```
def calculate_target_pbs():
  # Connect to MongoDB
  client = pymongo.MongoClient("mongodb://localhost:27017/")
  database = client["HeartFailure"]
  collection = database["Patients"]

  # Retrieve all patients from the collection
  patients = list(collection.find())

  # Load the trained logistic regression model
  # model = LogisticRegression(C=0.01, max_iter=5000, random_state=0, solver='liblinear')

  # Iterate over each patient and calculate target and PBS
  for patient in patients:
    input_data = [
      int(patient['age']), int(patient['sex']), int(patient['cp']), int(patient['trestbps']), int(patient['chol']),
      int(patient['fbs']), int(patient['restecg']), int(patient['thalach']), int(patient['exang']),
      (patient['oldpeak']), int(patient['slope']), int(patient['ca']), int(patient['thal'])

    ]

    # Convert input data to a numpy array and reshape for prediction
    input_data_as_numpy_array = np.asarray(input_data, dtype=np.float64)
    input_data_reshaped = input_data_as_numpy_array.reshape(1, -1)

    # Perform prediction using the logistic regression model
    prediction = model.predict(input_data_reshaped)

    # Update the patient's target value
    patient["id"] = int(patient["id"])  # Convert the ObjectId to a string
    patient["target"] = int(prediction[0])

    # Calculate PBS as the probability of target class 1
    prediction_proba = model.predict_proba(input_data_reshaped)
    pbs = round(prediction_proba[0][1] * 100, 2)
    patient["PBS"] = pbs

    # Make a POST request to update the patient's data in the database
    url = "http://localhost:5000/update_patient_data"  # Replace with your API endpoint URL
    headers = {"Content-Type": "application/json"}
    response = requests.post(url, json=patient, headers=headers)

    if response.status_code == 200:
      print(f"Patient {patient['_id']} data updated successfully.")
    else:
      print(f"Failed to update data for patient {patient['_id']}.")

schedule.every().day.at("23:00").do(calculate_target_pbs)

# Keep the script running and continuously check the schedule
while True:
    schedule.run_pending()
    time.sleep(1)
```