```python
import socket
from time import ctime

# Caesar Cipher Functions
def caesar_encrypt(text, shift=3):
    encrypted = ""
    for char in text:
        if char.isalpha():
            shift_base = ord('A') if char.isupper() else ord('a')
            encrypted += chr((ord(char) - shift_base + shift) % 26 + shift_base)
        else:
            encrypted += char
    return encrypted

def caesar_decrypt(text, shift=3):
    return caesar_encrypt(text, -shift)

# File Functions
def load_users(filename="users.dat"):
    users = {}
    try:
        with open(filename, "r") as f:
            for line in f:
                username, encrypted_password = line.strip().split(":")
                users[username] = encrypted_password
    except FileNotFoundError:
        pass  # No users file yet
    return users

def save_user(username, password, filename="users.dat"):
    encrypted_password = caesar_encrypt(password)
    with open(filename, "a") as f:
        f.write(f"{username}:{encrypted_password}\n")

def log_query(country, leader, filename="query_log.txt"):
    with open(filename, "a") as f:
        f.write(f"{ctime()}: {country} -> {leader}\n")

# Authentication
def save_user(username, password, filename="users.dat"):
    encrypted_password = caesar_encrypt(password)
    with open(filename, "a") as f:
        f.write(f"{username}:{encrypted_password}\n")
def load_users(filename="users.dat"):
    users = {}
    try:
        with open(filename, "r") as f:
            for line in f:
                username, encrypted_password = line.strip().split(":")
                users[username] = encrypted_password
    except FileNotFoundError:
        pass  # No users file yet
    return users
def authenticate_user(users):
    print("Login or Register")
    choice = input("Type 'login' to log in or 'register' to create an account: ").strip().lower()

    if choice == "register":
        username = input("Enter a username: ").strip()
        if username in users:
            print("Username already exists. Please try logging in.")
            return authenticate_user(users)
        password = input("Enter a password: ").strip()
        save_user(username, password)
        print("Account created successfully. Please log in.")
        return authenticate_user(users)

    elif choice == "login":
        username = input("Username: ").strip()
        password = input("Password: ").strip()

        # Debugging output for stored and entered passwords
        print(f"Checking username '{username}'...")
        if username in users:
            stored_password = users[username]
            if caesar_decrypt(stored_password) == password:
                print("Login successful!")
                return True
            else:
                print("DEBUG: Password mismatch!")
        else:
            print("DEBUG: Username not found!")

        print("Invalid login. Try again.")
        return authenticate_user(users)
    else:
        print("Invalid choice. Try again.")
        return authenticate_user(users)


# Main Server Code
countries = [
    "Venezuela", "Brazil", "Argentina", "Colombia", "Chile",
    "Peru", "Ecuador", "Uruguay", "Paraguay", "Bolivia",
    "Guyana", "Suriname"
]
leaders = [
    "Maduro", "Lula da Silva", "Fernandez", "Petro", "Boric",
    "Boluarte", "Lasso", "Lacalle Pou", "Pena", "Arce",
    "Ali", "Santokhi"
]

HOST = 'localhost'
PORT = 5000
BUFSIZE = 1024
ADDRESS = (HOST, PORT)

server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
server.bind(ADDRESS)
server.listen()

print("Server is running. Waiting for connection...")

users = load_users()  # Load users from file

while True:  # Outer loop to accept multiple clients
    client, address = server.accept()
    print(f"Connected by {address}")

    if not authenticate_user(users):  # Authenticate user before continuing
        client.close()
        continue
```

```
client.send("Leaders of South American countries".encode())
client.send(bytes(f"{ctime()}\nInput a country, and I'll tell you the leader! ", "ascii"))

while True:  # Inner loop to interact with a single client
    try:
        data = client.recv(BUFSIZE).decode('utf-8')
        if not data:
            print("Client disconnected")
            break

        country = data.strip()
        if country in countries:
            index = countries.index(country)
            leader = leaders[index]
            log_query(country, leader)  # Log successful query
        else:
            leader = "Country doesn't exist!"

        client.send(leader.encode('utf-8'))

    except Exception as e:
        print(f"An error occurred: {e}")
        break

client.close()
```