College Admission Agent (EAMCET + Agentic AI)

This project helps students find colleges in Telangana based on their EAMCET rank and preferred location. It also includes a chatbot that answers admission-related queries using IBM's Granite foundation model (Agentic AI).

Features

- Rank-Based College Recommender
    - Filters based on EAMCET rank (up to 200000)
    - District-wise coverage: Hyderabad, Warangal, Karimnagar, Nizamabad, Nalgonda, Medak, etc.
- Agentic AI Chatbot (IBM Granite)
- Ask: *"What is the admission process for JNTU?"*
- Uses IBM watsonx.ai Granite Model via API
- Optional RAG Module (LangChain) for PDF Q&A

Dataset: `colleges.csv`

This file contains:

- College Name
- Location (Telangana-wide)
- Closing Rank
- Branch (CSE, ECE, IT, AI, etc.)

Designed for EAMCET students up to 2 lakh rank.

How to Run

1. Clone the repo or download ZIP

```python
import streamlit as st
import pandas as pd
from rag_chatbot import ask_ibm_granite
import os

st.set_page_config(page_title="College Admission Agent", layout="centered")

# Initialize session state
if "logged_in" not in st.session_state:
    st.session_state["logged_in"] = False
if "username" not in st.session_state:
    st.session_state["username"] = ""

# Load users
def load_users():
    if os.path.exists("users.csv"):
        return pd.read_csv("users.csv")
    else:
        return pd.DataFrame(columns=["username", "password"])

def save_user(username, password):
    users_df = load_users()
    new_user = pd.DataFrame([[username, password]], columns=["username", "password"])
    updated_users = pd.concat([users_df, new_user], ignore_index=True)
    updated_users.to_csv("users.csv", index=False)

# Load colleges
@st.cache_data
def load_college_data():
    return pd.read_csv("colleges.csv")

colleges_df = load_college_data()

# Sidebar - Login/Register
st.sidebar.title("ųŷ”ђ User Login/Register")

login_tab, register_tab = st.sidebar.tabs(["Login", "Register"])

# Login tab
with login_tab:
    username = st.text_input("Username", key="login_user")
    password = st.text_input("Password", type="password", key="login_pass")
    if st.button("Login"):
        users_df = load_users()
        if ((users_df['username'] == username) & (users_df['password'] == password)).any():
            st.session_state["logged_in"] = True
            st.session_state["username"] = username
            st.success(f"Welcome {username}!")
            st.rerun()  # вњ… FIXED
        else:
            st.error("Invalid credentials. Try again.")

# Register tab
with register_tab:
    new_user = st.text_input("New Username", key="reg_user")
    new_pass = st.text_input("New Password", type="password", key="reg_pass")
    if st.button("Register"):
        users_df = load_users()
        if new_user in users_df["username"].values:
            st.warning("Username already exists. Choose another.")
        elif new_user.strip() == "" or new_pass.strip() == "":
            st.warning("Username and password cannot be empty.")
        else:
            save_user(new_user, new_pass)
            st.success("Registered successfully! Please login.")
            st.rerun()  # вњ… FIXED

# Main App
if st.session_state["logged_in"]:
    st.title("ųђ“ Smart College Admission Agent")
    st.markdown(f"ųʻ¤ Logged in as **{st.session_state['username']}**")

    # Logout button
    if st.button("Logout"):
        st.session_state["logged_in"] = False
        st.session_state["username"] = ""
        st.success("Logged out successfully.")
        st.rerun()  # вњ… FIXED

    # User inputs
    rank = st.number_input("Enter EAMCET Rank:", min_value=0, max_value=200000)
    location = st.selectbox("Preferred Location:", options=colleges_df['District'].unique())
    subject = st.selectbox("Interested Subject:", options=['CSE', 'ECE', 'EEE', 'IT', 'MECH', 'CIVIL'])
    pcm_marks = st.slider("Average PCM Marks (%):", 0, 100, 60)

    if st.button("Get Recommendations"):
        filtered_df = colleges_df[
            (colleges_df['District'] == location) &
            (colleges_df['Branch'] == subject) &
            (colleges_df['ClosingRank'] >= rank)
        ]

        if not filtered_df.empty:
            st.success(f"Found {len(filtered_df)} eligible colleges")
            st.dataframe(filtered_df[['CollegeName', 'District', 'Branch', 'ClosingRank']])
        else:
            st.warning("No colleges found matching your criteria.")

    # Chatbot section
    st.divider()
    st.subheader("ųø– Admission Assistant (IBM Granite Chatbot)")
    user_question = st.text_area("Ask any admission-related question:")
    if st.button("Ask Chatbot") and user_question:
        with st.spinner("Contacting IBM Granite..."):
            response = ask_ibm_granite(user_question)
            st.success("Chatbot Response:")
            st.write(response)

else:
    st.warning("Please login to access the College Admission Agent.")
```

```python
import requests
import os
from dotenv import load_dotenv

load_dotenv()

API_KEY = os.getenv("IBM_API_KEY")
PROJECT_ID = os.getenv("IBM_PROJECT_ID")  # Make sure this is in .env

def get_ibm_token():
    url = "https://iam.cloud.ibm.com/identity/token"
    headers = {"Content-Type": "application/x-www-form-urlencoded"}
    data = {
        "apikey": API_KEY,
        "grant_type": "urn:ibm:params:oauth:grant-type:apikey"
    }
    res = requests.post(url, headers=headers, data=data)
    res.raise_for_status()
    return res.json()["access_token"]

def ask_ibm_granite(user_question):
    token = get_ibm_token()

    url = "https://au-syd.ml.cloud.ibm.com/ml/v2/inference"  # вњ… Correct

    headers = {
        "Content-Type": "application/json",
        "Authorization": f"Bearer {token}",
        "ML-Instance-ID": PROJECT_ID
    }

    body = {
        "model_id": "granite-3-8b-instruct",  # вњ… Change based on your Watsonx deployment
        "input": user_question,
        "parameters": {
            "decoding_method": "greedy",
            "max_new_tokens": 300,
            "temperature": 0.7
        }
    }

    response = requests.post(url, headers=headers, json=body)
    response.raise_for_status()
    return response.json()["results"][0]["generated_text"]
```

```python
import os
import requests
from dotenv import load_dotenv

load_dotenv()

API_KEY = os.getenv("IBM_API_KEY")
PROJECT_ID = os.getenv("IBM_PROJECT_ID")  # You must set this in your .env

def get_ibm_token():
    url = "https://iam.cloud.ibm.com/identity/token"
    headers = {"Content-Type": "application/x-www-form-urlencoded"}
    data = {
        "apikey": API_KEY,
        "grant_type": "urn:ibm:params:oauth:grant-type:apikey"
    }
    response = requests.post(url, headers=headers, data=data)
    response.raise_for_status()
    return response.json()["access_token"]

def ask_ibm_granite(prompt):
    token = get_ibm_token()

    url = "https://au-syd.ml.cloud.ibm.com/ml/v2/inference"  # Correct region

    headers = {
        "Content-Type": "application/json",
        "Authorization": f"Bearer {token}",
        "ML-Instance-ID": PROJECT_ID
    }

    payload = {
        "model_id": "granite-3-8b-instruct",  # Or whatever you're using
        "input": prompt,
        "parameters": {
            "decoding_method": "greedy",
            "max_new_tokens": 300,
            "temperature": 0.7
        }
    }

    response = requests.post(url, headers=headers, json=payload)
    response.raise_for_status()
    return response.json()["results"][0]["generated_text"]

# pµ"Ќ Run test
response = ask_ibm_granite("What is EAMCET?")
print("Generated Answer:\n", response)
```

streamlit
pandas
requests
python-dotenv

IBM_API_KEY=RMdzn4FxFyT_f3WnDO01kot3AYXX60C-alolyOi7mxF2
IBM_PROJECT_ID=67ad752c-7357-4540-884c-22d9f745b938

CollegeName	District	ClosingRank	Branch
JNTU Hyderabad	Hyderabad	15000	CSE
OU College of Engineering	Hyderabad	12000	IT
CVR College of Engineering	Hyderabad	52000	CSE
VNR VJIET	Hyderabad	58000	ECE
CBIT	Hyderabad	40000	AI
KMIT	Hyderabad	60000	CSE
MREC	Hyderabad	100000	EEE
Malla Reddy Engineering College	Hyderabad	85000	CSE
TITS	Karimnagar	110000	EEE
Jyothishmathi Institute	Karimnagar	150000	CSE
VITS	Nizamabad	175000	ECE
Kakatiya Institute of Tech	Warangal	60000	CSE
SR Engineering College	Warangal	70000	ECE
Vaagdevi Engineering College	Warangal	180000	EEE
Swarna Bharathi Institute	Khammam	190000	CSE
Nova Engineering College	Khammam	165000	IT
SITAM	Adilabad	140000	CSE
Trinity College of Engineering	Peddapalli	160000	ECE
Vignan Institute	Medak	170000	AI
Kamala Institute	Siddipet	150000	CSE
Anurag University	Hyderabad	20000	CSM
Mahatma Gandhi Institute	Nalgonda	155000	EEE
Princeton Engineering College	Ranga Reddy	125000	CSE
BVRIT	Medak	68000	ECE

username	password	
admin	admin	12333

username	password
Deekshi	Deekshi@23