

**TSE2102 Software Engineering Fundamental**

**Final Report**

Team Name: **BoBoBoyz**

1. **Ng Jing Keong (1151100169) -**[**james0523njk@gmail.com**](mailto:james0523njk@gmail.com)
2. **Goh Kun Shun (1151101980) -**[**kunshun225@gmail.com**](mailto:kunshun225@gmail.com)
3. **Christopher Too (1151101473) -**[**christophertwoweibin@gmail.com**](mailto:christophertwoweibin@gmail.com)
4. **Eric Chia Teck Han (1151101246) -**[**ericchia508@gmail.com**](mailto:ericchia508@gmail.com)
5. **Ong Koon Hua (1151102941) -**[**koonhua310@gmail.com**](mailto:koonhua310@gmail.com)

Table of Contents

[*1. Scenario / Problem Statements: 4*](#_Toc494752110)

[*1.1. Objective: 4*](#_Toc494752111)

[*1.1.1. Scope: 4*](#_Toc494752112)

[*1.2. Functional Requirements 5*](#_Toc494752113)

[*1.3. Gantt Chart 8*](#_Toc494752114)

[*2. System Overview 9*](#_Toc494752115)

[*2.1. Use Case Diagram 9*](#_Toc494752116)

[*2.2. ERD Diagram 11*](#_Toc494752117)

[*2.3. Quality Requirement 12*](#_Toc494752118)

[*2.4. Pseudocode 16*](#_Toc494752119)

[*2.4.1. Login module 16*](#_Toc494752120)

[*2.4.2. AddUser Module 16*](#_Toc494752121)

[*2.4.3. DeleteUser Module 16*](#_Toc494752122)

[*2.4.4. EditUserStatus 16*](#_Toc494752123)

[*2.4.5. Application Module 17*](#_Toc494752124)

[*2.4.6. Transaction Module 17*](#_Toc494752125)

[*2.4.7. Appointment Module 18*](#_Toc494752126)

[*2.4.8. File Management Module 18*](#_Toc494752127)

[*2.4.9. Subject Management Module 19*](#_Toc494752128)

[*3. Testing 20*](#_Toc494752129)

[*3.1. Test Data 20*](#_Toc494752130)

[*3.1.1. Login Module 20*](#_Toc494752131)

[*3.1.2. Profile Module 20*](#_Toc494752132)

[*3.1.3. Appointment Module 22*](#_Toc494752133)

[*3.1.4. Upload File Module 23*](#_Toc494752134)

[*3.1.5. Add Subject Module (Admin) 23*](#_Toc494752135)

[*3.1.6. Transaction Module 24*](#_Toc494752136)

[*3.1.7. Application Module 25*](#_Toc494752137)

[*3.2. Testing Strategy 28*](#_Toc494752138)

[*3.2.1. Types of Testing 28*](#_Toc494752139)

[*3.2.2. Example of Bottom Up Integration Testing 30*](#_Toc494752140)

[*4. Final Design 32*](#_Toc494752141)

[*4.1. Application Module 32*](#_Toc494752142)

[*4.1.1. Student 32*](#_Toc494752143)

[*4.1.2. Admin (Approving students) 33*](#_Toc494752144)

[*4.2. Login Module 33*](#_Toc494752145)

[*4.2.1. Login Page 33*](#_Toc494752146)

[*4.2.2. Home Page (Student/Supervisor) 34*](#_Toc494752147)

[*34*](#_Toc494752148)

[*4.2.3. Home Page (Admin) 34*](#_Toc494752149)

[*4.3. Profile​ ​Module 36*](#_Toc494752150)

[*4.3.1. Main Page 36*](#_Toc494752151)

[*4.3.2. Update Info 36*](#_Toc494752152)

[*4.3.3. Change​ ​password 37*](#_Toc494752153)

[*4.3.4. Search​ ​profile 37*](#_Toc494752154)

[*4.3.5. Update​ ​progress (For Admin) 38*](#_Toc494752155)

[*4.4. Demo​ ​(Subject​ ​Management​ ​Module) 39*](#_Toc494752156)

[*4.4.1. Main Page 39*](#_Toc494752157)

[*4.4.2. Choosing Subject 39*](#_Toc494752158)

[*4.5. File Management Module 40*](#_Toc494752159)

[*4.5.1. Main Page 40*](#_Toc494752160)

[*4.5.2. Upload File 40*](#_Toc494752161)

# Scenario / Problem Statements:

Today, universities use *post graduate management system (PG-MS)* to manage data of their student. However, most of the system used by these institutions are outdated as they are inconvenient to be used and view in mobile devices as the systems are built on obsolete technologies. Moreover, the system used by some of these universities are not integrated as different components are hosted on multiple domains, making it hard to be used as users have to navigate through different website for different functions.

## Objective:

To tackle the problems above, we decided to:

* Create an integrated PG-MS, with all functionalities all bundle in a centralized platform.
* Make a PG-MS with responsive web pages that is suitable for browsing on both computers and mobile devices of different size.

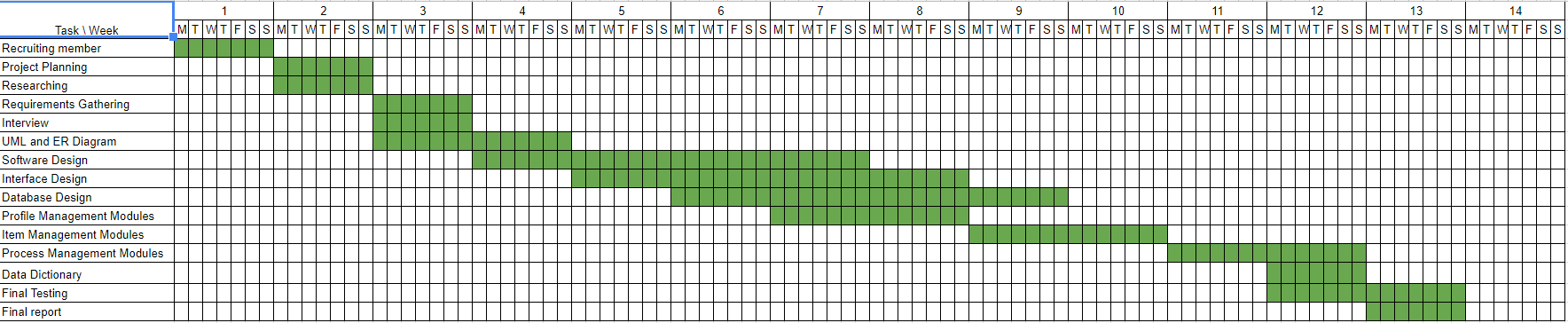
### Scope:

1. Profile Management
   * Login function
   * Logout function
   * View and edit profile information
2. Item Management
   * Add Item
   * Edit Item
   * Delete Item
   * Search Item
   * Item check in/check out process
3. Transaction/Process Management
   * Alert or notification process
   * Payment transaction
   * Report generation

## Functional Requirements

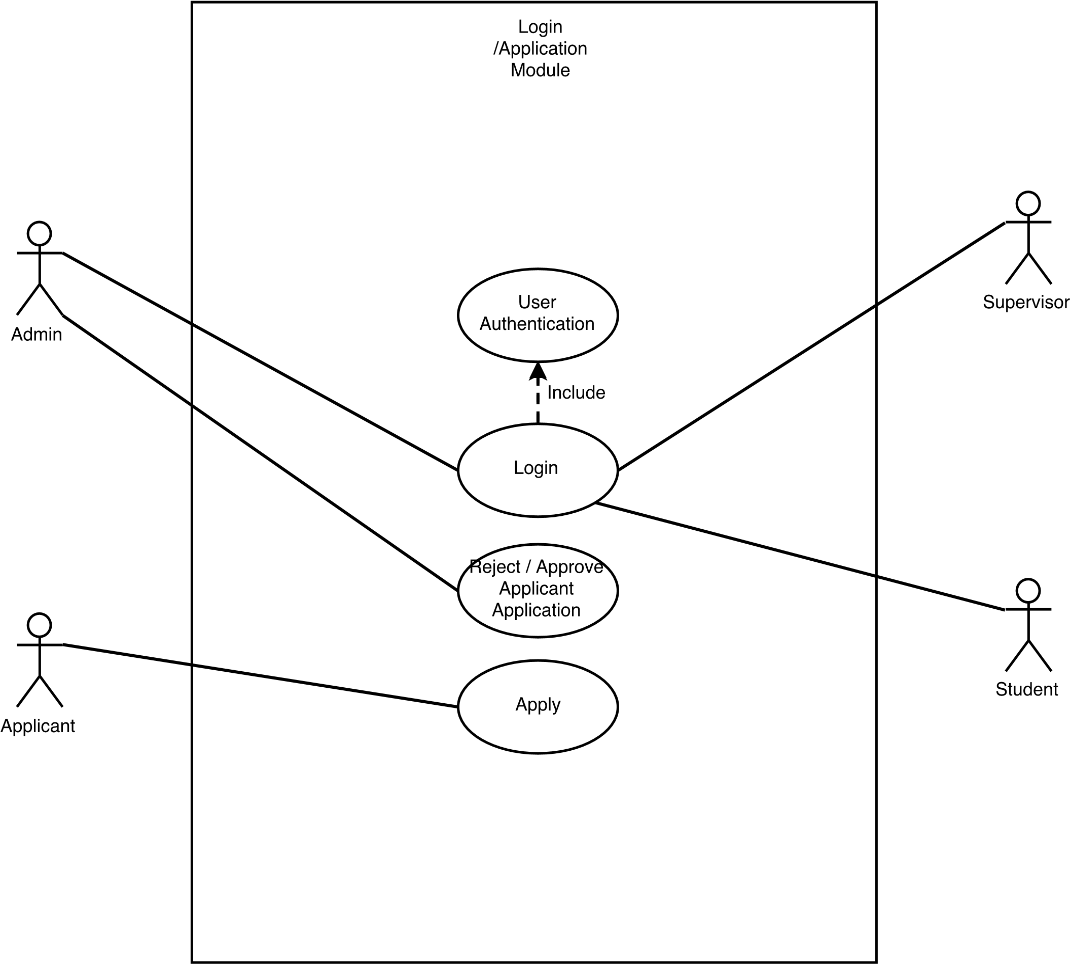
|  |  |  |  |
| --- | --- | --- | --- |
| No. | Module | Requirements | Actor Involved |
| 1 | Login | * **Login function**   Users will be able to login by entering the correct user id and password combination through the login page, and will be redirected to the user’s and admin home page respectively.   * **Logout function**   After logging in, users are able to log out at will by clicking at the logo button and they will be redirected to the website’s homepage. | Student, Supervisor, Admin |
| 2 | Profile Management | * **View and edit profile information**   Users are able to edit their information such as changing password, username and profile picture and also view their master’s progress. | Student, Supervisor, Admin |
| * **Profile search**   Students are able to search for supervisor profile and vice versa. | Student,  Supervisor |
| * **Update progress**   + Supervisors are able to update the progress of the master by research students they’re supervising.   + Supervisors are able to update master by coursework student’s academic report. | Supervisor |
| 3 | Application Management | * **Application function**   Applicant that apply for a post-graduate studies, an account will be created for the applicant only if they’re qualified and approved by admin. Student will need to provide all required information. For research student, they need to submit their research proposal. | Applicant |
| * **Application approval function**   Admin can review the applications and create a student account for the applicants if all the requirements are met. | Admin |
| 4 | File Management | * **Add File**   Student and supervisor are able to submit their documents such as journal, and thesis by uploading the pdf files to the server. | Student, Supervisor |
| * **Delete File**   Students and supervisors are able to delete the files they uploaded. |
| * **Download File**   Students and supervisors are able to download the files that they or their students/supervisors have uploaded to the system. |
| * **Search File**   Students and supervisors will be able to search for the documents they have uploaded in the system. |
| 5 | Appointment Management | * **Appointment Booking**   + Student will be able to make appointment with supervisor. | Student |
| * **Appointment Approval**   + Supervisor are able to approve appointment date booked by the student. | Supervisor |
| 6 | Subject Management | * **Enroll Subject**   Student is able to enroll subject | Student |
| * **Add Subject**   Admin is able to add subject and assign supervisor to that subject. | Admin |
| 7 | Transaction/ Process Management | * **Payment transaction**   + Student may make a payment through the system. | Student |
| * **Report generation**   + Generating report payment and result for student taking master by coursework. |
| 8 | User Management | * **Add user**   Admin are able to add students and supervisors user. | Admin |
| * **Delete user**   Admin are able to delete any user. |
| * **Edit user status**   Admin are able to determine the status of students and supervisors. For students, there are 3 type of status, active, terminated and graduated. For supervisors, the statuses are active and retired. |
| * **Search user**   Admin are able to search any user by their user id and username. |

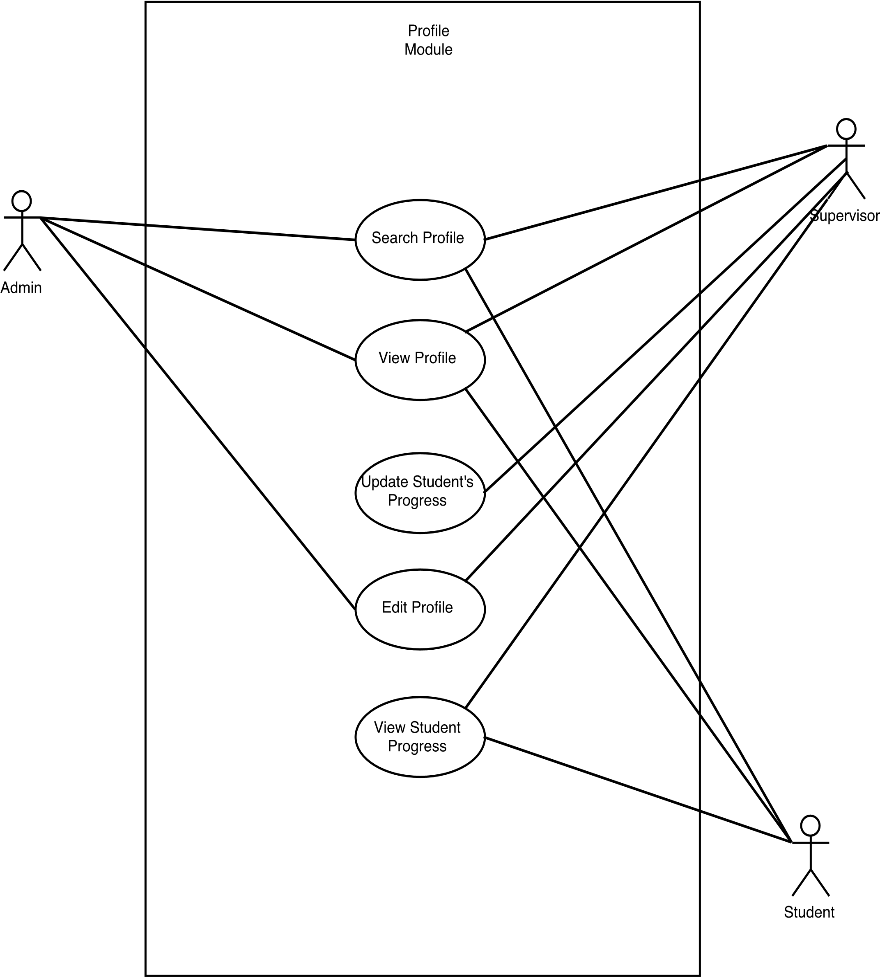
## Gantt Chart

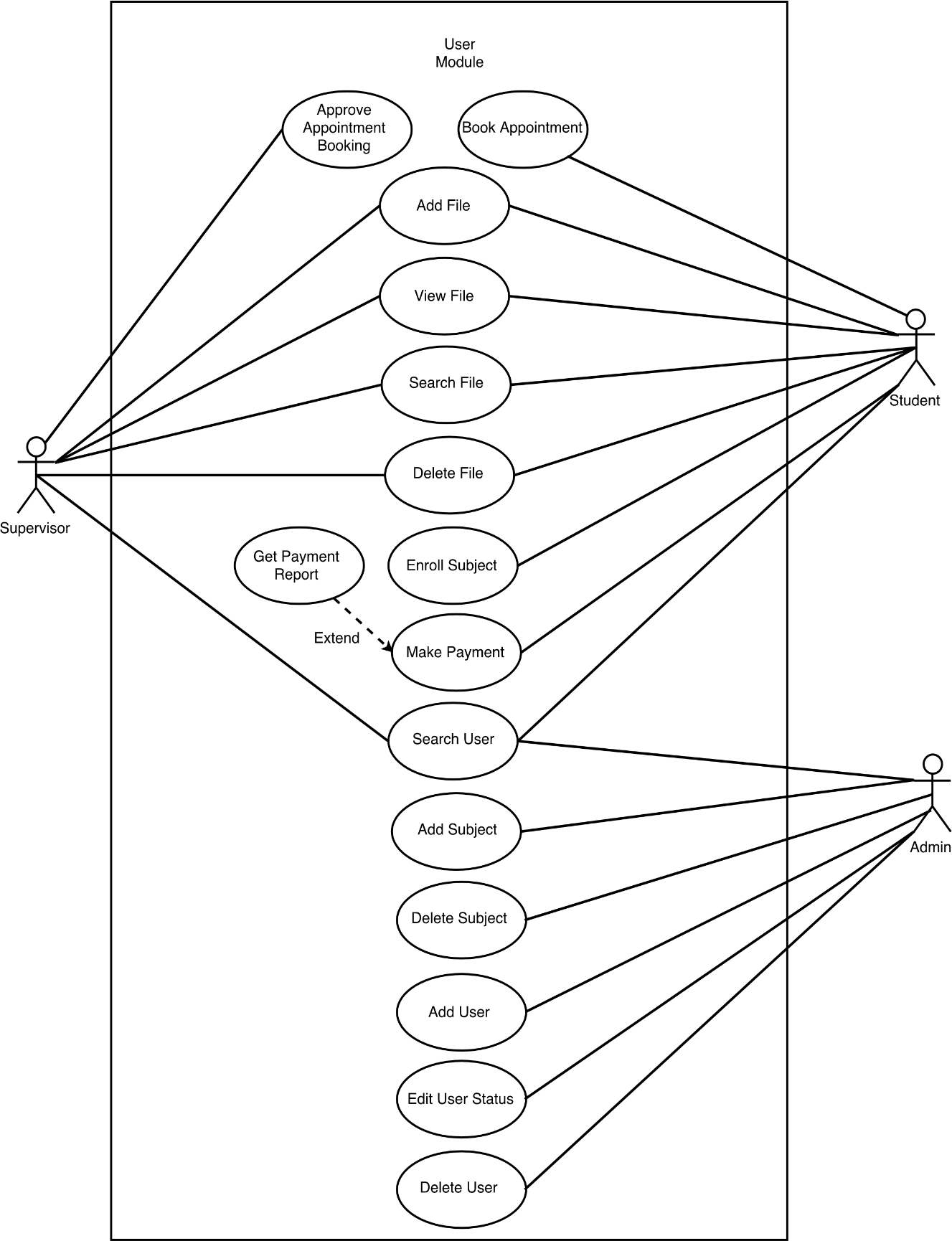


# System Overview

## Use Case Diagram







## C:\Users\Windows\Downloads\ERD - SEF.pngERD Diagram

## Quality Requirement

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Quality Requirement | Description / Metric | Member |
| 1 | Responsive Design | * **Description**   Our web system layout scale will be adjusted automatically with the screen size. Certain element will be hide and display according to the screen size of the device where the application is accessed. The arrangement of elements will be adjusted by this 3 screen size:   * 768px * 992px * 1200px | Goh Kun Shun |
| * **Metric**   Make sure all elements display properly in all 3 screen size |
| 2 | Data Integrity | * **Description**   Accessibility is constrained by the user type. | James |
| * **Metric**   User can only access to specific function that is allow by their user type. For example, admin cannot access or view document uploaded by the student or supervisor. |
| 3 | Security | * **Description**   All users password will be not be saved in raw text, but will be hashed and salted. Each user will be given a different salt. | Eric |
| * **Metric**   User password are encrypted and can’t be access by even the admin. Different salt for different user are given and the first few character of the salt are shown as the proof that each user have different salt even though they have the same password, resulting in different hash. |
| 4 | Portability | * **Description**   Our system will be accessible by laptops, desktop and mobile devices through website domain. | Ong Koon Hua |
| * **Metric**   System testing will be done on a few different computer and phone to make sure that they display the correct output. |
| 5 | Functionality Testing | * **Description**   Our website will include exception handling and error checking to ensure that the user will receive correct response from our system when requested. | Christopher |
| * **Metric**   Wrong input will be put to test the system error handling and ensure that the system give the correct response. |
| No. | Quality Requirement | Description / Metric | Member |
| 1 | Responsive Design | * **Description**   Our web system layout scale will be adjusted automatically with the screen size. Certain element will be hide and display according to the screen size of the device where the application is accessed. The arrangement of elements will be adjusted by this 3 screen size:   * 768px * 992px * 1200px | Goh Kun Shun |
| * **Metric**   Make sure all elements display properly in all 3 screen size |
| 2 | Data Integrity | * **Description**   Accessibility is constrained by the user type. | James |
| * **Metric**   User can only access to specific function that is allow by their user type. For example, admin cannot access or view document uploaded by the student or supervisor. |
| 3 | Security | * **Description**   All users password will be not be saved in raw text, but will be hashed and salted. Each user will be given a different salt. | Eric |
| * **Metric**   User password are encrypted and can’t be access by even the admin. Different salt for different user are given and the first few character of the salt are shown as the proof that each user have different salt even though they have the same password, resulting in different hash. |
| 4 | Portability | * **Description**   Our system will be accessible by laptops, desktop and mobile devices through website domain. | Ong Koon Hua |
| * **Metric**   System testing will be done on a few different computer and phone to make sure that they display the correct output. |
| 5 | Correctness | * **Description**   Our website will include exception handling and error checking to ensure that the user will receive correct response from our system when requested. | Christopher |
| * **Metric**   Wrong input will be put to test the system error handling and ensure that the system give the correct response. |

## Pseudocode

### Login module

**Start**

**if** loginBtn is pressed:

**get** username

**get** password

**if** authenticate(username, password) is success:

Redirect to home\_page

**else**

Print error\_message

**end if**

**end if**

**End**

### AddUser Module

**Start**

**if** loginBtn is pressed:

**get** username

**get** password

**get** usertype

**get** email

**get** dob

**get** gender

**get** address

**if** info is valid:

addUser(username, password,usertype**,** email**,** dob,gender,address)

**end if**

**end if**

**End**

### DeleteUser Module

**Start**

**if** deleteBtn is pressed

deleteUser(username)

**end if**

**End**

### EditUserStatus

**Start**

**if** editBtn is pressed:

editStatus(username)

**get** Status

updateStatus(username, status)

**end if**

**End**

### Application Module

#### Applicant

**Start**

**If** applyBtn pressed:

**get** name

**get** dob

**get** gender

**get** ic

**get** nationality

**get** address

**get** qualification

**get** document

**get** paymentstatus

**get** programme

**get** photo

**get** type

**get** timestamp

**If** application form is full fill:

print success\_message

**else**

Print error\_message

**end** else

**End**

#### Admin

**Start**

**display** application list

**If** approvalBtn pressed:

process to **AddUser Module**

**end if**

**End**

### Transaction Module

**Start**

**if** ( verified payment is selected )

**get** photoFile

**get** receiptRefNo

**while** (photoFile format is not PNG, PDF, GIF, JPEG or receiptRefNo is empty)

printerror message

**get** photoFile

**get** receiptRefNo

**end while**

**else**

**Break**

**end else**

**End**

### Appointment Module

#### Student

**Start**

**get** userSearch

**print** all lecture names

**get** selectedUser

User is directed to lecture profile page

**if** ( make appointment is selected )

**get** time

**get** date

While ( time is less than 1 hour or date is before current date )

**get** time

**get** date

**end while**

**else**

Break

**end if**

Database is updated with time, date, studentID and LecID

// LecID and studentID will be autofilled

#### Lecture

**Start**

**if ( approve is selected )**

Update status of appointment in database to approve

**else if** ( reject is selected)

Update status of appointment in database to reject

**else**

Break

**end if**

**End**

### File Management Module

**Start**

**If** uploadBtn pressed:

**get** file

**get** file\_name

**If** file is valid:

upload(file, file\_name)

**else**:

showError(“File should be in .pdf, .docx format.)

**end if**

**else If** deleteBtn pressed:

**get** file.id

file.delete(pk=id)

**else if** downloadBtn pressed:

**get** file.id

**get** file(pk = id)

**else if** searchBtn pressed:

**get** search

**for** result in file:

**if** search == result

display result

**end if**

**end for**

**end if**

**End**

### Subject Management Module

#### Admin

**Start**

**If** addSubjectBtn pressed:

**get** subject\_name

**get** subject\_fee

**get** subject\_creditHour

subject(subject\_name, subject\_fee, subject\_creditHour )

**End**

#### Student

**Start**

**If** enrollBtn pressed:

**get** subject.id

enroll(student.id, subject.id)

**else If** deleteBtn pressed:

**get** enroll.id

enroll.delete(pk=id)

**end if**

**End**

# Testing

## Test Data

### Login Module

* **Point to take note**
  + **Login with ID number as credentials belong to student account**
  + **Login with Name as credentials belong to Lecturer account**
  + **Login with “admin” as credentials belong to admin account**

|  |  |  |
| --- | --- | --- |
| **Input** | | **Expected Output** |
| **Username** | **Password** |
| 1151101980 | pass1234 | Login success. Redirect to student homepage. |
| 1151101987 | pass1234 | Login fail. Error message“Account disabled”displayed. |
| Chris | pass1234 | Login success. Redirect to supervisor homepage. |
| admin | pass1234 | Login success. Redirect to admin page. |
| admin | 1234 | Login fail. Error message“Invalid login”displayed. |

### Profile Module

|  |  |  |  |
| --- | --- | --- | --- |
| **Change Password** | | | |
| **Input** | | | **Expected Output** |
| **Old Password** | **New Password** | **New Password Confirmation** |
| pass1234 | pass1234 | pass1234 | Valid. |
| pass1234 | pass5678 | pass5678 | Valid. |
| pass1234 | pass1234 | pass5678 | Invalid.  The two password fields didn't match. |
| pass2345 | pass1234 | pass1234 | Invalid.  Old password entered incorrectly. |
| pass1234 | 1234 | 1234 | Invalid.  New password need to be at least 8 character. |
| pass1234 | 12345678 | 12345678 | Invalid.  New password contain only numeric. |
| pass1234 | password | password | Invalid.  New password too common. |
| pass1234 | pass1234 | pass1234 | Login fail. Error message“Invalid login”displayed. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Edit Profile** | | | | |
| **Input** | | | | **Expected Output** |
| **Profile Picture** | **First Name** | **Last Name** | **Address** |
| - | Kun | Shun | 18, Jalan Dewasa,  Batu Tiga | Valid. |
| me.jpg | Kun | Shun | 18, Jalan Dewasa,  Batu Tiga | Valid. |
| demo.pdf | Kun | Shun | 18, Jalan Dewasa,  Batu Tiga | Invalid.  Invalid profile picture file. |
| me.jpg | - | Shun | 18, Jalan Dewasa,  Batu Tiga | Invalid.  First name must not be empty field. |
| me.jpg | Kun | - | 18, Jalan Dewasa,  Batu Tiga | Invalid.  Last name must not be empty field. |
| me.jpg | Kun | Shun | - | Invalid.  Address must not be empty field. |

### Appointment Module

**Points to note**

* **Assuming the details at the time of booking is as follow:**
  + **Date : 10/9/2017**
  + **Time : 12:00 (using 24 hour format)**
* **Booking time slot that is clashing will not be check, it is assume that the lecturer will manage his/her own time schedule**
* **LecID will be done as drop down menu, therefore, no invalid input for lecturer is possible, unless there is none selected**
* **Reason does not have invalid input, unless there is no input.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Add Appointment** | | | | |
| **Input** | | | | **Expected Output** |
| **Date** | **Time** | **LecID** | **Reason** |
| 12/9/2017 | 16:00 | [valid selected ID] | [Reason to consult Lec] | Valid |
| 10/9/2017 | 13:59 | Invalid,  Booking time must be 2 hour of current time |
| 10/9/2017 | 14:00 | Valid |
| 9/9/2017 | 16:00 | Invalid,  Booking date cannot be before today’s date |
| - | 16:00 | Invalid,  Booking date cannot be empty |
| 10/9/2017 | - | Invalid,  Booking time cannot be empty |
| 10/9/2017 | 16:00 | - | Invalid,  LecID cannot be empty |
| 10/9/2017 | 16:00 | [valid selected ID] | - | Invalid,  Reason cannot be empty |

### Upload File Module

|  |  |  |
| --- | --- | --- |
| **Input** | | **Expected Output** |
| **File name** | **File type** |
| Research | pdf, docx | Upload success, redirect to file page. |
| Research | jpg, mp3 | Upload fail. File type must be in .pdf or .docx. |
| - | pdf, docx | Upload fail. File name can’t be blank. |
| - | jpg, mp3 | Upload fail. File name can’t be blank and file type must be in .pdf or .docx. |

### Add Subject Module (Admin)

|  |  |  |  |
| --- | --- | --- | --- |
| **Input** | | | **Expected Output** |
| **Subject Name** | **Credit Hour** | **Subject Fee** |
| Software Engineering Fundamental | 4 | 1234.50 | Add subject success. Redirect to subject management page. |
| Software Engineering Fundamental | 4.0 | 1234.50 | Add subject success. Redirect to subject management page. |
| Software Engineering Fundamental | 4 | 1234 | Add subject success. Redirect to subject management page. |
| Software Engineering Fundamental | 4.1 | 1234.50 | Add subject fail. Credit Hour must be in integer. |
| Software Engineering Fundamental | 4 | 1234.500 | Add subject fail. Subject Fee must be in two decimal places. |
| - | 4 | 1234.50 | Add subject fail. Subject name can’t be blank. |

### Transaction Module

**Points to take note**

* **File type that are accepted are.**
  + **pdf, PNG, jpg and gif only.**
* **RefNo does not have invalid input unless there are no input.**

|  |  |  |
| --- | --- | --- |
| **Make Transaction** | | |
| **Input** | | **Expected Output** |
| **[uploaded file type]** | **RefNo** |
| .pdf | [Resit Reference Number] | Valid |
| .gif | Valid |
| .mp4 | Invalid,  File type invalid |
| - | Invalid,  No file is uploaded |
| .PNG | - | Invalid,  RefNo cannot be empty |

### Application Module

* **Points to take note**
  + **This module is separated into 3 table to make it simpler to identify the error, however, all details in 3 table will be submitted together from a single-page view.**
  + **Number is not allow for Name input.**

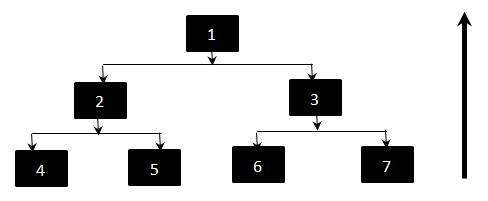
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Application** | | | | |
| **Input** | | | | **Expected Output** |
| **Name** | **Nationality** | **Type** | **Programme** |
| Joker | Malaysia | By Research | Master of Business | Valid,return to login page |
| - | Malaysia | By Research | Master of Business | Invalid,Please enter your name |
| 123 | Malaysia | By Research | Master of Business | Invalid,Name only consist of alphabet and space |
| Joker | - | By Research | Master of Business | Invalid, Please select your nationality |
| Joker | Malaysia | - | Master of Business | Invalid, Please select your type |
| Joker | Malaysia | By Research | - | Invalid,Please select the programme you interest |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Application** | | | | |
| **Input** | | | | **Expected Output** |
| **Birthday Date** | **Gender** | **IC** | **Qualification** |
| - | Male | 960310105777 | Qualification.pdf | Invalid,Please insert your date of birth |
| 03/10/1996 | - | 960310105777 | Qualification.pdf | Invalid,Please select your gender |
| 03/10/1996 | Male | - | Qualification.pdf | Invalid,Please enter your identity number and do not include - |
| 03/10/1996 | Male | dasdasdas | Qualification.pdf | Invalid,The identity number only consist number |
| 03/10/1996 | Male | 3123 | Qualification.pdf | Invalid,Please enter value between 12 and 12 character long |
| 03/10/1996 | Male | 960310105777 | - | Invalid,Please select a file and file only accept pdf file |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Application** | | | | | |
| **Input** | | | | | **Expected Output** |
| **Document** | **Photo** | **Mobile number** | **Email Address** | **Address** |
| - | photo.png | 0123456789 | example@gmail.com | 18, Jalan Dewasa,  Batu Tiga | Invalid,Please select a file and file only accept pdf file |
| Document.pdf | - | 0123456789 | example@gmail.com | 18, Jalan Dewasa,  Batu Tiga | Invalid,Please select an image and image only accept png and jpg type |
| Document.pdf | photo.png | dsadsa | example@gmail.com | 18, Jalan Dewasa,  Batu Tiga | Invalid,The phone number only consist number |
| Document.pdf | photo.png | 01123 | example@gmail.com | 18, Jalan Dewasa,  Batu Tiga | Invalid,Please enter value between 10 and 11 character long |
| Document.pdf | photo.png | - | example@gmail.com | 18, Jalan Dewasa,  Batu Tiga | Invalid,Please insert your phone number |
| Document.pdf | photo.png | 0123456789 | - | 18, Jalan Dewasa,  Batu Tiga | Invalid,Email address is required and cannot be empty |
| Document.pdf | photo.png | 0123456789 | example@ | 18, Jalan Dewasa,  Batu Tiga | Invalid,Email address is invalid |
| Document.pdf | photo.png | 0123456789 | example@gmail.com | - | Invalid,Please insert address |

## Testing Strategy

To fulfill our quality requirement, we will be performing two types of testing on our system: unit testing and integration testing. The overall testing strategy is **bottom up integration testing**. Following this strategy, our modules will divided into submodule and form a tree. Each modules at lower hierarchy is tested individually (unit testing) and then the modules that rely upon these modules are tested.



Following the diagram above, we will be testing 4, 5, 6 and 7 individually (unit testing). After testing all the modules at the bottom layer. We will start integration testing in pair with 4 and 2, 5 and 2 and so on. While testing 4 and 5, 2 are replaced with what is called a driver.

### Types of Testing

In order to satisfy each of our quality requirement, we will be applying different types of testing to test our system.

#### Responsive Design Testing

Ensure all the elements in the user interface are displayed properly and doesn’t affect the user’s ability to understand and interact with it.

These elements include:

* + Buttons
  + Images
  + Texts
  + Modals
  + Icons
  + Forms

To test this, we will be using Chrome Developer Tools to simulate how every web page will display in 4 screen-width settings:

* + Extra small devices Phones (<768px)
  + Small devices Tablets (≥768px)
  + Medium devices Desktops (≥992px)
  + Large devices Desktops (≥1200px)

#### Data Integrity Testing

* Testing for accessibility of function:
  + Testing whether or not all user type can only access the functions that system allowed.
  + Eg: File management page can only accessed by students and supervisors but admin.
  + We will copy the url of the function ( eg:/file/ ) and login as the user type that is not allowed to access the function to run the url. If the access is denied that means our testing is success.
* Testing for accessibility of data:
  + Testing whether or not user can access the data that related with them.
  + Eg: Student can view the file that upload by their supervisor or vice versa, but can’t view the file uploaded by other students or supervisors.
  + To test this, we will copy the file url( eg:file/<file\_id>/ ) of one of the user and run it using another user account. If the access is denied means our testing is success.

#### Security Testing

* Correct implementation of user login function
  + Username and password can login correctly
* Hashed and salted password will be checked if some users’ password are same
  + Raw password is tested not to show in the user table which open the file by using SQLite
* Perspective of admin
  + No password is allow to show to anyone
* The functionality of password encryption

#### Functionality Testing

* Forms checking
  + Make sure that all forms are fill with correct input (e.g. file uploaded must be in a specific format only), if the forms have been input with invalid input, error message will pop up prompting user to fill the forms with the correct details.
  + Scenario : If user decided to upload/change their profile picture, user must only use file format such as PNG, JPEG and GIF. If user use other file format such as PDF, the system will notify the user to use on PNG, JPEG and GIF for their file input.
* Test all links
  + Check links to ensure user is directed to the section/page that it was intended.

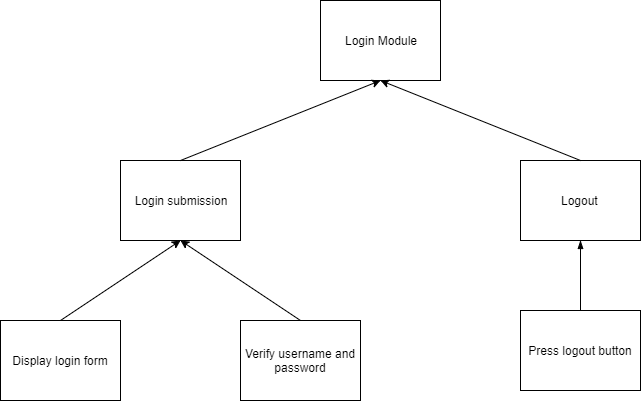
#### Portability Testing

* To ensure user can access by multiple type of device and different web browser through the website.

The device included:

* + Smartphone
  + Desktop
  + Laptop
  + Tablet
* The browser that will be tested on every device (if applicable) includes:
  + Google Chrome
  + Mozilla Firefox
  + Internet Explorer 11
  + Microsoft Edge
  + Safari
* To this test, we will use multiple device to access the website and ensure the output is works fine and display correctly as we expected in different type of device.

### Example of Bottom Up Integration Testing



In our login module, the module itself is subdivided into sub-modules, namely Login Submission and Logout. So in order to test our Login Submission modules, we will have to test the sub-module’s sub-modules, Display Login Form and Verify Username and Password.

We will start by testing the Display Login Form module. We will have to make sure the the labels for the form are displayed properly, and the forms work properly, eg. the username form will show everything the user types, while password form will hide the characters with ‘\*’ character.

Now to test Verify Username and Password, we could’ve test this module by inserting the password and username in variables and pass it to the login function to test it.

Now that both sub-modules of Login Submissions are working, we now can test Login Submission itself. To test this module we will use the form from the Display Login Form submodule and press the Login button. The input from the login form will be passed to the Verify Username and Password module and user will be logged or not logged in depending on the inputs.

However, we still have to check the Logout sub-module before we declare the Login module tested. We have make sure the Logout button is displayed to the user. If the user clicked the Logout button and is redirect to the Login page, it’s considered check. To test if the Logout truly works, we can try to access the homepage again and see if we got redirected back to login page. If it does, it means the Login sub-module is working.

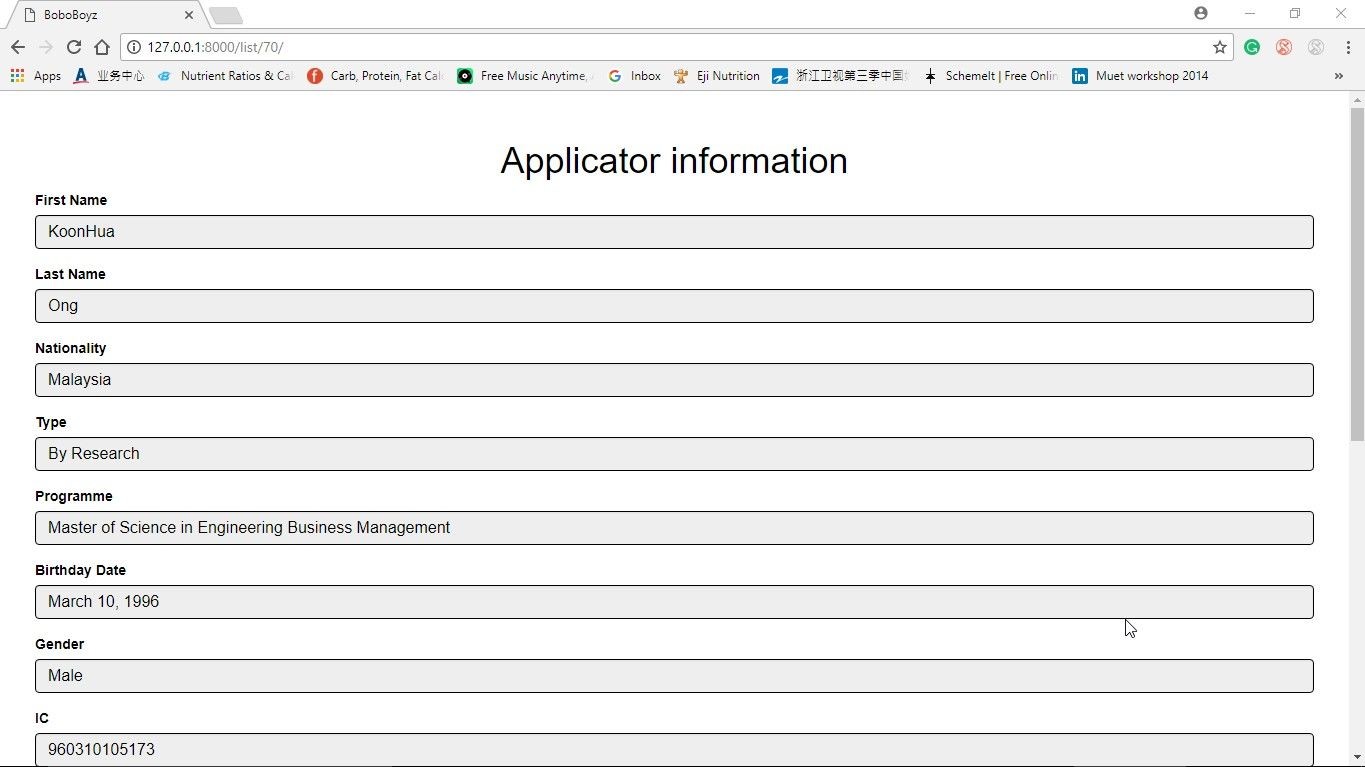
With all the sub-modules tested, we now can declare the Login module tested.

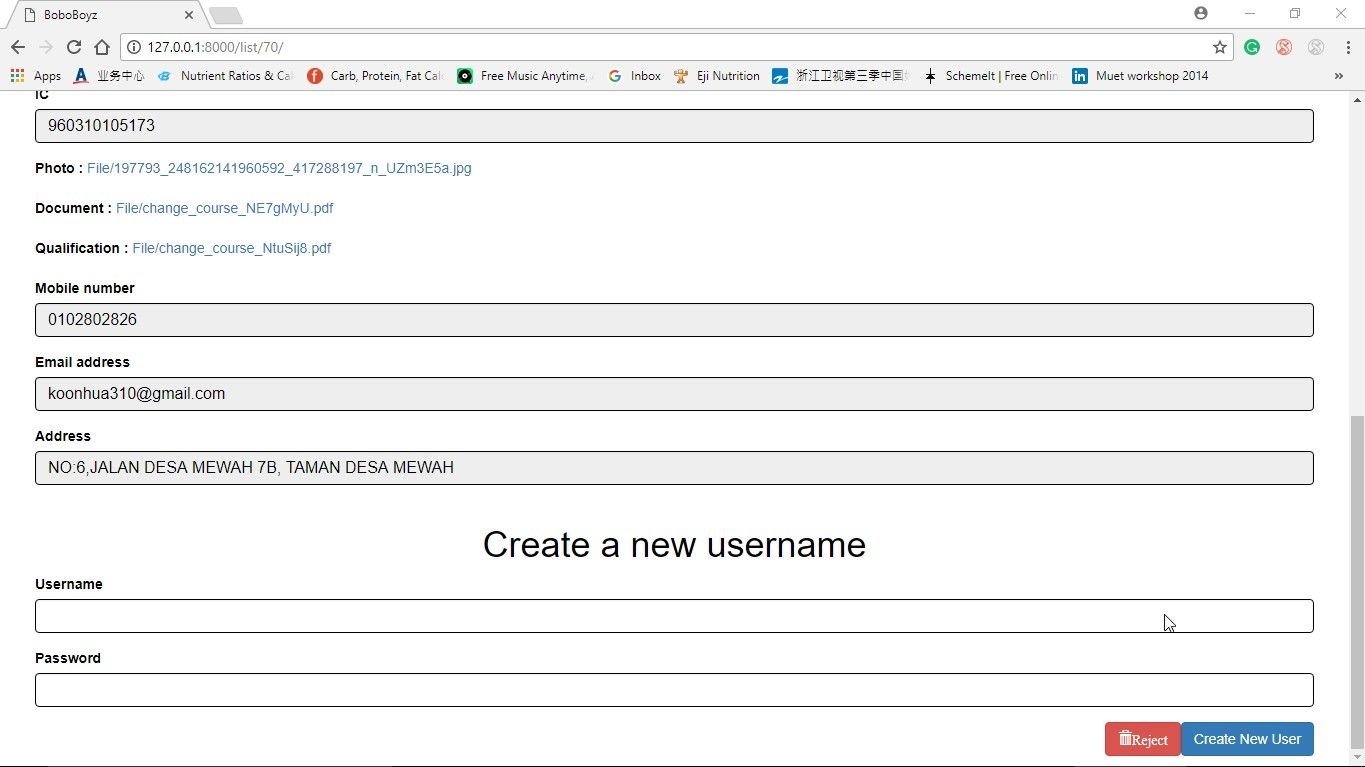
# Final Design

## Application Module

### Student

### Admin (Approving students)

​



## Login Module

### Login Page

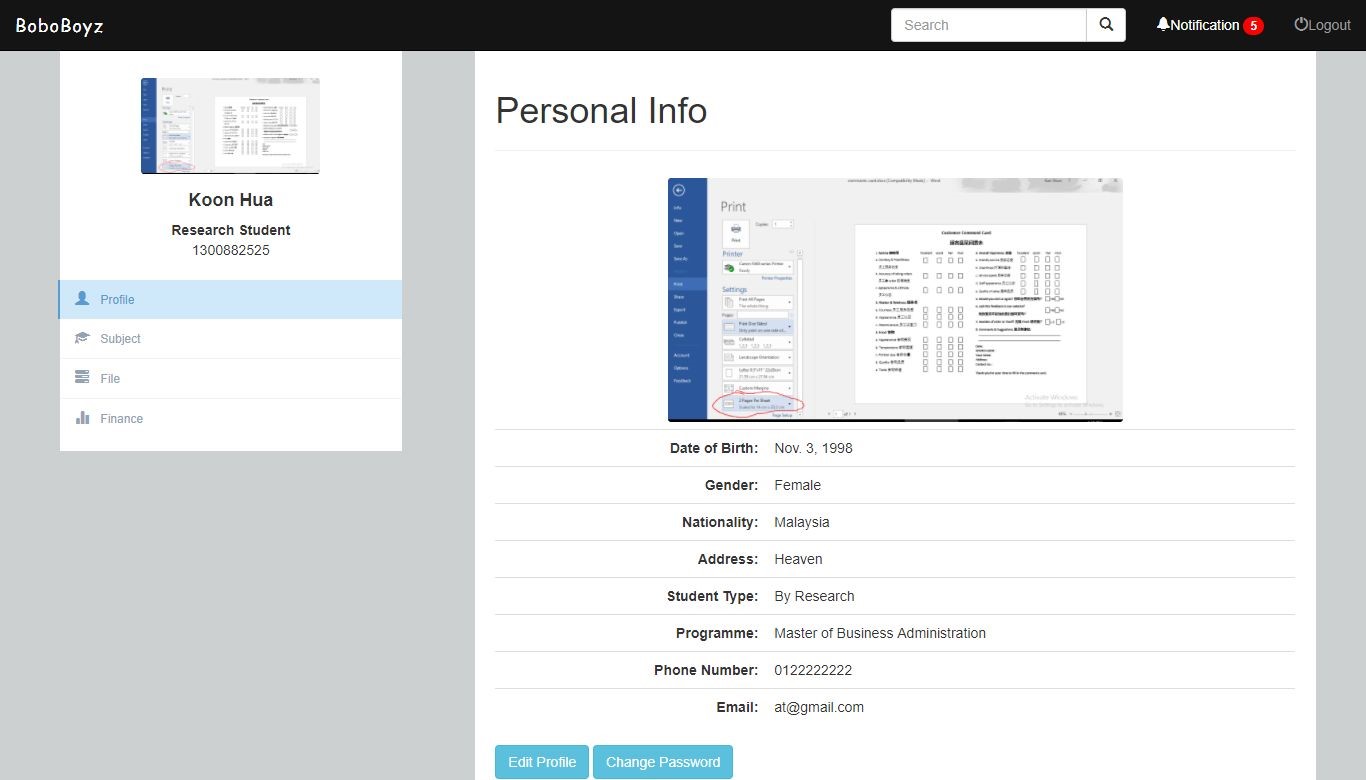
### Home Page (Student/Supervisor)

### 

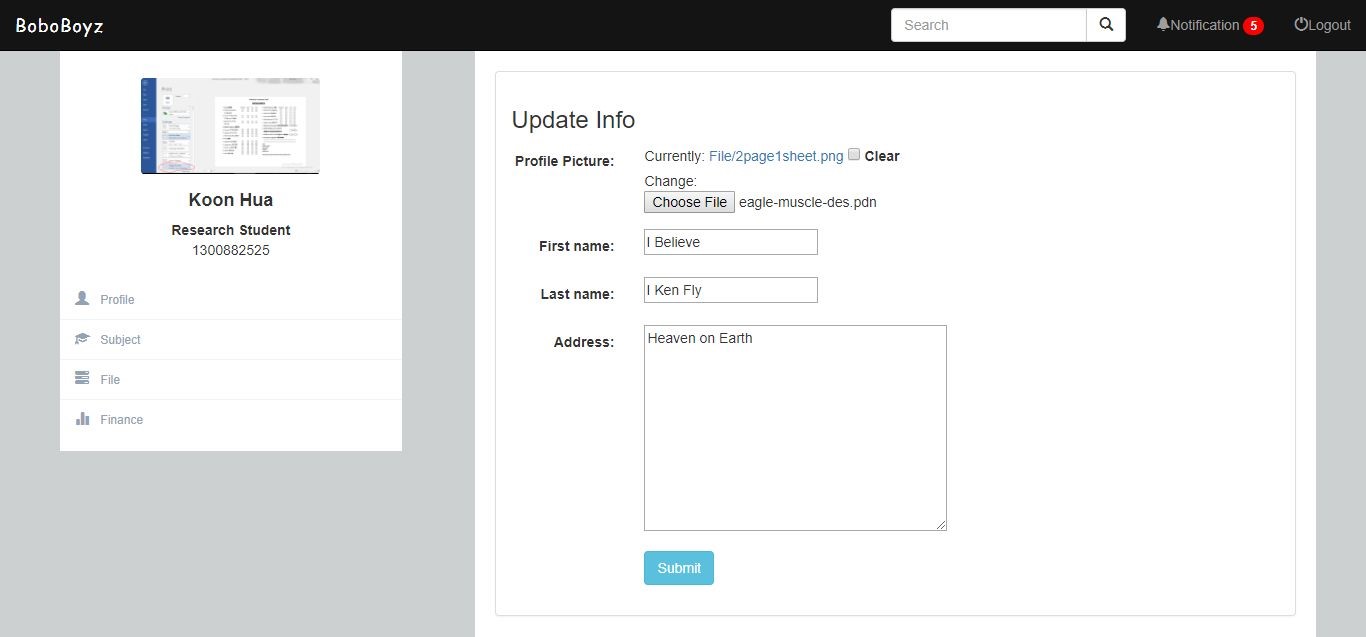
### Home Page (Admin)

## Profile​ ​Module

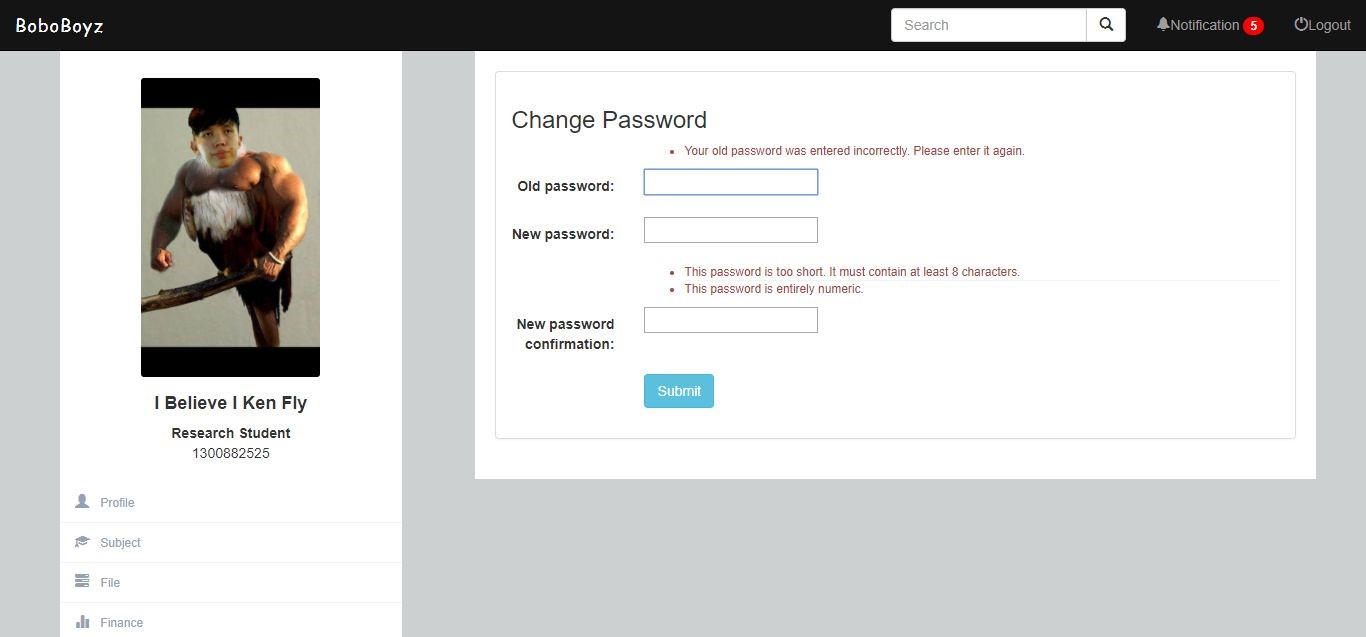
### Main Page



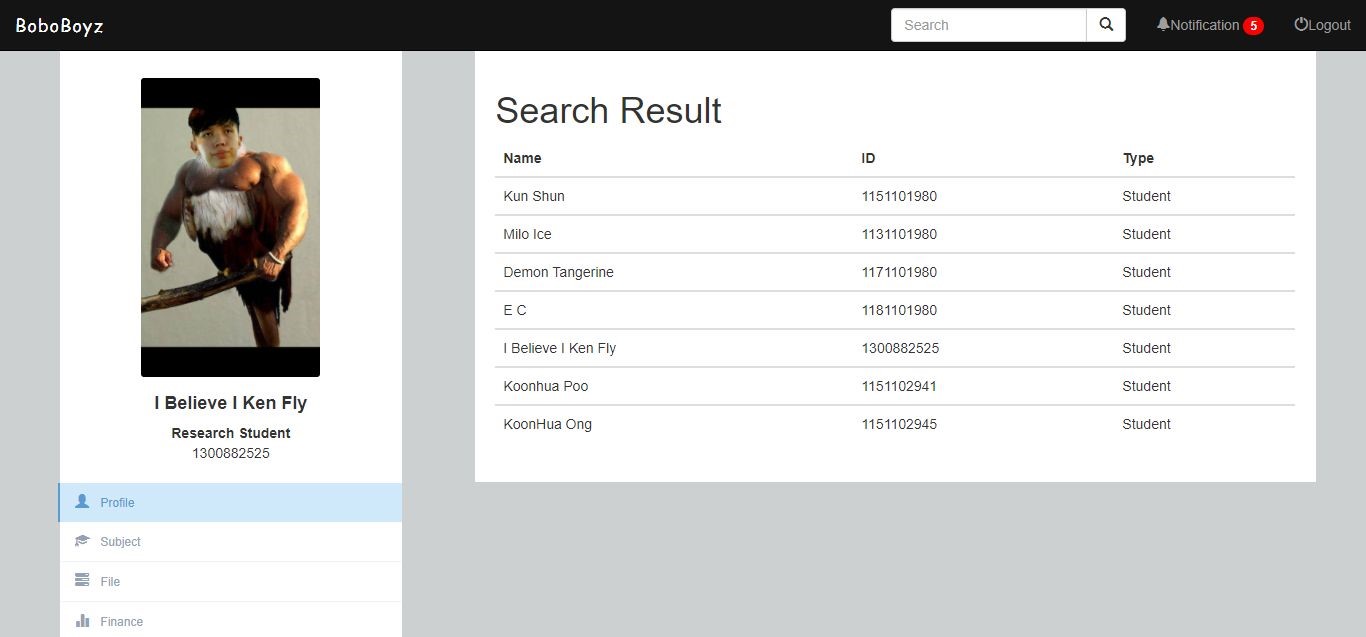
### Update Info

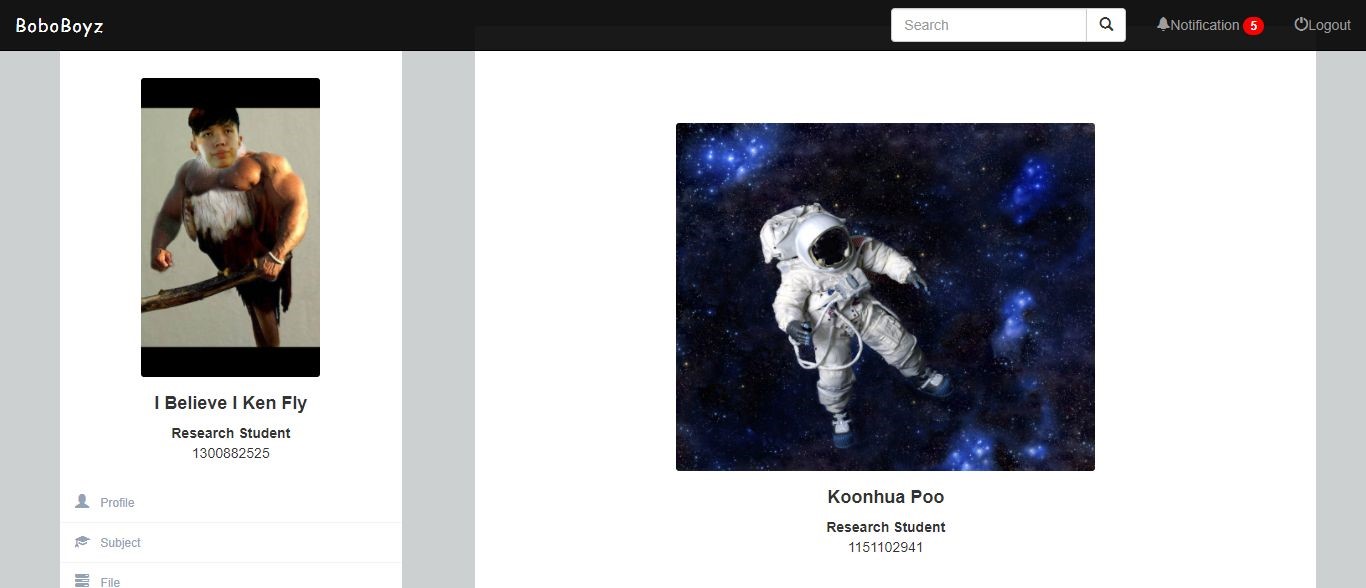


### Change​ ​password

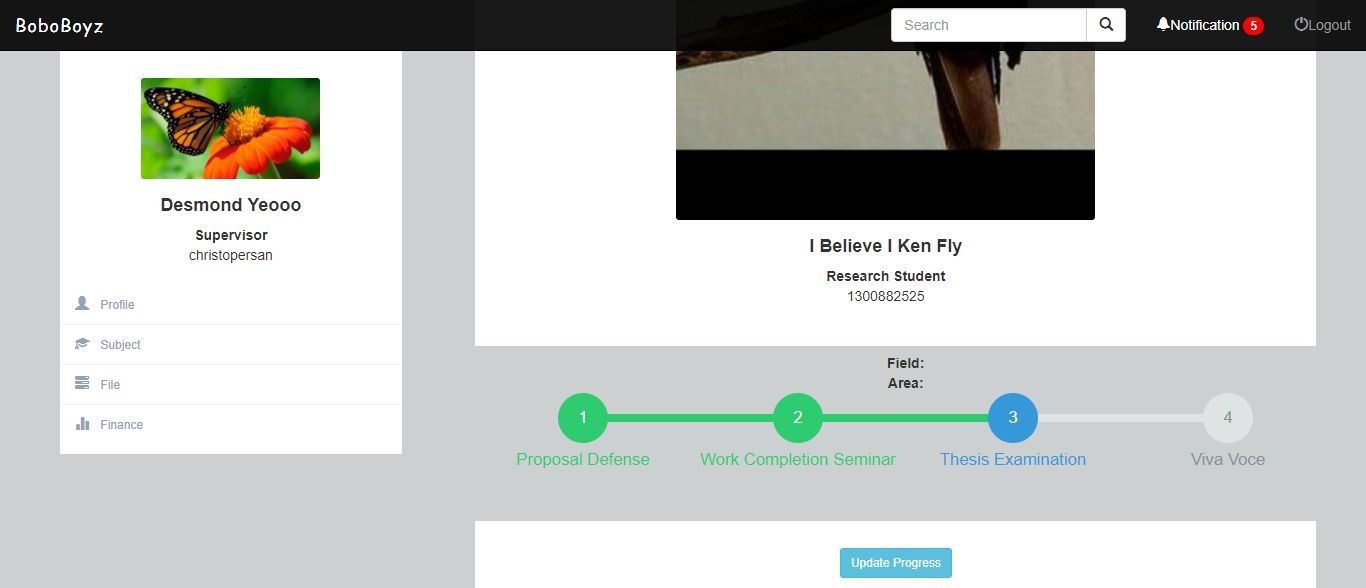


### Search​ ​profile

 ​



### Update​ ​progress (For Admin)



## Demo​ ​(Subject​ ​Management​ ​Module)

### Main Page

### Choosing Subject

## File Management Module

### Main Page

### Upload File

