import streamlit as st

import re

import numpy as np

from datetime import datetime

import json

# Set page config

st.set\_page\_config(

page\_title="Builders Coach AI",

page\_icon="🏗️",

layout="wide",

initial\_sidebar\_state="expanded"

)

# Clean OCR text function

def clean\_ocr\_text(text):

text = re.sub(r'\(truncated \d+ characters\)\.\.\.', '', text)

text = re.sub(r'0{10,}', '', text)

text = re.sub(r'<PAGE\d+>|<CONTENT\_FROM\_OCR>|</CONTENT\_FROM\_OCR>|</PAGE\d+>', '', text)

text = re.sub(r'<[^>]+>', '', text)

text = re.sub(r'\s+', ' ', text)

return text.strip()

# Book contents (replace with your full content)

@st.cache\_data

def load\_book\_content():

book1\_content = """

[PASTE YOUR FULL MILLION DOLLAR BUILDER CONTENT HERE]

"""

book2\_content = """

[PASTE YOUR FULL MILLION DOLLAR BUILDER 2.0 CONTENT HERE]

"""

corpus = clean\_ocr\_text(book1\_content + "\n\n" + book2\_content)

chunks = [chunk.strip() for chunk in re.split(r'\n{2,}', corpus) if chunk.strip()]

return chunks

# Coaching modules

COACHING\_MODULES = {

'profit': ['margin', 'profit', 'markup', 'cost', 'pricing', 'money'],

'cashflow': ['cash', 'payment', 'forecast', 'debtor', 'invoice'],

'sales': ['quote', 'client', 'conversion', 'tender', 'proposal', 'lead'],

'systems': ['process', 'workflow', 'delegate', 'system', 'manual', 'efficiency'],

'team': ['hire', 'staff', 'employee', 'culture', 'performance', 'management']

}

def identify\_coaching\_module(question):

question\_lower = question.lower()

for module, keywords in COACHING\_MODULES.items():

if any(keyword in question\_lower for keyword in keywords):

return module

return 'general'

# Search function

def search\_chunks(query, chunks, top\_k=3):

query\_words = set(re.findall(r'\w+', query.lower()))

scores = []

for chunk in chunks:

chunk\_words = set(re.findall(r'\w+', chunk.lower()))

score = len(query\_words & chunk\_words) / len(query\_words) if query\_words else 0

scores.append(score)

top\_indices = np.argsort(scores)[-top\_k:][::-1]

return [chunks[i] for i in top\_indices if scores[i] > 0.1]

# Enhanced answer function

def format\_coaching\_response(question, relevant\_content, module):

if not relevant\_content:

return """

🤔 \*\*I couldn't find specific content for that question.\*\*

Try rephrasing your question or ask about:

- Profit margins and pricing

- Cash flow management

- Sales processes

- Team building

- Business systems

\*\*Quick tip from Kurt\*\*: "Focus on the 20% of activities that drive 80% of your results!"

"""

content = relevant\_content[0][:400] + "..." if len(relevant\_content[0]) > 400 else relevant\_content[0]

# Module-specific tips

module\_tips = {

'profit': "🎯 \*\*TARGET\*\*: Minimum 25% gross margin (33% markup)",

'cashflow': "💰 \*\*RULE\*\*: Money in before money out - never finance your jobs",

'sales': "📈 \*\*GOAL\*\*: 90% conversion rate using Kurt's 6-Step Sales Process",

'systems': "⚙️ \*\*FOCUS\*\*: Systemize routine tasks, humanize exceptions",

'team': "👥 \*\*PRINCIPLE\*\*: Hire for attitude, train for skill"

}

return f"""

### 🎯 \*\*COACHING RESPONSE\*\*

\*\*📚 Kurt's Methodology:\*\*

{content}

\*\*✅ Key Action Items:\*\*

1. Apply the specific framework mentioned above

2. Measure your current performance against Kurt's benchmarks

3. Implement systematic improvements

4. Review progress weekly

\*\*{module\_tips.get(module, '💡 \*\*TIP\*\*: Focus on continuous improvement')}\*\*

---

\*Based on Million Dollar Builder methodology by Kurt Hegetschweiler\*

"""

# Calculators

def margin\_calculator():

st.subheader("📊 Margin & Markup Calculator")

col1, col2 = st.columns(2)

with col1:

cost = st.number\_input("Job Cost ($)", min\_value=0.0, value=80000.0, step=1000.0)

price = st.number\_input("Quote Price ($)", min\_value=0.0, value=100000.0, step=1000.0)

with col2:

if cost > 0 and price > cost:

margin = ((price - cost) / price) \* 100

markup = ((price - cost) / cost) \* 100

st.metric("Gross Margin", f"{margin:.1f}%",

f"{'✅ Good' if margin >= 25 else '⚠️ Below Target'}")

st.metric("Markup", f"{markup:.1f}%",

f"{'✅ Good' if markup >= 33 else '⚠️ Below Target'}")

if margin < 25:

target\_price = cost / 0.75 # 25% margin

st.warning(f"💡 \*\*Kurt's Advice\*\*: Increase price to ${target\_price:,.0f} for 25% margin")

# Main app

def main():

# Header

st.title("🏗️ Builders Coach AI")

st.subheader("Your 24/7 coaching assistant based on Kurt's Million Dollar Builder methodology")

# Load content

chunks = load\_book\_content()

# Sidebar

with st.sidebar:

st.header("🎯 Quick Tools")

if st.button("📊 Margin Calculator"):

st.session\_state.show\_calculator = True

st.markdown("---")

st.header("📚 Common Topics")

quick\_questions = [

"How do I improve my profit margins?",

"What's Kurt's 6-step sales process?",

"How do I manage cash flow better?",

"How do I delegate effectively?",

"What should I focus on first?"

]

for q in quick\_questions:

if st.button(q, key=q):

st.session\_state.current\_question = q

# Main content area

col1, col2 = st.columns([2, 1])

with col1:

# Chat interface

if "messages" not in st.session\_state:

st.session\_state.messages = []

# Display chat history

for message in st.session\_state.messages:

with st.chat\_message(message["role"]):

st.markdown(message["content"])

# Handle quick question clicks

if hasattr(st.session\_state, 'current\_question'):

question = st.session\_state.current\_question

delattr(st.session\_state, 'current\_question')

else:

question = st.chat\_input("Ask me anything about building a million-dollar construction business...")

if question:

# Add user message

st.session\_state.messages.append({"role": "user", "content": question})

with st.chat\_message("user"):

st.markdown(question)

# Generate response

with st.chat\_message("assistant"):

with st.spinner("Searching Kurt's methodology..."):

module = identify\_coaching\_module(question)

relevant = search\_chunks(question, chunks)

response = format\_coaching\_response(question, relevant, module)

st.markdown(response)

# Add to chat history

st.session\_state.messages.append({"role": "assistant", "content": response})

with col2:

# Calculator section

if hasattr(st.session\_state, 'show\_calculator') and st.session\_state.show\_calculator:

margin\_calculator()

else:

st.info("💡 \*\*Quick Tip\*\*\n\nKurt's #1 Rule: Focus on profit margin before revenue. A smaller business with 25% margins beats a larger one with 10% margins every time!")

st.markdown("---")

st.subheader("📈 Key Benchmarks")

st.markdown("""

- \*\*Gross Margin\*\*: Minimum 25%

- \*\*Net Profit\*\*: Target 10-15%

- \*\*Sales Conversion\*\*: 70-90%

- \*\*Work Hours\*\*: Maximum 50/week

- \*\*Cash Flow\*\*: Always positive

""")

if \_\_name\_\_ == "\_\_main\_\_":

main()