

**BIT302**

**Software Engineering Principles**

## DESIGN AND TESTING DOCUMENT ITERATION 1

***MK SYSTEM***

**Student Name:** **LIM PEIR WEI Student ID: B1401800 E-mail: kevinklzai@gmail.com**

**Student Name: ZHOU YU JIE Student ID: B1402332 E-mail: 1539883891mitchie@gmail.com**

#### 

**15 Jalan Sri Semantan 1, Off Jalan Semantan, Bukit Damansara, 50490 Kuala Lumpur, Malaysia.**

**Tel : 603-2716 2000, Fax : 603 - 2095 7100**

[1. Introduction to Design Documentation 3](#_Toc3209028)

[Introduction 3](#_Toc3209029)

[Use Case Diagram to show the division of work done by individual team member 4](#_Toc3209030)

[2. Design specification: 4](#_Toc3209031)

[Design Class Diagram 4](#_Toc3209032)

[Entity Relationship Diagram (ERD) 5](#_Toc3209033)

[Site Map 6](#_Toc3209034)

[Generic User Experience/Wireframe 7](#_Toc3209035)

[System Sequence Diagram of the use cases to be developed for Iteration 1 (updated) 13](#_Toc3209036)

[Database Design (Data Dictionary) 14](#_Toc3209037)

[3. Iteration 1 18](#_Toc3209038)

[Test Plan 18](#_Toc3209039)

[Test Objective 18](#_Toc3209040)

[Test Scope 18](#_Toc3209041)

[Type of Testing 18](#_Toc3209042)

[Test Strategy 19](#_Toc3209043)

[Testing Tools 20](#_Toc3209044)

[Testing Analysis Report 21](#_Toc3209045)

[Test Case 22](#_Toc3209046)

[Updated Gantt Chart 68](#_Toc3209047)

[Appendix: 71](#_Toc3209048)

###### Introduction to Design Documentation

Introduction

The design and testing document will describe the information about the system development process and also contains the process for our system how to develop.

In this document will provide the description and graphical documentation for MK System. Architectural Design will be designed and recorded about the use cases which will be built up in Iteration 1, such as design class diagram, ERD, site map and wireframe. Here are also offered the system design which based on database design and system sequence diagram of use cases to be developed for Iteration 1.

Use Case Diagram to show the division of work done by individual team member



Figure 1Use Case Diagram to show division of work done by individual team member

###### Design specification:

Design Class Diagram



Figure 2Design Class Diagram of Iteration 1

Entity Relationship Diagram (ERD)

Lf\_gbl\_user (comp\_code, User\_id, User\_name, Password, Type, Email, Contact\_no, Status, Lf\_uid\_created, Lf\_Date\_created, Lf\_last\_modified, Lf\_date\_modified,)

Primary Key: comp\_code, User\_id

Foreign Key: --

Lf\_gbl\_University (comp\_code, Uni\_code, Uni\_desc, Status, Locked, Lf\_uid\_created, Lf\_last\_modified, Lf\_date\_modified)

Primary Key: comp\_code, Uni\_code

Foreign Key: --

Lf\_gbl\_qualification (comp\_code, Qualification\_code, Qualification\_desc, Average\_best\_of, Min\_score, Max\_score, Grade\_system, Grade\_subject, Lf\_uid\_created, Lf\_Date\_created, Lf\_last\_modified, Lf\_date\_modified)

Primary Key: comp\_code, Qualification\_code

Foreign Key: --

Site Map

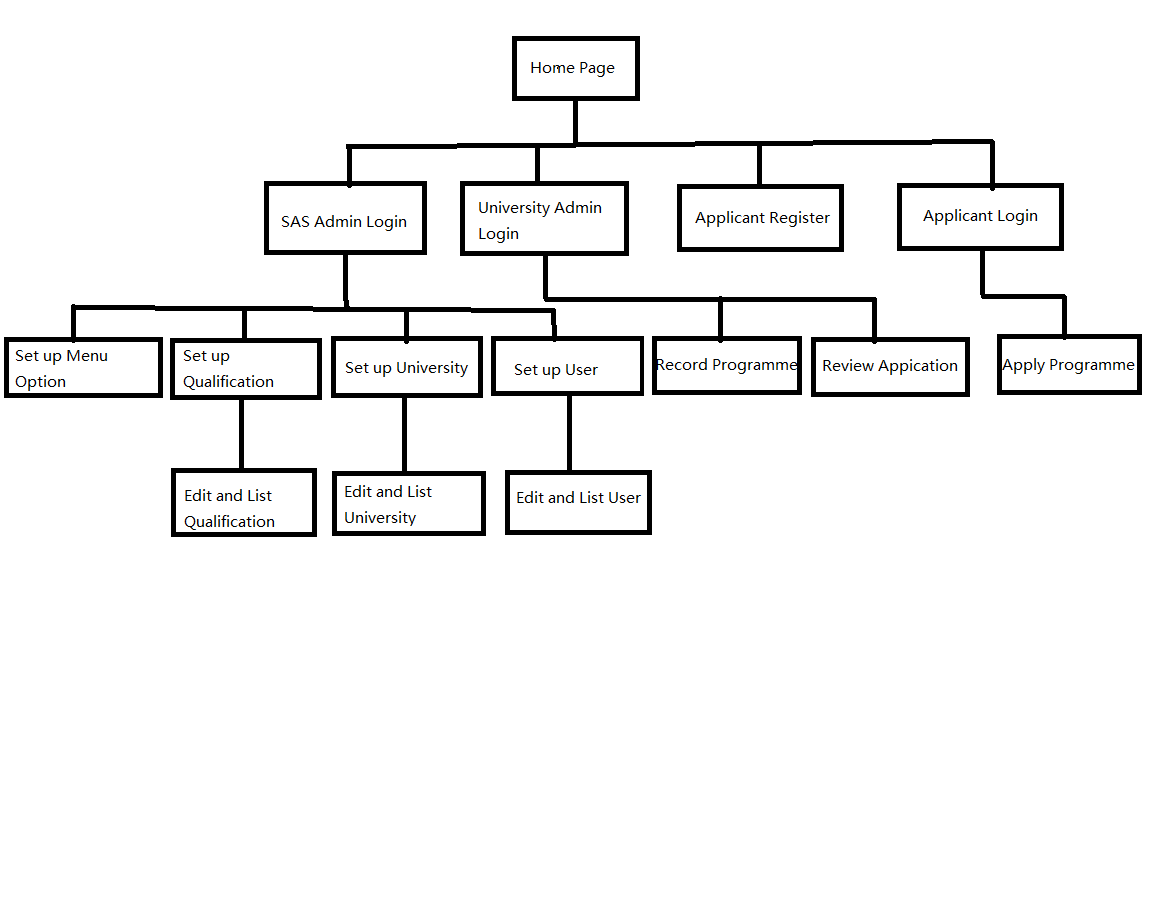
****

Figure 3Site Map Diagram

Generic User Experience/Wireframe

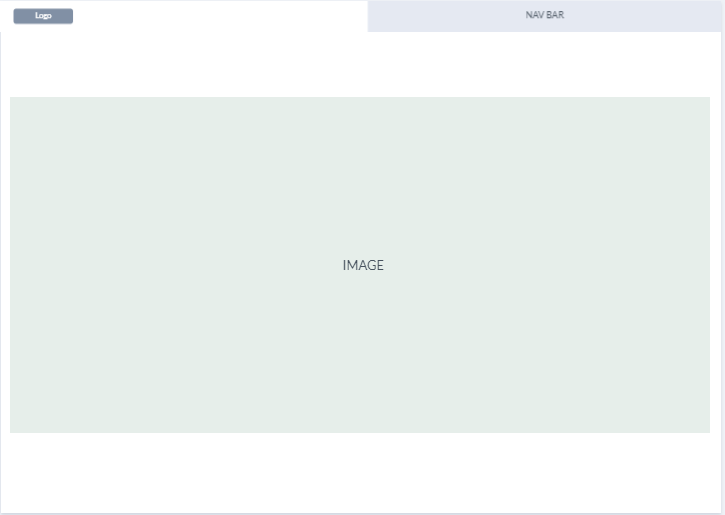


Figure 4Home Page

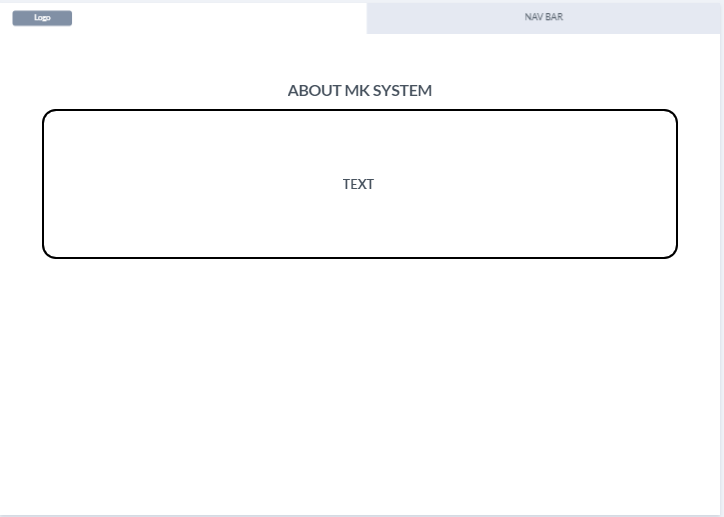


Figure 5about Us Page

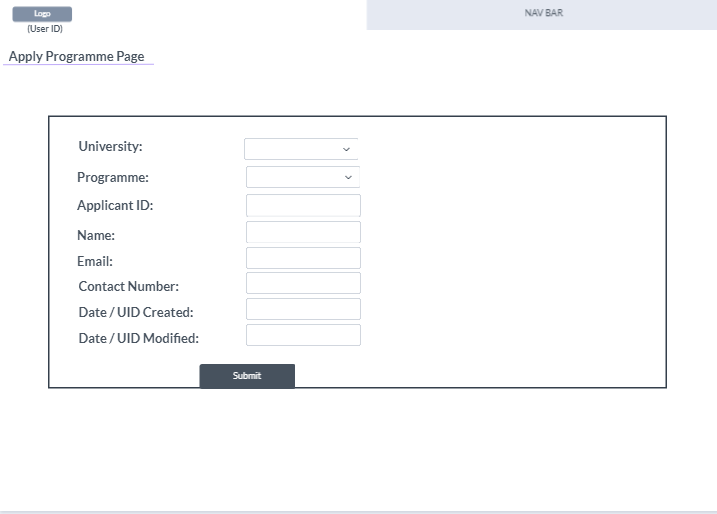


Figure 6Apply Programme Page

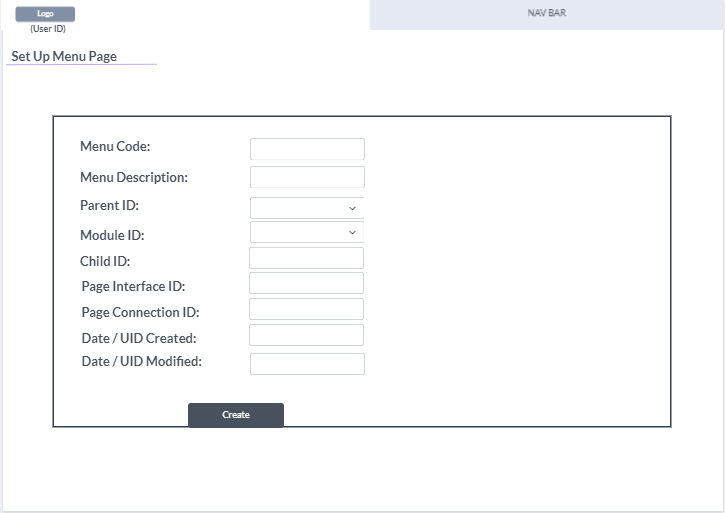


Figure 7Set Up Menu Page

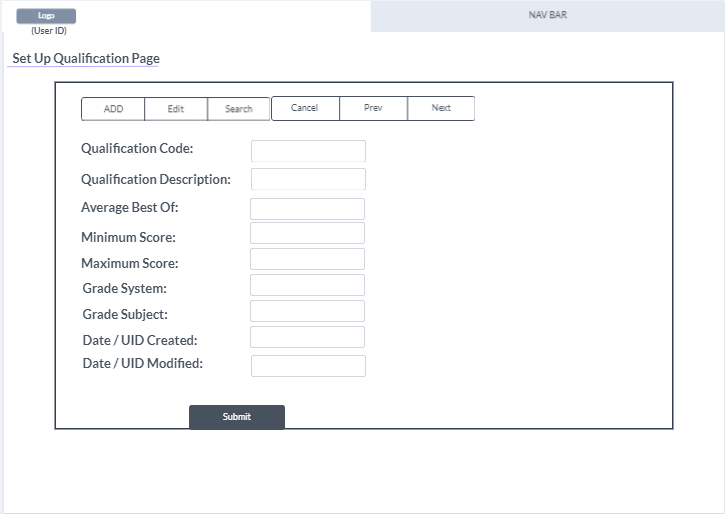


Figure 8Set Up qualification Page

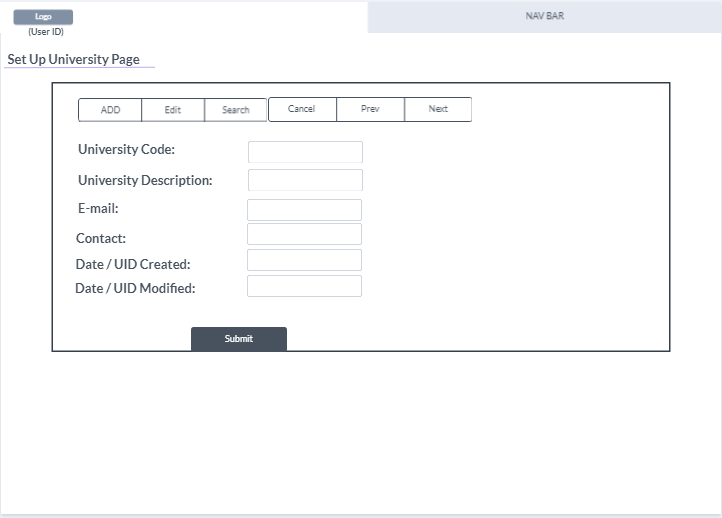


Figure 9Set Up University Page

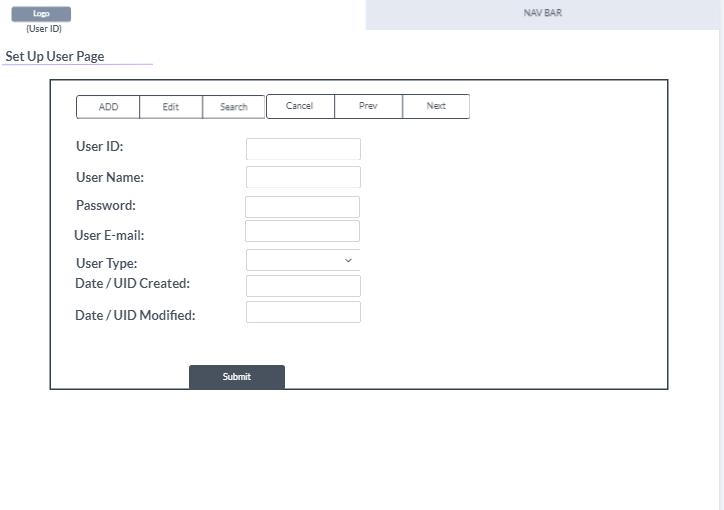


Figure 10Set Up User Page

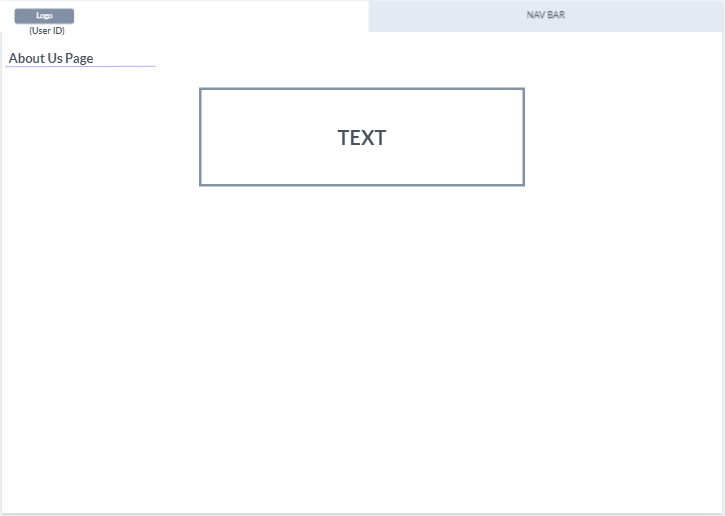


Figure 11About Us Page

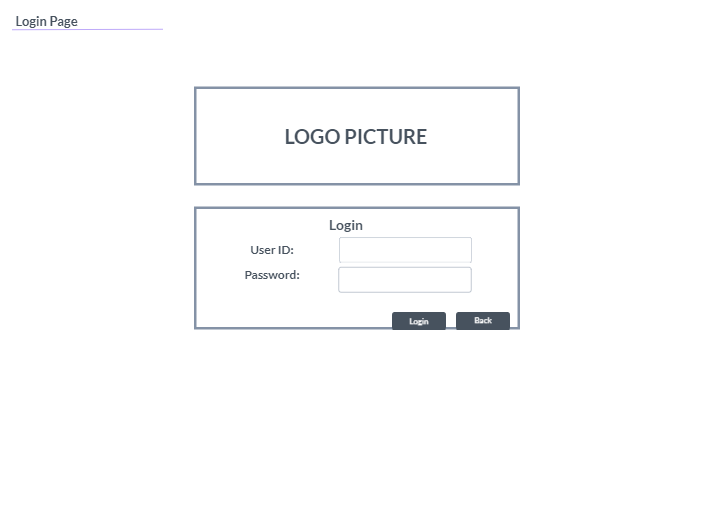


Figure 12Login Page

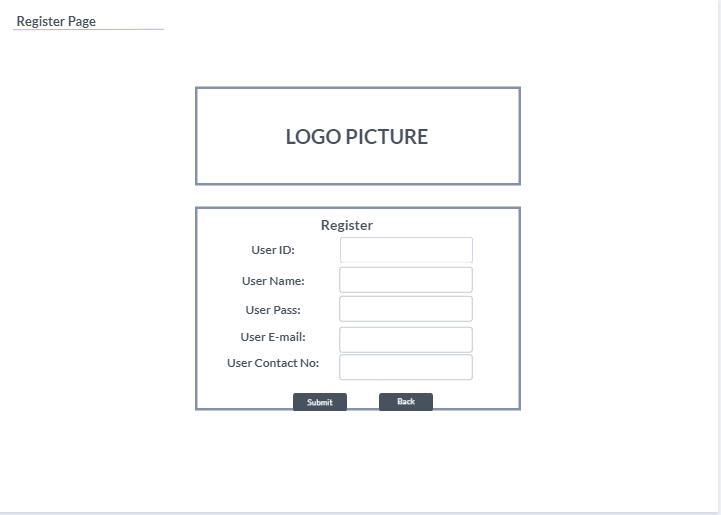


Figure 13Register Page

System Sequence Diagram of the use cases to be developed for Iteration 1 (updated)

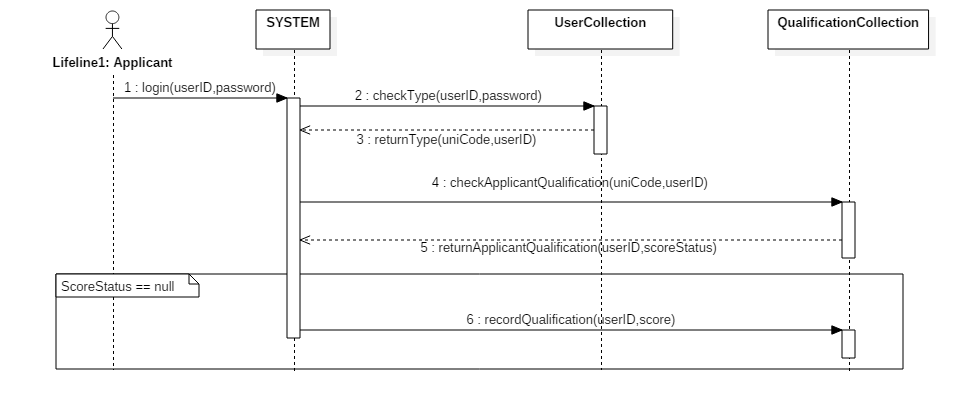


Figure 14SSD for Set Up Qualification

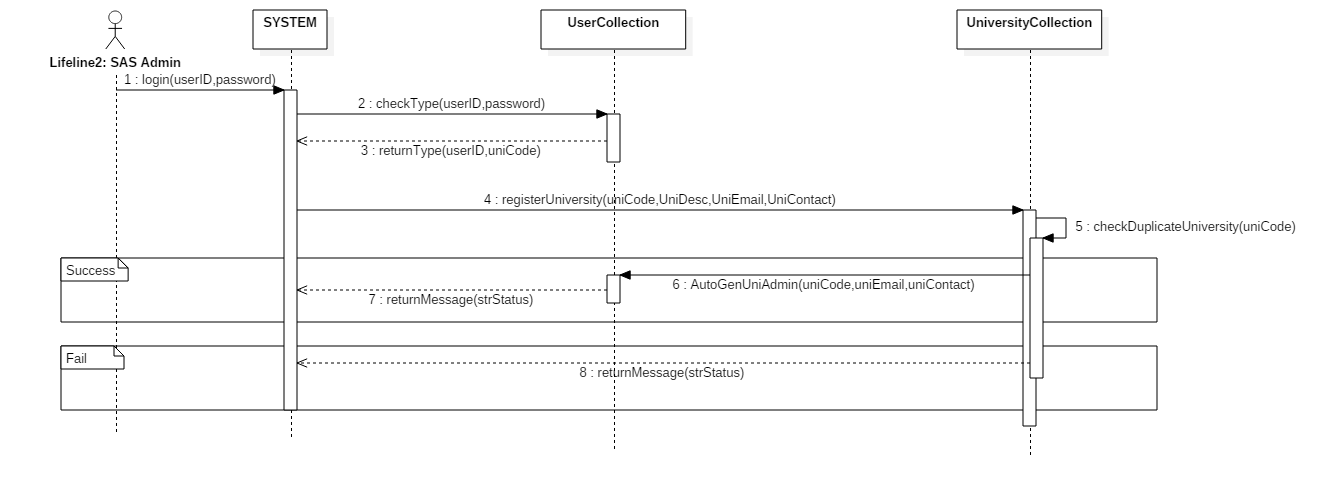


Figure 15SSD for Set Up University

Database Design (Data Dictionary)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Table | Description | DataType | Notes | Example | Description |
| Lf\_gbl\_user | comp\_code | Varchar(10) | Primary Key  Not Null | MY |  |
| User\_id | Varchar(20) | Primary Key  Not Null | HELPAdmin | This is the unique ID for every user. |
| User\_name | Varchar(60) | Not Null | HELP ADMIN | Description of the user. |
| Password | Varchar(60) | Not Null | 123456 | Store the password user key in |
| Type | Varchar(20) | Not Null | UNI | Store the type of user belong to (APP = Applicant, UNI = University) |
| Email | Varchar(100) | Not Null | helpadmin@help.edu.my | This is the unique email for each user in the system. |
| Contact\_no | Varchar(20) | Not Null | 012-3456789 | Store the user Contact Number |
| Status | Varchar(3) | Not Null | N | This is to Indicate the user status is (T = Terminate, N = Normal) |
| Lf\_uid\_created | Varchar(50) | Not Null | SYSTEM | Automatic Generate by the SYSTEM |
| Lf\_Date\_created | Datetime | Not Null | 2019-03-04 | This date is belong to the user create follow the time in the server. |
| Lf\_last\_modified | Varchar(50) | Not Null | - | When Edit will update the user in this field |
| Lf\_date\_modified | Datetime | Not Null | - | When Edit will update the user in this field |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Table | Description | DataType | Notes | Example | Description |
| Lf\_gbl\_University | comp\_code | Varchar(5) | Primary Key  Not Null | MY |  |
| Uni\_code | Varchar(5) | Primary Key  Not Null | HELP | This is the unique ID for every University. |
| Uni\_desc | Varchar(255) | Not Null | HELP Univeristy | Description of the University. |
| Status | Varchar(3) | Not Null | N | This is to Indicate the user status is (T = Terminate, N = Normal) |
| Locked | Varchar(3) | Not Null | 0 | To check this Uni\_code does bind with other stuff |
| Lf\_uid\_created | Varchar(50) | Not Null | SYSTEM | Automatic Generate by the SYSTEM |
| Lf\_Date\_created | Datetime | Not Null | 2019-03-04 | This date is belong to the user create follow the time in the server. |
| Lf\_last\_modified | Varchar(50) | Not Null | - | When Edit will update the user in this field |
| Lf\_date\_modified | Datetime | Not Null | - | When Edit will update the user in this field |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Table | Description | DataType | Primary Key | Example | Description |
| Lf\_gbl\_qualification | comp\_code | Varchar(10) | Primary Key  Not Null | MY |  |
| Qualification\_code | Varchar(20) | Primary Key  Not Null | STPM | This is the unique ID for every qualification. |
| Qualification\_desc | Varchar(20) | Not Null | Sijil Tinggi Persekolahan Malaysia | Description of the Qualification. |
| Average\_best\_of | Int(11) | Not Null | 3 | Average the grade by using how many subject. |
| Min\_score | Double | Not Null | 0.0 | Store the type of user belong to (APP = Applicant, UNI = University) |
| Max\_score | Double | Not Null | 4.0 | This is the unique email for each user in the system. |
| Grade\_system | Text | Not Null | A=4.00;A-=3.67;B+=3.33;B=3.00;B-=2.67;C+=2.33;C=2.00;C-=1.67;D+=1.33;D=1.0;F=0.00; | The grade indicate how many point obtain. |
| Grade\_subject | Text | Not Null | Science,Mathematic,Business | This qualification contain subject for user select. |
| Lf\_uid\_created | Varchar(50) | Not Null | SYSTEM | Automatic Generate by the SYSTEM |
| Lf\_Date\_created | Datetime | Not Null | 2019-03-04 | This date is belong to the user create follow the time in the server. |
| Lf\_last\_modified | Varchar(50) | Not Null | - | When Edit will update the user in this field |
| Lf\_date\_modified | Datetime | Not Null | - | When Edit will update the user in this field |

###### Iteration 1

Test Plan

Test Objective

1. To get confidence and offer information about the quality level
2. To find the defects that developers may make during developing website.
3. To prevent defects and errors.
4. To verify the consistency of functional requirements and plan with determinations and structure particulars.
5. To ensure that the outcomes meet the functional requirements.
6. To make sure that all components work in accordance with functional specifications.

**Test Scope**

The testing scope will cover the testing for Iteration 1 about the use cases Set Up Qualification, Set Up University, Set Up User and Login. The navigation also will be tested.

**Type of Testing**

Unit testing:

The main purpose of unit testing is that find some types of defects such as computing faults and algorithmic errors in the website system. The JavaScript and PHP function for submitting the several forms, to ensure that the value will return the correct parameters.

Integration Testing:

Ensure that there is a firm stream when our unit testing is consolidated and make it test as a group. We will start with the top module which is the signup and login part, actualize with the top down integration, trailed by the next module which is post user’s profiles, and lastly testing the last module which submits the profile details. For example, the userID what she/he was submitted would display in the navigation bar with the system name in the left.

System testing:

The main aim is that testing with the function and process of the whole website system. It will be imitated with likely that a real user applying the system in the system testing. This is to limit the possibility of some major errors before the following stage. And we will base on the website use case to process the system testing.

Test Strategy

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Technique | Approach | Use Case | Schedule | Aim |
| Unit Testing | White Box | Set Up Qualification & Set Up University | After the section of code had been completed individually | * It's easier to find errors in the code of small parts. * Provide flexibility while encoding components, especially when changing code. * Make sure all the parts of the code are correct before consolidating them. |
| Integration Testing | Black Box | Login & Sign Up | After merging the component that have same function | * To make sure that different modules can still cope with one error and without any error. * To make sure that proper transition to other pages in correct way. |
| System Testing | White Box & Black Box | Sign Up & Login & Set Up Qualification & Set Up University | After the whole application is completed | * To ensure that MK System website enable to meet the functional and non-functional requirements. |

Testing Tools

In the Web based system, more or less are using different tools to develop the coding such as (Microsoft Visual Code, Notepad, Atom, NetBean, and others). For the web based testing using the browser such as (Google Chrome, Firefox, Microsoft Edge, and others), testing the code in website, most of the browser will have their own debug mode call as (Tools for Web Developer), this is to inspect the code which line when wrong, and doing a live adjustment testing without change the current programme UX resolution. For Example of using tools for web developer, to check the certain field resolution want to adjust big and small, can use on the top left corner button name as (select an element in the page to inspect it) to click on the element and adjust their size until the developer satisfying on the changing, and go back to the file and change the code. In currently we are using Google Chrome(72.0.3626.121) to testing the programme, we test the resolution, function of the page, Javascript and css are running without error.

Testing Analysis Report

Login

Sign Up

e

Main Menu

Top-down integration testing is performed as our MK system integration testing. Top-down integration can provide benefits for early testing because it simulates the behaviour of low-level modules that have not yet been integrated.

As results, there is no stubs that required which make us easier to test our function, yet we still manage to create and test module that have similar function before moving to the next part.

Top down integration will be using both white box testing and black box testing. For integration testing will use the black box testing to find the errors. For unit testing part will use white box testing to find out any error code or any logical error, because need to ensure there is no error occur again at future progression. Black box testing will be carried out for every end of phase which objective is to let tester to test MK System without any guidance from developer.

Test Case

**Unit Testing**

| **Unit Testing** | 1 |
| --- | --- |
| **Test Case Objective** | Checking duplicate qualification code |
| **Test Case Description** | This testing to test the University code if is existing, it will prompt error message “Duplicate Qualification Found”. |
| **Source Code** | if (isset($txtQualificationCode)){  $chkQualification = $cf->chkDupMaster($txtQualificationCode,"lf\_gbl\_qualification","qualification\_code");    if($chkQualification){  $strError .= "Duplicate Qualification Found";  }  }  echo "<script type='text/javascript'>alert('$strError');</script>";  echo "<script type='text/javascript'>alert('Record has been failed... Please try again...');window.location = '$redirect\_url\_fail'</script>";  public function chkDupMaster($str,$table,$col)  {  $txtReturn = '';  $cond = "where comp\_code = 'MY' and $col = '$str'";  $sqlChkDup = "SELECT COUNT(\*) AS countRow FROM $table $cond";  $stmt = $this->conn->prepare($sqlChkDup);  $stmt->execute();  if ($stmt->execute()) {  $stmt-> bind\_result($token2);  if($stmt-> fetch() )  {  $txtReturn = $token2;  }  // user existed  $stmt->close();  if($txtReturn == 1){  return true;  }else{  return false;  }  } else {  // user not existed  $stmt->close();  return false;  }  } |
| **Expected Result** | Pop up message: ‘Duplicate Qualification Found’  Pop up message: ‘Record has been failed... Please try again...’  Redirect to user main page. |
| **Outcome** |  |

| **Unit Testing** | 2 |
| --- | --- |
| **Test Case Objective** | Setup Qualification (Edit Mode) Amend or add new Grade |
| **Test Case Description** | This testing is to test the qualification amend or add new on the grade without clear off whole line. |
| **Source Code** | function addGrade(){      var strGrade = document.getElementById("txtGrade").value.toUpperCase();      var strScore = document.getElementById("txtScore").value;      var strGradeSystem = document.getElementById("txtGradeSystem").value;      var strFlag = true;        var array = "";      var arrayCount = "";      var arryItem = "";      var tmpStr = "";      var newItem = false;      var piority = false;      var i;      if(strGrade.trim() != ""){              if(strScore.trim() != ""){       if(strGradeSystem.trim() != ""){          array = strGradeSystem.split(";");          arrayCount = array.length -1;          for(i=0;i<arrayCount;i++){              arrayItem = array[i].split("=");              if(strGrade == arrayItem[0]){                  tmpStr += strGrade + "=" + strScore + ";";                  piority = true;                  newItem = false;              }else{                  tmpStr += array[i] + ";";                  if(!piority){                      newItem = true;                  }              }          }          if(newItem){              tmpStr += strGrade + "=" + strScore + ";";          }          document.getElementById("txtGradeSystem").value = "";          document.getElementById("txtGradeSystem").value += tmpStr;          document.getElementById("txtGrade").value = "";          document.getElementById("txtScore").value = "";          strFlag=false;      }      if(strFlag){              document.getElementById("txtGradeSystem").value += strGrade + "=" + strScore + ";";              document.getElementById("txtGrade").value = "";              document.getElementById("txtScore").value = "";      }          }else{  alert("Score Cannot be Empty");  }          }else{              alert("Grade Cannot be Empty");          }  } |
| **Expected Result** | From Source : A=4.00;B=3.00;C=2.00;D=1.00;E=0.00; To Source : A=5.00;B=4.00;C=3.00;D=2.00;E=1.00;F=0.00; Popout Message : ‘Record Has been Successful’. |
| **Outcome** |  |

| **Unit Testing** | 3 |
| --- | --- |
| **Test Case Objective** | Setup Qualification (Edit Mode) Amend or add new Grade |
| **Test Case Description** | This testing is to test the qualification amend or add new on the grade without clear off whole line. |
| **Source Code** | function addGrade(){      var strGrade = document.getElementById("txtGrade").value.toUpperCase();      var strScore = document.getElementById("txtScore").value;      var strGradeSystem = document.getElementById("txtGradeSystem").value;      var strFlag = true;        var array = "";      var arrayCount = "";      var arryItem = "";      var tmpStr = "";      var newItem = false;      var piority = false;      var i;      if(strGrade.trim() != ""){              if(strScore.trim() != ""){       if(strGradeSystem.trim() != ""){          array = strGradeSystem.split(";");          arrayCount = array.length -1;          for(i=0;i<arrayCount;i++){              arrayItem = array[i].split("=");              if(strGrade == arrayItem[0]){                  tmpStr += strGrade + "=" + strScore + ";";                  piority = true;                  newItem = false;              }else{                  tmpStr += array[i] + ";";                  if(!piority){                      newItem = true;                  }              }          }          if(newItem){              tmpStr += strGrade + "=" + strScore + ";";          }          document.getElementById("txtGradeSystem").value = "";          document.getElementById("txtGradeSystem").value += tmpStr;          document.getElementById("txtGrade").value = "";          document.getElementById("txtScore").value = "";          strFlag=false;      }      if(strFlag){              document.getElementById("txtGradeSystem").value += strGrade + "=" + strScore + ";";              document.getElementById("txtGrade").value = "";              document.getElementById("txtScore").value = "";      }          }else{  alert("Score Cannot be Empty");  }          }else{              alert("Grade Cannot be Empty");          }  } |
| **Expected Result** | From Source : A=4.00;B=3.00;C=2.00;D=1.00;E=0.00; To Source : A=5.00;B=4.00;C=3.00;D=2.00;E=1.00;F=0.00; |
| **Outcome** |  |

| **Unit Testing** | 4 |
| --- | --- |
| **Test Case Objective** | Setup Qualification (Edit Mode) Amend or add new Subject |
| **Test Case Description** | This testing is to test the qualification amend or add new on the subject without clear off whole line. |
| **Source Code** | function addSubject(){  var strSubjectBefore = document.getElementById("drpSubjectBefore").value.toUpperCase();  var strSubjectAfter = document.getElementById("txtSubjectAfter").value.toUpperCase();  var strGradeSubject = document.getElementById("txtGradeSubject").value;  var strFlag = true;    var array = "";  var arrayCount = "";  var editMode = false;  var tmpStr = "";  var newItem = false;  var piority = false;  var i;  if(strSubjectAfter.trim() != ""){  if(strGradeSubject.trim() != ""){  array = strGradeSubject.split(";");  arrayCount = array.length -1;  for(i=0;i<arrayCount;i++){  if(strSubjectBefore != ""){  if(strSubjectBefore == array[i]){  tmpStr += strSubjectAfter + ";";  piority = true;  newItem = false;  }else{  tmpStr += array[i] + ";";  if(!piority){ newItem = true; }  }  }else{  if(strSubjectAfter == array[i]){  tmpStr += strSubjectAfter + ";";  piority = true;  newItem = false;  }else{  tmpStr += array[i] + ";";  if(!piority){ newItem = true; }  }  }  }  if(newItem){  tmpStr += strSubjectAfter + ";";  }  document.getElementById("txtGradeSubject").value = "";  document.getElementById("txtGradeSubject").value += tmpStr;  document.getElementById("drpSubjectBefore").value = "";  document.getElementById("txtSubjectAfter").value = "";  strFlag=false;  }  if(strFlag){  document.getElementById("txtGradeSubject").value += strSubjectAfter + ";";  document.getElementById("drpSubjectBefore").value = "";  document.getElementById("txtSubjectAfter").value = "";  }  }else{ alert("Subject Cannot be Empty"); }  } |
| **Expected Result** | From Source : SC;MM;ASC;AMM; To Source : SC;MATHEMATIC;ASC;AMM; |
| **Outcome** |  |

| **Unit Testing** | 5 |
| --- | --- |
| **Test Case Objective** | Setup Qualification (Edit Mode) Subject Amend Button |
| **Test Case Description** | This testing is to test the qualification amend button validation the txtSubjectAfter is empty and prompt message to user what was error occur. |
| **Source Code** | function addSubject(){  var strSubjectBefore = document.getElementById("drpSubjectBefore").value.toUpperCase();  var strSubjectAfter = document.getElementById("txtSubjectAfter").value.toUpperCase();  var strGradeSubject = document.getElementById("txtGradeSubject").value;  var strFlag = true;    var array = "";  var arrayCount = "";  var editMode = false;  var tmpStr = "";  var newItem = false;  var piority = false;  var i;  if(strSubjectAfter.trim() != ""){  if(strGradeSubject.trim() != ""){  array = strGradeSubject.split(";");  arrayCount = array.length -1;  for(i=0;i<arrayCount;i++){  if(strSubjectBefore != ""){  if(strSubjectBefore == array[i]){  tmpStr += strSubjectAfter + ";";  piority = true;  newItem = false;  }else{  tmpStr += array[i] + ";";  if(!piority){ newItem = true; }  }  }else{  if(strSubjectAfter == array[i]){  tmpStr += strSubjectAfter + ";";  piority = true;  newItem = false;  }else{  tmpStr += array[i] + ";";  if(!piority){ newItem = true; }  }  }  }  if(newItem){  tmpStr += strSubjectAfter + ";";  }  document.getElementById("txtGradeSubject").value = "";  document.getElementById("txtGradeSubject").value += tmpStr;  document.getElementById("drpSubjectBefore").value = "";  document.getElementById("txtSubjectAfter").value = "";  strFlag=false;  }  if(strFlag){  document.getElementById("txtGradeSubject").value += strSubjectAfter + ";";  document.getElementById("drpSubjectBefore").value = "";  document.getElementById("txtSubjectAfter").value = "";  }  }else{ alert("Subject Cannot be Empty"); }  } |
| **Expected Result** | From Source : SC;MATHEMATIC;ASC;AMM;  Input : (empty string);  Popout message :’Subject Cannot Be Empty’ |
| **Outcome** |  |

| **Unit Testing** | 6 | |
| --- | --- | --- |
| **Test Case Objective** | Setup Qualification (Edit Mode) Amend or add new Grade | |
| **Test Case Description** | This testing is to test the qualification amend or add new on the grade without clear off whole line. | |
| **Source Code** | function addGrade(){      var strGrade = document.getElementById("txtGrade").value.toUpperCase();      var strScore = document.getElementById("txtScore").value;      var strGradeSystem = document.getElementById("txtGradeSystem").value;      var strFlag = true;        var array = "";      var arrayCount = "";      var arryItem = "";      var tmpStr = "";      var newItem = false;      var piority = false;      var i;      if(strGrade.trim() != ""){              if(strScore.trim() != ""){       if(strGradeSystem.trim() != ""){          array = strGradeSystem.split(";");          arrayCount = array.length -1;          for(i=0;i<arrayCount;i++){              arrayItem = array[i].split("=");              if(strGrade == arrayItem[0]){                  tmpStr += strGrade + "=" + strScore + ";";                  piority = true;                  newItem = false;              }else{                  tmpStr += array[i] + ";";                  if(!piority){                      newItem = true;                  }              }          }          if(newItem){              tmpStr += strGrade + "=" + strScore + ";";          }          document.getElementById("txtGradeSystem").value = "";          document.getElementById("txtGradeSystem").value += tmpStr;          document.getElementById("txtGrade").value = "";          document.getElementById("txtScore").value = "";          strFlag=false;      }      if(strFlag){              document.getElementById("txtGradeSystem").value += strGrade + "=" + strScore + ";";              document.getElementById("txtGrade").value = "";              document.getElementById("txtScore").value = "";      }          }else{  alert("Score Cannot be Empty");  }          }else{              alert("Grade Cannot be Empty");          }  } | |
| **Expected Result** | From Source : A=4.00;B=3.00;C=2.00;D=1.00;E=0.00;  Input : Grade = (empty string);  Popout message :’Grade Cannot Be Empty’  Input : Grade: ‘A’ , Score = (empty string);  Popout message :’Score Cannot Be Empty’ | |
| **Outcome** |  | |
| **Unit Testing** | | 7 |
| **Test Case Objective** | | Setup Qualification (Edit Mode) Manipulate the Subject Drop Down List |
| **Test Case Description** | | This testing is to test the qualification when existing subject, and update or add new subject will auto re-generate the subject into the drop down list |
| **Source Code** | | function genDrpDwnListSubject(){  document.getElementById("drpSubjectBefore").options.length = 1;  var select = document.getElementById("drpSubjectBefore");  var strGradeSubject = document.getElementById("txtGradeSubject").value;  var i;    if(strGradeSubject.trim() != ""){  document.getElementById("drpSubjectBefore").disabled=false;  array = strGradeSubject.split(";");  arrayCount = array.length -1;  for(i = 0; i < arrayCount; i++) {  var option = document.createElement('option');  option.text = option.value = array[i];  select.add(option, 1);  }  }  } |
| **Expected Result** | | From Source : SC;MATHEMATIC;ASC;AMM;  Input : Subject = ‘ENGLISH’  To Source : SC;MATHEMATIC;ASC;AMM;ENGLISH; |
| **Outcome** | |  |

| **Unit Testing** | 8 | |
| --- | --- | --- |
| **Test Case Objective** | Setup Qualification Check Field Only Numeric | |
| **Test Case Description** | This testing is to test the qualification insert data, verified the certain field must be numeric value, and not for character. | |
| **Source Code** | function chkNumeric(str){  if(isNaN(str.value)){  alert("(" + str.value + ") is not a number format.");  str.value = "";  }else{  str.value = str.value;  }  return str.value;  } | |
| **Expected Result** | Input : Average best Of = A;  Popout message :’(A) is not a number format.’  Reset the input to empty . | |
| **Outcome** |  | |
| **Unit Testing** | | 9 |
| **Test Case Objective** | | Checking duplicate university code |
| **Test Case Description** | | This testing to test the University code if is existing, it will prompt error message “Duplicate University Code Found”. |
| **Source Code** | | if (isset($txtUniversityCode)){  $chkUniversity = $cf->chkDupMaster($txtUniversityCode,"lf\_gbl\_university","uni\_code");    if($chkUniversity){  $strError .= "Duplicate University Found";  }  }  echo "<script type='text/javascript'>alert('$strError');</script>";  echo "<script type='text/javascript'>alert('Record has been failed... Please try again...');window.location = '$redirect\_url\_fail'</script>";  public function chkDupMaster($str,$table,$col)  {  $txtReturn = '';  $cond = "where comp\_code = 'MY' and $col = '$str'";  $sqlChkDup = "SELECT COUNT(\*) AS countRow FROM $table $cond";  $stmt = $this->conn->prepare($sqlChkDup);  $stmt->execute();  if ($stmt->execute()) {  $stmt-> bind\_result($token2);  if($stmt-> fetch() )  {  $txtReturn = $token2;  }  // user existed  $stmt->close();  if($txtReturn == 1){  return true;  }else{  return false;  }  } else {  // user not existed  $stmt->close();  return false;  }  } |
| **Expected Result** | | Pop up message: ‘Duplicate University Found’  Pop up message: ‘Record has been failed... Please try again...’  Redirect to user main page. |
| **Outcome** | |  |

| **Unit Testing** | | 10 |
| --- | --- | --- |
| **Test Case Objective** | | Setup University Check Field is Email |
| **Test Case Description** | | This testing to test the Email is not a valid email, it will prompt error message “Invalid Email Address”. |
| **Source Code** | | function chkValidEmail(str){  if(str.value != ""){  if(!str.value.includes("@")){  alert("("+str.value+") Invalid Email Address");  str.value = "";  }else{  str.value = str.value;  }  return str.value;  }  } |
| **Expected Result** | | Input : E-mail = ‘help.edu.my’;  Popout message :’(help.edu.my) Invalid Email Address.’  Reset the input to empty . |
| **Outcome** | |  |
| **Unit Testing** | 11 | |
| **Test Case Objective** | Setup University Check Field is Contact Number | |
| **Test Case Description** | This testing to test the Email is not a valid email, it will prompt error message “Invalid Email Address”. | |
| **Source Code** | function chkValidNumber(str){  if(str.value != ""){  if(str.value.length > 9){  if(str.value.length < 12){  if(isNaN(str.value)){  alert("("+str.value+") Invalid Phone Number");  str.value = "";  }else{  str.value = str.value.replace(".","");  str.value = str.value.replace("+","");  str.value = str.value.replace("-","");  str.value = str.value;  }  }else{  alert("("+str.value+") Invalid Phone Number Format");  str.value = "";  }  }else{  alert("("+str.value+") Invalid Phone Number Format");  str.value = "";  }  return str.value;  }  } | |
| **Expected Result** | Input : Contact No = ‘012-3456789’ (below 10 digits) (no symbol)  Popout message :’(012-3456789) Invalid Phone Number Format.’  Reset the input to empty . | |
| **Outcome** |  | |

**System Testing**

| **Unit Testing** | 1 |
| --- | --- |
| **Test Description:** | Applicant sign up for the system |
| **Input** | $txtUserID = ‘B1901000’  $txtUserName = ‘Test User’  $txtPassword = ‘123456’  $txtUserType = ‘APP’ (Hidden Field)  $txtEmail =’testuser@gmail.com’  $txtContact = ‘0123456789’ |
| **Source Code** | $dt = new DateTime();  $txtUserID = strtoupper($\_REQUEST['txtUserID']);  $txtUserName = strtoupper($\_REQUEST['txtUserName']);  $txtPassword = $\_REQUEST['txtPassword'];  $txtUserType = strtoupper($\_REQUEST['drpUserType']);  $txtEmail = $\_REQUEST['txtEmail'];  $txtContact = $\_REQUEST['txtContact'];  $txtStatus = 'N';  $txtUIDCreated= "SYSTEM";  $txtDateCreated = $dt->format('Y/m/d');  $stmt = $conn->prepare("INSERT INTO lf\_gbl\_user(comp\_code,user\_id,user\_name,password,type,status,email,contact\_no,lf\_date\_created,lf\_uid\_created)VALUE(?,?,?,?,?,?,?,?,?,?)");  $stmt->bind\_param("ssssssssss",$txtComCode,$txtUserID,$txtUserName,$txtPassword,$txtUserType,$txtStatus,$txtEmail,$txtContact,$txtDateCreated,$txtUIDCreated);  $result = $stmt->execute();  $stmt->close();    echo "<script type='text/javascript'>alert('Record has been successful...');window.location = '$redirect\_url'</script>"; |
| **Expected Outcome** | Pop up message: ‘Record Has Been Successful…’  Database : record each of the attributes is successful stored.  Redirect to Login page. |
| **Actual Outcome** |  |

**System Testing**

| **Unit Testing** | 2 |
| --- | --- |
| **Test Description:** | Applicant Login |
| **Input** | $txtUserID = ‘B1901000’  $txtPassword = ‘123456’ |
| **Source Code** | $txtUserID = $\_REQUEST['txtUserID'];  $txtPassword = $\_REQUEST['txtPassword'];  $stmt = $conn->prepare("SELECT user\_id,user\_name,type from lf\_gbl\_user WHERE user\_id = ? and password = ?");  $stmt->bind\_param("ss",$txtUserID,$txtPassword);  $stmt->execute();  $stmt->bind\_result($token2,$token3,$token4);  if ($stmt->fetch()) {  // user existed       $loginUserID = $token2;       $loginUserType = $token4;       $stmt->close();  $result = true;  } else {  // user not existed  $stmt->close();  $result = false;  }  if($result)  {  header("Location: $redirect\_url\_logon\_mainpage");  exit();  } |
| **Expected Outcome** | Redirect to user main page with user information. (B1901000) |
| **Actual Outcome** |  |

**System Testing**

| **Unit Testing** | 3 |
| --- | --- |
| **Test Description:** | Register University with Admin |
| **Input** | $txtUniversityCode = ‘TARUC’  $txtUniversityName = ‘Tunku Abdul Rahman University College’  $txtUniversityEmail = ‘taruc@gmail.com’  $txtUniversityContact = ‘0123456789’ |
| **Source Code** | $txtUniversityCode = strtoupper($\_REQUEST['txtUniversityCode']);  $txtUniversityName = strtoupper($\_REQUEST['txtUniversityDesc']);  $txtUniversityEmail = strtoupper($\_REQUEST['txtUniversityEmail']);  $txtUniversityContact = strtoupper($\_REQUEST['txtUniversityContact']);  $txtStatus = 'N';  $txtLocked = 0;  $txtUIDCreated= $Session\_UserID;  $txtDateCreated = $dt->format('Y/m/d'); |
| **Expected Outcome** | Pop up message: ‘Record Has Been Successful…’  University Database : record each of the attributes is successful stored.  User Database: Based on the attributes, create an user base on the university code.  Redirect to user main page. |
| **Actual Outcome** |  |

**System Testing**

| **Unit Testing** | 4 |
| --- | --- |
| **Test Description:** | Setup Qualification (Edit Mode) |
| **Input** | $txtQualificationCode = ‘A-Levels’ (unable to Edit)  $txtQualificationDesc = ‘A-Levels Certificate’  $txtAverageBestOf = ‘3’  $txtMinScore = ‘2’  $txtMaxScore = ‘4’  $txtGradeSystem = ‘A=4.00;B=3.00;C=2.00;D=1.00;E=0.00;’  $txtGradeSubject = ‘ENGLISH;MATHEMATIC;ADD.MATHEMATIC;SCIENCE;ADD.SCIENCE;’ |
| **Source Code** | $dt = new DateTime();  $txtComCode = 'MY';  $txtQualificationCode = strtoupper($\_REQUEST['txtQualificationCode']);  $txtQualificationDesc = strtoupper($\_REQUEST['txtQualificationDesc']);  $txtAverageBestOf = $\_REQUEST['txtAverageBestOf'];  $txtMinScore = $\_REQUEST['txtMinScore'];  $txtMaxScore = $\_REQUEST['txtMaxScore'];  $txtGradeSystem = strtoupper($\_REQUEST['txtGradeSystem']);  $txtGradeSubject = strtoupper($\_REQUEST['txtGradeSubject']);  $txtStatus = 'N';  $txtUIDCreated= $Session\_UserID;  $txtDateCreated = $dt->format('Y/m/d');  $stmt = $conn->prepare("UPDATE lf\_gbl\_qualification SET qualification\_desc = '$txtQualificationDesc', average\_best\_of = '$txtAverageBestOf', min\_score = '$txtMinScore', max\_score = '$txtMaxScore', grade\_system = '$txtGradeSystem', grade\_subject = '$txtGradeSubject',lf\_uid\_last\_modified = '$txtUIDCreated', lf\_date\_last\_modified='$txtDateCreated' WHERE comp\_code = ? AND qualification\_code = ?");  $stmt->bind\_param("ss",$txtComCode,$txtQualificationCode);  $result = $stmt->execute();  $stmt->close();  echo "<script type='text/javascript'>alert('Record has been successful...');window.location = '$redirect\_url'</script>"; |
| **Expected Outcome** | Search : the record user want to modify.  Pop up message: ‘Record Has Been Successful…’  Database : record each of the attributes is successful updated.  Redirect to user main page. |
| **Actual Outcome Before Edit** |  |
| **Actual Outcome After Edit** |  |

**System Testing**

| **Unit Testing** | 5 |
| --- | --- |
| **Test Description:** | Setup University (Edit Mode) and Update University Admin Information. |
| **Input** | $txtUniversityCode = ‘TARUC’ (unable to edit)  $txtUniversityName = ‘TAR University College’  $txtUniversityEmail = ‘taruc@edu.my’  $txtUniversityContact = ‘0123456789’ |
| **Source Code** | $txtUniversityCode = strtoupper($\_REQUEST['txtUniversityCode']);  $txtUniversityName = strtoupper($\_REQUEST['txtUniversityDesc']);  $txtUniversityEmail = strtoupper($\_REQUEST['txtUniversityEmail']);  $txtUniversityContact = strtoupper($\_REQUEST['txtUniversityContact']);  $txtStatus = 'N';  $txtLocked = 0;  $txtUIDCreated= $Session\_UserID;  $txtDateCreated = $dt->format('Y/m/d');  $stmt = $conn->prepare("UPDATE lf\_gbl\_university SET uni\_desc = '$txtUniversityName',uni\_email ='$txtUniversityEmail',uni\_contact='$txtUniversityContact',lf\_uid\_last\_modified = '$txtUIDCreated', lf\_date\_last\_modified='$txtDateCreated' WHERE comp\_code = ? AND uni\_code = ?");  $stmt->bind\_param("ss",$txtComCode,$txtUniversityCode);  $result = $stmt->execute();  if($result){  $cf->autoGenerateAdmin($txtUniversityCode,$txtUniversityContact,$txtUniversityEmail,$actionType);  }  public function autoGenerateAdmin($uniCode,$uniContact,$uniEmail,$task)  {  $sqlUniAdmin = "UPDATE lf\_gbl\_user SET email = '$uniEmail',contact\_no = '$uniContact',lf\_uid\_modified = 'SYSTEM',lf\_date\_modified = now() WHERE comp\_code = 'MY' and user\_id = '$uniAdmin'";  $stmt = $this->conn->prepare($sqlUniAdmin);  $stmt->execute();  } |
| **Expected Outcome** | Search : the record user want to modify.  Pop up message: ‘Record Has Been Successful…’  University Database : record each of the attributes is successful updated.  User Database : update the information base on the university record.  Redirect to user main page. |
| **Actual Outcome Before Edit** |  |
| **Actual Outcome After Edit** |  |

**System Testing**

| **Unit Testing** | 6 |
| --- | --- |
| **Test Description:** | Record Qualification |
| **Input** | txtQualificationCode = ‘STPM’  txtQualificationDesc = ‘Sijil Tinggi Persekolahan Malaysia’  txtAverageBestOf= ‘3’  txtMinScore =’0’  txtMaxScore = ‘4’  txtGradeSystem = ‘A=4.00;A-=3.67;B+=3.33;B=3.00;B-=2.67;C+=2.33;C=2.00;C-=1.67;D+=1.33;D=1.0;F=0.00;’  txtGradeSubject = ‘BM,BI,BC,HISTORY,MATHEMATIC,ADDITIONAL MATHEMATIC,SCIENCE,ADDITIONAL SCIENCE;’ |
| **Source Code** | $dt = new DateTime();  $txtQualificationCode = strtoupper($\_REQUEST['txtQualificationCode']);  $txtQualificationDesc = strtoupper($\_REQUEST['txtQualificationDesc']);  $txtAverageBestOf = $\_REQUEST['txtAverageBestOf'];  $txtMinScore = $\_REQUEST['txtMinScore'];  $txtMaxScore = $\_REQUEST['txtMaxScore'];  $txtGradeSystem = strtoupper($\_REQUEST['txtGradeSystem']);  $txtGradeSubject = strtoupper($\_REQUEST['txtGradeSubject']);  $txtStatus = 'N';  $txtUIDCreated= $Session\_UserID;  $txtDateCreated = $dt->format('Y/m/d');  $stmt = $conn->prepare("INSERT INTO lf\_gbl\_qualification(comp\_code,qualification\_code,qualification\_desc,average\_best\_of,min\_score,max\_score,grade\_system,grade\_subject,status,lf\_date\_created,lf\_uid\_created)VALUE(?,?,?,?,?,?,?,?,?,?,?)");  $stmt->bind\_param("sssssssssss",$txtComCode,$txtQualificationCode,$txtQualificationDesc,$txtAverageBestOf,$txtMinScore,$txtMaxScore,$txtGradeSystem,$txtGradeSubject,$txtStatus,$txtDateCreated,$txtUIDCreated);  $result = $stmt->execute();  echo "<script type='text/javascript'>alert('Record has been successful...');window.location = '$redirect\_url'</script>"; |
| **Expected Outcome** | Pop up message: ‘Record Has Been Successful…’  Database : record each of the attributes is successful stored.  Redirect to User Main page. |
| **Actual Outcome** |  |

**System Testing**

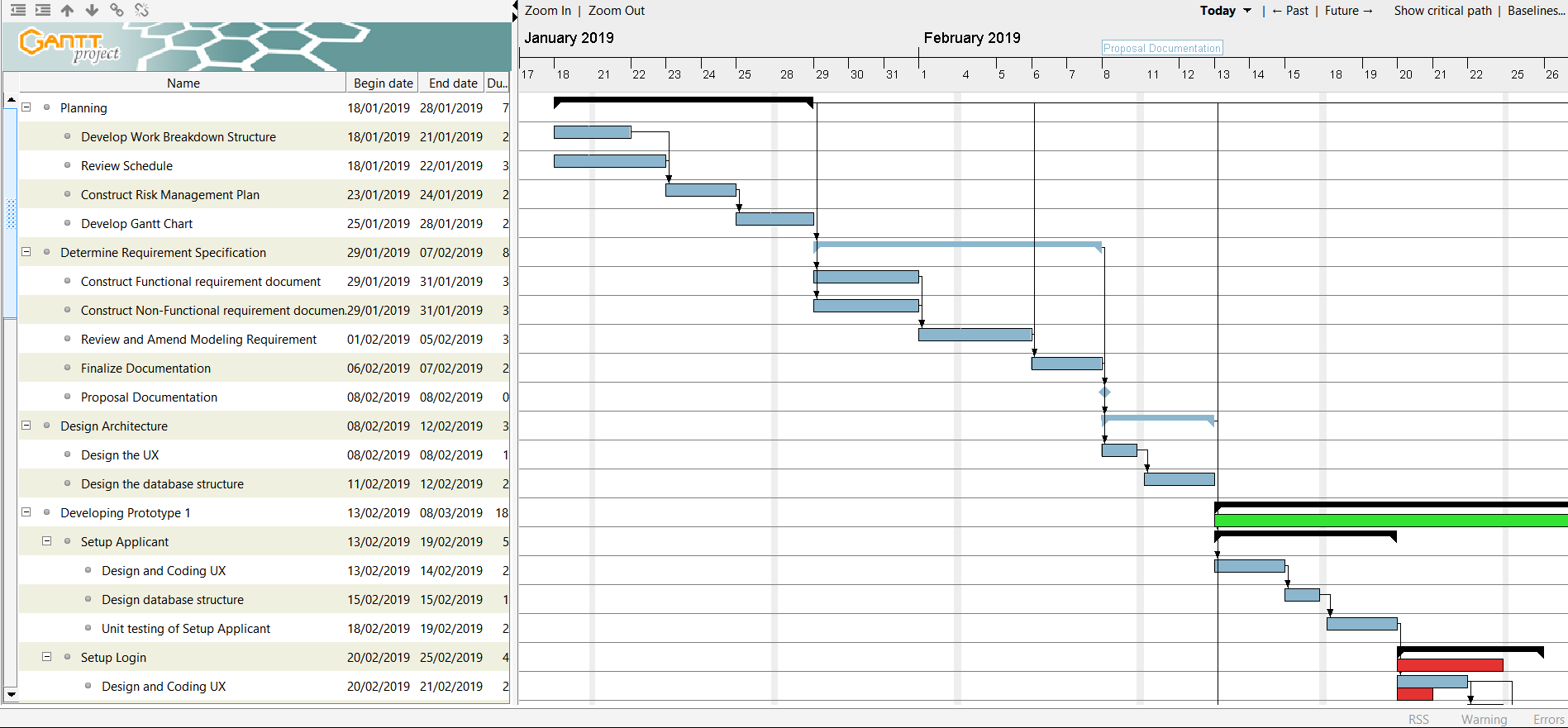
| **Unit Testing** | 7 |
| --- | --- |
| **Test Description:** | Register University |
| **Input** | University\_code = ‘TARUC’  University\_desc = ‘Tunku Abdul Rahman University College’  University\_email = ‘taruc@gmail.com’  University\_contact = ‘0123456789’ |
| **Source Code** | $txtUniversityCode = strtoupper($\_REQUEST['txtUniversityCode']);  $txtUniversityName = strtoupper($\_REQUEST['txtUniversityDesc']);  $txtUniversityEmail = strtoupper($\_REQUEST['txtUniversityEmail']);  $txtUniversityContact = strtoupper($\_REQUEST['txtUniversityContact']);  $txtStatus = 'N';  $txtLocked = 0;  $txtUIDCreated= $Session\_UserID;  $txtDateCreated = $dt->format('Y/m/d'); |
| **Expected Outcome** | Pop up message: ‘Record Has Been Successful…’  Database : record each of the attributes is successful stored.  Redirect to user main page. |
| **Actual Outcome** |  |

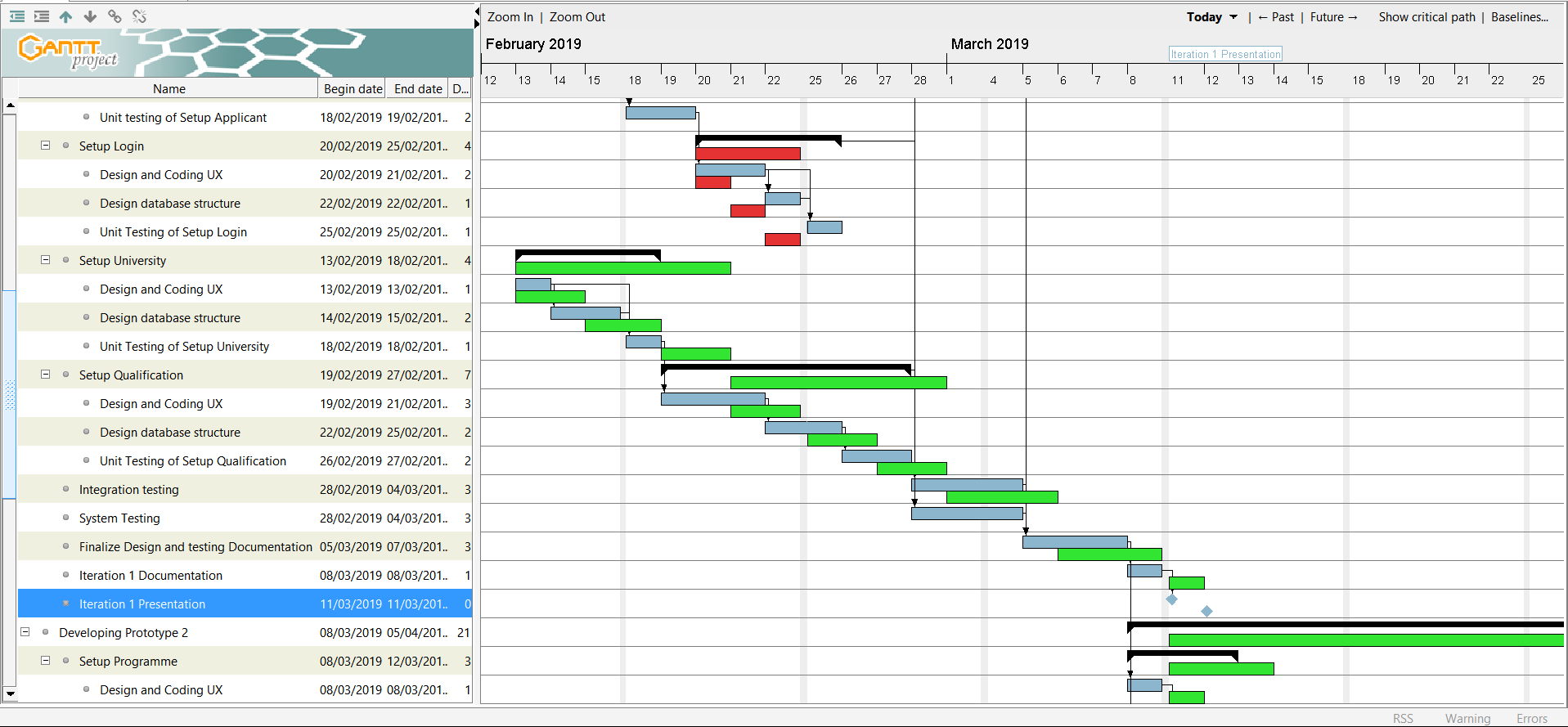
**Integration Test**

|  |
| --- |
| **Integration Test Plan** |
| **Test Case-- Integration Testing-1** |
| Login button for login page |
| **Test Case--Description** |
| To test whether the login page will be directed lead to user’s specific page when clicked the login button. |
| **Test Case--Diagram** |
|  |
| **Test Case--Test Data** |
| Login button is clicked |
| **Test Case--Expected Result** |
| Direct lead to user’s specific page |
| **Test Case--Test Outcome** |
| Direct lead to user’s specific page |
| **Test Case--Remark** |
| Page directed lead to user’s specific page without any problem. |

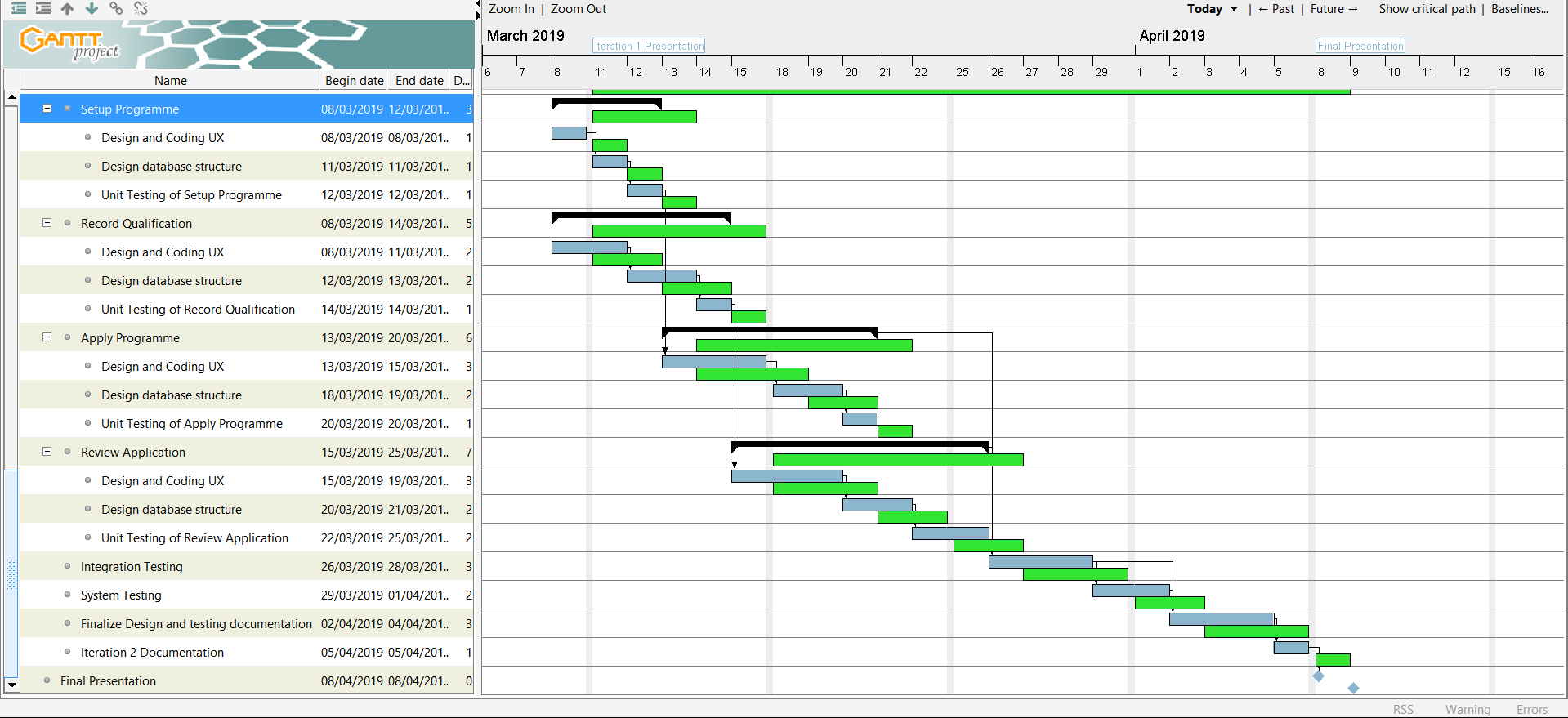
|  |
| --- |
| **Integration Test Plan** |
| **Test Case--Integration Testing-2** |
| Register button for register page |
| **Test Case--Description** |
| To test whether the register page will be displayed success message when clicked the register button. |
| **Test Case--Diagram** |
|  |
| **Test Case--Test Data** |
| Register button is clicked |
| **Test Case--Expected Result** |
| “successfully registered” displayed |
| **Test Case--Test Outcome** |
| “successfully registered” message showed |
| **Test Case--Remark** |
| Register Page enable to connect with database that lead the data enable to store inside database and no error. |

Updated Gantt Chart

 **(Part 1)**



**(Part 2)**

 **(Part 3)**

Appendix:

|  |  |
| --- | --- |
| **Use Case Name** | Record Qualification |
| **Goal in Context** | Allow applicant to enter their subject’s qualification in the study |
| **Primary Actor** | Applicant |
| **Trigger** | When applicant login to the user main page, will automatic redirect to the setup user qualification. |
| **Typical Course of Events**  **Actor Action** | **System Response** |
| 1.   When applicant login to the user main page, will automatic redirect to the setup user qualification. |  |
| 2.   Applicant enter the subject qualification in the specific field. | 3. System will verify the applicant enter, and store inside database for the next time viewing. |
| **Alternative Course of Events** |  |
|  | |

|  |  |
| --- | --- |
| **Use Case** | Register University |
| **Goal in Context** | To allow the SAS Admin to record details about a University and assign an administrator |
| **Primary Actor****Secondary Actor** | SAS Admin |
| **Trigger** | The SAS Admin has received information about a new university |
| **Typical Course of Events****Actor Action** | **System Response** |
| 1. This use case begins when the SAS Admin logs in to record information about a new university |  |
| 2. The SAS Admin enters the name of the university. | The University is recorded. A universityID is automatically generated. |
| 3. The SAS Admin enters the username, password, name and email for the University Admin. | The University Admin is recorded for the University. |
| **Alternative Course of Events** | |
| Line 3: If there are more University Admins, repeat line 3. | |

**Work Breakdown Structure**

1. Planning
   1. Develop Work Breakdown Structure
   2. Develop Gantt Chart
   3. Review Schedule
   4. Construct Risk management Plan
2. Determine Requirement Specification
   1. Construct Functional requirement Document
   2. Construct non-Functional requirement Document
   3. Review and Amend modelling Requirement
   4. Finalize Documentation
   5. Proposal Documentation
3. Design Architecture
   1. Designing the UX
   2. Designing the database concept
4. Developing Prototype 1
   1. Setup Application
      1. Design and Coding UX
      2. Design Database Structure
      3. Unit Testing of Setup Application
   2. Setup Login
      1. Design and Coding UX
      2. Design Database Structure
      3. Unit Testing of Setup Login
   3. Setup University
      1. Design and Coding UX
      2. Design Database Structure
      3. Unit Testing of Setup University
   4. Setup Qualification
      1. Design and Coding UX
      2. Design Database Structure
      3. Unit Testing of Setup Qualification
   5. Integration Testing
   6. System Testing
   7. Finalise Design and testing documentation
   8. Iteration 1 Documentation
   9. Iteration 1 Presentation
5. Developing Prototype 2
   1. Setup Programme
      1. Design and Coding UX
      2. Design Database Structure
      3. Unit Testing of Setup Programme
   2. Record Qualification
      1. Design and Coding UX
      2. Design Database Structure
      3. Unit Testing of Record Qualification
   3. Apply Programme
      1. Design and Coding UX
      2. Design Database Structure
      3. Unit Testing of Apply Programme
   4. Review Application
      1. Design and Coding UX
      2. Design Database Structure
      3. Unit Testing of Review Application
   5. Integration Testing
   6. System Testing
   7. Finalise Design and testing documentation
6. Final Presentation