



**Enterprise Web Software Development**

Group 5: Trần Ngọc Nhất Linh (GCS18518), Ngô Triều Hảo, Trần Trung Tính, Mai Xuân Duy, Lê Thanh Đức (GCS18688), Phạm Mạnh Hải (GCS18586)

Class: GCS0704A

Subject code: 1640

Assessor name: **HO NGUYEN PHU BAO**

Assignment due: Assignment submitted:

|  |  |  |
| --- | --- | --- |
| COMP1640 (2020/21) | **Enterprise Web Software Development** | **Contribution: 100% of course** |
| **Course Leader: Matt Prichard** | **Coursework** | **Deadline Date: 5th April 2021** |

|  |
| --- |
| **Plagiarism is presenting somebody else's work as your own. It includes: copying information directly from the Web or books without referencing the material; submitting joint coursework as an individual effort; copying another student's coursework; stealing coursework from another student and submitting it as your own work.  Suspected plagiarism will be investigated and if found to have occurred will be dealt with according to the procedures set down by the University. Please see your student handbook for further details of what is / isn't plagiarism.** All material copied or amended from any source (e.g. internet, books) must be referenced correctly according to the reference style you are using.   Your work will be submitted for plagiarism checking.  Any attempt to bypass our plagiarism detection systems will be treated as a severe Assessment Offence. |

#### **Coursework Submission Requirements**

* An electronic copy of your work for this coursework should be fully uploaded by midnight (local time) on the Deadline Date.
* The last version you upload will be the one that is marked.
* For this coursework you must submit a single Acrobat PDF document. In general, any text in the document must not be an image (i.e. must not be scanned) and would normally be generated from other documents (e.g. MS Office using "Save As .. PDF").
* **For this coursework you must also upload a single ZIP file containing supporting evidence.**
* There are limits on the file size. The current limits are displayed on the coursework submission page on the Intranet
* Make sure that any files you upload are virus-free and not protected by a password or corrupted otherwise they will be treated as null submissions.
* Comments on your work will be available from the Coursework page on the Intranet. The grade will be made available in the portal.
* You must NOT submit a paper copy of this coursework.
* All coursework must be submitted as above

The University website has details of the current Coursework Regulations, including details of penalties for late submission, procedures for Extenuating Circumstances, and penalties for Assessment Offences.  See <http://www2.gre.ac.uk/current-students/regs>for details.

#### **Scenario**

This is a group coursework. You will be assigned to a group.

You need to adopt agile scrum working practices and document your meetings appropriately. Ideally you need a database designer, an information architect, a programmer, a web designer and a tester, as well as a scrum master and product owner, but more than one person can be in any technical role. No one is to take the role of project manager, but there could be a technical team leader.

You will get an individual grade (40%) based on your report. There is also a group grade (60%), weighted by your contribution to the team effort (from 0% to 100%).

**Specification**

You are required to build a web-based secure role-based system for collecting student contributions for an annual university magazine in a large university.

The system must meet the following criteria:

* The University has a Marketing Manager to oversee the process.
* All Faculties have a Marketing Coordinator who is responsible for managing the process for their Faculty.
* All students have the opportunity to submit one or more articles as Word documents to the magazine.
* All students can also upload high quality images, e.g. photographs.
* All new contributions are disabled after a closure date for new entries, but updates can continue to be done until a final closure date.
* All students must agree to Terms and Conditions before they can submit.
* Once a contribution is submitted the system emails a notification to the Faculty’s Marketing Coordinator, who must make a comment within 14 days.
* A Marketing Coordinator can only access contributions by students in their Faculty.
* Each Marketing Coordinator needs to be able to interact with the students in their Faculty in order to edit the contributions and to select those for publication.
* The University Marketing Manager can view all the selected contributions but cannot edit any. They need to be able to download all the selected contributions after the final closure date in a ZIP file for transfer out of the system.
* An administrator maintains any system data, e.g. closure dates for each academic year.
* A guest account for each Faculty can be used to view the selected reports.
* Statistical analysis (e.g. number of contributions per Faculty) needs to be available.
* The interface must be suitable for all devices (eg mobile phones, tablets, desktops).

**Assumptions**

You must clearly state any assumptions you make.

**Reports**

A number of reports need to be made available. For example

* Statistics
  + Number of contributions within each Faculty for each academic year.
  + Percentage of contributions by each Faculty for any academic year.
  + Number of contributors within each Faculty for each academic year.
* Exception reports
  + Contributions without a comment.
  + Contributions without a comment after 14 days.

**Tasks**

1. Work as a team using agile scrum methods to develop and test a secure web-based system to meet the above specification.
2. Create a Panopto screencast recording (including screen and sound) demonstrating the key functionalities of the system.
3. Present the finished product to a non-technical audience to try to persuade them to purchase your system.
4. Document the system to an appropriate standard, including an evaluation of the design process you followed and your reflection on the finished product, and on the contributions of your team members.

**Deliverables**

1. A **Group Report based on a Group Repository** containing all the artefacts produced by the team (eg ERD, minutes, test log, product backlog) with a menu allowing easy access to its content. The repository must be secure, but accessible by your tutors. The Scrum Master is responsible to ensure this gets uploaded by the due date. It is not essential that all members upload a copy of the group report but it must be clear which students are in which group.
2. An **Individual** **PDF Report**  
   The report must give the **URL** of the Group Repository, the Screencast and the website and any **usernames or passwords** needed to access it. The **individual component of the marking will be based on your report,** so ensure this has evidence that your system meets the specified requirements. **The text in your individual report must be entirely your own words.**
3. A **Presentation** and **Screencast**  
   You must be present aspart of the team that presents the finished product to your tutor, and should contribute to the screencast. The **presentation** should be pitched at a non-technical audience to try to persuade them to purchase the product; the **screencast** should demonstrate the functionality of the system

**Assessment Breakdown**

**Group Component (60%)**This will be assessed based on a group report and a group repository created by the group on a secure shared area accessible to the Greenwich moderator. Password and URL must be provided in individual reports. Must be suitably structured with a menu. Suggested location: GitHUB, SharePoint 365, own website, DropBox or other repository.

Database 10%  
Expect: Security, appropriate data types and validation, clear ERD, referential integrity implemented, enables roles to be implemented

Site design 10%  
Expect: Responsive design, clear information architecture for both mobile and desktop, aesthetically pleasing, good usability, meets accessibility criteria

Functionality 10%  
Expect: Role based security, upload of documents and images, email notification, summary and exception reports, UML diagrams, code snippets, analytics of use  
  
Testing 10%  
Expect: Test plan, test log, sufficient data to fully test, evidence of testing finding errors, test items linked to user stories in the product backlog

Agile methods followed 10%  
Expect: Burn down chart, minutes of meetings, user stories, sprints, product backlogs

Screencast and Presentation 10%  
Expect: Professional standard of presentation promoting the product, with contributions by all the team members, Screencast demonstrating all the main features of the product. Screencast can be done by one person.

**Weighting factor for each student (scale 0 to 10) to be applied by the tutor to determine the group grade awarded to each student.**

|  |  |
| --- | --- |
| **Commitment** | **Weight** |
| Fully committed | 10 |
| Committed | 8 |
| Contributed substantially | 6 |
| Contributed partially | 4 |
| Minimal contribution | 2 |
| No contribution | 0 |

**Individual Component (40%)**N.B.: No shared content in the report, i.e. must be entirely in your own words, Must include title page with a list of team members and roles, URL and password of group repository, site and screencast.

Evaluation of product and process 10%  
Expect: Appropriate screen shots and commentary, with cross references to group documents, evaluative comments on the product and on the agile process used to build it

Evaluation of team 10%  
Expect: A weighted scoring model of the entire team (including yourself) with own choice of criteria and weighting, supported by commentary on each individual member. Model is expected to produce a range of scores for the individual members.

Self-evaluation 10%  
Expect: Honest description of own contribution, and reflection on own performance and any lessons learnt  
  
Quality of documentation 10%  
Expect: NO SHARED CONTENT, professional standard, header page, page numbers, table of contents, headings, cropped images, figure captions, no spelling or grammatical errors.

**Indicative Grading Criteria**

>=70%   
Well designed system to fully meet the requirements  
Professional standard of report, with appropriate documentation  
High level of individual commitment  
High level of evaluative commentary  
  
60-69%  
Well designed system to meet most of the requirements  
Professional standard of report  
High level of individual commitment  
Limited evaluative commentary

50-59%  
Well designed system to meet most of the requirements  
Acceptable standard of report  
Good level of individual commitment  
Limited evaluative commentary

40-49%  
Acceptable system to meet most of the requirements  
Acceptable standard of report  
Acceptable level of individual commitment  
Limited evaluative commentary

<40%  
Poorly designed system   
Few requirements met  
Poor standard of report  
Limited individual commitment  
No evaluative commentary

Contents

[1. **Introduction** 3](#_Toc65699320)

[2. **Database** 3](#_Toc65699321)

[2.1. Entity relationship diagrams 3](#_Toc65699324)

[2.2. Relational schema 3](#_Toc65699325)

[3. **Functionality** 3](#_Toc65699326)

[3.1. Product backlogs 3](#_Toc65699330)

[3.2. Use case diagrams 3](#_Toc65699331)

[3.2.1. Admin 3](#_Toc65699332)

[3.2.2. Manager 4](#_Toc65699333)

[3.2.3. Coordinator 4](#_Toc65699334)

[3.2.4. Student 4](#_Toc65699335)

[4. **Design** 4](#_Toc65699336)

[4.1. Website wireframes 4](#_Toc65699341)

[5. **Testing** 4](#_Toc65699342)

[5.1. Test plan 4](#_Toc65699344)

[5.2. Test log 4](#_Toc65699345)

[6. **Agile method** 4](#_Toc65699346)

[7. **Product** 5](#_Toc65699347)

[8. **Links** 5](#_Toc65699348)

[9. **References** 5](#_Toc65699349)

# **Introduction**

This project is a group coursework. The project is a web-based secure role-based system for collecting student contributions for an annual university magazine in a large university. This project implements the processes according to criteria using scrum agile work.

The project is built with ASP.NET Core which allow us retain flexibility while constructing solutions.

In this project, group 5 will create a website where students in the school can talk to each other called… ... There will be 3 main users of this website: Admin, Staff, and Student. With this website, all students in the school can post and comment on their opinions below that post. Staff will manage the posts posted to the website. Admin will decentralize for each user object. Not only that, users who are not under the management of the school can also visit

Below is the requirement specification as obtained. A system that accompishes the following:

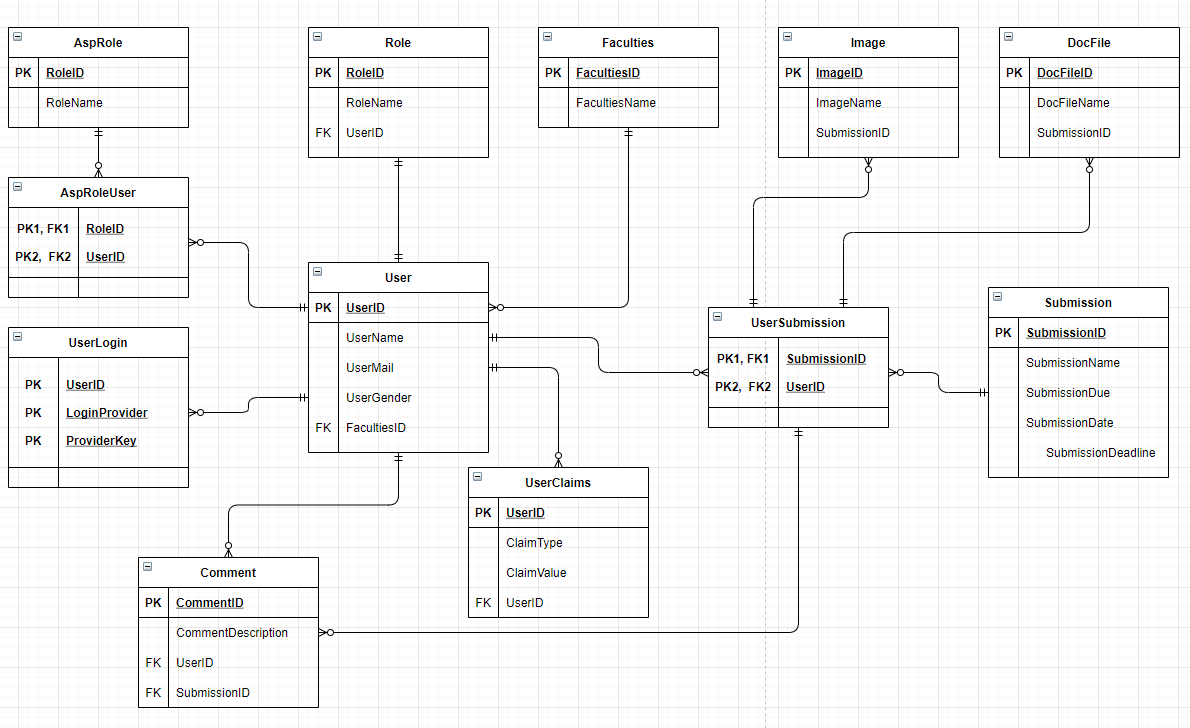
* Student can post on the system
* The coordinator can moderate the student's post before deciding whether to post it on the system or not
* Student and coordinator can comment in post.
* Admin can edit all data on the system as well as decentralize use for each object
* At every step an email notification must be sent to the relevant so action can be taken where required

# **Database**

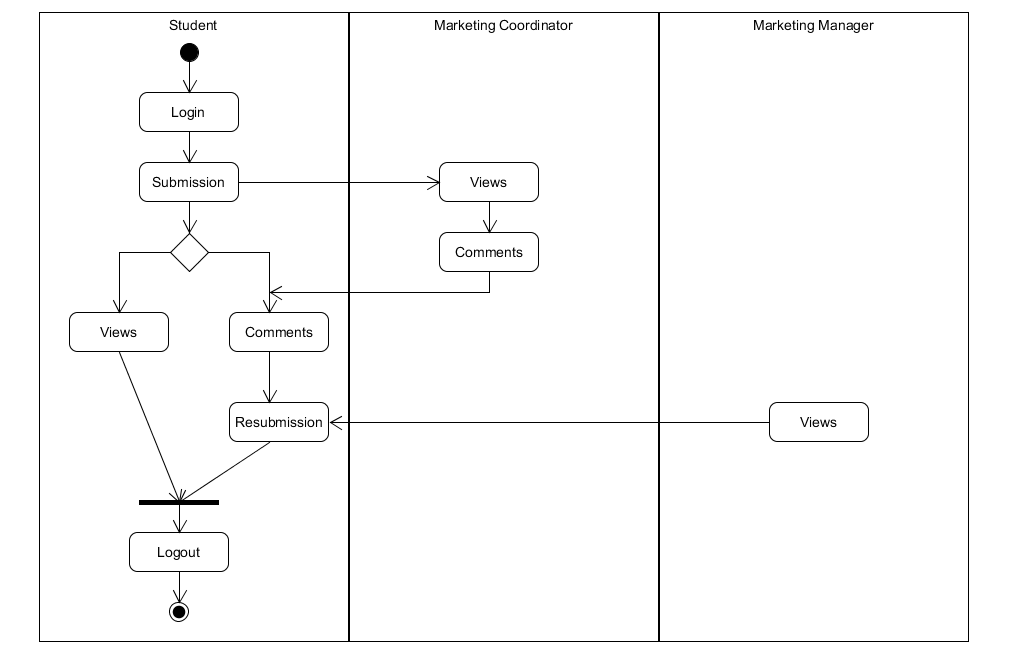


## Entity relationship diagrams

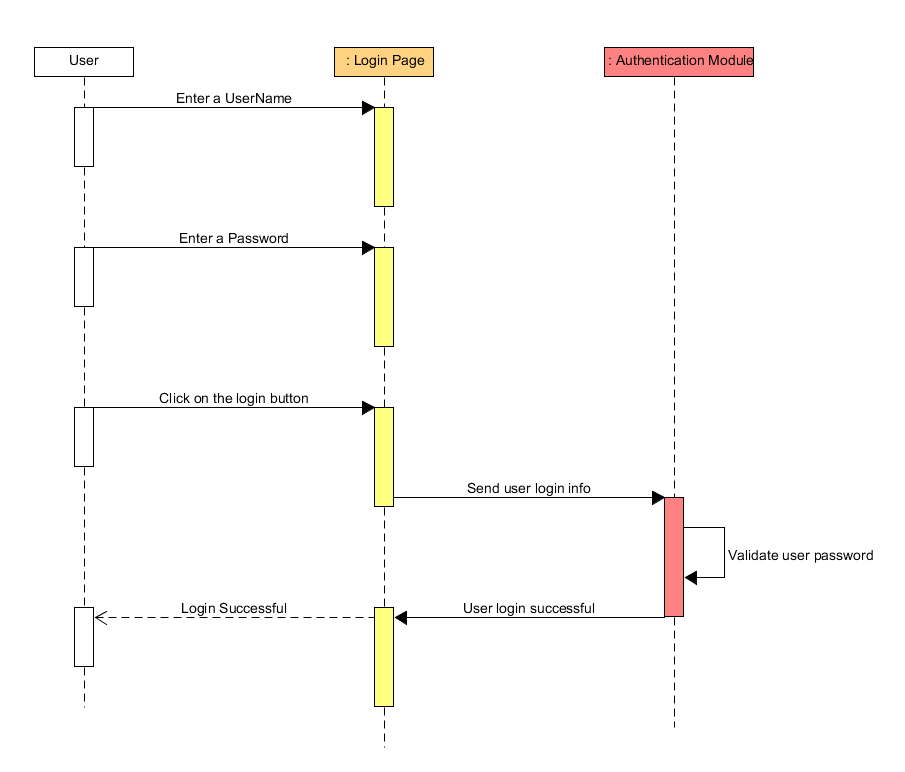
* ERD Diagram:



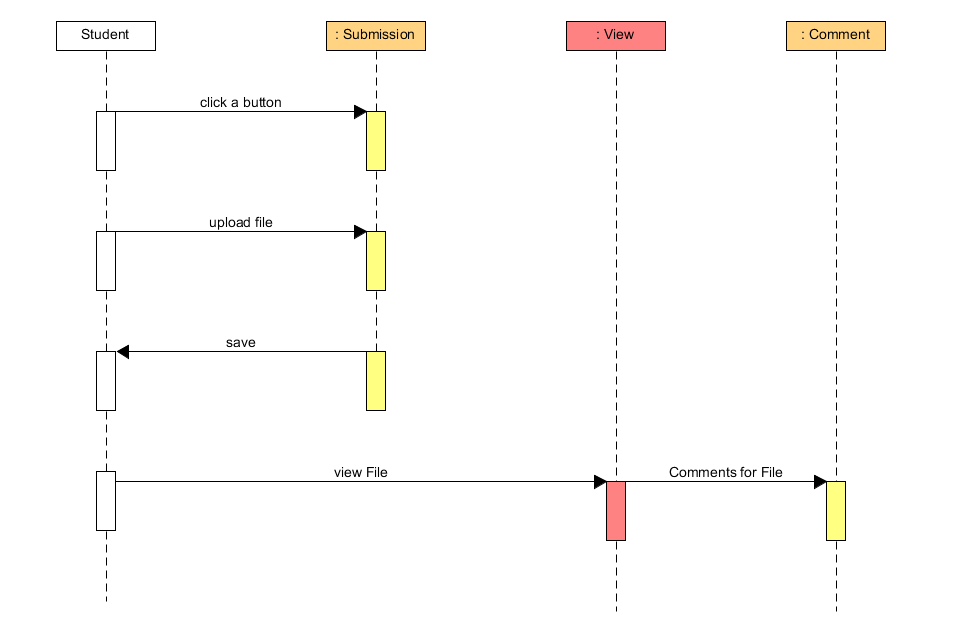
* Activity Diagram:

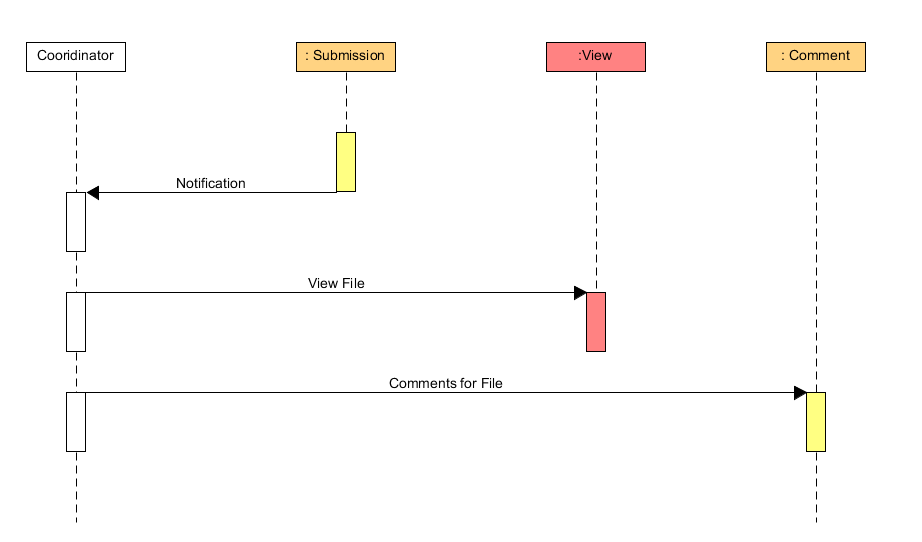


In the Activity Diagram, students will first log into the system with the account provided by the Admin. After successfully logging in, students can post to the system. From there, the Coordinator will view that post and then send the comment back to the students. Students will have 2 options: view the post then exit or view comments, then edit the post and then submit it again. Now the Manager will see and then decide whether that post is posted on the forum or not.



The above diagram shows in turn the steps that the user needs to do when starting to log in to the system such as: Enter User name, Enter Password and the system will give the results.





## Relational schema

* User( UserId, Firstname, Lastname)
* Faculty( FacultyId, FacultyName, Year)
* Submission
* Comment
* File
* Picture
* FacultiesUser( FacultyId, UserId)
* SubmissionUser

# **Functionality**

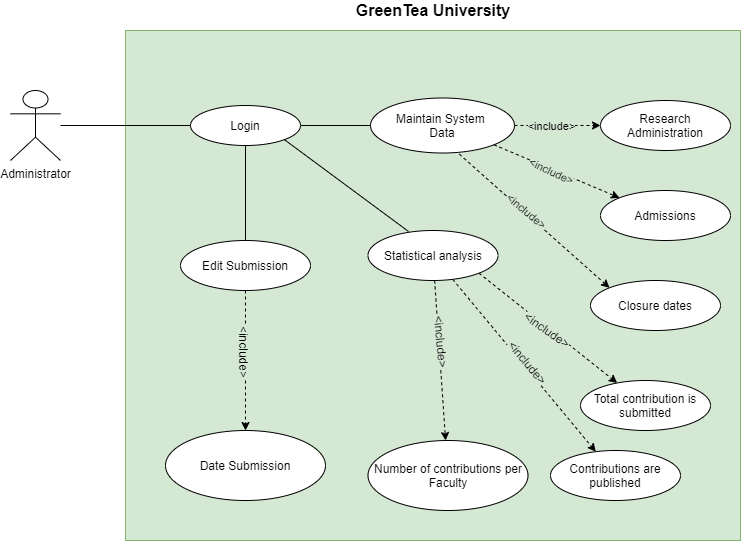


## Product backlogs

## Use case diagrams

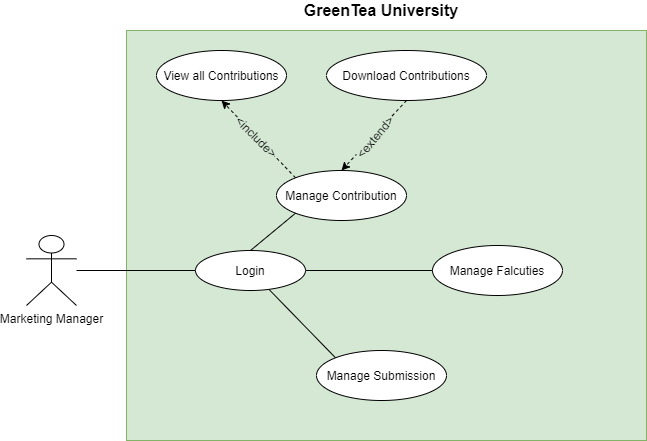
We choose use case diagrams for our design.The diagrams will describes the design of our project. Use case diagrams are made up of users, use cases, and their relationships.

### Admin



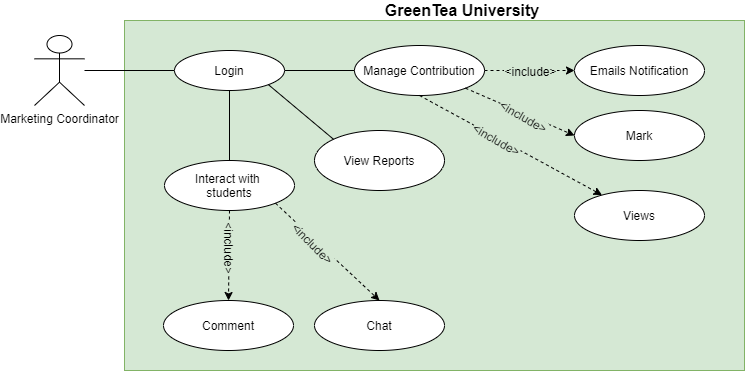
Admin has full rights to use the system such as logging into the system, correcting information on the system, and delegating authority to each type of system user

### Manager



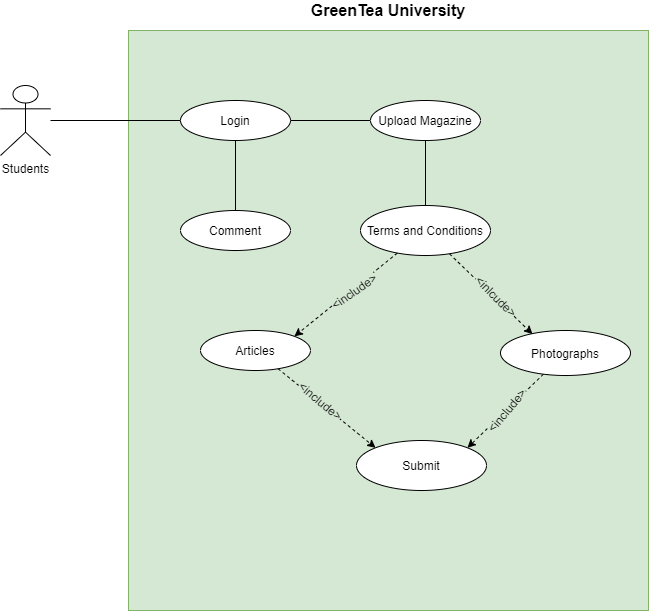
Manager can login to system and Manage Falcuties, Manage Submission, Manage Contribution

### Coordinator



Coordinator will log into the system from which to manage Contribution including Emails Notification, Mark, Views. Coordinator can view reports, Interact with students. In Interact with student coordinator can comment and chat with student

### Student



Student can login to system and upload Magazine have a articles and photographs or comment to post.

|  |  |
| --- | --- |
| **Use case ID:** | LG-1.1 |
| **Use case name:** | Login |
| **Brief description:** | As a user, administrator, marketing coordinator, marketing manager, I want to log into the application to use the service from the application. |
| **Actor:** | Marketing coordinator, marketing manager, students, administrator |
| **Priority:** | High |
| **Trigger:** | User wants to log into the Green Tea University application |
| **Pre-condition(s):** | * User account has been created * The user account has been authorized * The user's device was connected to the internet when logging in |
| **Post-condition(s):** | * The user has successfully logged in to the application * The system records successful login activity |
| **Basic Flow:** | 1. Users access the Green Tea University application 2. The user chooses the method to log in with the account 3. The user enters the account and selects the login command 4. The system verifies login information successfully and allows users to access the application 5. The system records successful login activity |

**Use case Specification**

**1.1 Login**

**1.2 Create Account**

|  |  |
| --- | --- |
| **Use case ID:** | CA-1.1 |
| **Use case name:** | Create Account |
| **Brief description:** | This function is used to create an account. For example, the admin will create an account for the students and marketing manager. |
| **Actor:** | Admin |
| **Priority:** | High |
| **Trigger:** | The required account is authorized to be an admin. |
| **Pre-condition(s):** | * Only Admin can use this function. * Admin can create account for Students, Marketing Manager and Marketing Coordinator. |
| **Post-condition(s):** | * The system will save the account data that Admin has created. |
| **Basic Flow:** | 1. The use case starts when the Admin chooses to create an account on the main System menu. 2. The system provides the Admin with a form that includes all the information to fill out. 3. If entered, the required fields are not valid    * The system requires the user to re-enter the incorrectly entered fields.    * The system validates the fields after re-entering. 4. If the information entered is valid, the account will be created successfully. 5. The system adds new accounts |

**1.3 Create Date Submission**

|  |  |
| --- | --- |
| **Use case ID:** | CDS-1.1 |
| **Use case name:** | Create Date Submission |
| **Brief description:** | Coordinator created that date and student based on that submission. |
| **Actor:** | Marketing Coordinator |
| **Priority:** | High |
| **Trigger:** | Coordinator wants to create a place for students to submit |
| **Pre-condition(s):** | * The Coordinator account has been successfully logged into the system |
| **Post-condition(s):** | * Create Date Submission was successfully * The system records the successful creation of the date submission to the Activity Log. |
| **Basic Flow:** | 1. Coordinator access the Green Tea University website  2. Coordinator chooses the Create Date Submission method  3. Coordinator will fill in information such as:  - Name of class, Course, ID Course number  - Select a date for the student to submit and a date to finish submitting  4. Save and Create Date Submission was successfully |

**1.4 Assign role for user**

|  |  |
| --- | --- |
| **Use case ID:** | AS-1.1 |
| **Use case name:** | Assign |
| **Brief description:** | This use case describes the function assign each account in the system. |
| **Actor:** | Admin |
| **Priority:** | Medium |
| **Trigger:** | Admin wants to authorize an account such as: This account will be Marketing Coordinator, Students, … |
| **Pre-condition(s):** | * The admin account has been successfully logged into the system |
| **Post-condition(s):** | * Authorization for each account is successful * The system records the successful creation of the date submission to the Activity Log. |
| **Basic Flow:** | 1. Admin access the website. 2. The administrator chooses the Assign method 3. Admin will assign authority to each account (One account will take over certain rights)    * Marketing Manager    * Marketing Coordinator    * Student 4. Assign and save was successful |

**1.5 Comments**

|  |  |
| --- | --- |
| **Use case ID:** | CM-1.1 |
| **Use case name:** | Comments |
| **Brief description:** | This function will allow Marketing Coordinator to want to comment on submissions from students, so that students can comment from the Marketing Coordinator and vice versa |
| **Actor:** | Marketing Coordinator, Students |
| **Priority:** | Medium |
| **Trigger:** | Marketing Coordinator wants to give feedback to students in order to create interaction between coordinator and students  Students can respond to comments from Coordinator |
| **Pre-condition(s):** | * The Student and Marketing Coordinator account has been successfully logged into the system * The student must submit the previous articles |
| **Post-condition(s):** | * Marketing Coordinator successfully commented * Students have successfully commented * After a successful comment, the system will notify the parties to know who commented to whom. * The system will save to Activity Log |
| **Basic Flow:** | 1. The Marketing Coordinator (or Students) accesses the Green Tea University website  2. Select method:  2.1 Marketing Coordinator will choose View Submission  2.2 Students will select My Course and click on the Course they submitted (or Students can click the notification from Coordinator comment on that course).  3. After you have finished commenting  3.1 Marketing Coordinator will send comments to students on that post. The system will save that activity and will send an email notification to that student  3.2 Student will submit again and this comment will be reviewed by Coordinator again  4. The data will be saved automatically and sent to the database |

**1.6 Views**

|  |  |
| --- | --- |
| **Use case ID:** | VW-1.1 |
| **Use case name:** | Views |
| **Brief description:** | When Students and Guests want to see the report and articles, this Views function will help them see all the articles and reports |
| **Actor:** | Students, Guests |
| **Priority:** | Medium |
| **Trigger:** | Students (or Guests) want to Views their articles or reports |
| **Pre-condition(s):** | * The students (or guests) account has been successfully logged into the system |
| **Post-condition(s):** | * The students view the articles (guest view reports) successfully |
| **Basic Flow:** | 1. Students and Guests access the Green Tea University website  2. Select the methods:  2.1 Students choose My Course and just click on the articles they want to see  2.2 Guests choose Views and just click on the reports they want to see  3. The system records student view the articles (guests view the reports) sending in the Activity Log. |

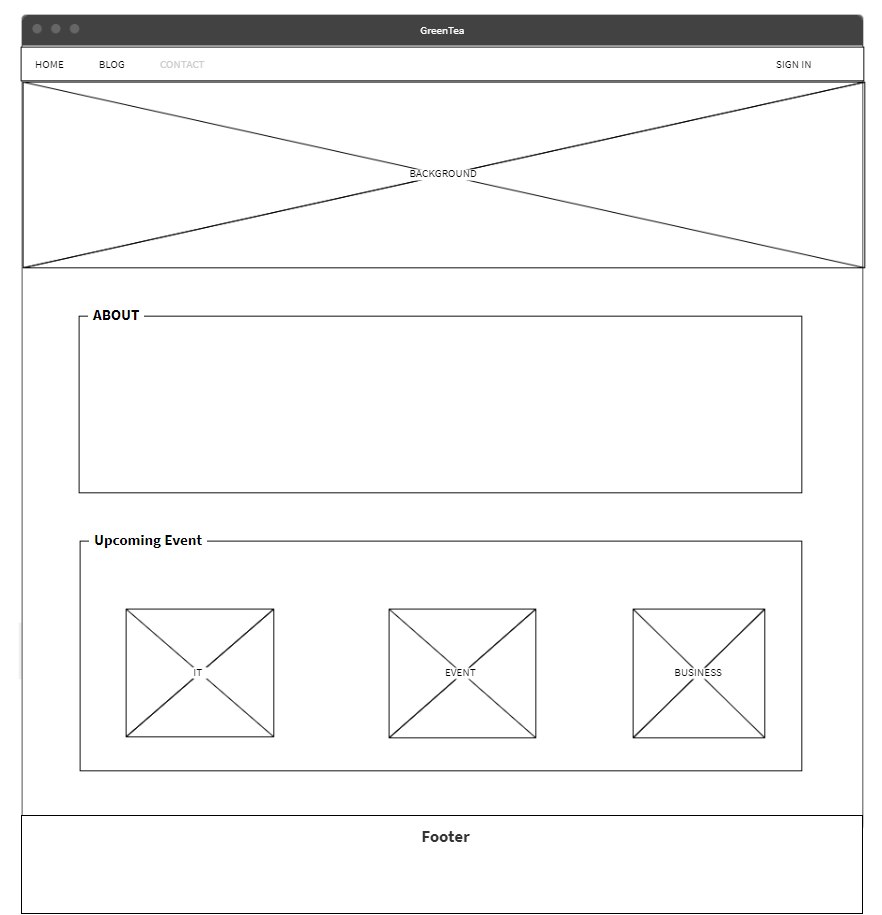
# **Design**



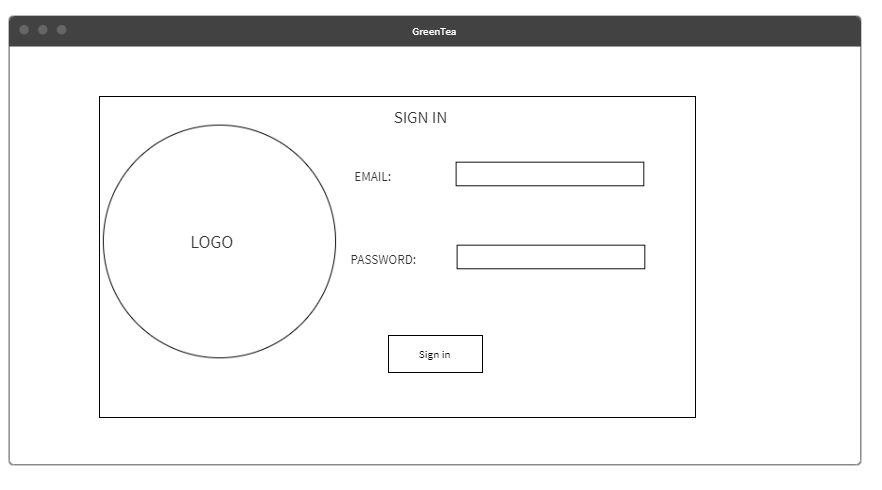
## Website wireframes

In this session, we will show a design of the homepage, login pages and other pages

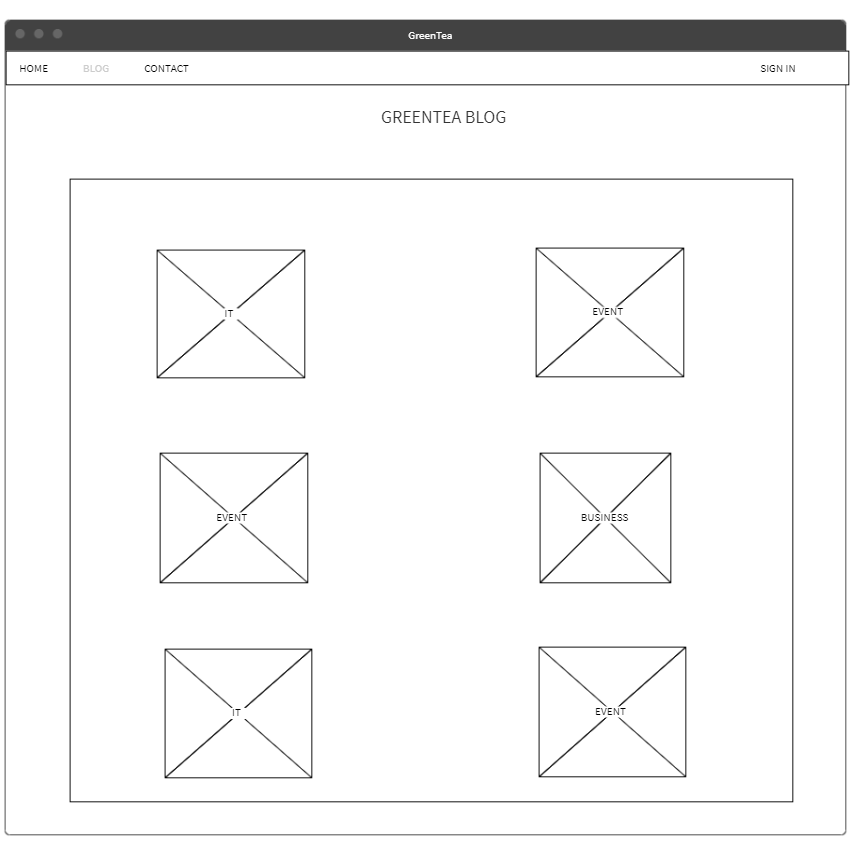
* Home Page:



* Login:



* Blog:



# **Testing**



## Test plan

This session will provides the test methods have been implemented

## Test log

# **Agile method**

Product Backlogs

Sprint Backlogs

Schedule

Progress

Daily Meetings

Sprint Review & Retrospective Meetings

Burndown Charts

# **Product**

Screenshots of the website

# **Links**

# **References**