ALCPT
線上測驗系統

113年班專案實作發表

04.29.2024

國防大學 | 管理學院 | 資訊管理學系

指導老師：袁葆宏老師

目錄

01 系統緣起

系統緣起

- 有鑑於提升各軍事院校學生的英語聽讀能力，每年各軍事院校會有ALCPT (American Language Course Placement Test)測驗，評量各校學生英文能力。

- 本院為加強學生英文能力，訂定每週進行ALCPT小考及練習，而校部會於每月進行ALCPT模擬測驗並於測驗後寄發個人的測驗狀況給學生。

02 系統目的

系統目的

- 為提供本院學生更多的個人練習機會及更便捷的平台，同時讓老師及隊職官能夠有效的掌握同學的練習狀況。因此開發此ALCPT線上練習平台供學生自主練習。
- 本平台亦提供模擬測驗的功能，提供老師、隊職官檢閱學生的測驗成績狀況，學生也可藉此調整自身學習的方針與措施。本平台持續精進原先功能並增加新功能，以提高使用者的使用意願，期能提升本院全體學生ALCPT的成績之外並增強英文聽讀的能力。

03 專案組織

專案組織

指導老師

袁葆宏 老師

專案經理

林柔妤

程式設計、資料庫

高昀、張語瑄

資料庫維護、文件

田德御、段懷智

系統測試、文件

王名碩、陳彥瑋

04 系統分析與設計

現階段系統需求

1. 重構程式碼，使系統更易維護並提高重用性
2. 透過上線及註冊人數，隨時了解系統使用率
3. 刪去重複題目並新增題目，豐富題庫
4. 檢視並修正系統架構設計，期能提升運作效率
5. 改善系統使用者介面，增強使用體驗與可及性

本系統區分為六種角色
每個使用者可同時具備多重角色

1

系統管理員
主要功能為管理使用者單位、類別等資料

2

考試管理員
管理模擬測驗、考卷
及受測者群組相關資料

3

題庫管理員
管理題庫內試題，
審核由題庫操作員新增的試題

4

題庫操作員
新增題目及送審給題庫管理員

5

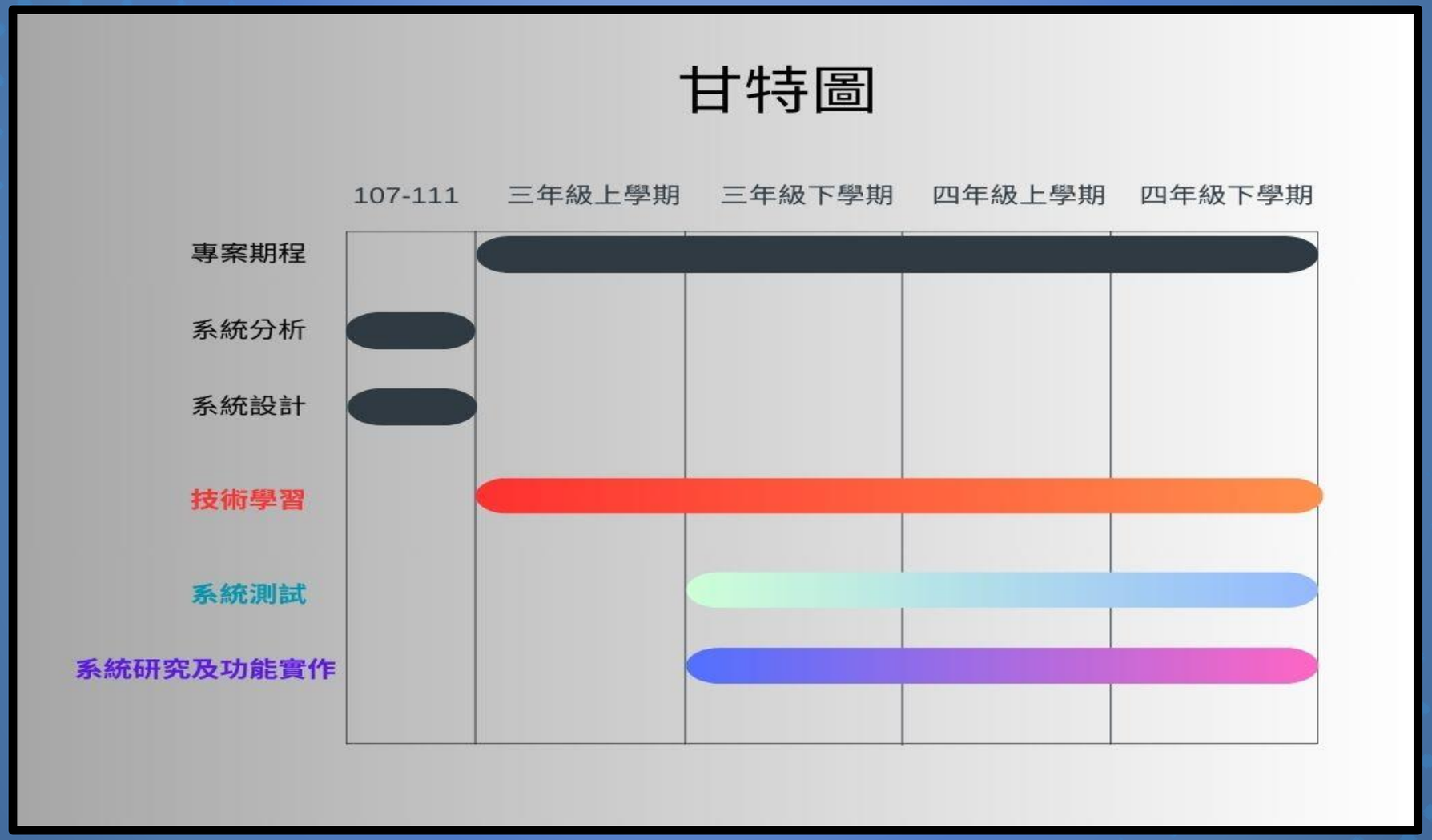
成績檢閱者
檢閱模擬測驗受測狀況
及了解受測者考試學習狀況

6

受測者
可實施各項練習、模擬測驗、
觀看歷次模擬測驗成績

05 開發期程

甘特圖


甘特圖
107-111
三年級上學期
三年級下學期
四年級上學期
四年級下學期
專案期程
系統分析
系統設計
技術學習
系統測試
系統研究及功能實作

ER model介紹

實體	alcpt_onlinestatus	
功能	儲存使用者的上線狀態	
關聯	alcpt_user	
屬性	id(主鍵)	online_status

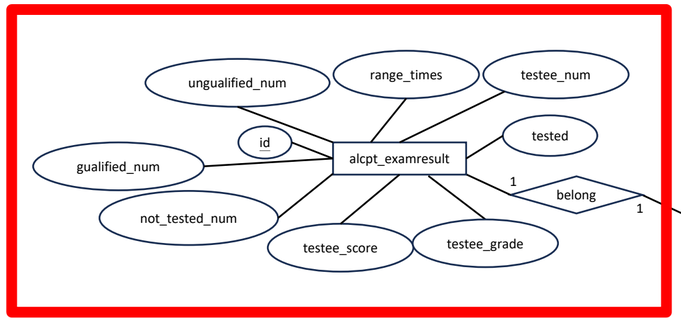
實體	alcpt_locationurl		
功能	儲存類別名稱及url		
關聯	無		
屬性	id(主鍵)	form_class	url_class

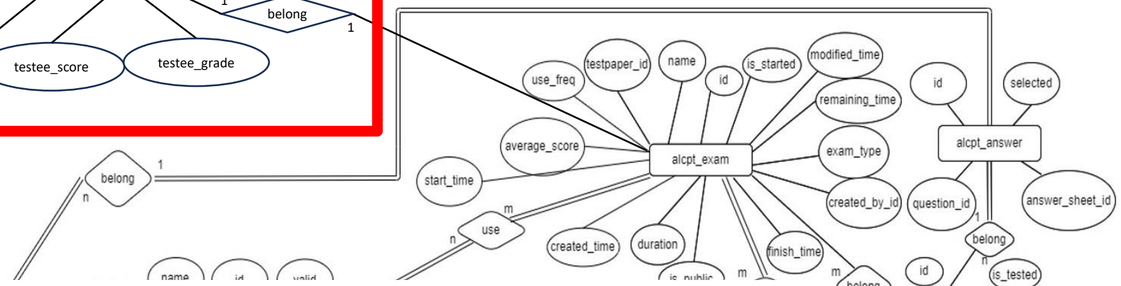
實體	alcpt_examresult		
功能	儲存模擬鑑測中統整後的考試結果		
關聯	alcpt_exam		
屬性	id(主鍵)		range_times
	unqualified_num		qualified_num
	testee_num		not_tested_num
	tested	testee_score	testee_grade

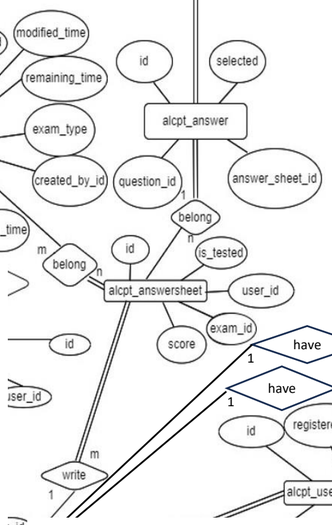
實體	alcpt_scorerecord	
功能	儲存使用者在某類型測驗 (模擬鑑測、聽力、閱讀)中 合格及不合格次數	
關聯	alcpt_user	
屬性	id(主鍵)	exam_type
	qualified_times	unqualified_times

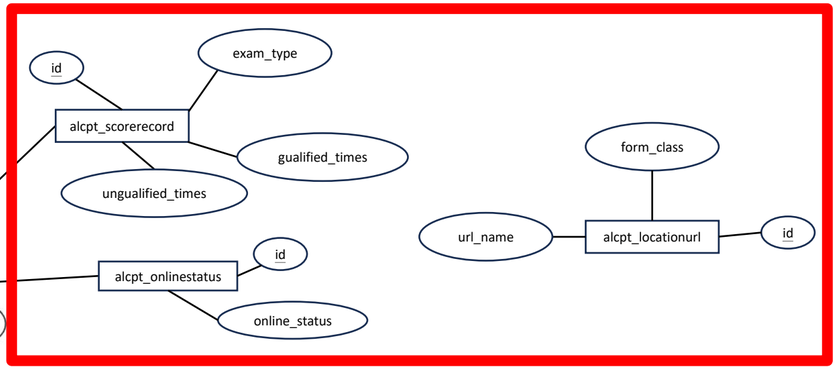
ER model

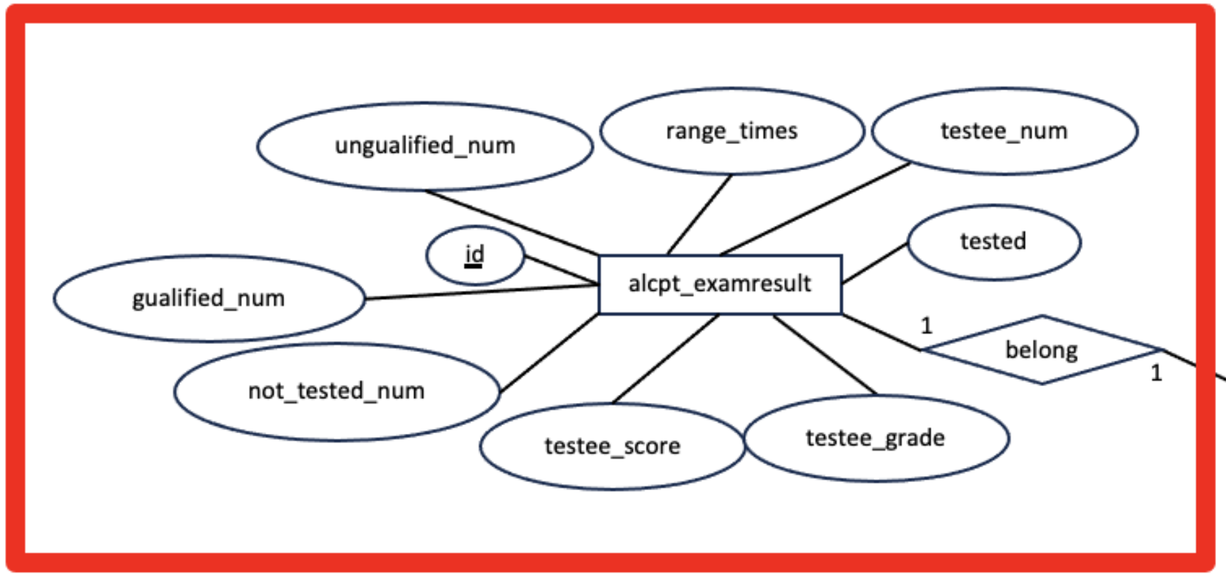
紅框標示為本次新增之系統功能

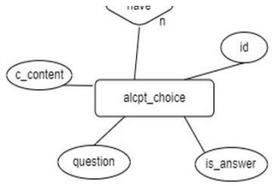


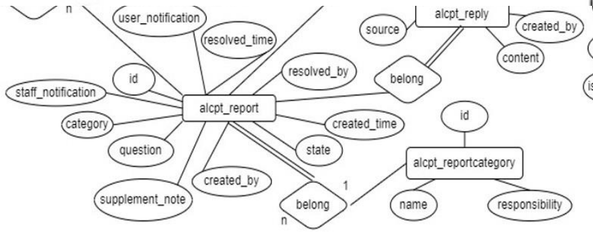


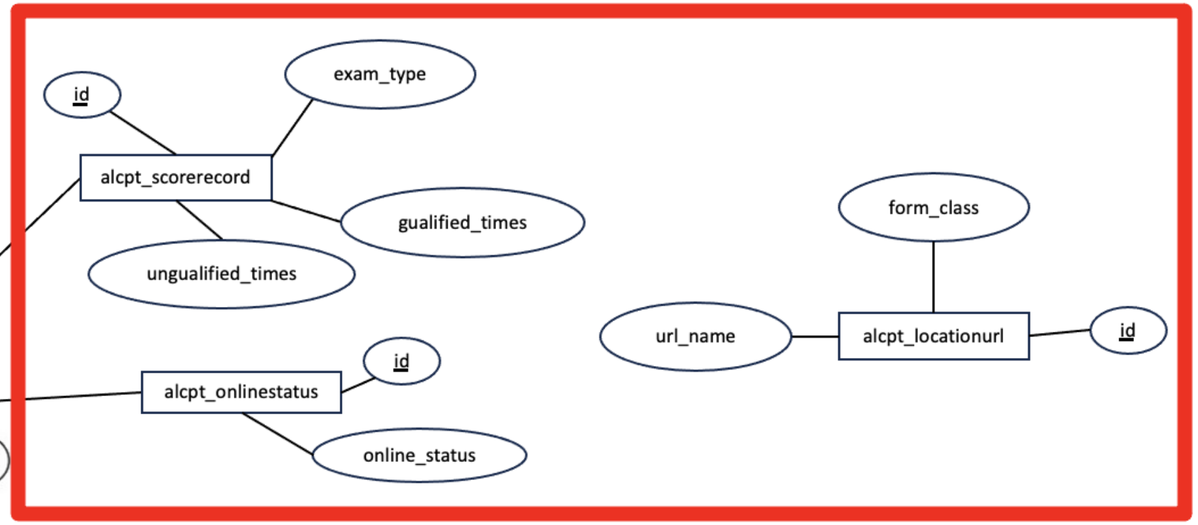












06 系統環境與開發工具

系統環境與開發工具

開發語言

- Python3.10
- HTML5
- CSS2.1
- JavaScript1.8

資料庫

- MySQL8.0.19

前端設計工具

- Jinja2
- Bootstrap4.5.2
- Font Awesome5.15

版本控制

- Git
- SourceTree
- Github

Web框架

- Django3.2

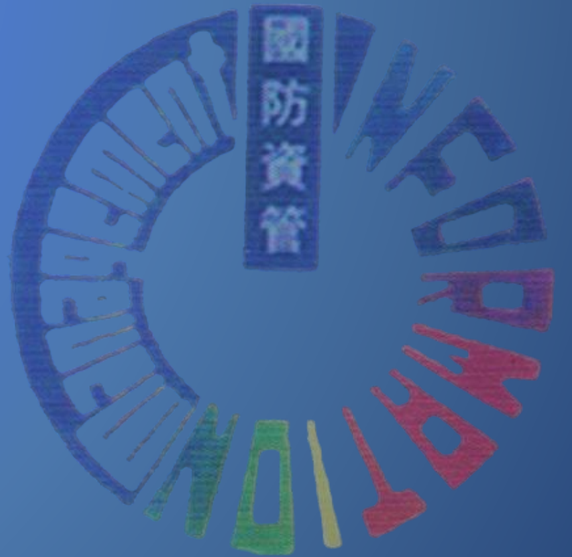
程式編輯器

- PyCharm
- VS Code

07 完成功能

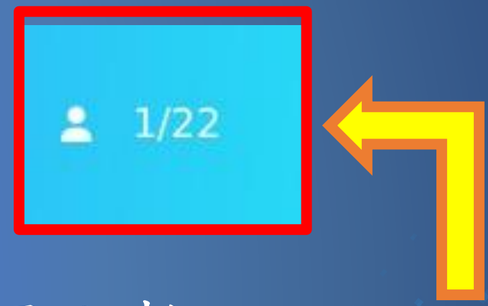
完 成 功 能

① 網頁顯示註冊暨線上人數
② 以Class Based View (CBV)重構程式碼
③ 頁數正常點選及更動
④ 重新確認題目，並將重複題目刪去
⑤ 考試起始時間設立防呆機制
⑥ 頁面跳轉錯誤排除
⑦ 判斷試卷是否已被使用
⑧ 優化成績計算方式
⑨ 優化歷次成績顯示方式
⑩ 各操作新增輔助介面

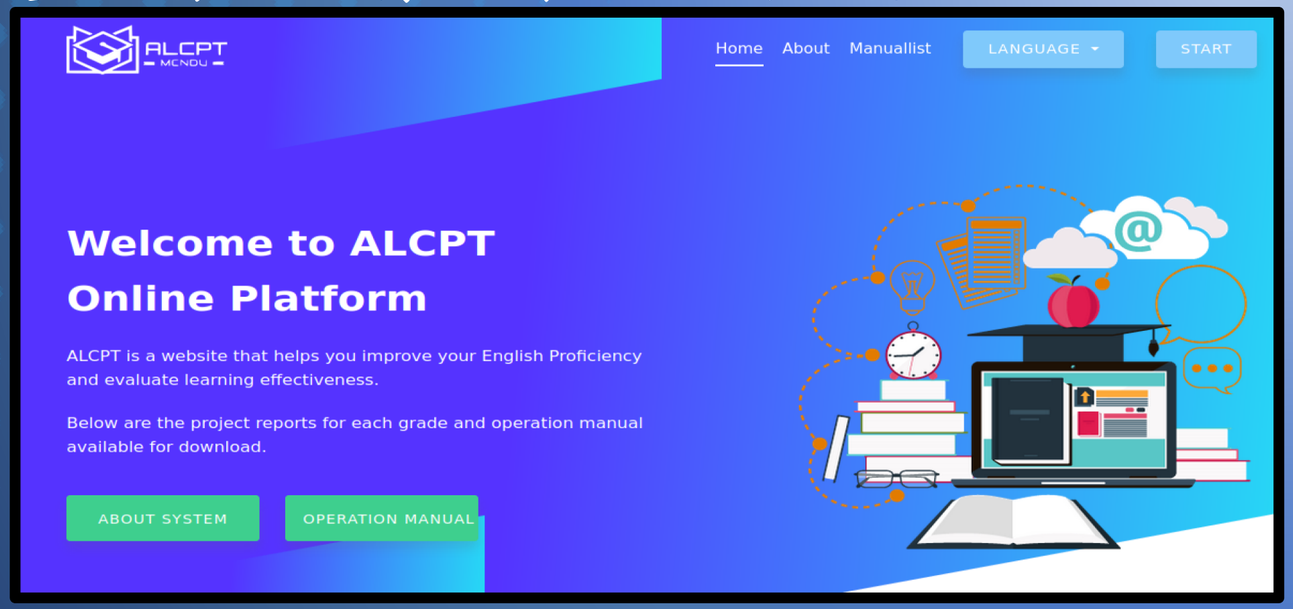


網頁顯示註冊暨上線人數

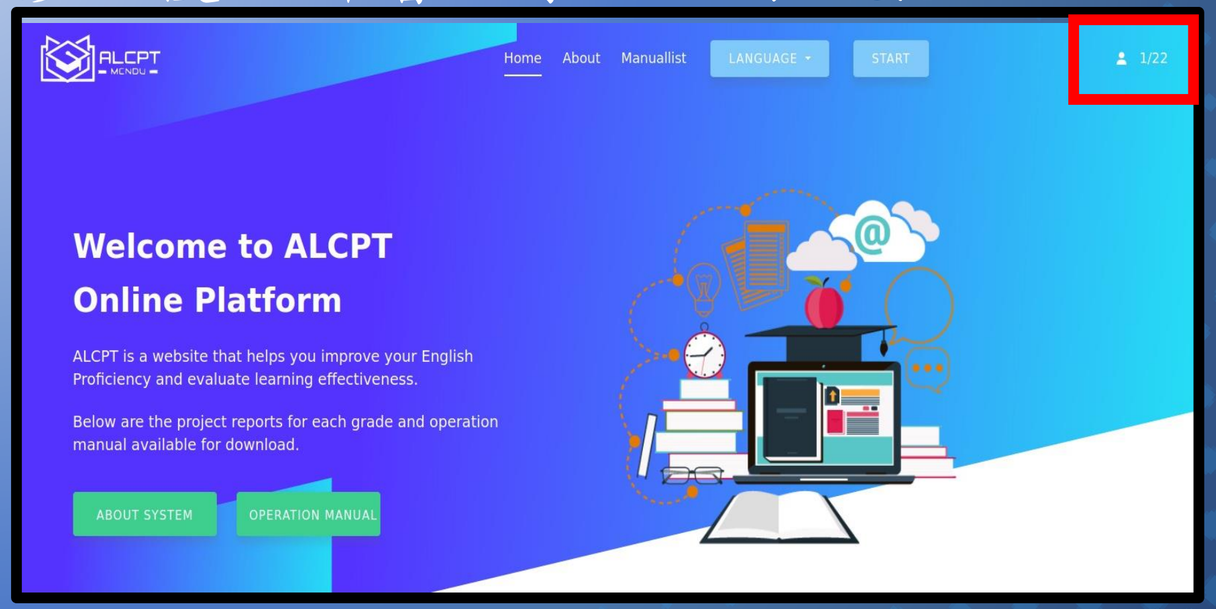
- 為了解系統使用率及供大家隨時了解使用人數，因而建立此功能。



修改前：沒有上線及註冊人數



修改後：新增上線及註冊人數



以ClassBasicView(CBV)重構程式碼

- 為了顯示註冊及上線人數的功能，需在原系統中每個相關的view function新增相同的程式碼，在系統維護及修改上造成極大的困擾。
- Django提供class basic view (CBV)，以物件導向的方式設計系統，使程式碼容易維護並且提高重用性。
- 以CBV重構系統中所有相關的view function。在父類別中處理計算及顯示上線及註冊人數，再以子類別繼承實現個別的功能，以避免重複撰寫相同功能的程式碼，未來若需要新增或修改共同功能(父類別)也會相對容易。

以ClassBasicView(CBV)重構程式碼

➢ View function：

```
@login_required
 def report(request):
    if request.method == 'POST':
       try:
          category = ReportCategory.objects.get(id=
                        int (request.POST.get ('category',)))

       except ObjectDoesNotExist:
           messages.error(request, 'Category does not exist, category name:
           { }'.format(category))

            return render(request, 'report/report.html', locals( ))

            return redirect('report_list')
       else:
           return render(request, 'report/report.html', locals( ))
```

➢ CBV：

```
@method_decorator(login_required,name='get')
class ReportCreate(View,OnlineUserStat):

  template_name = 'report/report.html'

   def do_content_works(self,request):
      return dict(categories = ReportCategory.objects.all())

   def post(self,request):
      try:
          category = ReportCategory.objects.get
          (id=int(request.POST.get('category',)))

      except ObjectDoesNotExist:
          messages.error(request, 'Category does not exist, category name:
          { }'.format(category))
          return dict(categories = ReportCategory.objects.all())
                      ...
      return redirect('report_list')
```

以ClassBasicView(CBV)重構程式碼

➢ View function：

父類別

```
class OnlineUserStat:
    template_name = ' '

    def get(self,request,*args,**kwargs):
        online_num = OnlineStatus.objects.filter(online_status=True).count( )
        reg_num = len(User.objects.all())
        contents = {'reg_num':reg_num, 'online_num':online_num}

        contents_dict =  self.do_content_works(request,*args,**kwargs)
         #do_content_works( ) return dict. if not do anything return { }.

        contents.update(contents_dict)

        return render(request, self.template_name, contents)
```

➢ CBV：

```
@method_decorator(login_required,name='get')
class ReportCreate(View,OnlineUserStat):

    template_name = 'report/report.html'

    def do_content_works(self,request):
        return dict(categories = ReportCategory.objects.all())

    def post(self,request):
        try:
            category = ReportCategory.objects.get
            (id=int(request.POST.get('category',)))

        except ObjectDoesNotExist:
            messages.error(request, 'Category does not exist, category name:
            { }'.format(category))
            return dict(categories = ReportCategory.objects.all())
                        ...
        return redirect('report_list')
```

頁數正常點選及更動

- 問題：點選試題列表下頁按鈕時，頁面無法正常更改至選取頁數，且會跳回第一頁。

- 解決方法：點選下頁按鈕時，頁數更改至正確數值，讓使用者知道目前頁數。

考試起始時間設立防呆機制

- 問題：原系統對於考試起始時間無設立防呆機制，考試設定時間早於系統當下時間仍可建立模擬測驗，造成無效考試。

- 解決方法：設立防呆機制，防止考試管理員新增模擬測驗時設定的考試時間早於當下時間。

頁面跳轉錯誤排除

- 問題：考試管理員欲刪除考試時如點選cancel按鈕會跳到錯誤頁面。

- 解決方法：更改exam_create.html中url指向的位址。

判斷試卷是否已被使用

- 問題：使用過的試卷仍可被更改考試內容（如答案選項），將造成該份考卷無效。

- 解決方法：testpaper_unvalid函數新增判斷「是否使用」的功能，若該試卷已被使用，則試卷將不可再修改。

優化成績計算方式

- 問題：受測者每一次看自己成績及相關統計資料時，系統都會重新計算該次考試成績。

- 解決方法：將成績存入table，使受測者查看成績時，不再需要重新計算成績。

優化成績計算方式

```
#統整user practices/all testees exam的結果
class IntegrateTestResults:

    #更新user在某類型(exam_type)考試中合格次數
    def score_records(self, user, exam_type):
            score_record = ScoreRecord.objects.get(user=user,
exam_type=exam_type)
            ...

    #record Exam result
    def exam_results(self, exam, score):
            exam_result =  ExamResult.objects.get(exam=exam)
            ...

     def grade(score, breakpoints=[60,70,80,90], grades='FDCBA'    ):
    #將分數劃分等級
             i = bisect.bisect(breakpoints, score)
             return grades[i]
             ...
```

```
class ExamScoreDetail(View,OnlineUserStat):
        template_name = 'viewer/exam_score_detail.html'

        def do_content_works(self,request,exam_id):
            try:
                exam = Exam.objects.get(id=exam_id)
                exam_result = ExamResult.objects.get(exam=exam)
                testees = exam.testeeList.all( )
                ...

                context={'exam':exam,
                  'testeeData':testeeData,
                  'testee_number':exam_result.testee_num,
                  'testee_not_tested':exam_result.not_tested_num,
                  'qualified':exam_result.qualified_num,
                  'unqualified':exam_result.unqualified_num,
                  'pie_chart':pie_chart,
                  'bar_chart':bar_chart}

                  return context

            except ObjectDoesNotExist:
                  messages.error(request, 'Exam does not exist, exam id -
{ }'.format(exam_id))
                  return redirect('exam_score_list')
```

優化歷次成績顯示方式

- 問題：原長條圖計次方式無法讓受測者知悉進步幅度。

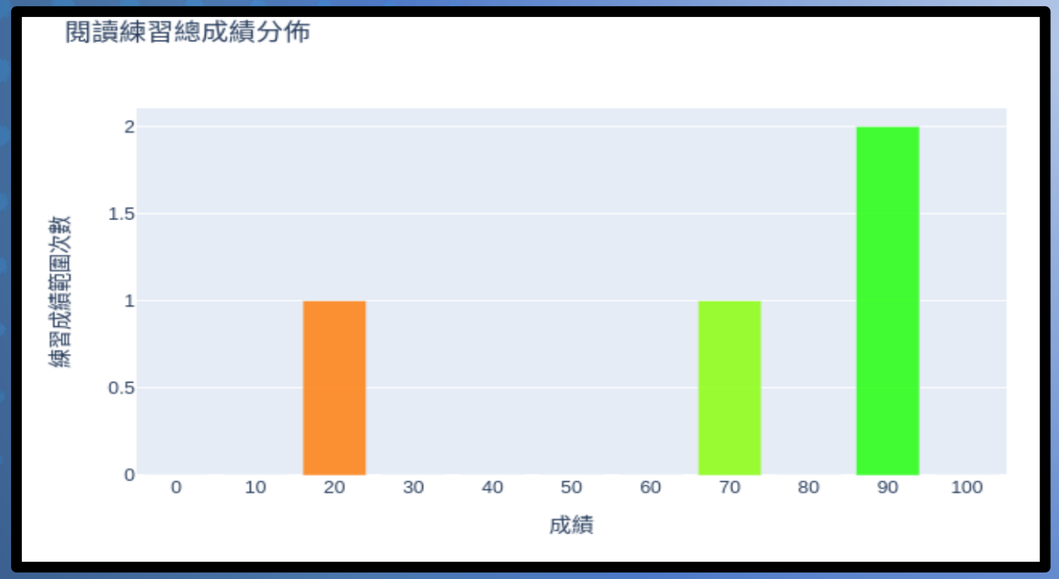
- 解決方法：建立成績曲線圖，提升系統效能，使受測者快速掌握進步幅度，進而改善自身學習方針。

優化歷次成績顯示方式

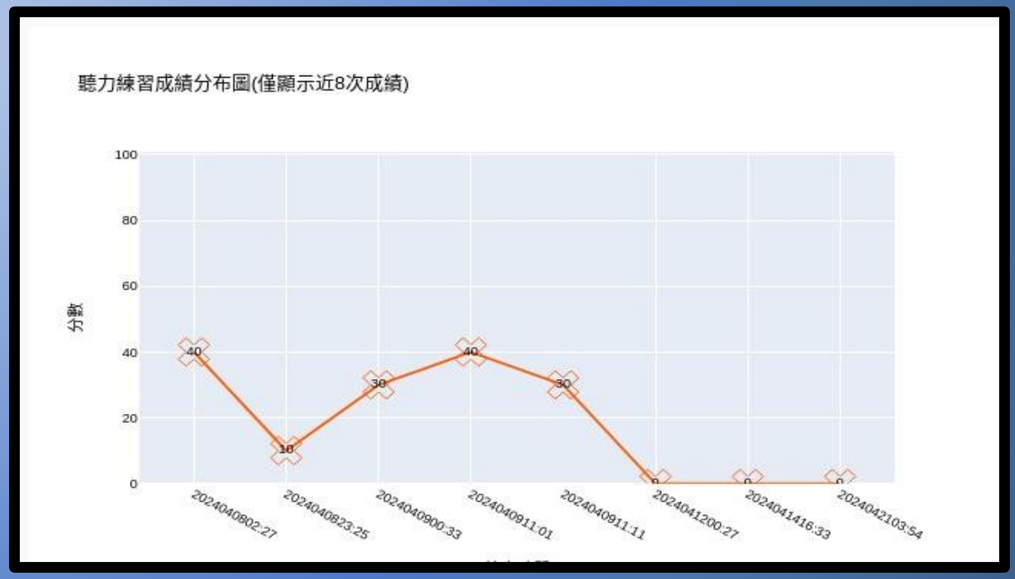
無法分辨歷次成績是否有提高！！！

透過曲線圖更能瞭解學習成效！！！

修改前：長條圖計次方式

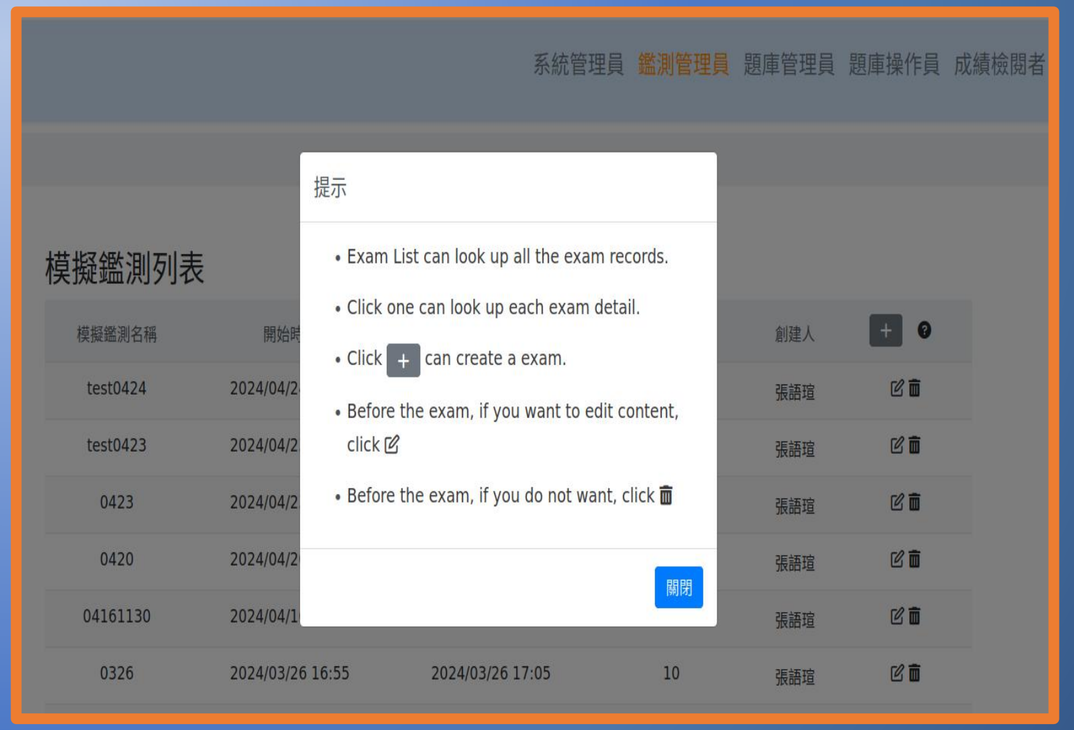


修改後：曲線圖方式呈現歷次成績



各操作新增輔助介面

- 為了提供系統使用者更完善的使用體驗，在每個操作的旁邊新增？按鈕，提供初次使用的人員更快熟悉系統。



08 待檢討事項及開發功能

檢討事項及待開發功能

項次	檢討事項&待開發功能	項次	檢討事項&待開發功能
1	cookie (記錄使用者設定，如：自動登入、中英語言設定等事項)	5	英聽檔逐字稿(方便受測者受測後檢討)
2	模擬考及聽力練習音檔重複問題 (正式鑑測時聽力只播放一次，但目前題目可重複播放)	6	過濾及刪除題目功能 1.過濾出重複的題目，並將該題刪除 (如該題已被使用於鑑測或練習卷中，則不可刪除，因為回看題卷時會少題目) 2.避免系統把重複的題目，出現在同一份試卷中
3	將收藏的題目生成試卷，供受測者反覆練習	7	修改題目正確解答時，新增防呆機制 (跳出確認視窗，並列出修改前、修改後的選項)
4	列出鑑測特定成績範圍的受測者 (方便檢閱者以成績篩選受測者)	8	修改題目正確解答時，新增防呆機制 (跳出確認視窗，並列出修改前、修改後的選項)

09 系統展示

10 結論

結論

本專案是由61期的學長承接至67期的我們。在持續精進改善專案的過程中，我們經歷了許多阻礙。雖然無法解決所有問題，但齊心協力解決困難的過程著實是一大收穫。這是我們第一次參與專案系統的開發，因缺乏實作經驗，也花費較多時間在研讀及練習相關技術上，故在構思與使用便利性上仍有待加強。最後我們也會將此專案傳承給68期的學弟妹，期望這項專案能開發的愈加完善，未來可以正式上線，供管院的學生使用。