國立臺中教育大學資訊工程學系

112級資訊專題期末報告

題目：基於DevOps之程式作業品質測試分析系統

組員：

ACS108122 賴冠銘

ACS108118 巫信緯

ACS108102 許弘麒

執導教授：  
徐國勛教授

目錄

(一)專題內容製作動機與預期目標

(二)專題整體時程與進度評估

(三) 執行情形與討論

(四) 後續工作規劃與調整

(五) 經費使用情形

(六) 參考文獻

一、專題內容製作動機與預期目標  
在大一程式設計這堂課程中，使用了系上老師自己開發的線上自動評分系統，這套系統只能正確的判斷程式執行的結果，當時的我們也就直覺的認為程式只要結果對就好。到了大二，在軟體工程這堂課中，我們了解到了軟體程式品質的重要性，如果只在乎結果，程式中可能存在許多不必要的變數或結構，而不好的變數的宣告方式及糟糕的函式寫法都有可能產生安全性問題、影響可讀性，從而降低程式的品質。  
藉由系上資訊專題課程的契機，我們預計開發出一套能判斷程式執行結果正確性，且能夠分析程式品質的教學輔助系統。對於這套系統，我們將重點放在如何檢測程式品質以及如何提升使用者程式碼品質上。檢測品質的部分，會利用SonarQube來檢測上面所使用的三項指標：安全性、可靠性以及可維護性[1]。提升程式碼品質的部分，在檢測完後，系統會將可改進的部分標記起來，並提供範例給使用者參考。此外，系統也會將每次結果紀錄下來，生成程式碼的品質曲線圖，以便了解學習狀況。  
  
目前預計的系統功能：

(1) 提供單元測試以及品質檢測。

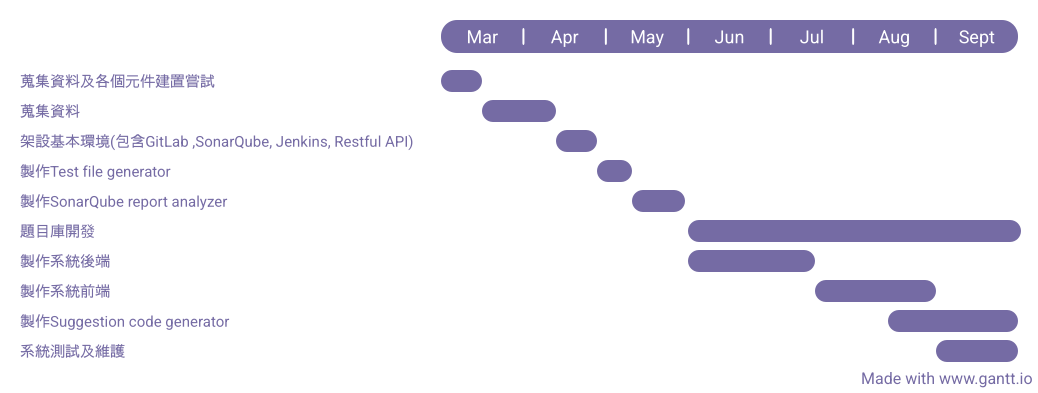
(2) 根據學生程式碼錯誤提供對應的建議。

(3) 老師在出題時可以自由選擇自己出題或是從題庫出題。

(4) 提供品質曲線圖以便學生了解進步狀況。

(5) 報告結果整合到儀表板上。

二、專題整體時程與進度評估



|  |  |  |  |
| --- | --- | --- | --- |
|  | 時間 | 預計時程 | 標的 |
| 1. | 3/1~3/14 | 蒐集資料及各個元件建置嘗試 |  |
| 2. | 3/15~4/11 | 蒐集資料 |  |
| 3. | 4/12~4/25 | 架設基本環境(包含GitLab, SonarQube, Jenkins, Restful api) | 能運行基本功能並將各個部件連接，蒐集題目庫題目,產生基本Maven pom.xml |
| 4. | 4/26~5/9 | 製作Test file generator | 能將題目庫的測資做成Maven中的測試程式 |
| 5. | 5/10~5/23 | 製作SonarQube report analyzer | 能把SonarQube報告中的特定測試結果提出並存入Database中 |
| 6. | 5/24~6/20 | 題目庫開發、製作系統後端 | 架設題目庫，並輸入題目敘述，題目測資會在後面陸續加入 |
| 7. | 6/7~6/20 |
| 8. | 6/21~7/4 | 題目庫開發、製作系統後端 | 架設系統後端Server，指派題目、Jenkins pipeline project的建立、GitLab project的建立、課程管理、系統帳號管理、自行出題等服務 |
| 9.. | 7/5~7/18 |
| 10. | 7/19~8/1 | 製作系統前端、題目庫開發 | 能顯示系統儀表板、包含系統登入介面,老師出題介面,學生答題介面,儀表板,管理員介面 |
| 11. | 8/2~8/15 | 製作系統前端、題目庫開發、系統測試及維護 |  |
| 12. | 8/16~8/29 |
| 13. | 8/29~9/12 | 系統測試及維護、製作  Suggestion code generater |  |
| 14. | 9/13~9/30 | 系統測試及維護 |  |

以上是我們預定的整體時程，三到五月，是我們系統架設的初期，主要工作是架設Server的基礎環境，其中包含蒐集資料、架設基本環境(包含Gitlab、SonarQube、Jenkins、Restful api)以及製作Test file generator、SonarQube report analyzer，題目庫的架設也會在這段時間中進行，但在後面的時間也會陸續增加題目來增加題數，六到八月之間，是我們系統架設的中期，會開始著手系統網頁的前、後端，到了九月就會是系統架設的後期，這段時間，會將重心放在製作Suggestion code generater以及系統測試及維護中，預計到了十月我們的專題就會結束。

三、執行情形與討論

專題執行到目前為止完成了四個系統所需要的元件，分別是:

Jenkins Pipeline Script、SonarQube Report Analyzer、Test File Generator、系統題目庫。

1. Jenkins Pipeline Script：我們撰寫Jenkins Pipeline script，將建置過程分為多個stage並依序執行，能有效掌控專案建置流程，其中包含Maven Unit Test，及SonarQube的品質分析。

2.SonarQube Report Analyzer：我們利用SonarQube Web API擷取SonarQube生成之報告中系統所需內容，並將資料寫入資料庫中。其中擷取內容包含三項指標：安全性(Bugs數目)，可靠性(Vulnerability數目)，可維護性(Code smell的數目)的評估等級。

3.Test File Generator：為了隱藏題目測資提防學生作弊和直接生成測試程式減少老師出題負擔，我們利用Restful api，生成Maven單元測試的測試程式，並在建置的過程中替換學生提交的測試程式。

4.系統題目庫：我們利用動態網頁爬蟲將題目從目標網站擷取下來，並且自行生成隨機測資，存入系統資料庫中，讓老師可以從系統選題，減少出題負擔。  
以下是我們討論會遇到的難題及報告時老師對我們的提問:

1.關於系統上未來可以改進的地方：

目前我們的系統目前預計支援的語言就只有JAVA，如果未來學弟妹想要升級系統的話，可以從增加支援語言和增加題目庫資料量著手。

2.關於如何讓初學者透過我們的系統得知程式品質的好壞：  
 我們預計採用SonarQube對各項標準評估的等級(包含安全性、可維護性及可靠性)[1]，並依據此等級給予使用者實際的量化分數，使其對於程式品質的認知有了初步的認識。除此之外我們也會根據SonarQube提供的Code smell給予使用者實際的軟體品質建議。

四、後續工作規劃與調整  
後續預計完成系統前、後端及Suggestion code generator的實作。由於系統後端的複雜度較高因此我們將原先的前端預計時程後移，並且由於前端需要較長的工作時程較長，我們決定先著手進行前端的開發，Report generator的部分我們將其與前端結合，故從時程表中去除。  
1.系統後端的開發：此部分主要處理GitLab project的建立、Jenkins pipeline project的建立、課程管理、系統帳號管理。為了能讓系統前端獲得顯示的內容此部分會優先製作。  
  
2.系統前端的開發：此部分主要處理使用者介面、管理者介面的顯示。目前已經針對兩者應該顯示的介面進行初步的模板製作，但其中細節部分考慮尚未完全，因此預計時常較長。  
  
3.Suggestion code generator的實作：目前計畫參考Diego Marcilio,Carlo Alberto Furia,Rodrigo Bonifacio,Gustavo Pinto "Automatically Generating Fix Suggestions in Response to Static Code Analysis Warnings"的實作方式，在其研究中提供了SonarQube中最常被手動更正的11個靜態代碼分析規則，依據這些規則，我們會製作每條規則的代碼轉換模板(code-transformation templates)，利用這些模板套用到使用者的程式碼中，即可產生建議程式碼。而在系統確實產生建議程式碼前會先針對SonarQube檢測產生的結果進行檢查，若確實違反了其中某項規則，才會套用模板。  
以下為Suggestion Code Generator運作方法：

a.針對SonarQube的報告內容偵測是否違反預先決定的11條規則。

b.若有違反規則，找出使用者程式碼中違反規則的程式碼片段。

c.套用代碼轉換模板，產生建議程式碼。

以下為論文[2]提供的C8規則的代碼轉換模板(code-transformation templates)：

refactoredExp = parse(#Expression, "<beforeFunc>.isEmpty()");

此代碼轉換模板能套用在違反規則C8的程式碼片段上，能將Collection.size()==0替換為Collection.isEmpty(),如此便可消除SonarQube品質檢測所產生的issues。

五、經費使用情形  
本次專題會將Server架設在學校提供的虛擬機上，且我們無需額外的設備。因此無申請任何經費，故無經費使用情形。

六、參考文獻

[1]SonarQube url:[Code Quality and Code Security | SonarQube](https://www.sonarqube.org/)

[2]Diego Marcilio,Carlo Alberto Furia,Rodrigo Bonifacio,Gustavo Pinto "Automatically Generating Fix Suggestions in Response to Static Code Analysis Warnings" 2019 DOI:10.1109/SCAM.2019.00013