

**GROUP COURSEWORK REPORT ON**

**ENTERPRISE WEB SOFTWARE DEVELOPMENT**

**(COMP 1640)**

Group member: **Trinh Nguyen Huong Kieu**

**Huynh Tan Phat**

**Ho Ngoc Tan (ht8367s)**

**Vo Vu Duy (vd2973u)**

**Ngo Chi Kiet**

**Le Do Minh Triet (lt2314o)**

Word counts:

Date of submition:

# **CONTENT**

[**CONTENT** 2](#_Toc65885583)

[**1.** **Introduction** 3](#_Toc65885584)

[**2.** **Database** 4](#_Toc65885585)

[**3.** **Functionality** 8](#_Toc65885586)

[**3.1.** **Use Case Diagram** 8](#_Toc65885587)

[**3.2.** **Use Case Specification** 8](#_Toc65885588)

[**3.3.** **Class Diagram** 8](#_Toc65885589)

[**3.4.** **Activity Diagram** 8](#_Toc65885590)

[**3.5.** **Sequence Diagram** 8](#_Toc65885591)

[**4.** **Design** 8](#_Toc65885592)

[**5.** **Testing** 8](#_Toc65885593)

[**5.1.** **Test plan** 8](#_Toc65885594)

[**5.2.** **Smoke Testing** 14](#_Toc65885595)

[**5.3.** **Integration Testing** 15](#_Toc65885596)

[**5.4.** **Test log** 15](#_Toc65885597)

[**6.** **Agile Method** 16](#_Toc65885598)

[**6.1.** **Daily Meetings** 16](#_Toc65885599)

[**6.2.** **Sprint Review** 16](#_Toc65885600)

[**6.3.** **Burndown Charts** 17](#_Toc65885601)

[**7.** **Some image of product** 17](#_Toc65885602)

[**8.** **Appendix** 17](#_Toc65885603)

[**9.** **Reference** 17](#_Toc65885604)

# **FIGURE CONTENT**

[Figure 1. ERD relationships 5](https://uogcloud-my.sharepoint.com/personal/ht8367s_gre_ac_uk/Documents/COMP%201640%20Enterprise%20Web/Team/Report/Coursework%20COMP%201640%20Enterprise%20Web_Team.docx#_Toc65922515)

[Figure 2. ERD diagram for system 6](https://uogcloud-my.sharepoint.com/personal/ht8367s_gre_ac_uk/Documents/COMP%201640%20Enterprise%20Web/Team/Report/Coursework%20COMP%201640%20Enterprise%20Web_Team.docx#_Toc65922516)

[Figure 3. Use case Login 9](https://uogcloud-my.sharepoint.com/personal/ht8367s_gre_ac_uk/Documents/COMP%201640%20Enterprise%20Web/Team/Report/Coursework%20COMP%201640%20Enterprise%20Web_Team.docx#_Toc65922517)

# **TABLE CONTENT**

[Table 1. Multiplicity constraints 5](#_Toc65922527)

[Table 2. Description Role table 6](#_Toc65922528)

[Table 3. Description Account table 7](#_Toc65922529)

[Table 4. Description Comment table 7](#_Toc65922530)

[Table 5. Description Faculty table 7](#_Toc65922531)

[Table 6. Description Contribution table 8](#_Toc65922532)

[Table 7. Description File table 8](#_Toc65922533)

[Table 8. Description Magazine table 8](#_Toc65922534)

[Table 9. Description AcademicYear table 8](#_Toc65922535)

[Table 10. Use Case: Login 9](#_Toc65922536)

[Table 11. Use Case: Manage account 10](#_Toc65922537)

[Table 12. Use Case: Manage role 12](#_Toc65922538)

[Table 13. Use Case: Manage faculty 14](#_Toc65922539)

[Table 14. Use Case: Manage academic year 16](#_Toc65922540)

[Table 15. Use Case: Manage magazine 17](#_Toc65922541)

[Table 16. Use Case: Manage contribution 20](#_Toc65922542)

[Table 17. Test log 28](#_Toc65922543)

[Table 18. Links 30](#_Toc65922544)

# **Introduction**

With the scenario some universities now need a system to collect and manage student contributions to the University’s annually published magazine, through this course, We decided to form a team to develop the “Contribution Collection System (CCS)” system building project.

This document aims to present the results of the team's efforts to conceptualize, design and build a CSS: system that enables universities to collect and manage contributions from faculty students, interact with student contributions for the best articles and photos for the University’s annual magazine publishing.

By taking advantage of the Agile Scrum methodology's advantages, the team shortened the software release time, completed the system in 6 weeks, and ensured the best product quality during deployment. Thereby also improving team spirit, optimizing the efficiency and efforts of the development team.

In particular, we would like to express special gratitude to lecturer Ho Nguyen Phuoc Bao, who has supported us throughout the course.

**About Team**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Name** | **User** | **Role** |
| 1 | Huynh Tan Phat |  | Scrum master; Back-end developer |
| 2 | Ho Ngoc Tan | ht8367s | Product Owner; Tester |
| 3 | Vo Vu Duy | vd2973u | Back-end developer |
| 4 | Trinh Nguyen Huong Kieu |  | Designer |
| 5 | Ngo Chi Kiet |  | Designer |
| 6 | Le Do Minh Triet | lt2314o | Front-end developer |

# **Database**

* **ERD relationships**

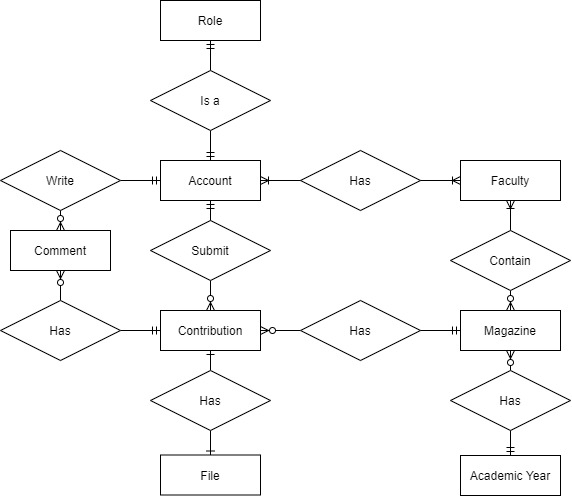


Figure 1. ERD relationships

Table 1. Multiplicity constraints

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entity** | **Multiplicity** | **Relationship** | **Multiplicity** | **Entity** |
| Role |  | Is a |  | Account |
| Account | 1…1 | Write | 0…\* | Comment |
| Account | 1…\* | Has | 1…\* | Faculty |
| Account | 1…1 | Submit | 0…\* | Contribution |
| Comment | 0…\* | Has | 1…1 | Contribution |
| Faculty | 1…\* | Contain | 0…\* | Magazine |
| Magazine | 1…1 | Has | 0…\* | Contribution |
| Magazine | 0…\* | Has | 1…1 | AcademicYear |
| Contribution | 1…1 | Has | 1…1 | File |

* **ERD Diagram**

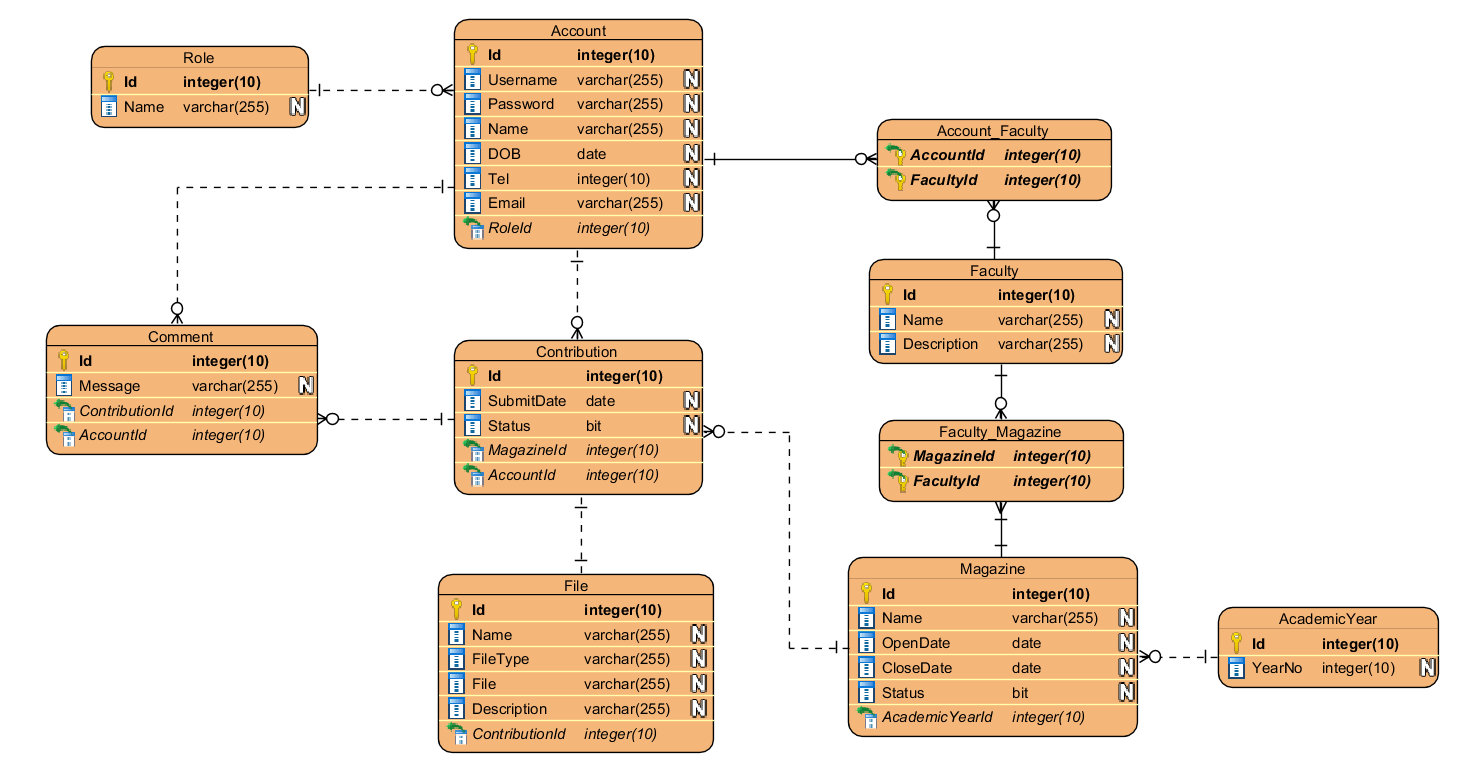


Figure 2. ERD diagram for system

Through user requirement analysis, we have designed an optimized database system consisting of 8 tables with different storage functions like: storage user information (Account), storage files that users have submitted to the system (File), storage information about journals that have been published or are being opened, ect. Below is a description of each table.

Table 2. Description Role table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute name** | **Description** | **PK/ FK** | **Data type** | **Length** | **Nulls** |
| Id | Id Role | PK | Integer | 10 | No |
| Name | Role name |  | Varchar | 255 | No |

Table 3. Description Account table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute name** | **Description** | **PK/ FK** | **Data type** | **Length** | **Nulls** |
| Id | Account id | PK | Integer | 10 | No |
| Roleld | Role of user (link to Role table) | FK | Integer | 10 | No |
| Username | Username to login |  | Varchar | 255 | No |
| Password | Password for account |  | Varchar | 255 | No |
| Name | Full name |  | Varchar | 255 | No |
| DOB | Date of birthday of use |  | DateTime |  |  |
| Tel | Phone number of user |  | Integer | 10 |  |
| Email | Email of user |  | Varchar | 255 | No |

Table 4. Description Comment table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute name** | **Description** | **PK/ FK** | **Data type** | **Length** | **Nulls** |
| Id | Comment id | PK | Integer | 10 | No |
| AccountId | Account id (link to Account table) | FK | Integer | 10 | No |
| ContributionId | Contribution id (link to Contribution table) | FK | Integer | 10 | No |
| Message | Content of comment |  | Varchar | 255 |  |

Table 5. Description Faculty table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute name** | **Description** | **PK/ FK** | **Data type** | **Length** | **Nulls** |
| Id | Faculty id | PK | Integer | 10 | No |
| Name | Name of faculty |  | Varchar | 255 | No |
| Description | Description |  | Varchar | 255 |  |
| MagazineId | Magazine id (link to Magazine table) | FK | Integer | 10 | No |

Table 6. Description Contribution table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute name** | **Description** | **PK/ FK** | **Data type** | **Length** | **Nulls** |
| Id | Contribution id | PK | Integer | 10 | No |
| SubmitDate | Date of submit |  | DateTime |  | No |
| Status | Status of contribution |  | Bit |  | No |
| MagazineId | Magazine id (link to Magazine table) | FK | Integer | 10 | No |
| AccountId | Account id (link to Account table) | FK | Integer | 10 | No |

Table 7. Description File table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute name** | **Description** | **PK/ FK** | **Data type** | **Length** | **Nulls** |
| Id | File id | PK | Integer | 10 | No |
| Name | Name of file |  | Varchar | 255 | No |
| ContributionId | Contribution id (link to Contribution table) | FK | Integer | 10 | No |
| FileType | File type (doc or img) |  | Varchar | 255 | No |
| File | File URL |  | Varchar | 255 | No |
| Description | File description |  | Varchar | 255 |  |

Table 8. Description Magazine table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute name** | **Description** | **PK/ FK** | **Data type** | **Length** | **Nulls** |
| Id | Magazine id | PK | Integer | 10 | No |
| Academic YearId | Academic Year Id (link to AcademicYear table) | FK | Integer | 10 | No |
| Name | Name of magazine |  | Varchar | 255 | No |
| OpenDate | Date of open magazine |  | DateTime |  | No |
| CloseDate | Date of close magazine |  | DateTime |  | No |
| Status | Status of magazine |  | bit |  | No |

Table 9. Description AcademicYear table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute name** | **Description** | **PK/ FK** | **Data type** | **Length** | **Nulls** |
| Id | Academic Year id | PK | Integer | 10 | No |
| Name | Year of academic |  | DateTime | 255 | No |

# **Functionality**

## **Use Case Diagram**

* **Login**

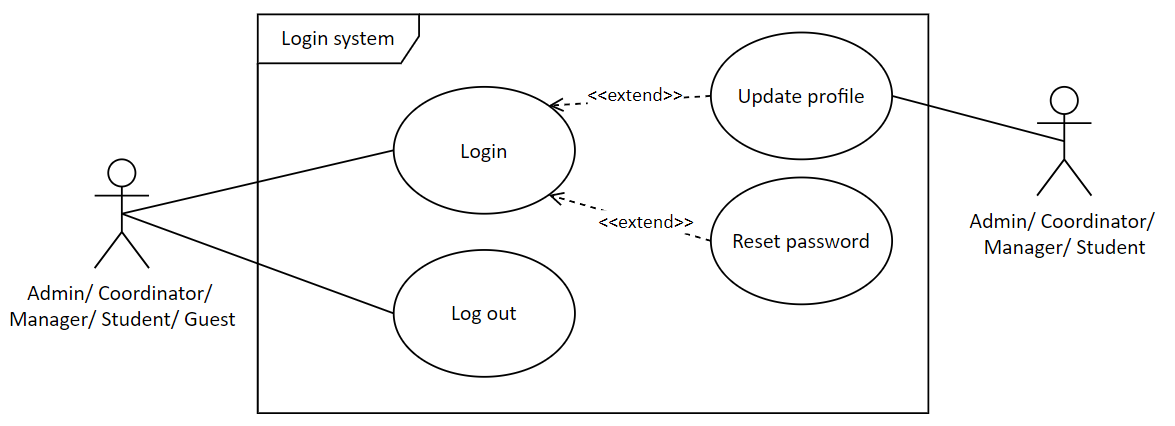


Figure 3. Use case Login

Table 10. Use Case: Login

|  |  |
| --- | --- |
| **Use Case Name** | Login |
| **Description** | As a user, I want to login to the system to use the service of the application |
| **Actor(s)** | Admin, Coordinator, Manager, Student, Guest |
| **Priority** | Must have |
| **Trigger** | User want to login to the application |
| **Pre-Condition(s)** | User’s account has been created  User’s account has been authenticated  User’s device is connected to the internet |
| **Post-Condition(s)** | User login successfully |
| **Basic Flow** | 1. User accesses to Url of the web application  2. User types username/password  3. User clicks login button  4. System checks account and allows user access to the web application. |
| **Alternative Flow** | None |
| **Exception Flow** | 4b. System checks account failed and announces to user.  4b1. User chooses “Forget password”  4b2. User types registered email with their account and click Reset password button  4b3. User logins email and reset password  Use case return step 1 |

* **Manage account**

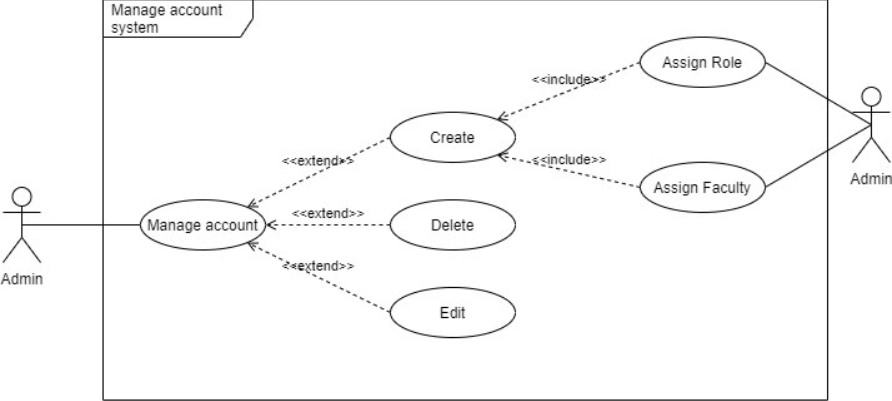


Table 11. Use Case: Manage account

|  |  |
| --- | --- |
| **Use Case Name** | Manage account |
| **Description** | As a user, I want to manage account of the system |
| **Actor(s)** | Admin |
| **Priority** | Medium |
| **Trigger** | User wants to Create, delete or edit account |
| **Pre-Condition(s)** | User must login successfully by account assigned with Admin role  User must choose Manage account functions. |
| **Post-Condition(s)** | User will be moved page that have account management functions. |
| **Basic Flow** | 1. User chooses manage account in Admin page. |
| **Alternative Flow** | 2a. User clicks Create account button  2a1. User types account and password.  2a2. User assigns role to account.  2a3. User assigns faculty to account.  2a4. User clicks create new account button.  2a5. System adds new account to database.  2b. User click delete button of any account in the table.  2b1. User clicks confirm delete.  2b2. System deletes chosen account.  2c. User clicks Edit button of any account in the table.  2c1. User edits password/role/faculty of the account.  2c2. User clicks confirm edit button.  2c3. System edits account to database. |
| **Exception Flow** | 2a5.1. System adds new account failed because user leave blanks.  2a5.2. System announces to user the blank(s) field.  Return to step 2a1.  2b1.1. User clicks Cancel delete  Use case end.  2c2.2 User clicks Cancel edit button  Use case end. |

* **Manage role**

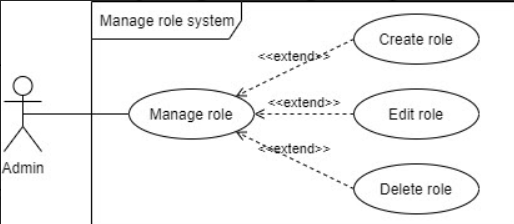


Table 12. Use Case: Manage role

|  |  |
| --- | --- |
| **Use Case Name** | Manage role |
| **Description** | As a user, I want to manage role of the system |
| **Actor(s)** | Admin |
| **Priority** | Medium |
| **Trigger** | User wants to Create, delete or edit role |
| **Pre-Condition(s)** | User must login successfully by account assigned with Admin role  User must choose Manage role functions. |
| **Post-Condition(s)** | User will be moved page that have role management functions. |
| **Basic Flow** | 1. User chooses manage role in Admin page. |
| **Alternative Flow** | 2a. User clicks Create role button.  2a1. User types role name and description.  2a2. User clicks create new role button.  2a3. System adds new role to database.  2b. User click delete button of any Role in the table.  2b1. User clicks confirm delete.  2b2. System deletes chosen Role.  2c. User clicks Edit button of any Role in the table.  2c1. User edits Role name and Description.  2c2. User clicks confirm edit button.  2c3. System edits account to database. |
| **Exception Flow** | 2a3.1. System adds new Role failed because user leave blanks.  2a3.2. System announces to user the blank(s) field.  Return to step 2a1.  2b1.1. User clicks Cancel delete  Use case end.  2c2.2. User clicks Cancel edit button  Use case end. |

* **Manage faculty**

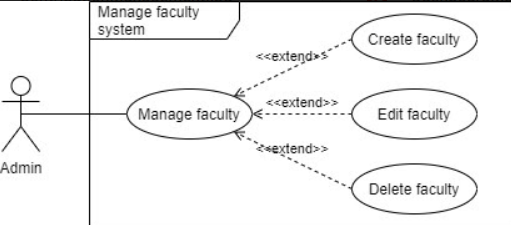


Table 13. Use Case: Manage faculty

|  |  |
| --- | --- |
| **Use Case Name** | Manage faculty |
| **Description** | As a user, I want to manage faculty of the system |
| **Actor(s)** | Admin |
| **Priority** | Medium |
| **Trigger** | User wants to Create, delete or edit faculty |
| **Pre-Condition(s)** | User must login successfully by account assigned with Admin role  User must choose Manage faculty functions. |
| **Post-Condition(s)** | User will be moved page that have faculty management functions. |
| **Basic Flow** | 1. User chooses manage faculty in Admin page. |
| **Alternative Flow** | 2a. User clicks Create faculty button.  2a1. User types faculty name and description.  2a2. User clicks create new faculty button.  2a3. System adds new faculty to database.  2b. User click delete button of any Faculty in the table.  2b1. User clicks confirm delete.  2b2. System deletes chosen Faculty.  2c. User clicks Edit button of any Faculty in the table.  2c1. User edits Role name and Description.  2c2. User clicks confirm edit button.  2c3. System edits account to database. |
| **Exception Flow** | 2a3.1. System adds new Faculty failed because user leave blanks.  2a3.2. System announces to user the blank(s) field.  Return to step 2a1.  2b1.1. User clicks Cancel delete  Use case end.  2c2.2. User clicks Cancel edit button  Use case end. |

* **Manage academic year**

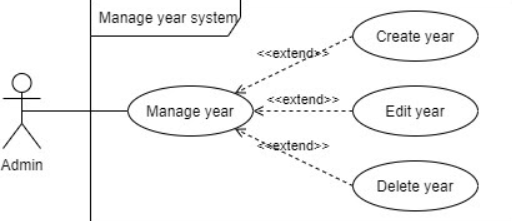


Table 14. Use Case: Manage academic year

|  |  |
| --- | --- |
| **Use Case Name** | Manage academic year |
| **Description** | As a user, I want to manage data of academic year |
| **Actor(s)** | Admin |
| **Priority** | Medium |
| **Trigger** | User wants to Create, delete or edit academic year |
| **Pre-Condition(s)** | User must login successfully by account assigned with Admin role  User must choose Manage academic year functions. |
| **Post-Condition(s)** | User will be moved page that have academic year management functions. |
| **Basic Flow** | 1. User chooses manage academic year in Admin page. |
| **Alternative Flow** | 2a. User clicks Create year button.  2a1. User types faculty Year number.  2a2. User clicks create new Year button.  2a3. System adds new Year to database.  2b. User click delete button of any Year in the table.  2b1. User clicks confirm delete.  2b2. System deletes chosen Year.  2c. User clicks Edit button of any Year in the table.  2c1. User edits Year number.  2c2. User clicks confirm edit button.  2c3. System edits Year to database. |
| **Exception Flow** | 2a3.1. System adds new Year failed because user leave blanks.  2a3.2. System announces to user the blank(s) field.  Return to step 2a1.  2b1.1. User clicks Cancel delete  Use case end.  2c2.2. User clicks Cancel edit button  Use case end. |

* **Manage magazine**

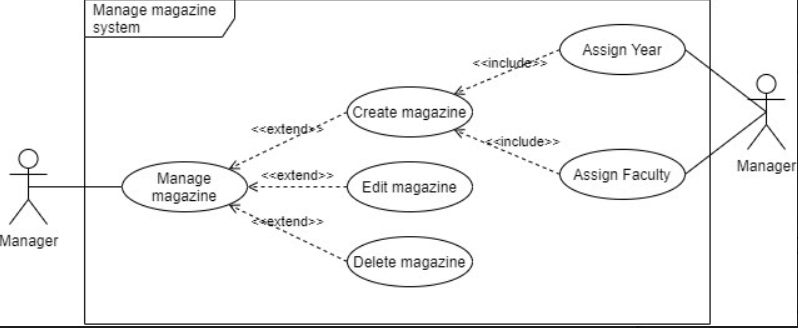


Table 15. Use Case: Manage magazine

|  |  |
| --- | --- |
| **Use Case Name** | Manage magazine |
| **Description** | As a user, I want to manage magazine of the system |
| **Actor(s)** | Manager |
| **Priority** | Medium |
| **Trigger** | User wants to Create, delete or edit magazine |
| **Pre-Condition(s)** | User must login successfully by account assigned with Manager role  User must choose Manage magazine functions. |
| **Post-Condition(s)** | User will be moved page that have account management functions. |
| **Basic Flow** | 1. User chooses manage magazine in Manager page. |
| **Alternative Flow** | 2a. User clicks Create magazine button  2a1. User types magazine name, sets open and close date, and sets Status.  2a2. User assigns Year to Magazine.  2a3. User assigns Faculty to Magazine.  2a4. User clicks create new Magazine button.  2a5. System adds new Magazine to database.  2b. User click delete button of any Magazine in the table.  2b1. User clicks confirm delete.  2b2. System deletes chosen Magazine.  2c. User clicks Edit button of any magazine in the table.  2c1. User edits basic info of the magazine.  2c2. User clicks confirm edit button.  2c3. System edits Magazine to database. |
| **Exception Flow** | 2a5.1. System adds new Magazine failed because user leave blanks.  2a5.2. System announces to user the blank(s) field.  Return to step 2a1.  2b1.1. User clicks Cancel delete  Use case end.  2c2.2 User clicks Cancel edit button  Use case end. |

* **Manage contribution**

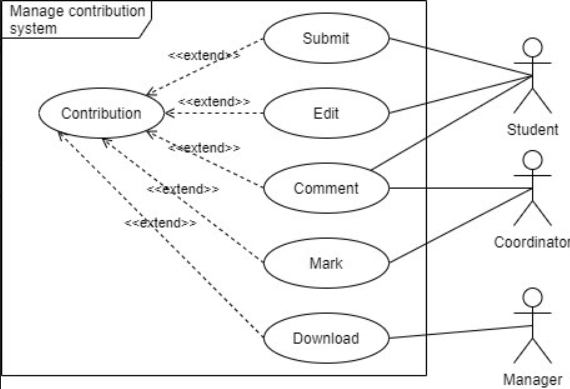


Table 16. Use Case: Manage contribution

|  |  |
| --- | --- |
| **Use Case Name** | Manage contribution |
| **Description** | As a user, I want to manage data of contribution |
| **Actor(s)** | Student, Coordinator, Manager |
| **Priority** | Medium |
| **Trigger** | User wants to Submit their article to the Magazine  User wants to Edit their contribution  User wants to give comment to the contribution  User wants to approve the contribution after deadline  User wants to download the file in contribution |
| **Pre-Condition(s)** | User must login successfully by account assigned with Student, Coordinator or Manager role to use the corresponded function.  User must choose functions corresponded to the user’s role. |
| **Post-Condition(s)** | User will be moved to page that have user’s needed functions. |
| **Basic Flow** | 1. Student chooses Submit contribution.  2. Student upload file Word and images to the contribution.  3. System saves Student’s contribution.  4. Student edits files in contribution before deadline.  5. Student and Coordinator exchange opinion in Comment section  6. Coordinator starts to mark the contribution after ending deadline.  7. Manager downloads the marked contribution. |
| **Alternative Flow** |  |
| **Exception Flow** | 6a1. Contribution is unapproved  6a2. Manager is not able to download the unapproved contribution  End use case |

## **Class Diagram**

## **Activity Diagram**

## **Sequence Diagram**

# **Design**

# **Testing**

## **Test plan**

**TEST PLAN**

**<< CONTRIBUTION COLLECTION SYSTEM >>**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Write by** | **Date** | **Outline** |
| 1.0 | Tan | 26.2.2021 | Create document |
| 2.0 | Tan | 16.3.2021 | Fix scope |
|  |  |  |  |

**1. Intro**

This document details the plan to test the stability of functions in the Contribution Collection System (CCS), to help us determine the effort required to evaluate the quality of the application under test. The document details: ***(i)*** define the scope of the test; ***(ii)*** identify items and features to be testing; ***(iii)*** identify tools and techniques used; ***(iv)*** Determine the risks during the test; ***(v)*** define the product quality assessment criteria, and ***(vi)*** Determine the resources and schedule needed to complete the test.

**2. Scope**

*2.1 In scope*

The functionality in the following table will be tested in the Testing phase

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Module Name** | **Applicable Roles** | **Description** |
| 1 | Login/ logout | Admin  Manager  Student  Coordinator | **Admin:** Admin can login and / log out from the system  **Manager:** AManager can login and / log out from the system  **Student:** A student can login and / log out from the system  **Coordinator:** A coordinator can login and / log out from the system |
| 2 | Create account | Admin | **Admin:** Admin can create an account for all people |
| 3 | Edit account | Admin  Manager  Student  Coordinator | **Admin:** Admin can edit his own account information and admin cannot edit for other accounts in the system  **Manager:** A **m**anager can edit his own account information except *username*  **Student:** A student can edit his own account information except username  **Coordinator:** Acoordinator can edit his own account information except *username* |
| 4 | Delete account | Admin | **Admin:** Admin can delete any account from the system |
| 5 | View detail profile | Admin  Manager  Student  Coordinator | **Admin:** Admin can view their own profiles and some basic information of the accounts in the system (username, name, email)  **Manager:** A manager can view their own profiles and other accounts in the system  **Student:** A coordinator can view their own profiles  **Coordinator:** A coordinator can view their own profiles and other accounts in his or her faculty students |
| 6 | Change password | Admin  Manager  Student  Coordinator | **Admin:** Admin can change the password of his or her account, they cannot change the password of other accounts  **Manager:** A manager can change the password of his or her account  **Student:** A student can change the password of his or her account  **Coordinator:** A coordinator can change the password of his or her account |
| 7 | Reset password | Admin  Manager  Student  Coordinator | **Admin:** Admin can reset the password of his or her account, they can reset the password of other accounts  **Manager:** A manager can reset the password of his or her account  **Student:** A student can reset the password of his or her account  **Coordinator:** A coordinator can reset the password of his or her account |
| 8 | Create  academic year | Admin | **Admin:** Admin can create academic year |
| 9 | Edit  academic year | Admin | **Admin:** Admin can edit academic year |
| 10 | Delete  academic year | Admin | **Admin:** Admin can delete academic year |
| 11 | Create magazine | Manager | **Manager:** A manager can create magazine topic |
| 12 | Edit magazine | Manager | **Manager:** A manager can edit magazine topic |
| 13 | Delete magazine | Manager | **Manager:** A manager can delete magazine topic |
| 14 | Create faculty | Admin | **Admin:** Admin can create faculty |
| 15 | Edit faculty | Admin | **Admin:** Admin can edit faculty |
| 16 | Delete faculty | Admin | **Admin:** Admin can delete faculty |
| 17 | Assign faculties | Manager | **Manager:** A manager can assign faculties that will be in the magazine |
| 18 | Assign account | Manager | **Manager:** A manager can assign accounts that will be in the faculty |
| 19 | Submit file | Student | **Student:** A student can submit file (doc, image) to the system from his or her faculty before the closure date |
| 20 | Edit file | Student | **Student:** A student can edit a file that he or she submitted after the closure date |
| 21 | Delete file | Student | **Student:** A student can delete a file that he or she submitted after the closure date |
| 22 | Comment File | Student  Coordinator | **Student:** A students can comment on files they submit  **Coordinator:** The coordinator can comment on files submitted by his or her faculty students |
| 23 | Mark status  for file | Coordinator | **Coordinator:** A coordinator can mark student's contribution after the closure date |
| 24 | Notify via email | Coordinator | **Coordinator:** A coordinator will receive an email notification when students in their faculty submit file |
| 25 | View file mark student's contribution | Manager  Student  Coordinator  Guest | **Manager:** A manager can view files that students submitted to the system have been approved in all faculty  **Student:** A student can view files that other students submitted to the system have been approved in faculty  **Coordinator:** A coordinator can view files that students submitted to the system have been approved in faculty  **Guest:** A guest can view files that students submitted to the system have been approved in faculty |
| 25 | View dashboard | Manager | **Manager:** A manager can view an overview of the activities of participating facilities contributing the magazine article via the dashboard |
| 26 | Download all file in zip | Manager | **Manager:** The manager can download all the approved files to his computer as a zip after the closure date |

*2.2 out scope*

Functions / non-functions not tested during the Test phase include:

* User interface
* Logical database
* Efficiency of the system
* Performance of the system
* The security of the system
* Compatibility with each hardware/software

**3. Testing Types**

The test phase uses 3 methods:

* Component Testing: A Website is all made up of several components, component testing refers to checking these components to ensure the expected output of the Component. Testing is performed on each individual component separately, without integration with other components.
* Smoke Testing: Smoke Testing is performed to verify if the basic functions are working.  In Smoke Testing, the selected test cases cover the most important function or component of the system. The goal is not to perform comprehensive testing but to verify that critical system functionality is working properly. Therefore, if the system passes Smoke Testing then it is considered a stable system.
* Integration Testing: Integration testing is an integrated testing method in which the components of the system are logically integrated and tested as a group. A system consists of many components code by different developers. The purpose of this level of inspection is to find defects in the interactions between components as they are integrated.

**4. Risk and Issues**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Risk** | **Probability** | **Impact** | **Mitigation Risk** |
| 1 | The test scope is not fully defined | Medium | High | Fully define the scope of the test |
| 2 | There is not enough manpower to do the testing which leads to delay in the testing phase | Medium | High | Ask other teams for help with human resources;  Give accurate estimates of testers' working hours including vacations and holidays and determine whether they will work throughout those vacations. |
| 3 | The product was delivered on schedule to go through the testing period | Low | Medium | Plan early for test cases to quickly get test work done as soon as the product starts the testing phase. |
| 4 | Testing was delayed because the functions had too many errors to resolve, causing a delay | High | High | Have a bug management plan in place, have a discussion with the development team about the error handling plan |
| 5 | The laptop is damaged due to many reasons | High | High | Prepare a backup computer with all the necessary tools pre-installed, ready to use in any case. |
| 6 | Natural disaster | Low | Medium | Members of all teams in different geographical areas are therefore responsible for assisting testers in case of unexpected events in the area. |

**5. Quality evaluation criteria**

*5.1 Quality criteria*

The system passes the test when it meets the following criteria:

- Test 100% of the functions mentioned in section 2.1

- Where some functionality is not being tested, you must give a clear reason

- 80% of tests must meet at least **PASS**

*5.2 Suspension Criteria*

During testing, if more than 40% of **FAIL** test cases occur, testing will be halted until the development team has corrected these **FAIL** cases.

**6. Resources and schedule**

*6.1 Resources*

*6.1.1 System resource*

|  |  |  |
| --- | --- | --- |
| **No.** | **Resource** | **Description** |
| 1 | 1 Laptop | *Laptop ASUS UX410UQ, intel core i5 -7200U, 12GB RAM, windows 10 pro 64bit*. To be fully installed all necessary software to launch the system and testing. |
| 2 | 1 PC | *intel core i7 -8700K, 32GB RAM, windows 10 pro 64bit*. To be fully installed all necessary software to launch the system and testing. This is a PC used when a laptop has a problem |
| 3 | Execl | Excel is used to record test results during Testing |

*6.1.2 Human resource*

|  |  |  |
| --- | --- | --- |
| **No.** | **Name** | **Task** |
| 1 | Ho Ngoc Tan  (Tan) | * Write Test plan for the system * Perform functional test and results * Make sure the testing process meets specific requirements * Write a report about test phase results |

*6.2 Schedule*

|  |  |  |
| --- | --- | --- |
| **Task** | **Member** | **Estimate effort** |
| Create Test case | Tan | 15 man-hour |
| Testing | Tan | 40 man-hour |
| Write Test report | Tan | 5 man-hour |
| Total | | **60 man-hour** |

## **Smoke Testing**

## **Login System**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nhập đúng tên user, sai mật khẩu | admin / 123457 | Your Password or Username is incorrect! | Pass |  |
| Nhập sai username, đúng mật khẩu | Duyvu/ 123456 | Your Password or Username is incorrect! | Pass |  |
| Nhập đúng username, mật khẩu | admin/123456 | Login success | pass |  |

## **Create Magazine**

Role MM

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## **Assign Faculty to Account**

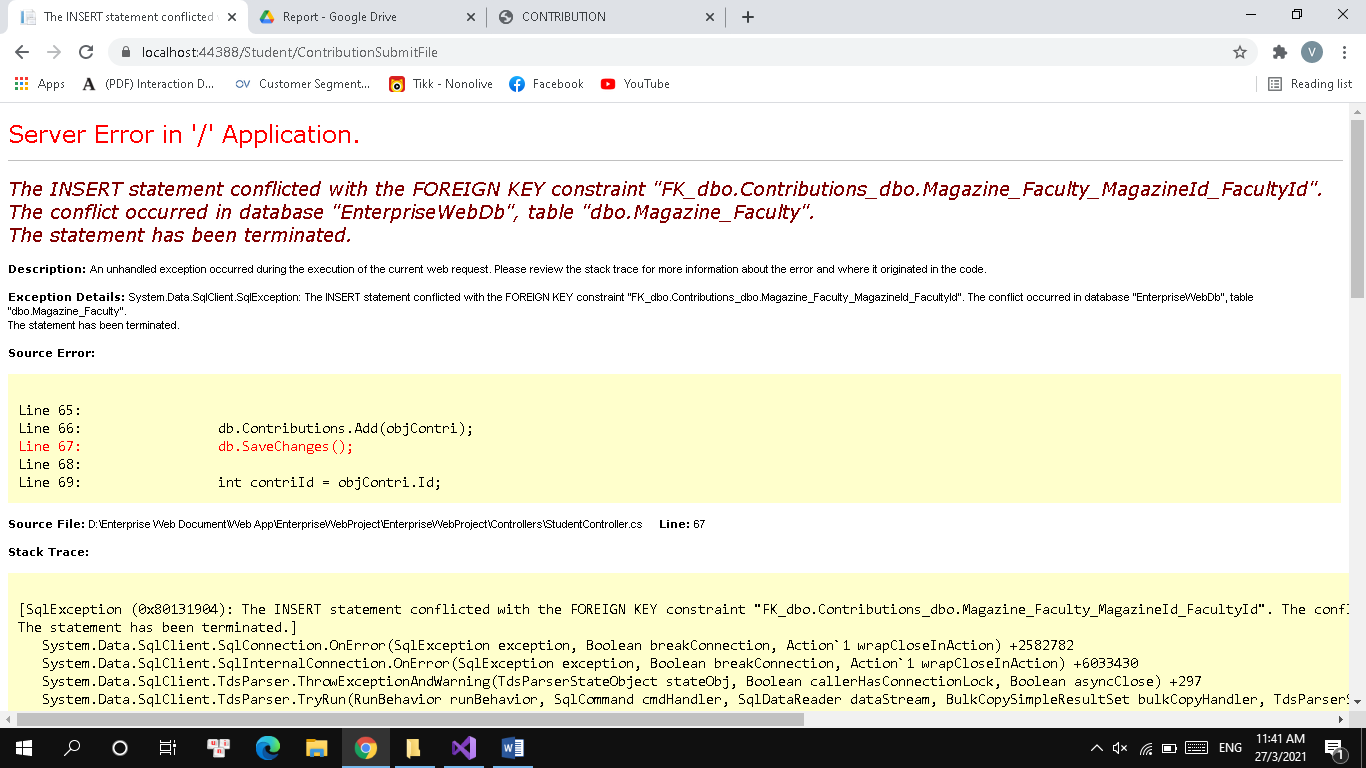
Role Admin

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Click update nhưng không chọn faculty |  | checkbox select is required! | Pass |  |
| Chọn faculty | Checkbox |  | Pass |  |
| Click update |  | Assign success | pass |  |

## **Update file**

Role Student

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Null |  | Thông báo lỗi | pass |  |
| Nhập title, không chọn file |  |  |  |  |
| Chọn file sai định dạng | .pdf  .html | Only doc and docx files are allowed! | pass | .jpg, .jpeg, .png, docx, .doc |
| Chọn file đúng, ko nhập title |  | The Name field is required. | pass |  |
| Nhập title, chọn file đúng định dạng |  | Update success | fail | Figure 1 |



## **Notify via email**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Gửi mail thông báo khi có file update |  | MC nhận được email | Fail | Update file function Fail |

## **View dashboard**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Chọn năm | 2020 | Hiển thị dữ liệu 2020 | Pass |  |
| Tùy chọn các faculty muốn xem | IT, Design, Event | Hiển thị dữ liệu các khoa đã chọn | pass |  |
| Tắt các faculty không muốn xem | Finance, Economic, police, Business | Không hiển thị dữ liệu các khoa đã tắt | Pass |  |

## **Download all file in zip**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Click select all |  | Các file trong 1 trang dc select | pass |  |
| Chọn các file cần tải | Nice Contribution, Super Contribution, Good Contribution | Các file dc chọn |  |  |
| Click Download |  | Download dưới dạng zip | pass |  |

## **Test log**

Table 17. Test log

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Module Name** | **Status (Pass/Fail)** | **Date** | **Note** |
| 1 | Login/ logout | Pass |  |  |
| 2 | Create account |  |  |  |
| 3 | Edit account | pass |  |  |
| 4 | Delete account |  |  |  |
| 5 | View detail profile | pass |  |  |
| 6 | Change password | pass |  |  |
| 7 | Reset password |  |  |  |
| 8 | Create academic year | pass |  |  |
| ~~9~~ | ~~Edit academic year~~ | ~~-~~ | **~~-~~** | **~~-~~** |
| 10 | Delete academic year |  |  |  |
| 11 | Create magazine |  |  |  |
| 12 | Edit magazine |  |  |  |
| 13 | Delete magazine |  |  |  |
| 14 | Create faculty |  |  |  |
| 15 | Edit faculty |  |  |  |
| 16 | Delete faculty |  |  |  |
| 17 | Assign faculties | pass |  |  |
| 18 | Assign account | pass |  |  |
| 19 | Submit file |  |  |  |
| 20 | Edit file |  |  |  |
| 21 | Delete file |  |  |  |
| 22 | Comment File |  |  |  |
| 23 | Mark status for file | pass |  |  |
| 24 | Notify via email | fail |  |  |
| 25 | View file mark student's contribution | pass |  |  |
| 25 | View dashboard | pass |  |  |
| 26 | Download all file in zip | pass |  |  |
| **Total** | |  |  |  |

## **Conclusion**

# **Agile Method**

*Product Backlogs, Sprint Backlogs, Schedule, Progress…(Can use Trello)*

* *Dẫn link đến trello của mình*

## **Daily Meetings**

*Lấy trên Teams về*

## **Sprint Review**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sprint:** 1 - Meeting after Tet holiday  **Date:** 23/2/2021  **Attended:** 6/6 | | | |
|  |  | |
| **Goal of sprint** | | **Sprint review** | |
| * Determine software requirements | | * 5/6 tasks have been completed | |
| **Feedback** | | **What’s next** | |
|  | | * Use-case specification * Draw Activity Diagram * Draw Sequence Diagram * Code, test Database * Draw high-level design * Writing Test plan | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Sprint:** 2  **Date:** 1/3/2021  **Attended:** 4/6 *(absent: Kieu and Kiet)* | | | |
|  |  | |  |
| **Goal of sprint** | | **Sprint review** | |
| * Design UX/UI and System | | * 4/6 task have been completed * The development team continues coding and testing the database | |
| **Feedback** | | **What’s next** | |
| * The front-end team needs to communicate more to get the job done | | * Code front – end * Code function   + Authentication   + Admin function   + Manager function   + Coordinator function   + Student function * Writing reports:   + 1. Intro   + 2. Database   + 3.3 User-case   + 5.1 Test plan | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Sprint:** 3  **Date:** 9/3/2021  **Attended:** 6/6 | | | |
|  |  | |  |
| **Goal of sprint** | | **Sprint review** | |
| * Code functions and frontend | | * Frontend completed the dashboard and Homepage * Backend completes the scheduled task | |
| **Feedback** | | **What’s next** | |
|  | | * Backend continues to code the validation functions * Frontend tries to complete all the remaining quests | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Sprint:** 4  **Date:** 15/3/2021  **Attended:** 6/6 | | | |
|  |  | |  |
| **Goal of sprint** | | **Sprint review** | |
| * Code functions and frontend | | * Frontend: 4/5 tasks have been completed * Backend completes check validation | |
| **Feedback** | | **What’s next** | |
| * The front end needs improvement | | * Frontend tries to complete all the remaining work, fixes the interface that the group commented on at the meeting on March 13, 2021. * Compiler front-backend * Writing report: Design | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Sprint:** 5  **Date:** 22/3/2021  **Attended:** 6/6 | | | |
|  |  | |  |
| **Goal of sprint** | | **Sprint review** | |
| * Code functions and frontend | | * Frontend completed * Compiler front-backend completed 50% of the task | |
| **Feedback** | | **What’s next** | |
|  | | * Backend continues to compiler * Testing | |

## **Burndown Charts**

# **Some image of product**

# **Appendix**

Table 18. Links

|  |  |  |
| --- | --- | --- |
| **No.** | **Name** | **Link** |
| 1 | Trello |  |
| 2 | Github |  |
| 3 | Google Drive |  |

# **Reference**

|  |  |  |
| --- | --- | --- |
|  | Group Report Outline: |  |
| [V] | 1. Cover Page; Table of Contents; etc. |  |
| [V] | 2. Project Introduction. | Tân |
| [V] | 3. Database: ERD, Relational Schema, etc. | Phát  Tân Duy |
|  | 4. Functionality:  [V]Use Case Diagram  [V]Use Case Specification  Class Diagram  Activity Diagram  Sequence Diagram | Kiều Kiệt |
|  | 5. Design | Kiều Kiệt |
|  | 6. Testing:  [V]Test Plan;  [V] Test Log;  [V]Sufficient Test Data to fully test;  Evidence of testing finding errors;  Test Items linked to user stories in the product backlog (xem case với backlog có phù hợp với nhau chưa) | Tân |
|  | 7.1 Agile Method: Product Backlogs, Sprint Backlogs, Schedule, Progress…(Can use Trello) | Phát |
|  | 7.2 Daily Meetings, Sprint Review & Retrospective Meetings, Burndown Charts. | Tân |
|  | 8. Product: Screenshots of your real application and its functions. | Tân |
|  | 9. Links: Trello, Github, Google Drive, etc. |  |
|  | 10. References: (if possible). |  |