected_months:
            fee_collection.update_one(
                {"student_id": student_id, "month": record["month"], "year": record["year"]},
                {"$set": {"paid": True}}
            )
            logging.info(f"{student_name} | Paid: {label}")

def fee_view():
    st.title("💰 Fee Ledger Viewer")

    # Load all students
    students = list(students_collection.find({}, {"_id": 0}))
    student_names = [s["Name"] for s in students]
    selected_name = st.selectbox("Select Student", student_names)
```

```
41
42      student = next((s for s in students if s["Name"] == selected_name),
    None)
43      if not student:
44          st.warning("Student not found.")
45          return
46
47      student_id = student["Student ID"]
48      existing = fee_collection.count_documents({"student_id": student_id})
49      if existing == 0:
50          create_fee_ledger(student)
51
52      records = fetch_fee_records(student_id)
53
54      st.subheader(f"📋 Fee Ledger for {student['Name']}")
55      for record in records:
56          status = "✅ Paid" if record["paid"] else "❌ Unpaid"
57          st.markdown(f"- {record['month']} {record['year']}: ₹
    {record['fee_due']} {status}")
58
59      unpaid_months = [r for r in records if not r["paid"]]
60      if unpaid_months:
61          unpaid_labels = [f"{r['month']} {r['year']}" for r in
    unpaid_months]
62          selected = st.multiselect("Mark Paid Months", unpaid_labels)
63
64          if st.button("Update Payment Status"):
65              update_payment_status(student["Name"], student_id, selected,
    unpaid_months)
66              st.success("✅ Payment status updated!")
67
68
```