حل الطلب الرابع:

إدخال بيانات الحساب

تم إدخال معلومات الحساب مباشرة داخل الكود لاستخدامها في استخراج التوصيات.

```
country_input = 'US'
solution_input = 'CRM'
display_top_actions(df, country=country_input, solution=solution_input)
```

استخراج Top 4 Actions حسب البلد والحل

تم استخدام دالة واحدة لاستخراج أفضل أربعة إجراءات حسب:

- **البلد**
- **نوع الحل**
- **البلد ونوع الحل معًا**

```
def get_top_actions(df, country=None, solution=None):
    subset = df.copy()
    if country:
        subset = subset[subset['country'] == country]
    if solution:
        subset = subset[subset['SourceSystem'].str.contains(solution)]
    return subset['types'].value_counts().head(4)
```

عرض النتائج للمستخدم

تم إنشاء دالة لعرض النتائج النهائية للنظام.

```
def display_top_actions(df, country=None, solution=None):

    top_by_country = get_top_actions(df, country=country) if country else None


    top_by_solution = get_top_actions(df, solution=solution) if solution else None


    top_by_country_solution = get_top_actions(df, country=country, solution=solution) if country and solution else None
```

إضافة إجراء جديد وإعادة حساب النتائج

عند إضافة إجراء جديد، يتم تحديث البيانات وإعادة تدريب النموذج وإعادة حساب أفضل أربعة إجراءات.

```
new_action = "Outbound Email"
top_actions, predicted_stage = update_weights_and_retrain(new_action, country=country_input, solution=solution_input, is_lead=1)
```

تحديد نتيجة الرحلة(Win / Loss)

تم استخدام نموذج شجرة القرار لتحديد ما إذا كان الإجراء يؤدي إلى فوز أو خسارة الفرصة.

```
action_type = 'Inbound Call'
print("Predicted outcome (Win/Loss) for the action 'Inbound Call':")
print(predict_win_or_loss(action_type, is_lead=1))
```