# 新增圖表

    def add\_view(self, request, form\_url="", extra\_context=None):

        # 檢查用戶是否有新增權限

        if not self.has\_add\_permission(request):

            raise PermissionDenied  # 如果沒有權限，拋出例外

        # 如果是POST請求，表示用戶提交了新增表單

        if request.method == "POST":

            form = self.get\_form(request)(request.POST)  # 填充表單

            if form.is\_valid():  # 檢查表單內容是否合法

                obj = form.save()  # 儲存新增的圖表資料

                self.message\_user(request, "圖表新增成功", messages.SUCCESS)  # 顯示新增成功訊息

                return HttpResponseRedirect(self.get\_object\_url(obj))  # 新增後重定向到該圖表的頁面

        else:

            # 如果是GET請求，顯示空白表單

            form = self.get\_form(request)()

        context = {

            "title": \_("Add %s") % self.model.\_meta.verbose\_name,

            "form": form,

            "opts": self.model.\_meta,

        }

        context.update(extra\_context or {})

        return TemplateResponse(request, "admin/change\_form.html", context)

    # 刪除圖表

    def delete\_view(self, request, object\_id, extra\_context=None):

        # 檢查用戶是否有刪除權限

        if not self.has\_delete\_permission(request, None):

            raise PermissionDenied  # 如果沒有權限，拋出例外

        obj = self.get\_object(request, unquote(object\_id))  # 根據ID獲取圖表物件

        if obj is None:

            return self.\_get\_obj\_does\_not\_exist\_redirect(request, self.opts, object\_id)  # 如果找不到圖表物件，重定向

        # 確認刪除操作

        if request.method == "POST":

            obj.delete()  # 執行刪除操作

            self.message\_user(request, "圖表已刪除", messages.SUCCESS)  # 顯示刪除成功訊息

            return HttpResponseRedirect(self.get\_changelist\_url())  # 刪除後重定向回圖表列表頁面

        # 如果是GET請求，顯示刪除確認頁面

        context = {

            "title": \_("Are you sure you want to delete %s?") % force\_text(obj),

            "object": obj,

            "opts": self.model.\_meta,

        }

        context.update(extra\_context or {})

        return TemplateResponse(request, "admin/delete\_confirmation.html", context)

    # 修改圖表

    def change\_view(self, request, object\_id, form\_url="", extra\_context=None):

        # 檢查用戶是否有修改權限

        if not self.has\_change\_permission(request, None):

            raise PermissionDenied  # 如果沒有權限，拋出例外

        obj = self.get\_object(request, unquote(object\_id))  # 根據ID獲取圖表物件

        if obj is None:

            return self.\_get\_obj\_does\_not\_exist\_redirect(request, self.opts, object\_id)  # 如果圖表物件不存在，重定向

        # 處理表單提交（POST請求）

        if request.method == "POST":

            form = self.get\_form(request)(request.POST, instance=obj)  # 填充表單並與圖表物件綁定

            if form.is\_valid():  # 檢查表單是否有效

                obj = form.save()  # 儲存修改後的圖表資料

                self.message\_user(request, "圖表修改成功", messages.SUCCESS)  # 顯示修改成功訊息

                return HttpResponseRedirect(self.get\_object\_url(obj))  # 修改後重定向回該圖表頁面

        else:

            # 如果是GET請求，顯示修改表單

            form = self.get\_form(request)(instance=obj)

        context = {

            "title": \_("Change %s") % force\_text(obj),

            "object": obj,

            "form": form,

            "opts": self.model.\_meta,

        }

        context.update(extra\_context or {})

        return TemplateResponse(request, "admin/change\_form.html", context)

主題 AI 夢境分析儀表板，幫助使用者透過 AI 解析夢境，獲得心理建議與壓力分析，進而更了解自己的內在情緒與壓力來源。

解決的問題：很多人會好奇自己夢到的內容是否與心理狀態有關，但不知道如何解釋，於是系統可以幫忙進行解夢，並透過AI進行個人化分析，為不同使用者提供個別建議。

主要功能：

使用者輸入夢境內容（AI 讀取並分析夢境的關鍵字）。例如「鬼魂」、「被追趕」。

AI 模型解析夢境（透過 NLP（自然語言處理）找出常見夢境象徵與情緒）。

心理分析與壓力指數（根據夢境內容提供心理狀態評估）。

個人化建議，根據夢境內容（AI 給出舒緩壓力、改善睡眠的建議）。

夢境趨勢儀表板，追蹤使用者過去夢境的變化，顯示壓力變化趨勢，幫助使用者發現規律