Subject Name: Enterprise Web Software Development

Subject Code: COMP1640

Tutor: Nguyen Tran Dinh Long

Team member (Group 4)	
Student Name	**Student ID**
Nguyen Minh Son	**001181266**
Nguyen Van Tuan	**001182276**
Pham Quang Minh	**001181234**
Bui Chi Huong	**001181211**
Nguyen Trung Doan Khang	**001181006**

Table of Contents

Table of Figure

I. Introduction

Group 4		
Role	**Task**	**Member**
Product owner	Evaluate requirement, rate requirement	Pham Quang Minh
Database designer	Design database	Nguyen Minh Son Pham Quang Minh
Interface designer	Design Interface	Nguyen Van Tuan
Tester	Test	Bui Chi Huong Nguyen Trung Doan Khang
Scrum master	Make a plan, Track project progress, Division of work.	Nguyen Minh Son
Developer	Front-end	Nguyen Van Tuan
	Back-end	Nguyen Minh Son Pham Quang Minh

- URL Functional demo video: https://drive.google.com/file/d/1rKz8q-A3tbGWUBSLmm-2AFnk3hs1g_KD/view?usp=sharing
- URL Group repository: https://github.com/minhpham8502/Final?fbclid=IwAR3KCGLlbP_lpc-DKd5ebWbB_rAiCgnabMfzvmoUBAEHGQS_9DkQR7k7-vQ
- URL Documentation: https://drive.google.com/drive/folders/1xj9ncGe1kdkZSGyow0j8Uh0CM4IpwPsW?usp=sharing
- URL Site: https://appcuatoi123.herokuapp.com/

List account				
Role	Email	Password	Faculty	FacultyID
Guest	guest@gmail.com	12345678	graphic design	GCHGD
Student	nguyenminhson09112000@gmail.com	12345678	graphic design	GCHGD
Marketing Coordinator	minhpqgch18572@fpt.edu.vn	12345678	graphic design	GCHGD
Marketing Manager	manager@gmail.com	12345678		
Admin	admin@gmail.com	12345678		

1. Technology:

Reasons choose NodeJs:

Node.js is a very strong foundation, and our project is to build a student management website, so it's very important because we are trying to make a big product, and we want to make sure it can. It expands rapidly and fully responds to a large number of users as the website grows.

Node.js is capable of handling multiple connections at once, while PHP does not. In addition to the performance and scalability benefits, we can also learn a little more about JavaScript, so it's easy to learn and learn a whole new language.

Reasons choose Mongo:

MongoDB uses to store data in the form of JSON Document. Each collection will have flexible sizes and documents; this flexibility is very good in storing data, so whatever we want, please insert it freely.

Data in MongoDB has no mutual binding, no join as in RDBMS, so when inserting, deleting, or updating, it does not need to take time to check whether data constraints like in RDBMS are satisfied.

MongoDB is very extensible.

High performance: MongoDB's query speed (find, update, insert, delete) is much faster than other relational database management systems. With large enough data, the test shows that MongoDB's insert speed can be faster than other candidates.

Reasons choose Heroku:

- Provide the best user experience.
- A service ecosystem.
- Support to connect to salesforce.

II. Requirement:

1. Functional

Id	Criteria	Type
1	The University has a Marketing Manager to oversee the process.	Function
2	All Faculties have a Marketing Coordinator who is responsible for managing the process for their Faculty.	Function
3	All students have the opportunity to submit one or more articles as Word documents To the magazine.	Function
4	All students can also upload high quality images, e.g. Photographs.	Function
5	All students must agree to Terms and Conditions before they can submit.	Function
6	A Marketing Coordinator can only access contributions by students in their Faculty.	Function
7	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.	Function
8	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.	Function
9	An administrator maintains any system data, e.g. Closure dates for each academic Year.	Function
10	A guest account for each Faculty can be used to view the selected reports.	Function
11	Statistical analysis (e.g. Number of contributions per Faculty) needs to be available.	Function
12	All new contributions are disabled after a closure date for new entries, but updates Can continue to be done until a final closure date.	Function
13	Once a contribution is submitted the system emails a notification to the Faculty's Marketing Coordinator, who must make a comment within 14 days.	Function

Figure 1: Functional

2. Non-functional

Id	Criteria	Type
1	The interface must be suitable for all devices (eg mobile phones, tablets, desktops).	Non-function

Figure 2: Non-functional

III. Database:

1. ERD

We use MongoDB to store one-to-many data and models; the data from the collection can be compared to the data in another collection.

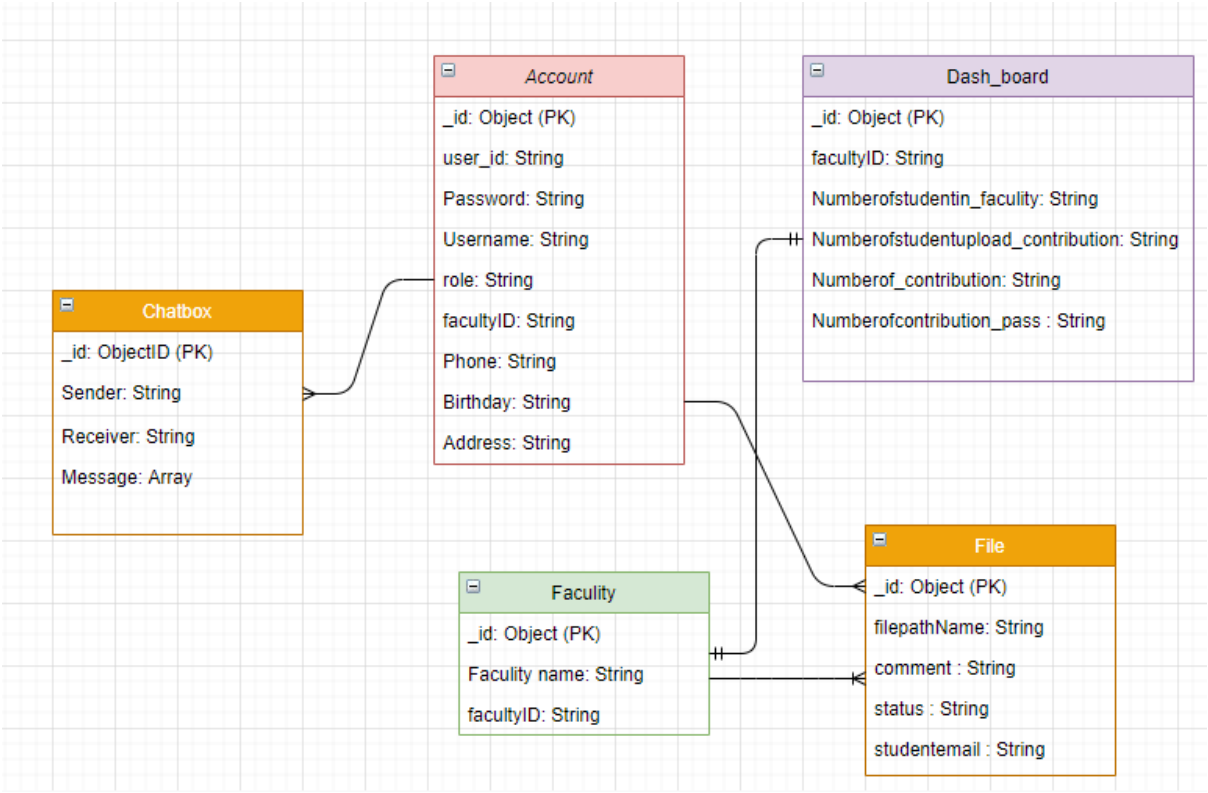


Figure 3:ERD

To meet the project requirements, we have designed five collections included:

_The account collection will contain attributes of the user, such as name, age, email, password, phone number, address, date of birth. The facultyID field we created is for the purpose of referencing other collections in the database. If the account role is a student, then a student can submit multiple files to the system.

_The faculty collection will contain the name and facultyID of the faculty. We use the facultyID to reference the student and the marketing coordinators available in that faculty. One faculty will have many students and only one marketing coordinator

_The chat collection will contain the recipient's email name and sender's email and message. The message field is an array; the data will be pushed when users chat with each other. One faculty will have many students and only one marketing coordinator. If the account's role is a student, then one student will only be able to chat with one marketing coordinator. If the account's role is the marketing coordinator, the marketing coordinator can chat with all students in the faculty

_ The dashboard collection will include the following properties: facultyID, the number of students in the department, the total number of students paid in the department, the total

contribution paid, and the total number of contributions passed. The facultyID field will refer to the faculty in the collection faculty in the database. Each dashboard will be associated with one faculty and vice versa

_ The file collection will include the path of the file, the file name, and the student's email. The path of file field will be the path to upload the file to the server, and the email field will reference the account collection to identify who uploaded the file.

IV. Design:

1. Sitemap:

Following are the links from the main page to the smaller pages of each role.

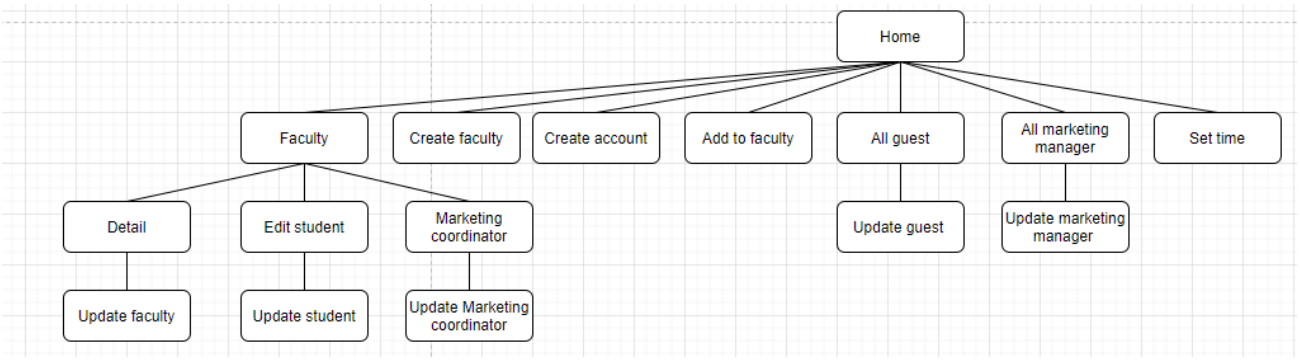


Figure 4: Sitemap admin

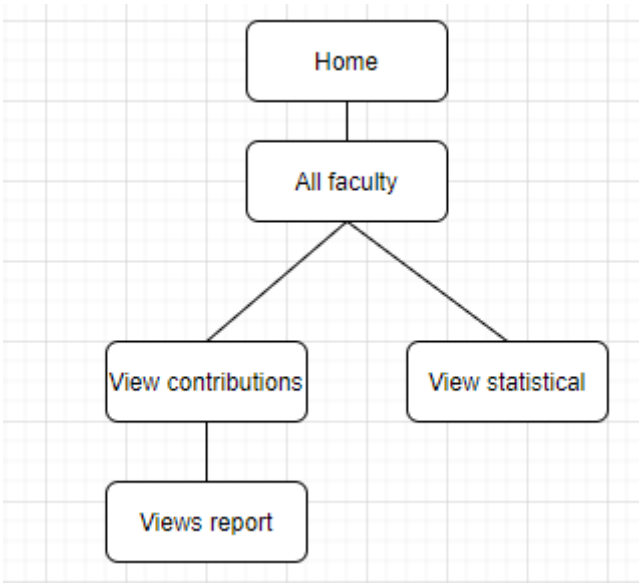


Figure 5:Sitemap marketing manager

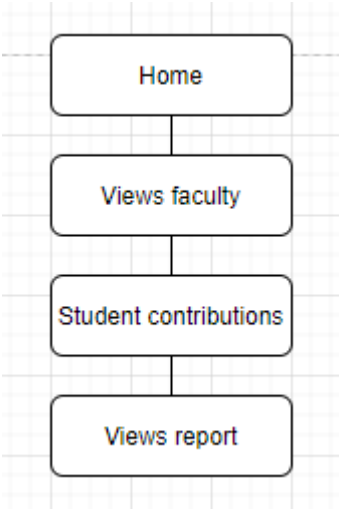


Figure 6:Sitemap guest

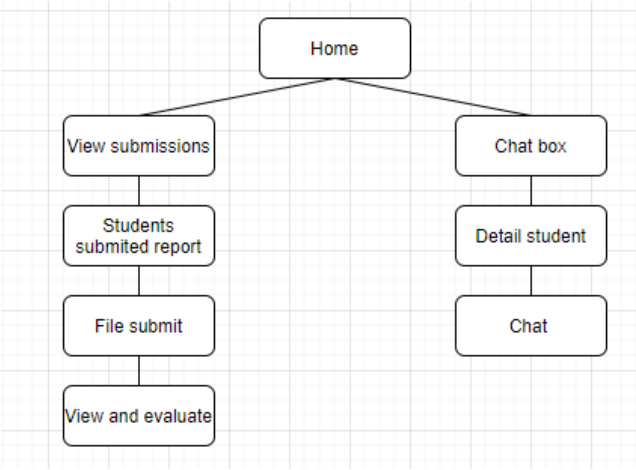


Figure 7:Sitemap marketing coordinator

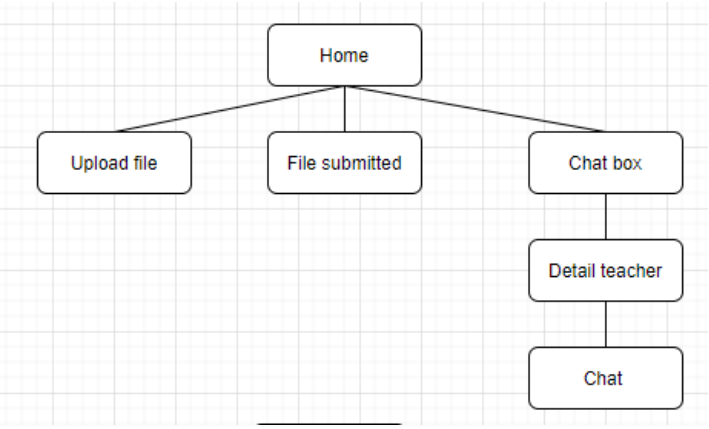


Figure 8:Sitemap student

2. Wireframe

According to the requirements of customers, we have to design interfaces for five different objects: administrator, Student, Marketing coordinator, Marketing manager, and Guest. For each object, we will design a separate interface, interface. The interface will be designed optimally and suitable for devices such as computers, mobile phones...

2.1 Login

First, when a user visits the website, the login page will be displayed first. Login is designed with a header in h tag with large clear font. Both the username and password fields are displayed in the text in the input area, making it easy for users to understand.

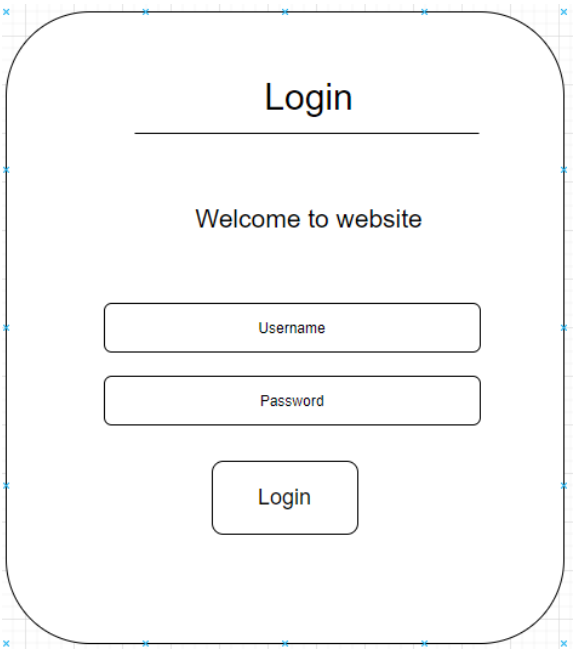


Figure 9:Wireframe login of computer and mobile

The layout of the interface is divided into four main parts: the header contains the “X” button to log out, the menu contains the functions that the user can use, and it is displayed on the left-hand side, the content contains the content, the Finally, the footer.

When entering the student interface, the menu will display student functions such as home, upload file, submitted file, and chatbox. Content will display personal information of students, including user name, email, faculty, Phone, Birthday, and address. Information will be included in the table to make it easier for the user to observe. The title section uses h-tags and large fonts

so that users can better understand the website they use. Pages displaying personal information are of the same structure.

2.2 Personal information

Desktop, laptop, pc:

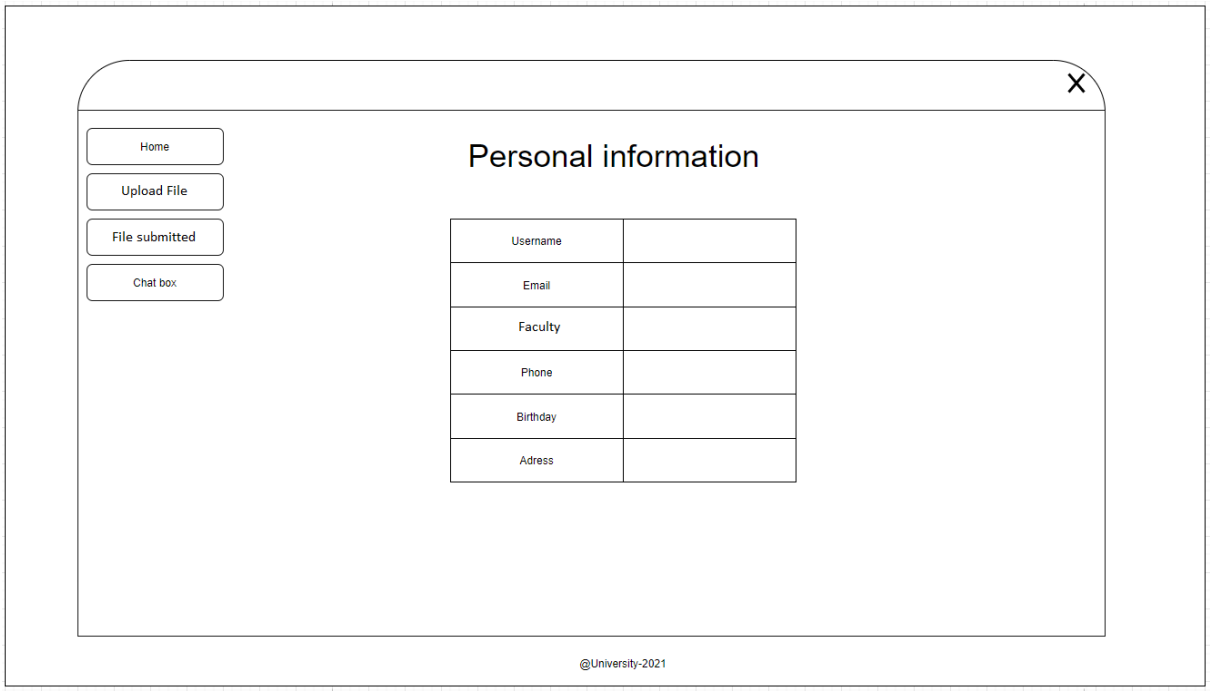


Figure 10:Wireframe home page of student

Mobile:

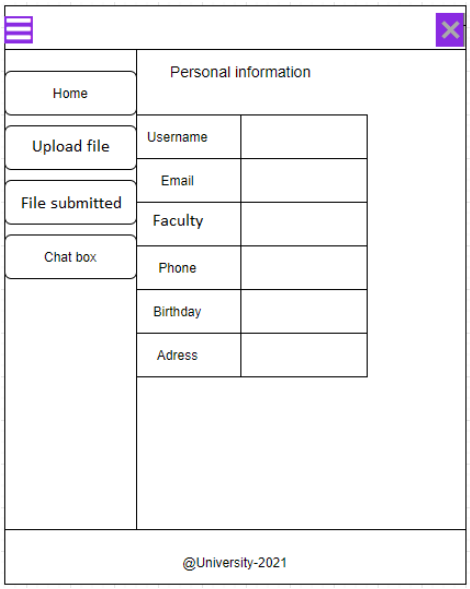


Figure 11:Wireframe home page of student mobile

To match mobile devices, the menu section is hidden and shows an icon for the menu on the left to save space for mobile devices. When you need to use the menu, just click on the icon of the menu, the menu will appear, and when the user clicks on the icon of the menu, the menu will be hidden.

2.3 Upload file

Desktop, laptop, pc:

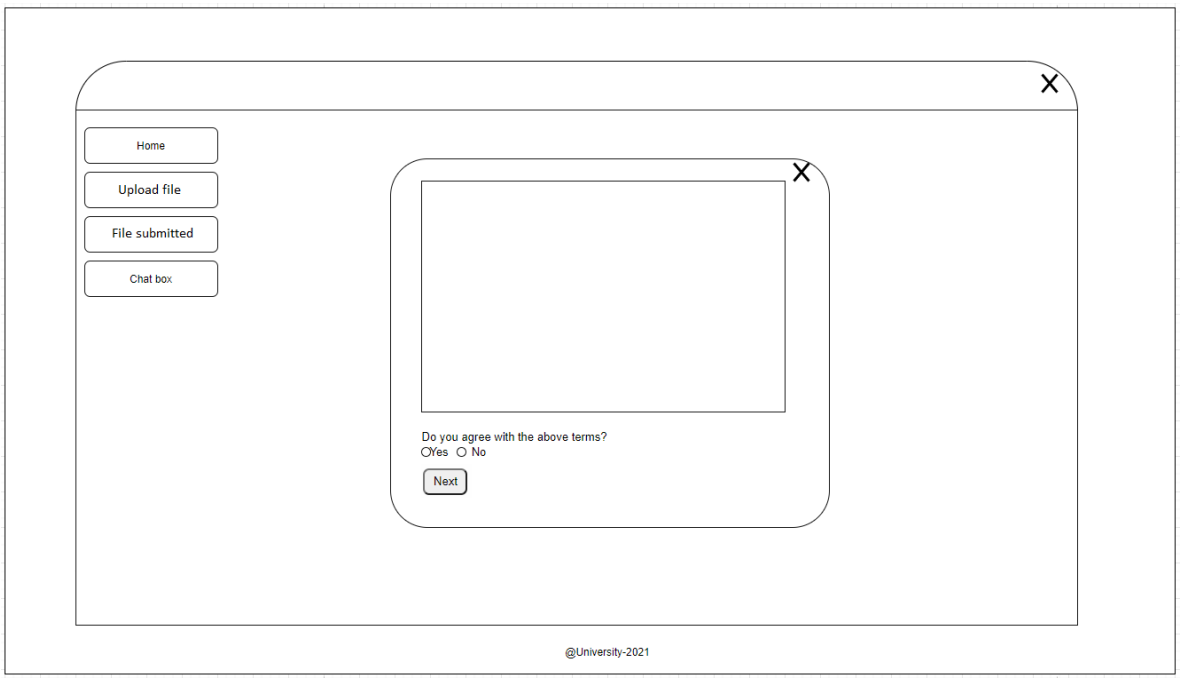


Figure 12:Wireframe popup terms

The terms display will use a popup to show the student's terms when trying to upload files.

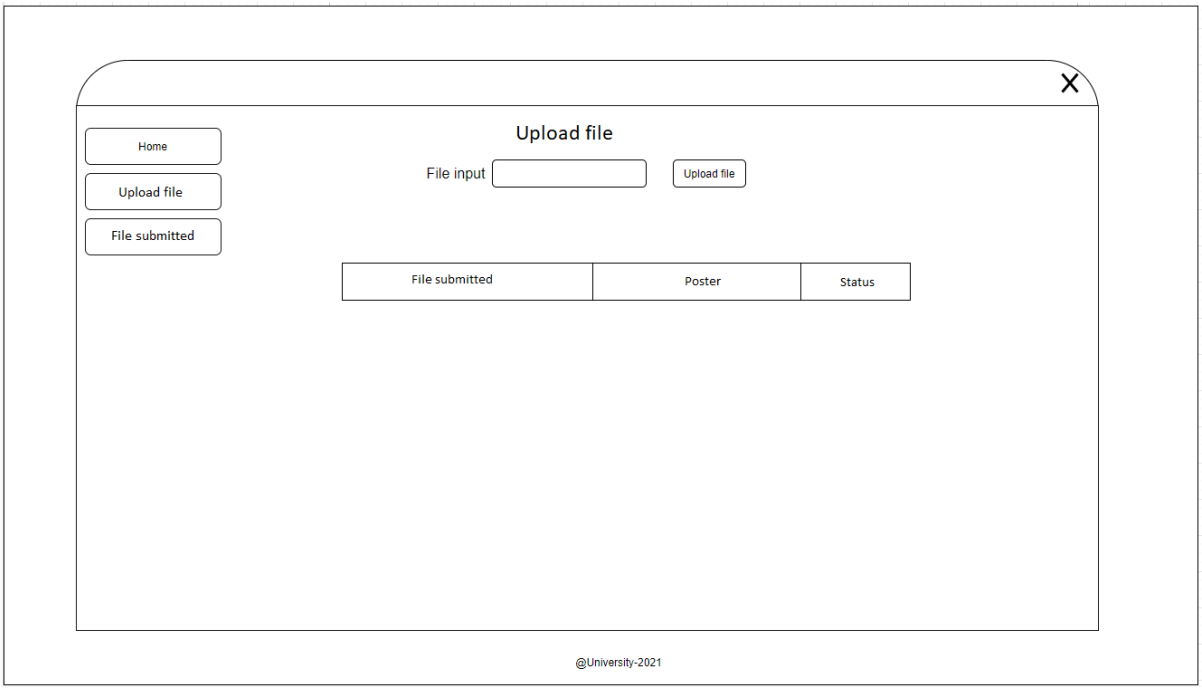


Figure 13:Wireframe upload file

Mobile:

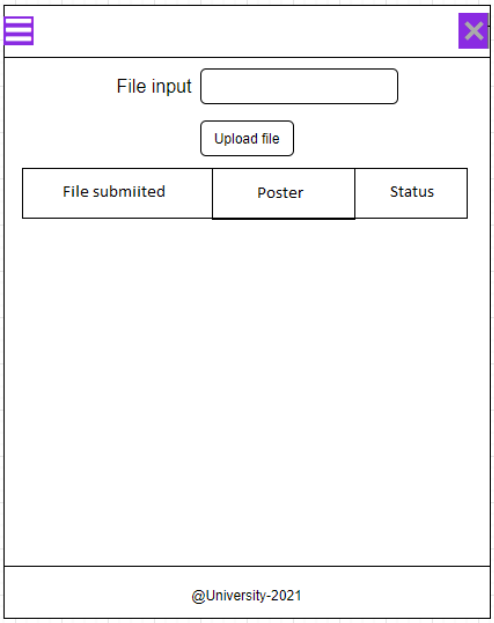


Figure 14:Wireframe upload file mobile

Using the input file header right before the file upload area will help users know the part they can upload files, and the file upload button is located next to the user so that users can enter files and easily upload files.

The submitted files are placed in tables for easier viewing and management by users.

Desktop, laptop, pc:

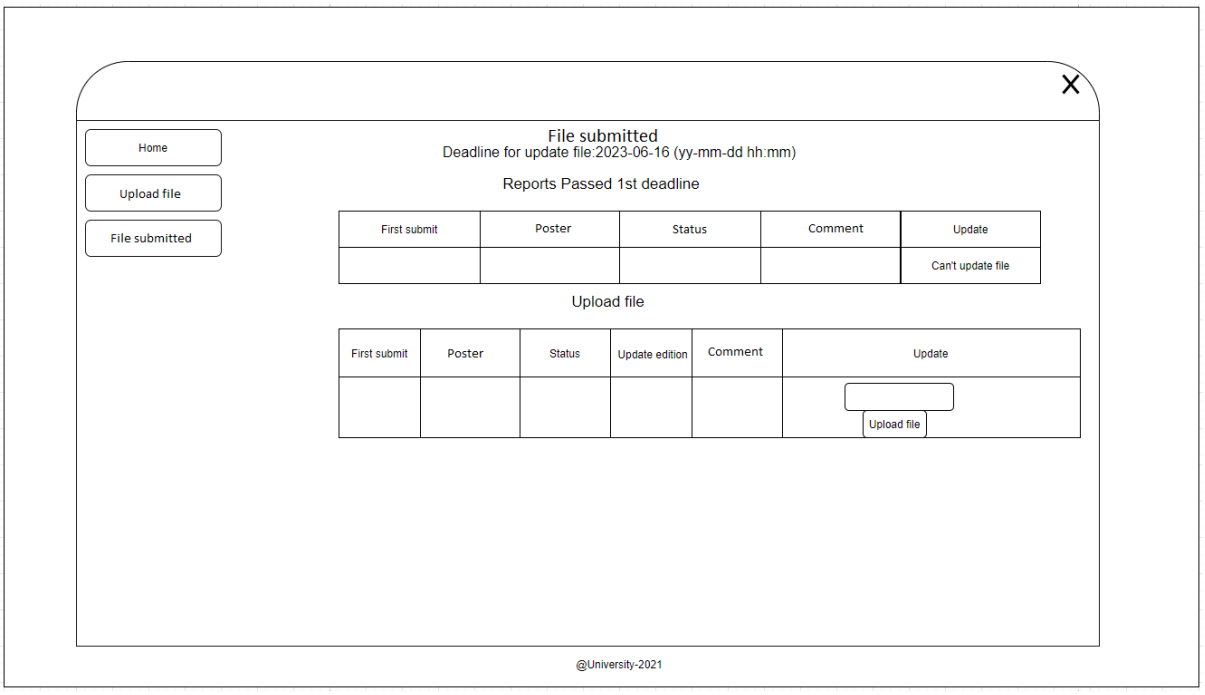


Figure 15:Wireframe file submitted

It shows the deadline for the report; table 1 shows the reports that have passed and the file could not be uploaded. Table 2 shows the reports that have not been graded or failed. If there is a deadline, users can upload another file.

2.4 Message

Desktop, laptop, pc:

Figure 16: Wireframe chatbox of Student and marketing coordinator

Mobile:

Figure 17: Wireframe chatbox of Student and marketing coordinator mobile

Students and coordinators can contact each other through the chatbox. The text chat section has: type your message here to help users know where to write the chatbox. Next to it is an icon for the button to send the message.

2.5 All faculty

Desktop, laptop, pc:

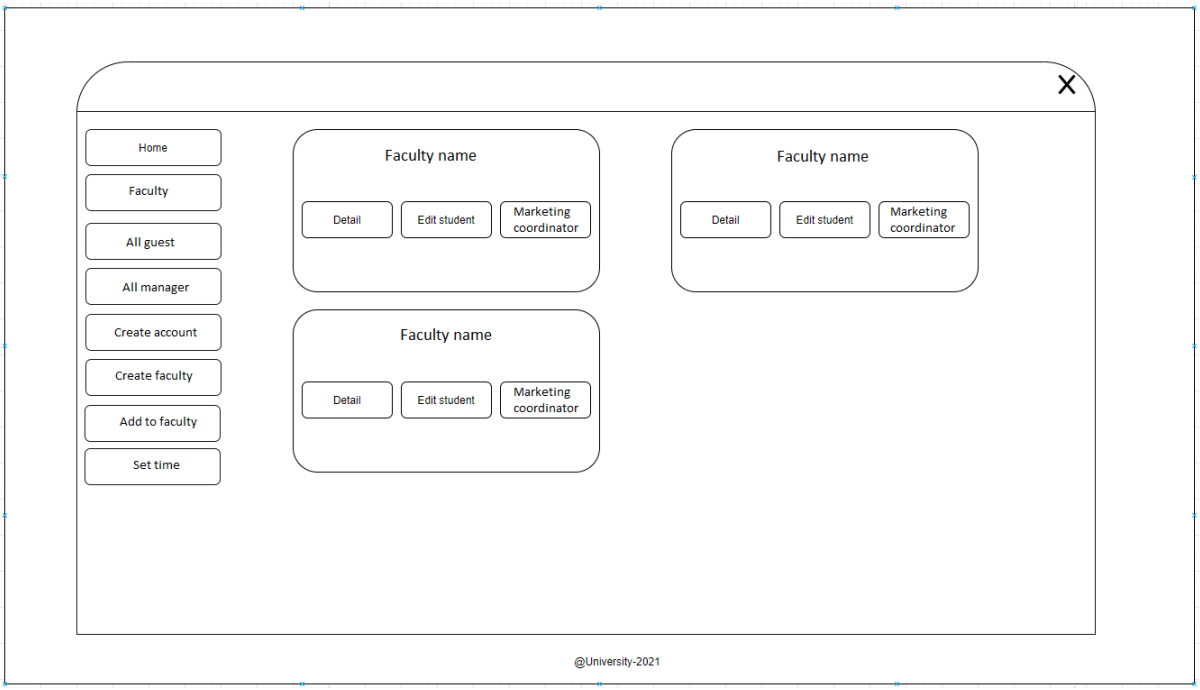


Figure 18:Wireframe all faculty of admin

Mobile:

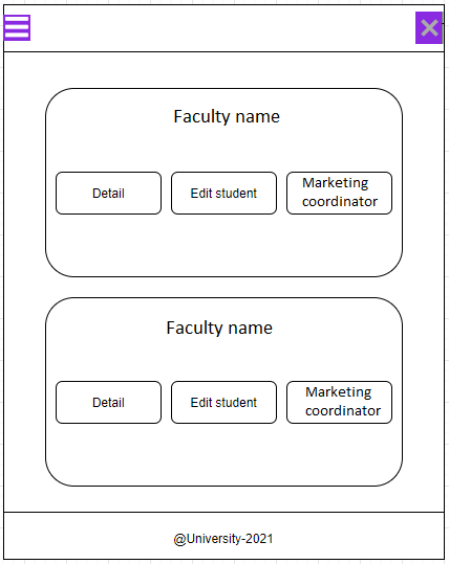


Figure 19:Wireframe all faculty of admin mobile

In faculty, it will display the faculty. The faculty will be wrapped into a frame, and functions are placed in the button so that the user can observe and understand its usage and functions.

Desktop, laptop, pc:

Figure 20:Wireframe all guest of the admin

Mobile:

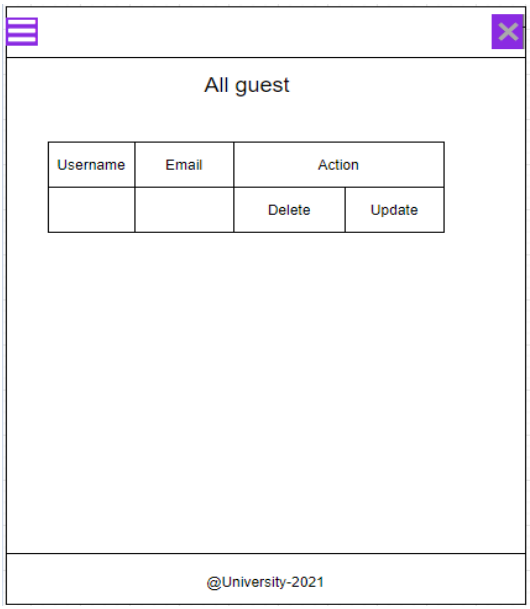


Figure 21:Wireframe all guest of admin mobile

Show all guest and marketing managers are included in the table for easier management.

2.6 Set time

Desktop, laptop, pc:

Figure 22:Wireframe set time of admin

Mobile:

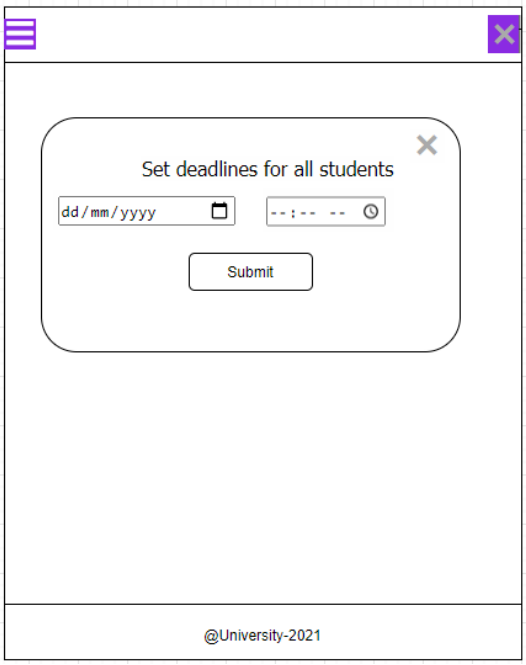


Figure 23:Wireframe set time of admin

Admin can set a time for the report. The date entry uses the date tag input for users to enter the correct category easily. The set time is also designed in a popup format that is both easy to use and saves an area.

2.7 View and evaluate

Desktop, laptop, pc:

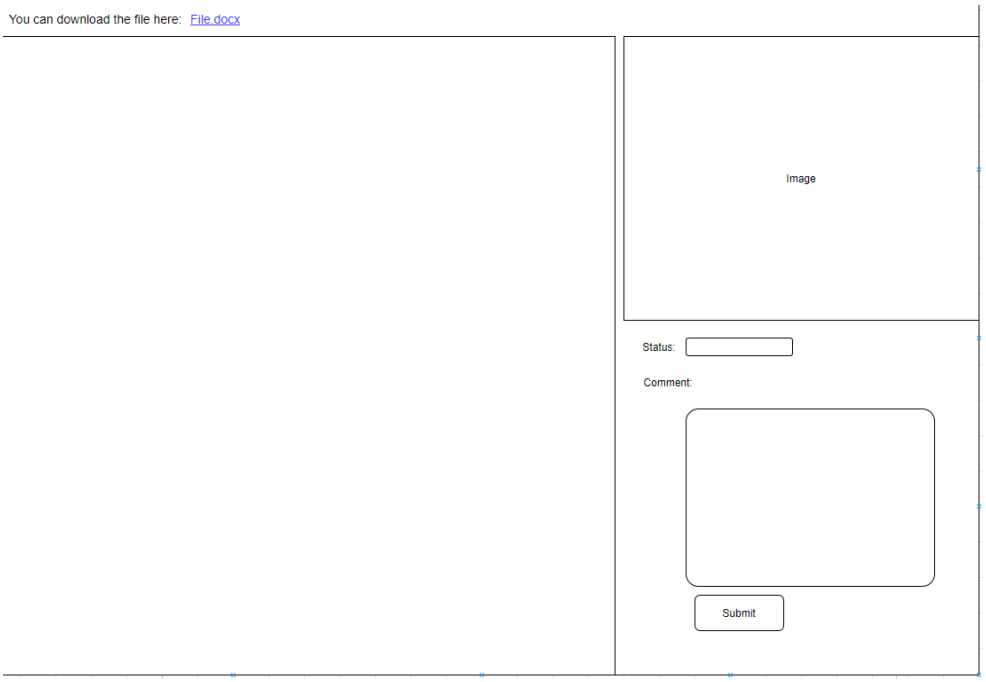


Figure 24:Wireframe view and evaluation of marketing coordinator

Mobile:

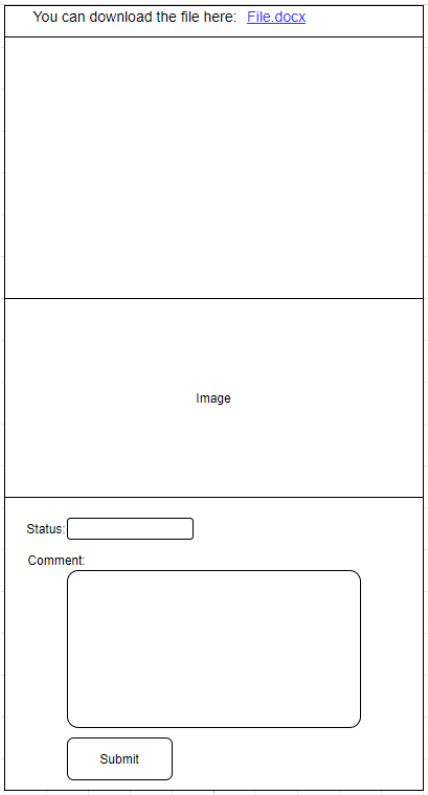


Figure 25:Wireframe view and evaluation of marketing coordinator mobile

The Marketing coordinator can view the student's uploaded files and rate them. The rating: status shows the current score and can choose the point to change without having to enter manually.

3. Interface

The colors used on the web interface are all light and gentle colors that do not cause inhibition.

When switching from desktop to mobile, the menu is hidden to save screen space. The fonts have all been adjusted to suit the phone.

Buttons will add effects.

3.1 Login

First, when a user visits the website, the login page will be displayed first. Then the user will enter the username and password; the system will check the username and password to provide an interface for each object above. It is suitable for both computers and mobile.

Desktop, laptop, pc:

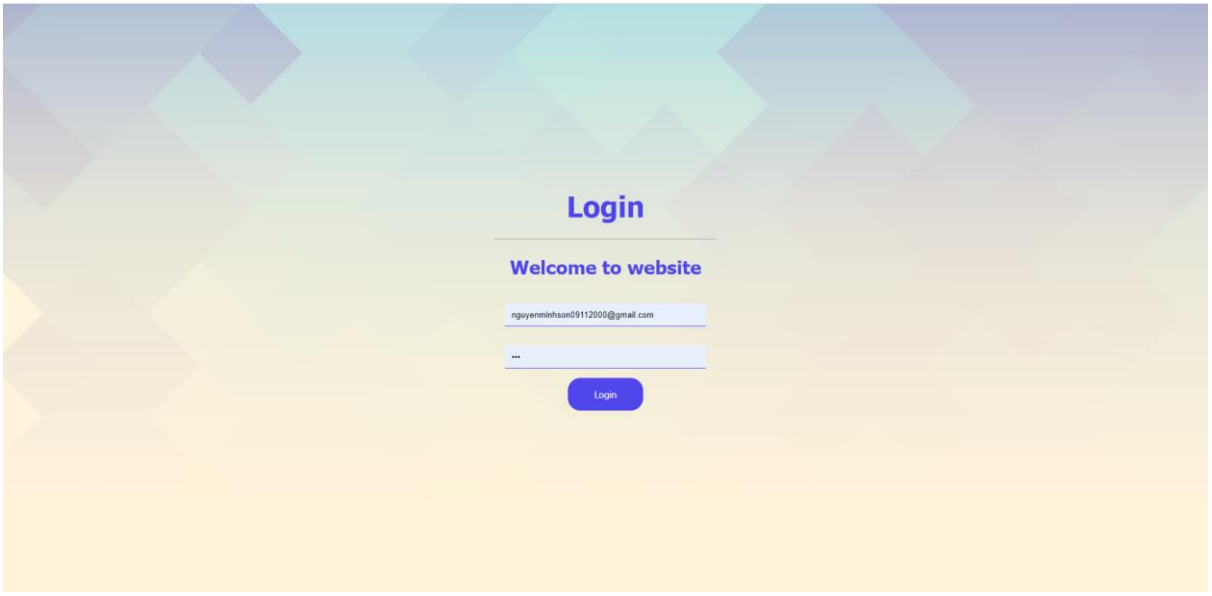


Figure 26:Login

Mobile:

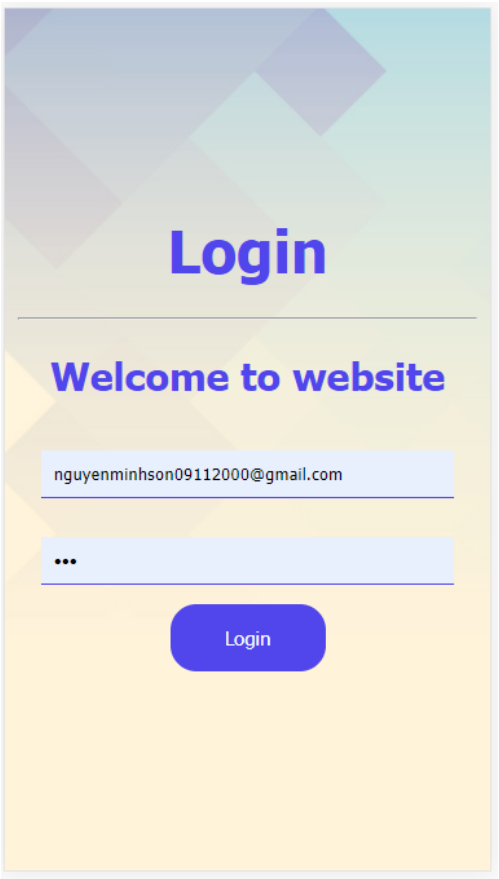


Figure 27:Login mobile

The layout of the interface is divided into four main parts: the header contains the “X” button to log out, the menu contains the functions that students can use, and it is displayed on the left-hand side, the content contains the content, the last part. The same is the footer.

When entering the student interface, the menu will display student functions such as home, upload file, submitted file, and chatbox. Content will display personal information of students, including user name, email, faculty, Phone, Birthday, and address.

3.2 Personal information

Desktop, laptop, pc:

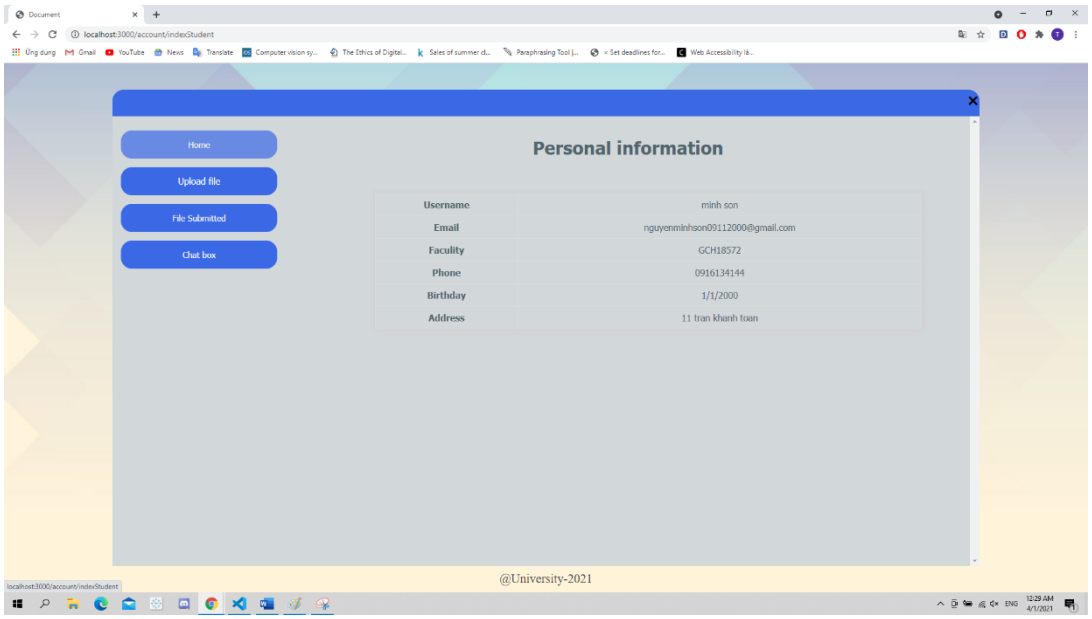


Figure 28:home page student

Mobile:

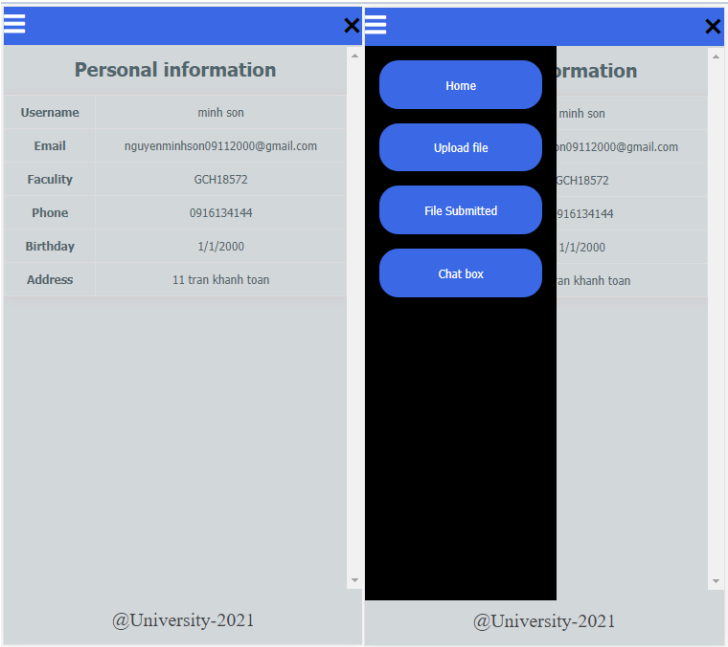


Figure 29:homepage student mobile

To be suitable for mobile, the menu section will be hidden and display an icon for the left-hand menu to save space for the mobile device. When you need to use the menu, just press on the icon for the menu, the menu will appear, and when the user presses the icon of the menu, the menu will be hidden.

3.3 Upload file

Desktop, laptop, pc:

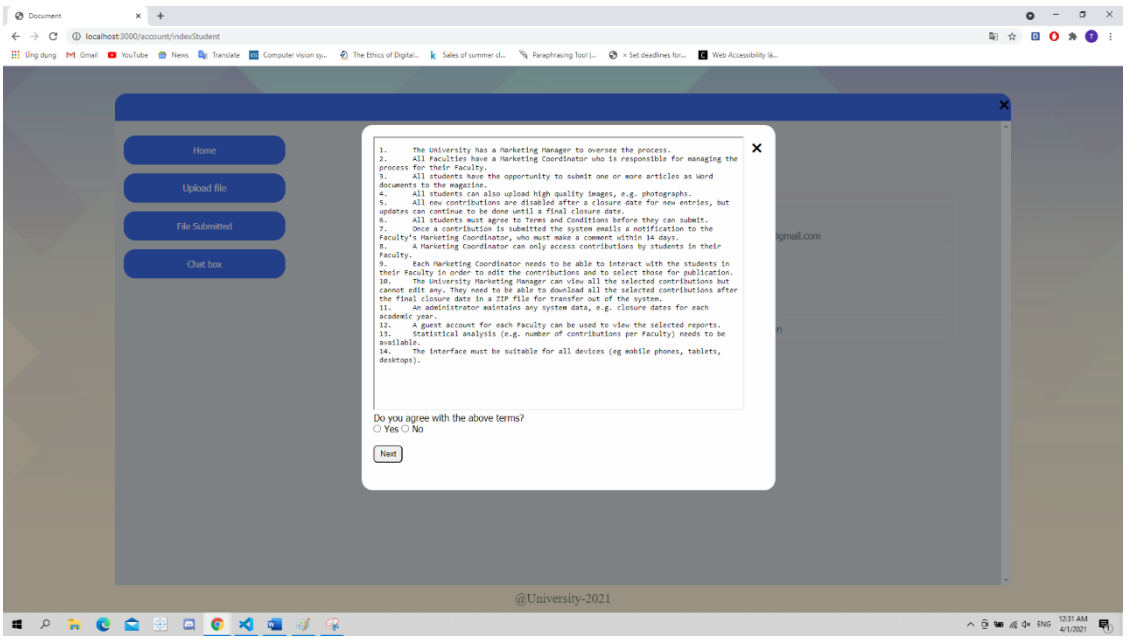


Figure 30:popup terms desktop

After clicking to Upload file, it will display the terms, and students need to agree with the terms before they can access the updated files.

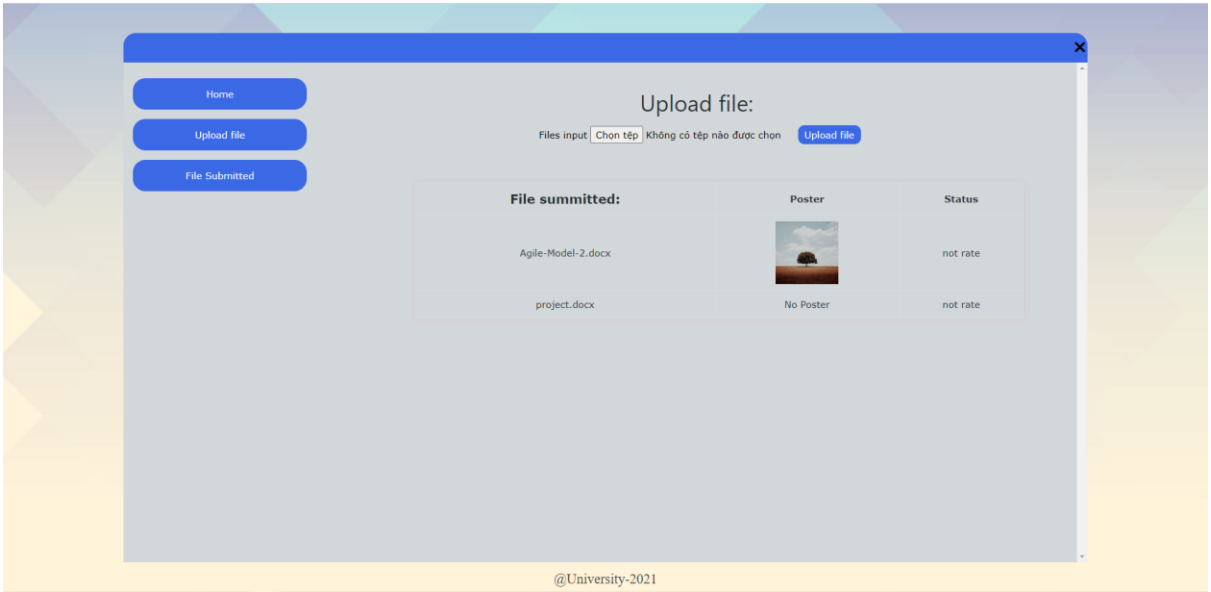


Figure 31:upload file

Mobile:

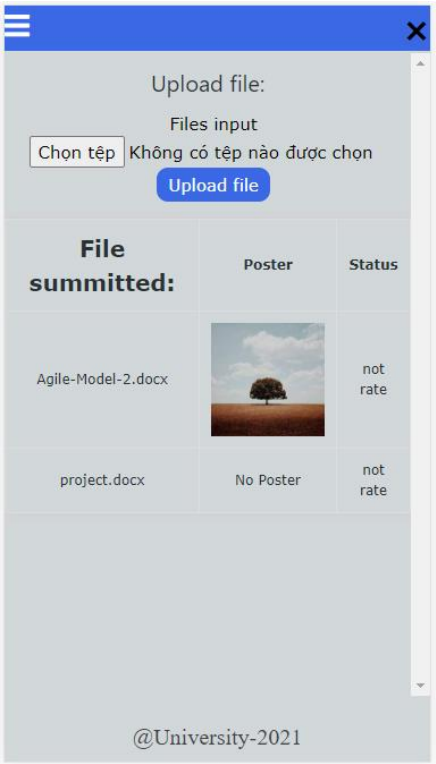


Figure 32:Upload file mobile

The input file section is for the user to import the files and images to upload the file. The table below shows the files that the user has uploaded. The content uses scroll, so the user can scroll down to see the content if the content is long.

Desktop, laptop, pc:

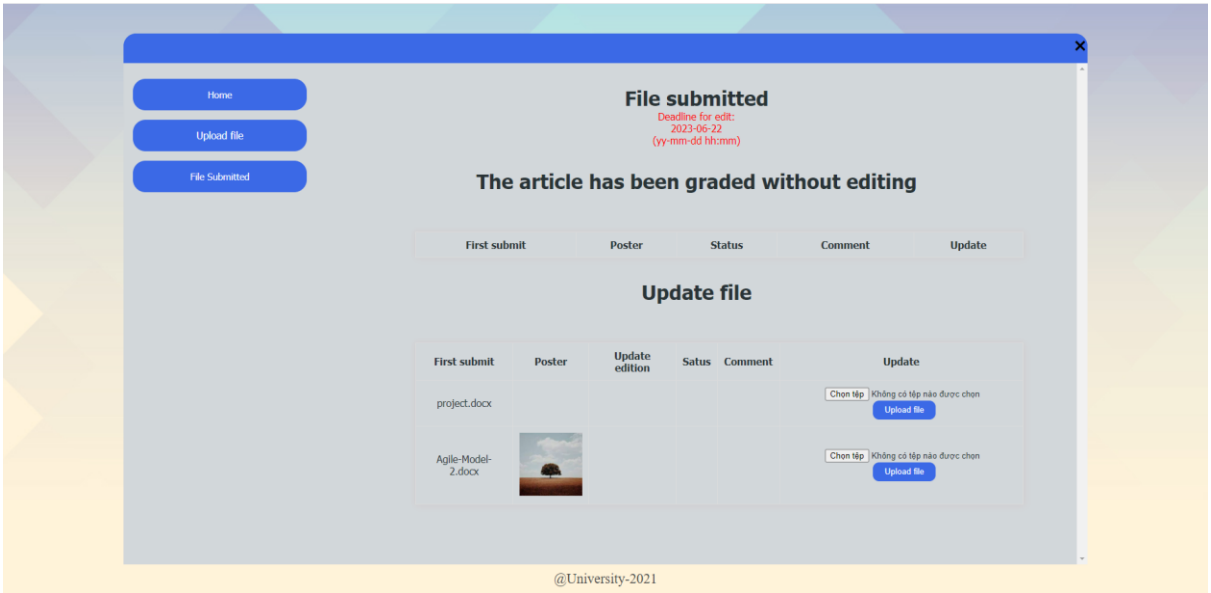


Figure 33:File submitted

Mobile:

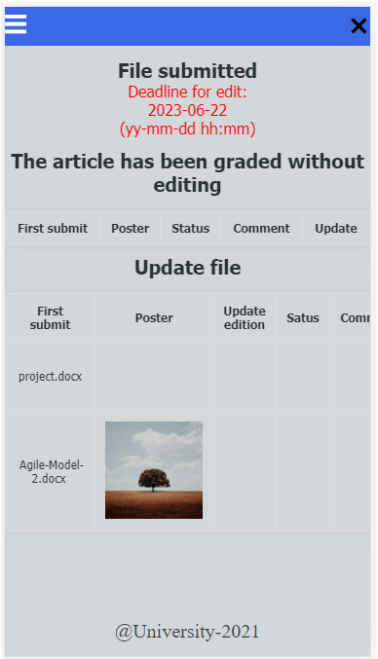


Figure 34:File submitted mobile

It shows the deadline for the report; table 1 shows the reports that have passed and the file could not be uploaded. Table 2 shows the reports that have not been graded or failed. If there is a deadline, users can upload another file.

3.4 Massage

Desktop, laptop, pc:

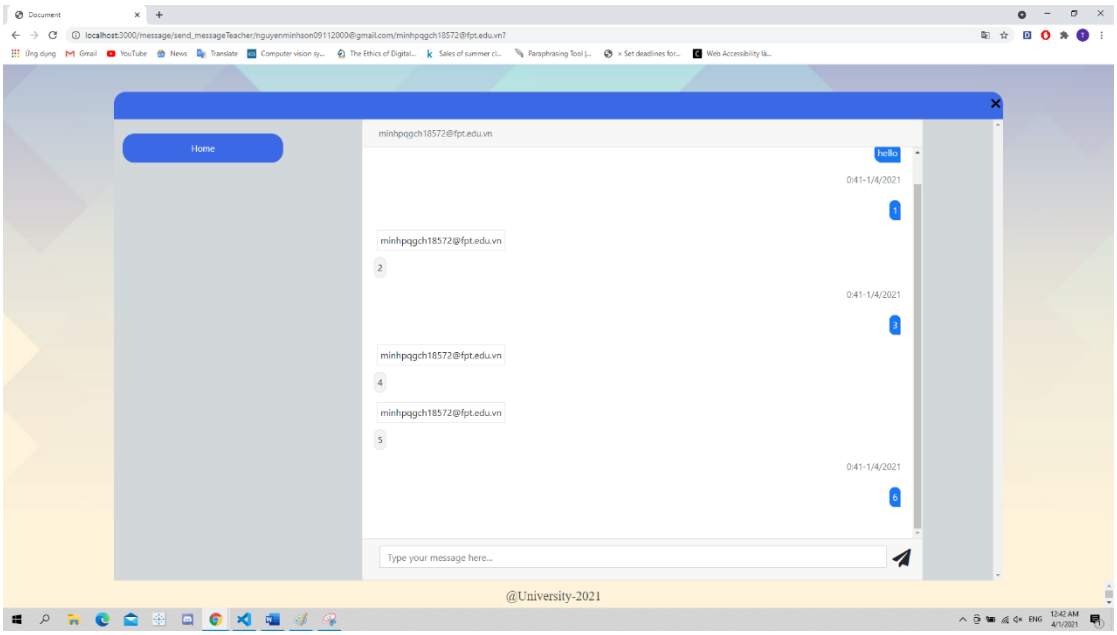


Figure 35:chat box of Student and marketing coordinator

Mobile:

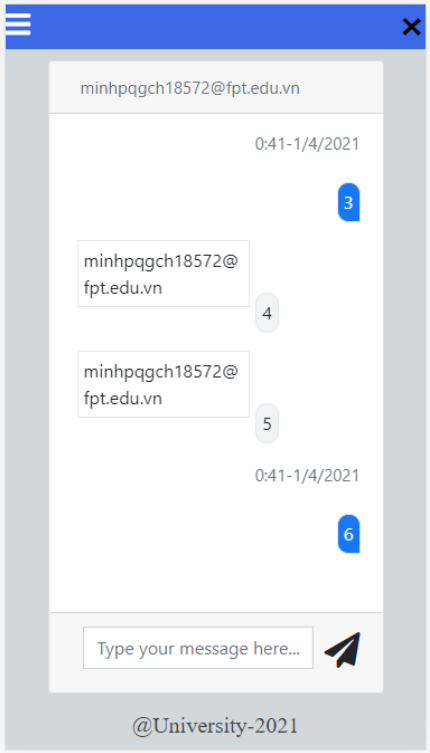


Figure 36:chat box of Student and marketing coordinator mobile

Students and coordinators can contact each other through the chatbox.

3.5 All faculty

Desktop, laptop, pc:

Figure 37:All faculty of admin

Mobile:

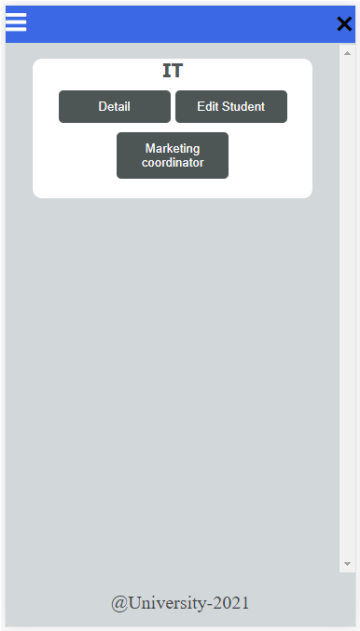


Figure 38:all faculty off admin mobile

In faculty, it will have the functions: create faculty, create student and create marketing coordinator. Content will display the Faculty name and can be edited by “Detail,” can view the students in the department, can view the marketing coordinator of that department.

Desktop, laptop, pc:

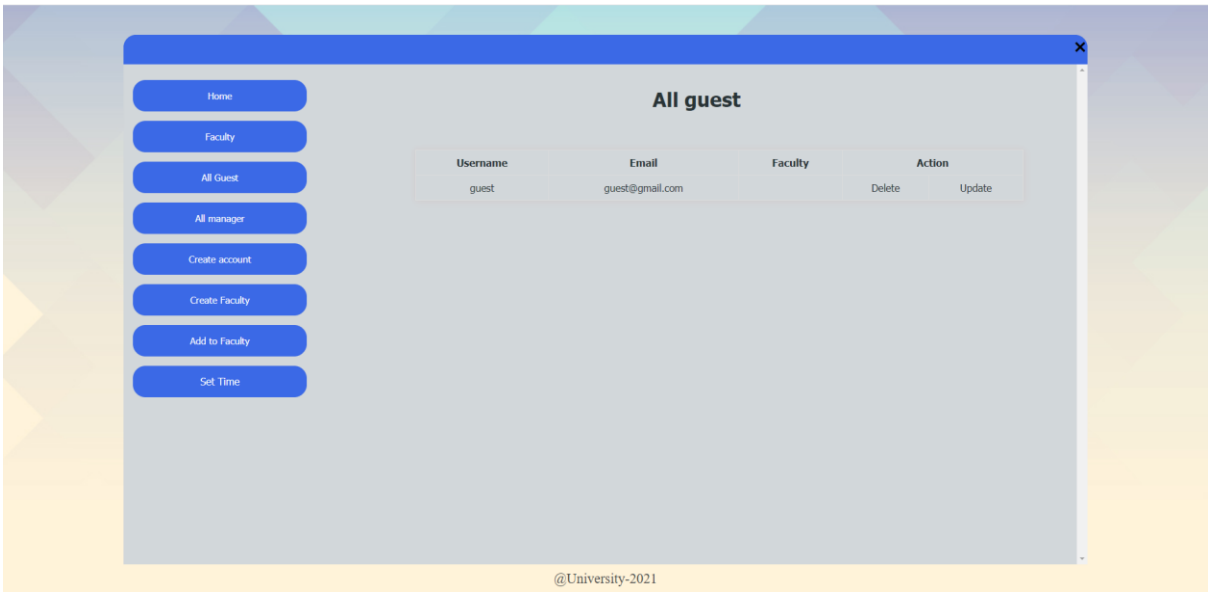


Figure 39:all guest of admin

Mobile:

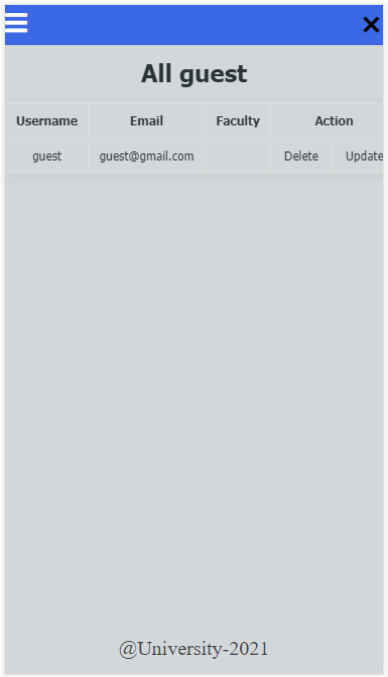


Figure 40:all guest of admin mobile

Show all guests. Admin can create guests.

3.6 View bar chart

Desktop, laptop, pc:

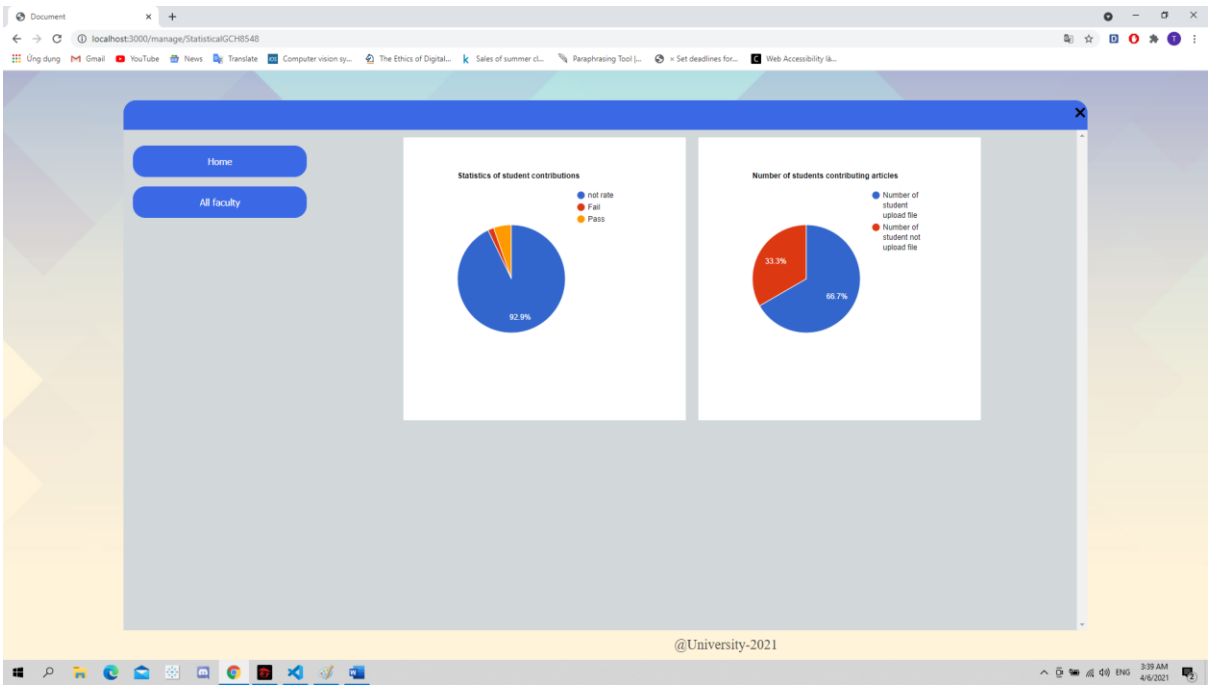


Figure 41:bar chart

Mobile:

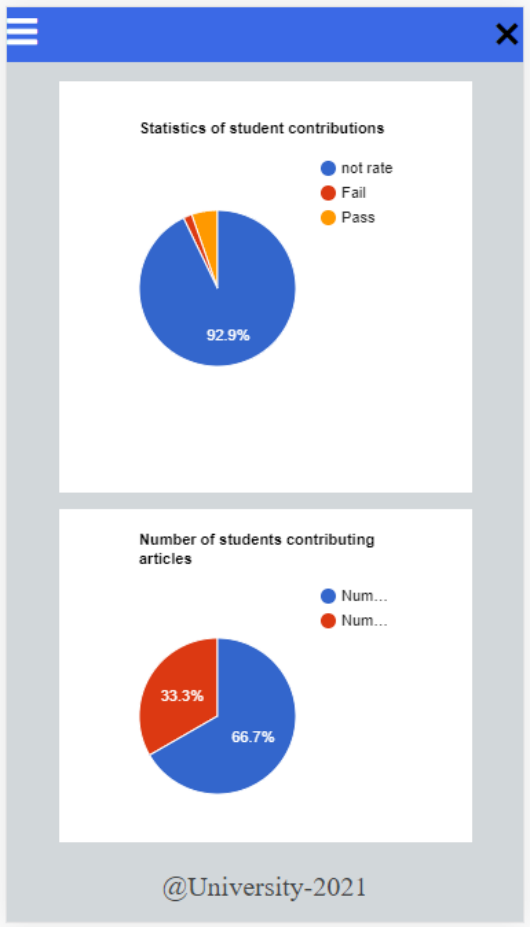


Figure 42:bar chart mobile

The marketing coordinator can view statistics about reports.

3.7 Set time

Desktop, laptop, pc:

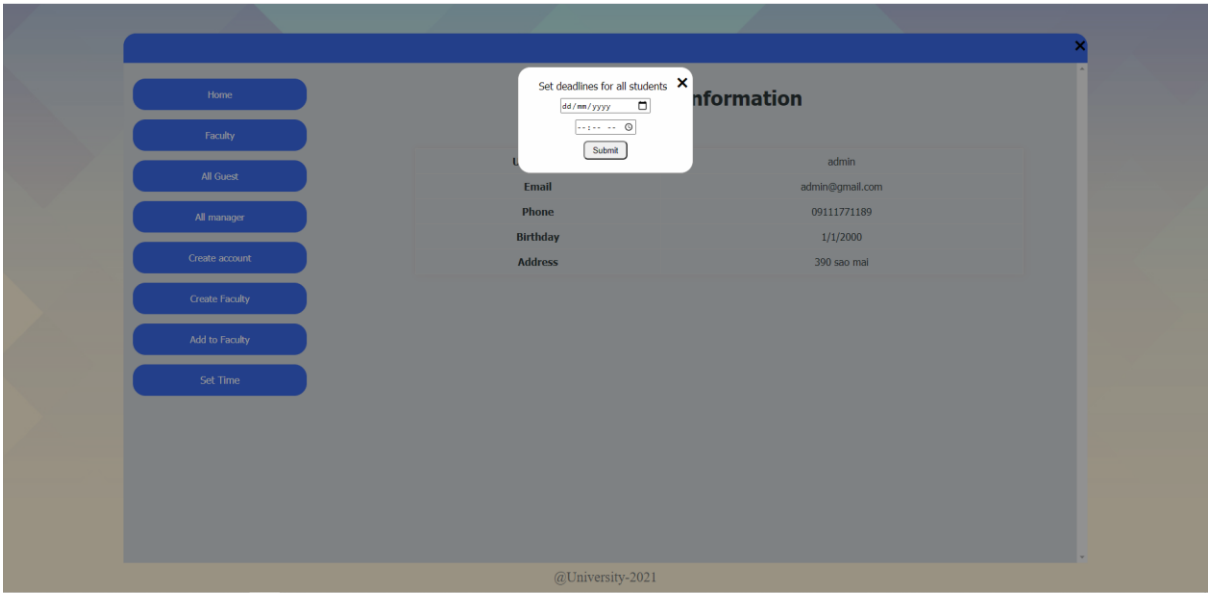


Figure 43:set time of admin

Mobile:

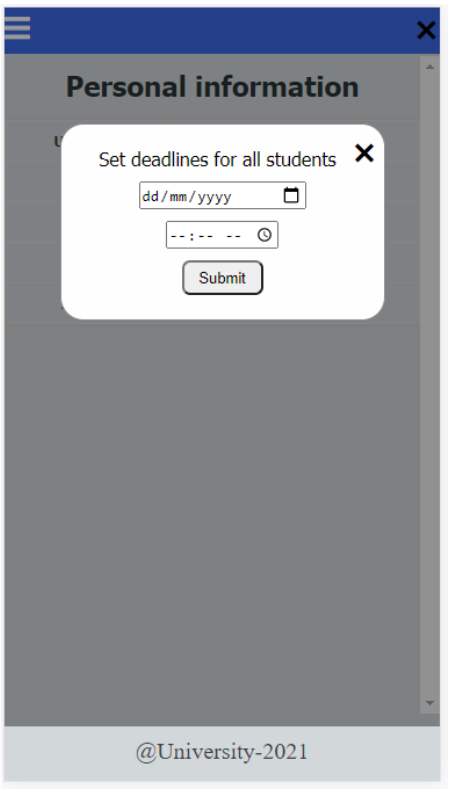


Figure 44:set time of admin

Admin can set a time for the report of students.

3.8 View and evaluate

The Marketing coordinator can view the student's uploaded files and rate them.

Desktop, laptop, pc:

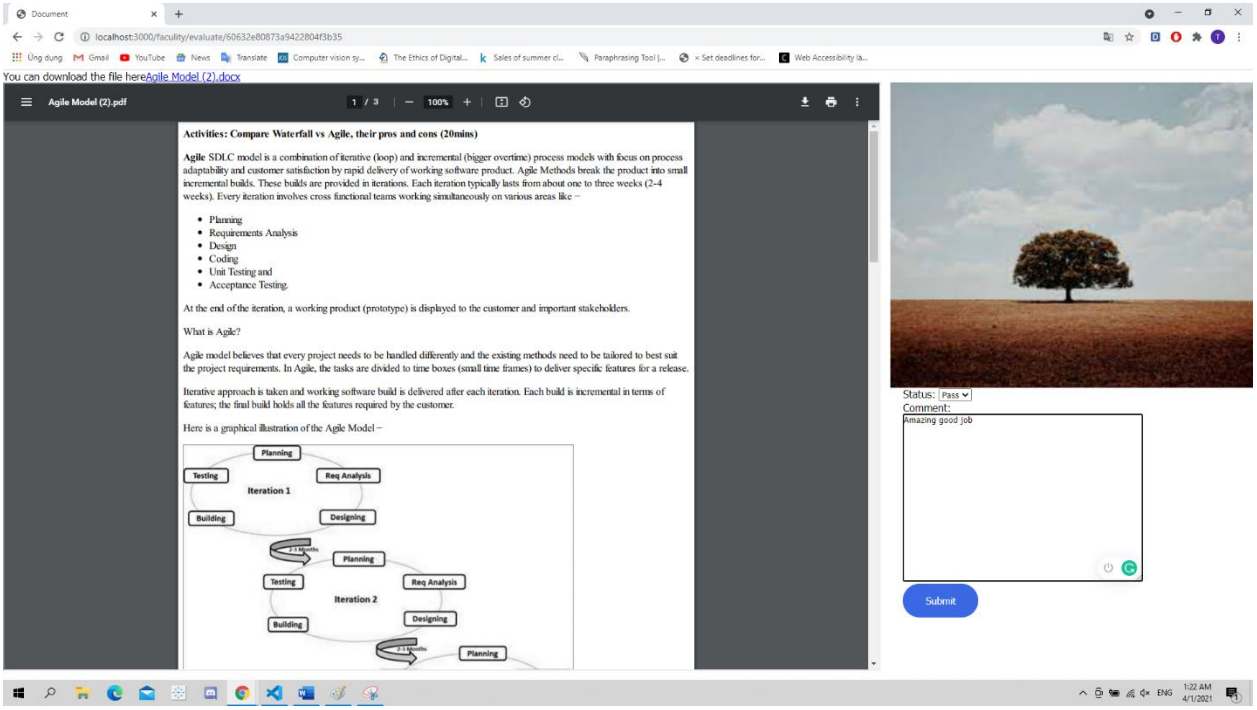


Figure 45:View and evaluate of marketing coordinator

Mobile:

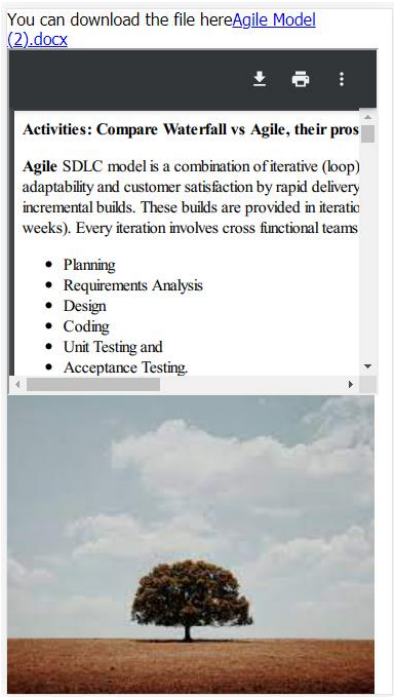


Figure 46:View and evaluate of marketing coordinator mobile

The interface uses bright colors, good-looking, and does not cause inhibition for the viewer. Buttons are added effects to let the user see where they are clicking and create an interesting feeling for the user. The clearly displayed functions make it easy to remember and use at the same time.

4. Usability

The website's interface is easy to recognize and easy to use functions:

- The menu is fixed on the left-hand side, and all functions are displayed in the menu bar so the user can use it easily.
- When using on the phone, the menu is hidden and displays the menu icon so the user can click on it to display the menu. The fonts will also be adjusted to suit the phone screen size.
- The website has headers for the user to see what function they are using.
- Naming functions are quite simple so that users can remember the functions in the website easily.
- When entering incorrect data, the types of data to be entered will be printed out so that the user can easily recognize and modify.
- The interface is used in creative colors that is comfortable and does not cause inhibition to the viewer.

5. Accessibility

- The headings use h-tags and are bold, large font easy to see for everyone to see.
- Display the correct data format when input is incorrect.
- Not only using the mouse to navigate and use the website, but the keyboard can also use those functions.

Image tags all have an alt attribute

V. Implementation (Functionality):

1. Use case Diagram:

Below is an overview diagram of the whole system. The diagram includes five actors that are guests, students, admin, marketing coordinator, and marketing manager. They are the people who interact directly with the system. With a clear decentralization, each task and their way of interacting is different on the system. For example, guests can view the selected report, but Students do not have permission to do this.

1. Guest use case diagram:

With guest authorization, there are three functions allowed to use and interact with the system, namely login/logout, view profile, view the selected report.

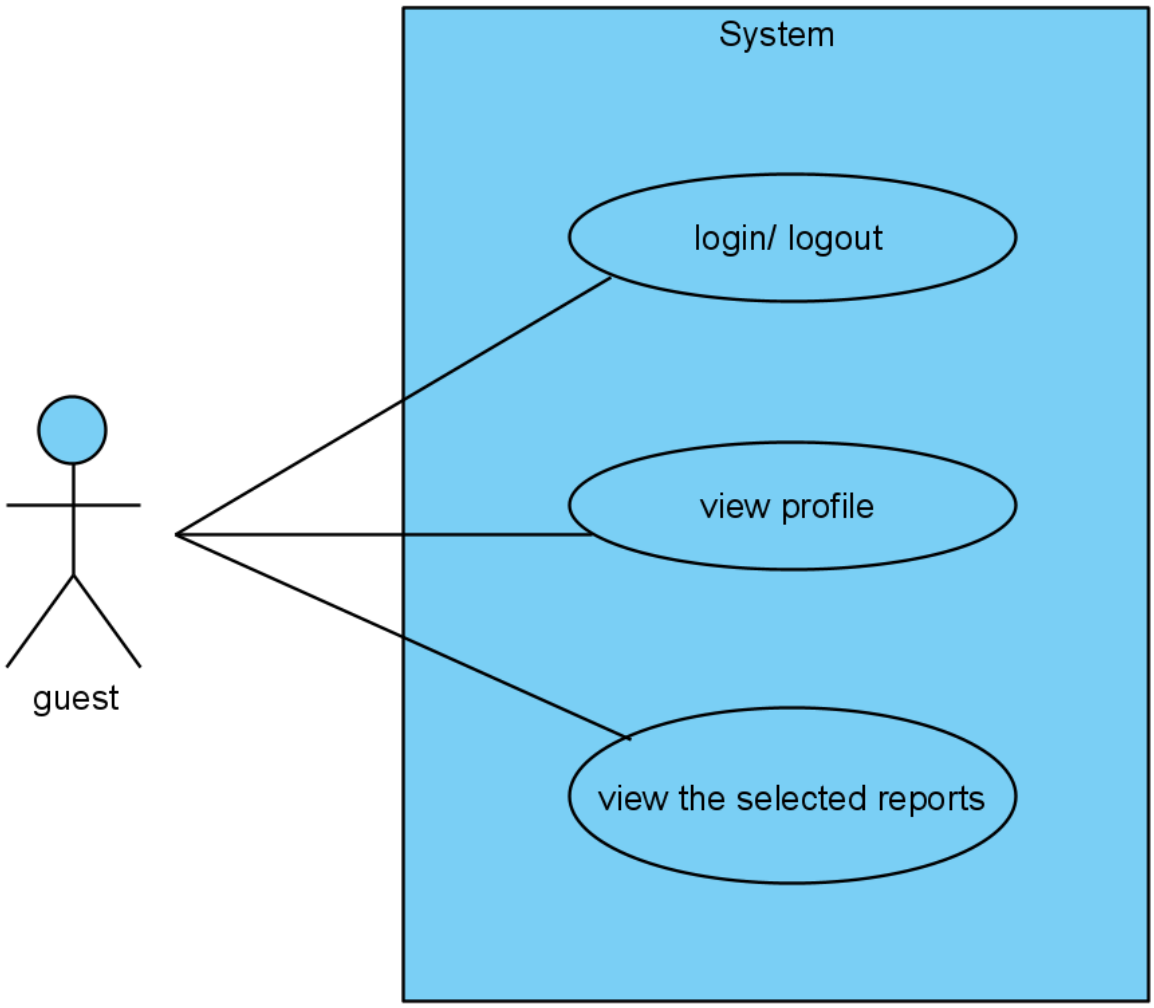


Figure 48 Guest use case diagram

2. The administrator uses a case diagram

With the Administrator decentralization, there are three functions allowed to use and interact with the system; they are any system data, Statistical analysis, and account management, set time.

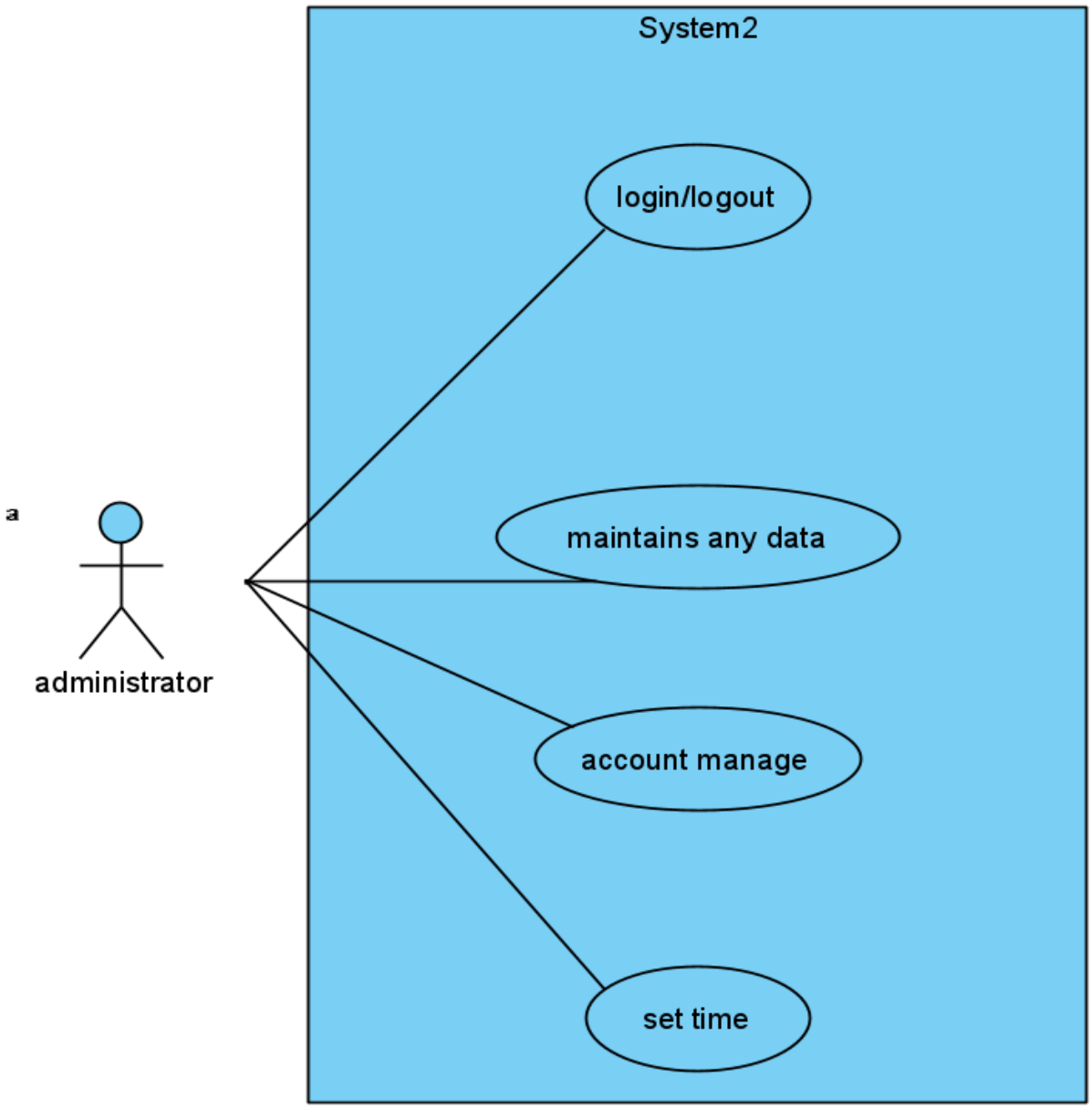


Figure 49 Administrator use case diagram

3. The marketing coordinator use a case diagram

With Administrator privileges, there are five functions allowed to use and interact with the system, namely login/logout, chatbox, view the selected reports, give comments.

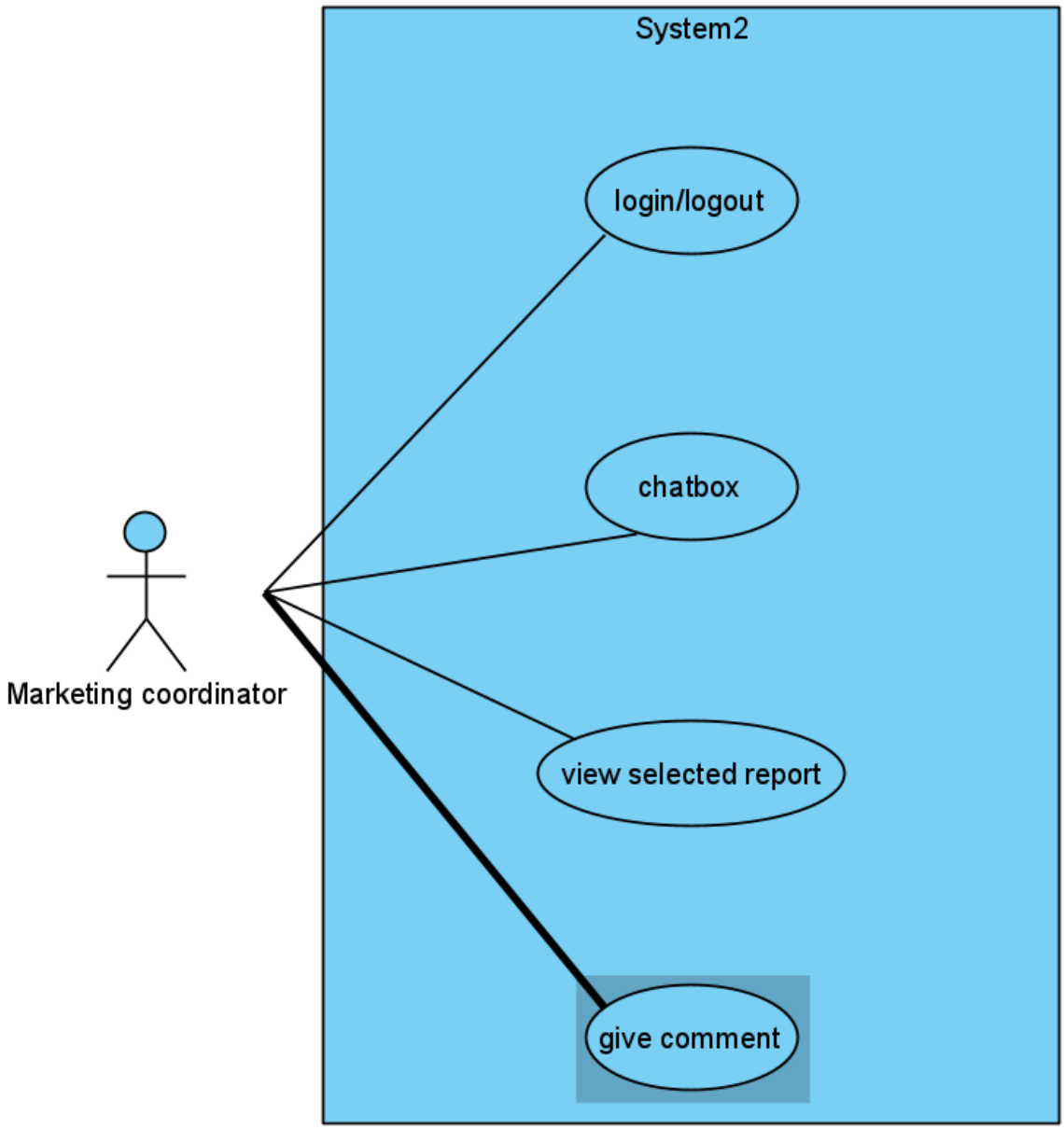


Figure 50 Marketing coordinator use case diagram

4. Students use a case diagram

With Administrator rights, there are five functions that are allowed to use and interact with the system, namely login/logout, chatbox, view profile, submit one or more articles as word documents and pictures, update articles.

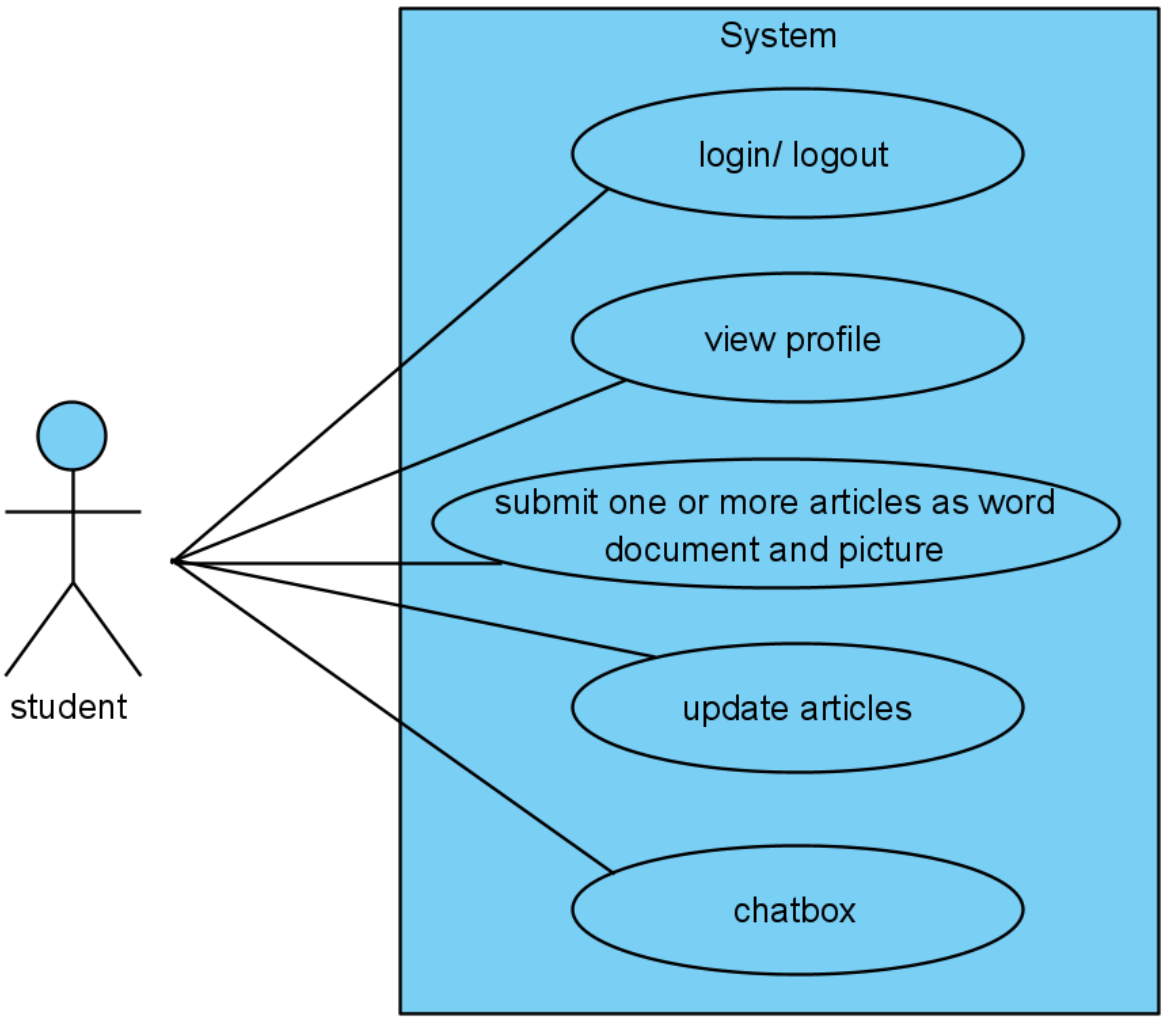


Figure 51 Student use case diagram

5. The marketing manager uses a case diagram

With Administrator privileges, there are six functions allowed to use and interact with the system: login/logout, download the selected reports, view profile, view the selected reports, Statistical Analysis.

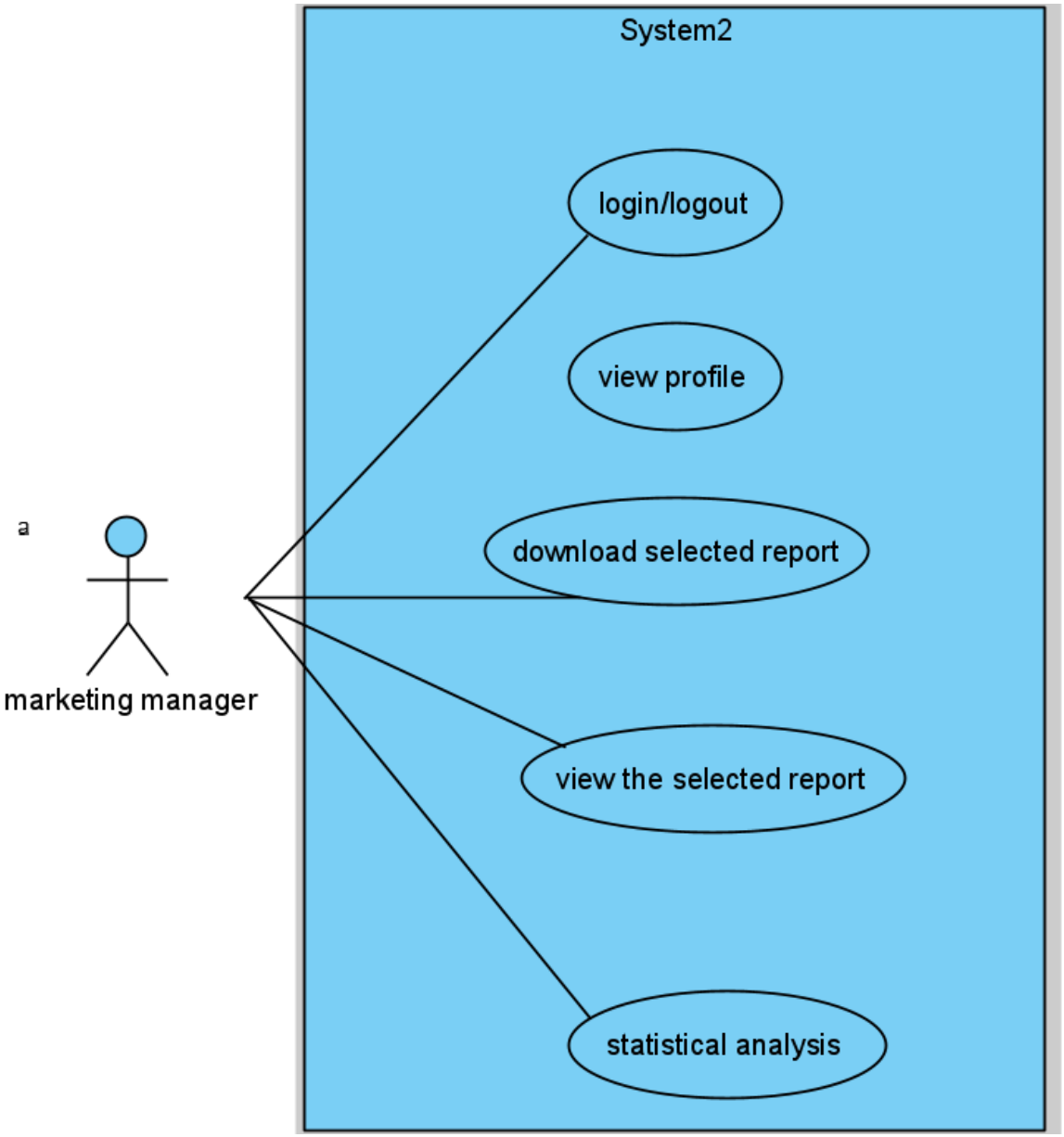


Figure 52 Marketing manager use case diagram

2. Activity Diagram:

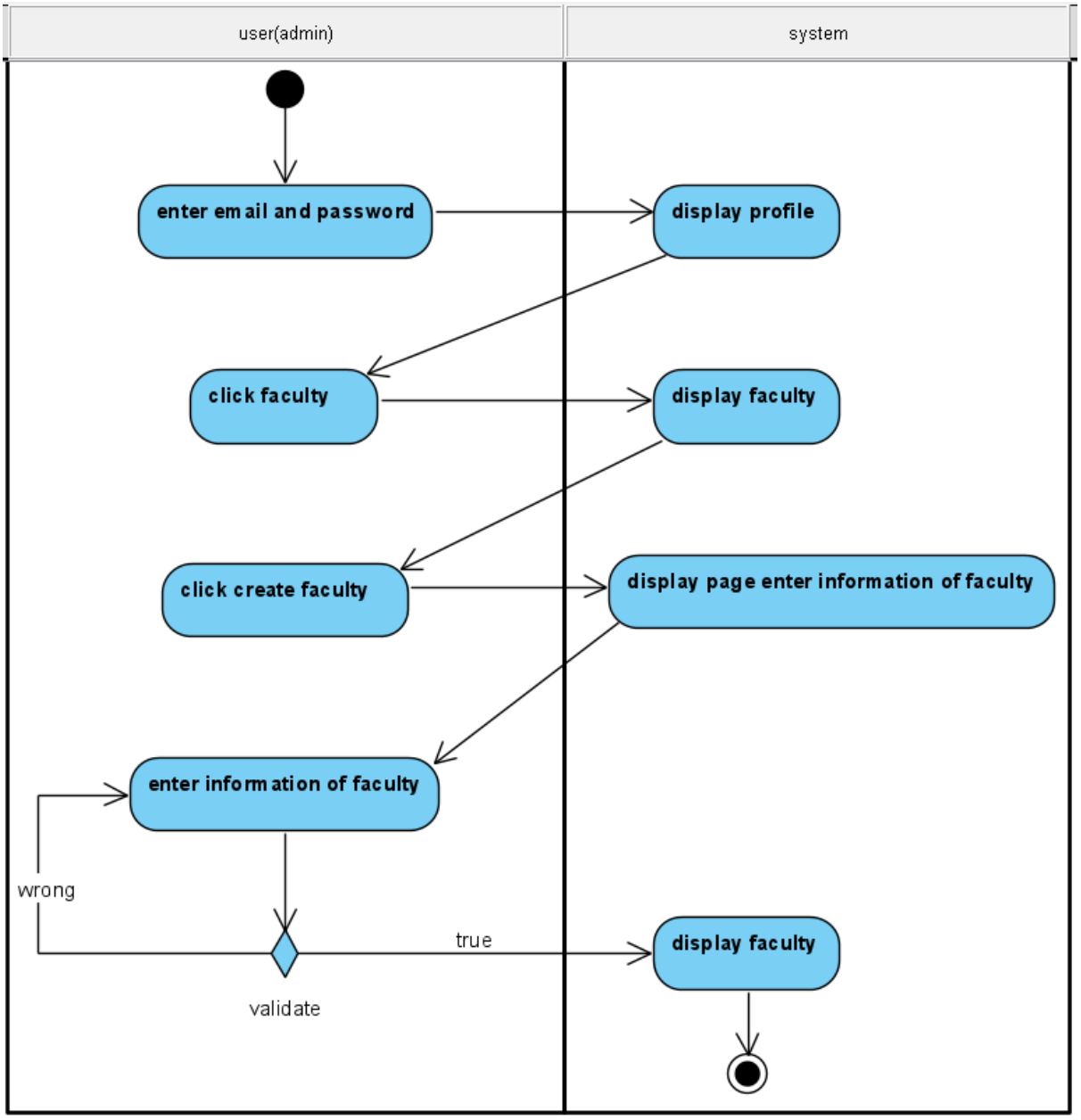


Figure 53 Create a faculty activity diagram

Activity creates faculty

Creating a new course is admin's right, so the only person interacting with the system here is admin. First, the admin logs in to his existing account, which has been created on the database, then the admin information screen appears, click on faculty, the screen displays the faculty page, click on create faculty, the screen displays the page to Enter information for faculty. Then it will divide two cases where an admin filled correctly and enough information, the system will now successfully create the class and send you to the faculty of all, and if wrong, the system will report an error and ask you to correct it it. It's correct.

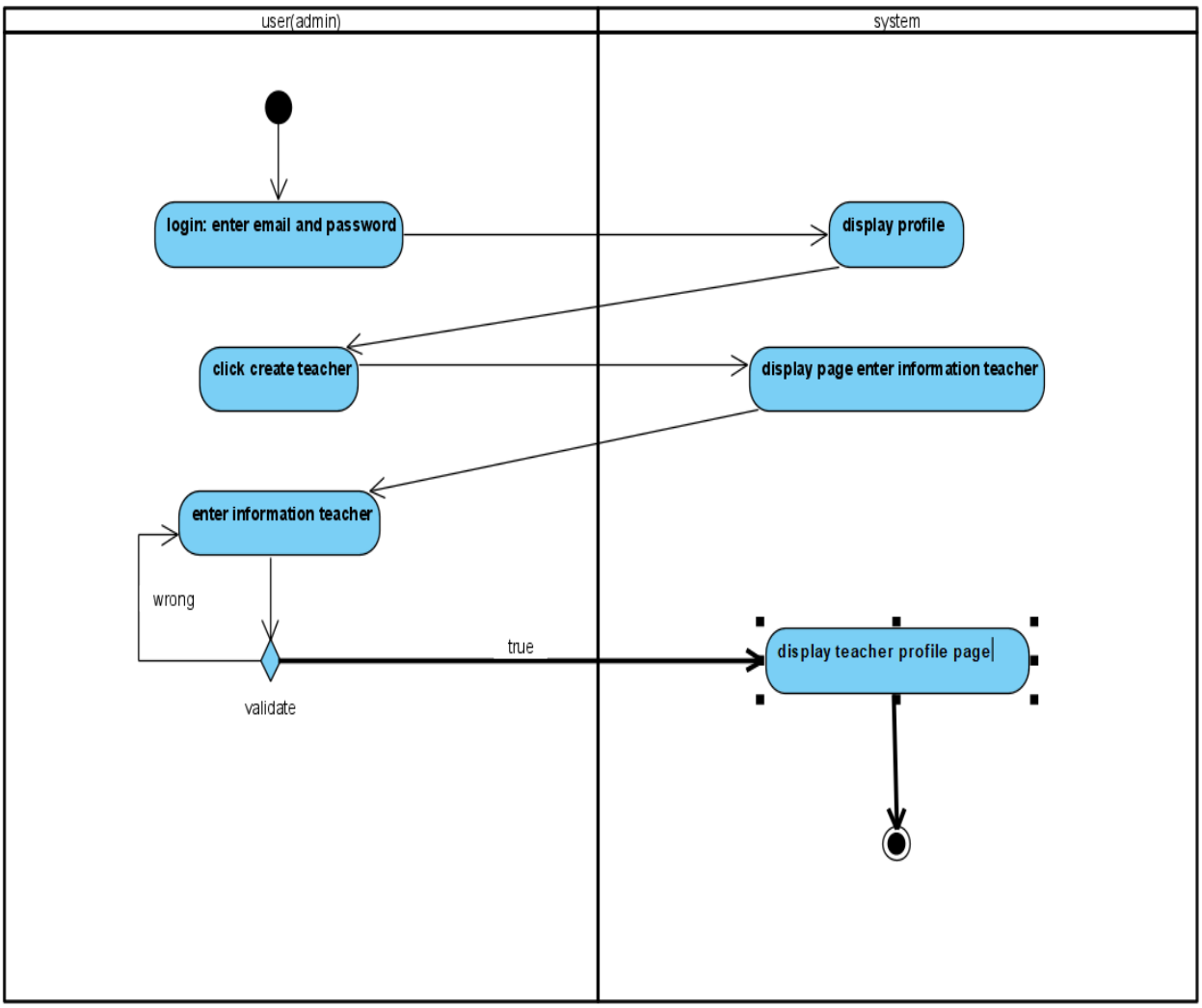


Figure 54 Create a teacher activity diagram

Activity creates teacher

Creating a teacher account is an admin function; first, the admin logs in to an available account exclusively for the admin on the system. The admin information screen appears. Click on create teacher; the screen will display the teacher information page. Here appear two problems; the first problem if entering the correct screen will take us to the account information page of all teachers. Also, if it is wrong, the system will notify you of a mistake and ask to correct the wrong input.

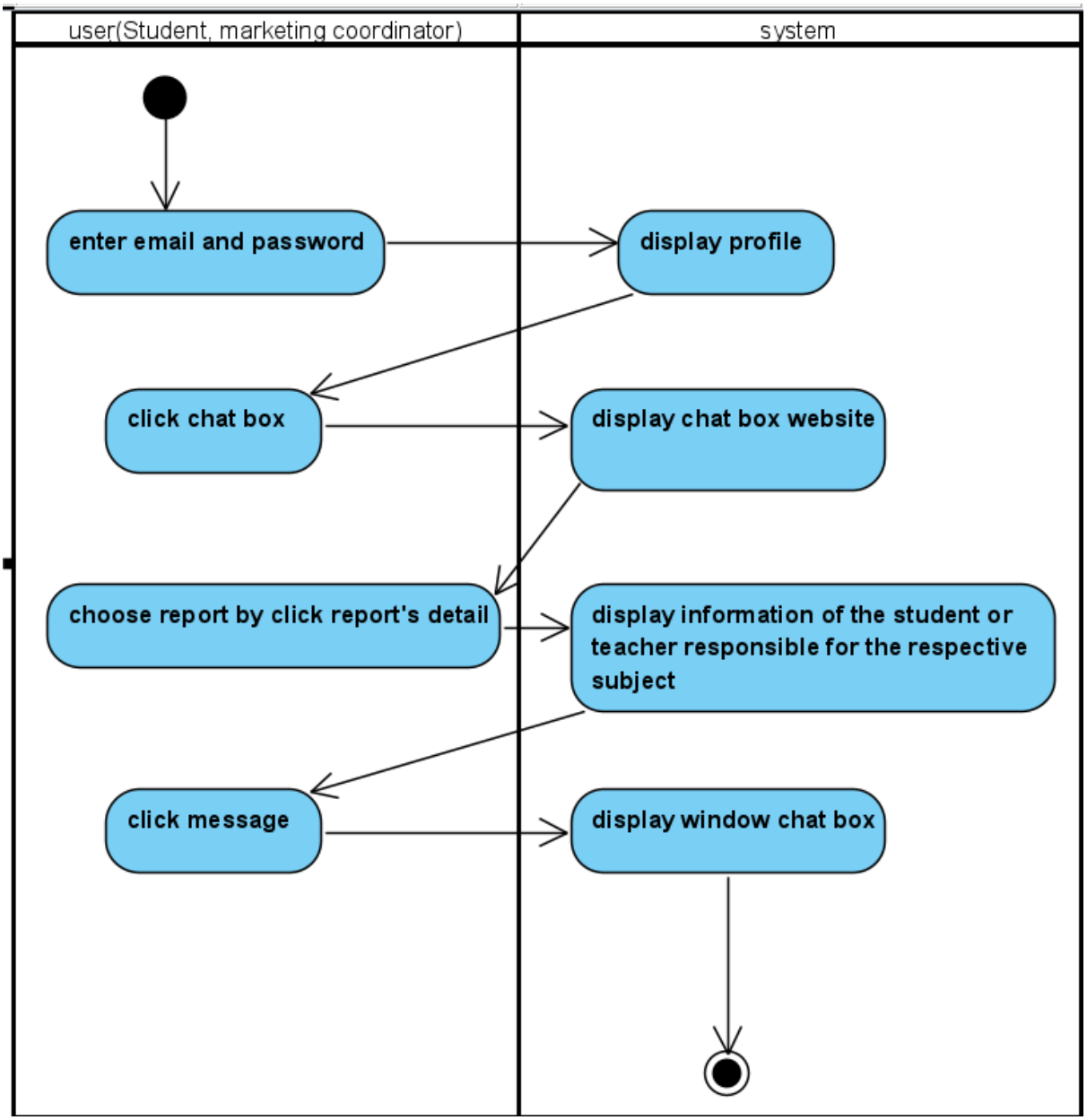


Figure 55 Chatbox activity diagram

Activity chatbox

For the chat box function, we created two roles: Marketing Coordinator and Student. The way it works is very simple; at the beginning, users logged in to the system and go straight to their profile page. After that, the user can click on the “chat box” button and will be lead to the “chat box display” website. On that page, users can choose to see which chat box they want to discuss between submitter and marker by click on “report’s detail.” By clicking on that button, the system will display information about the individual responsible for that submission. The last thing they need to do is click on “message,” and the system will show the chatbox for them to interact.

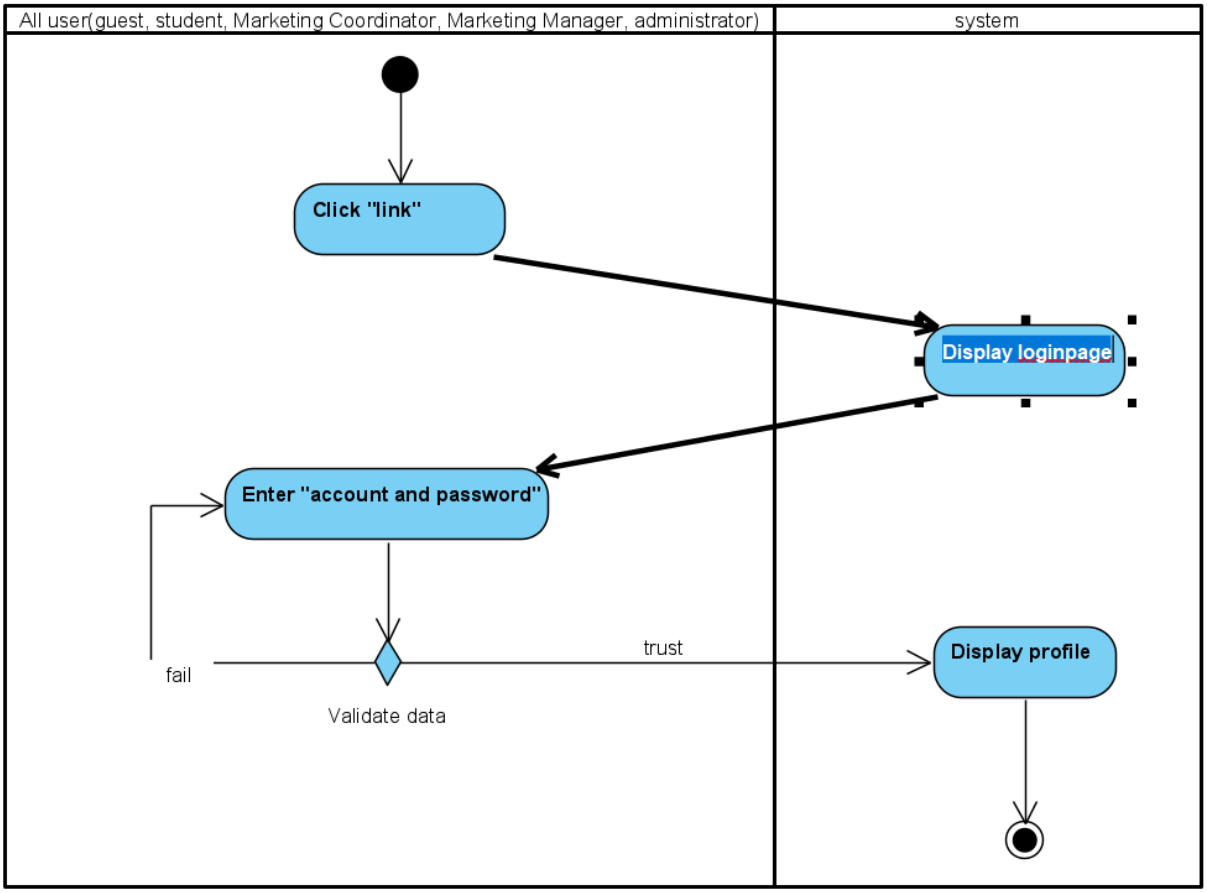


Figure 57 Login activity diagram

Activity login

The login function is created for all users. The purpose is very simple, to distinguish their role. First, they need to click on the link which leads to the website; the system will immediately show them the login page. After they put in the correct email and password of their account, the system will let them in the system and show up their profile.

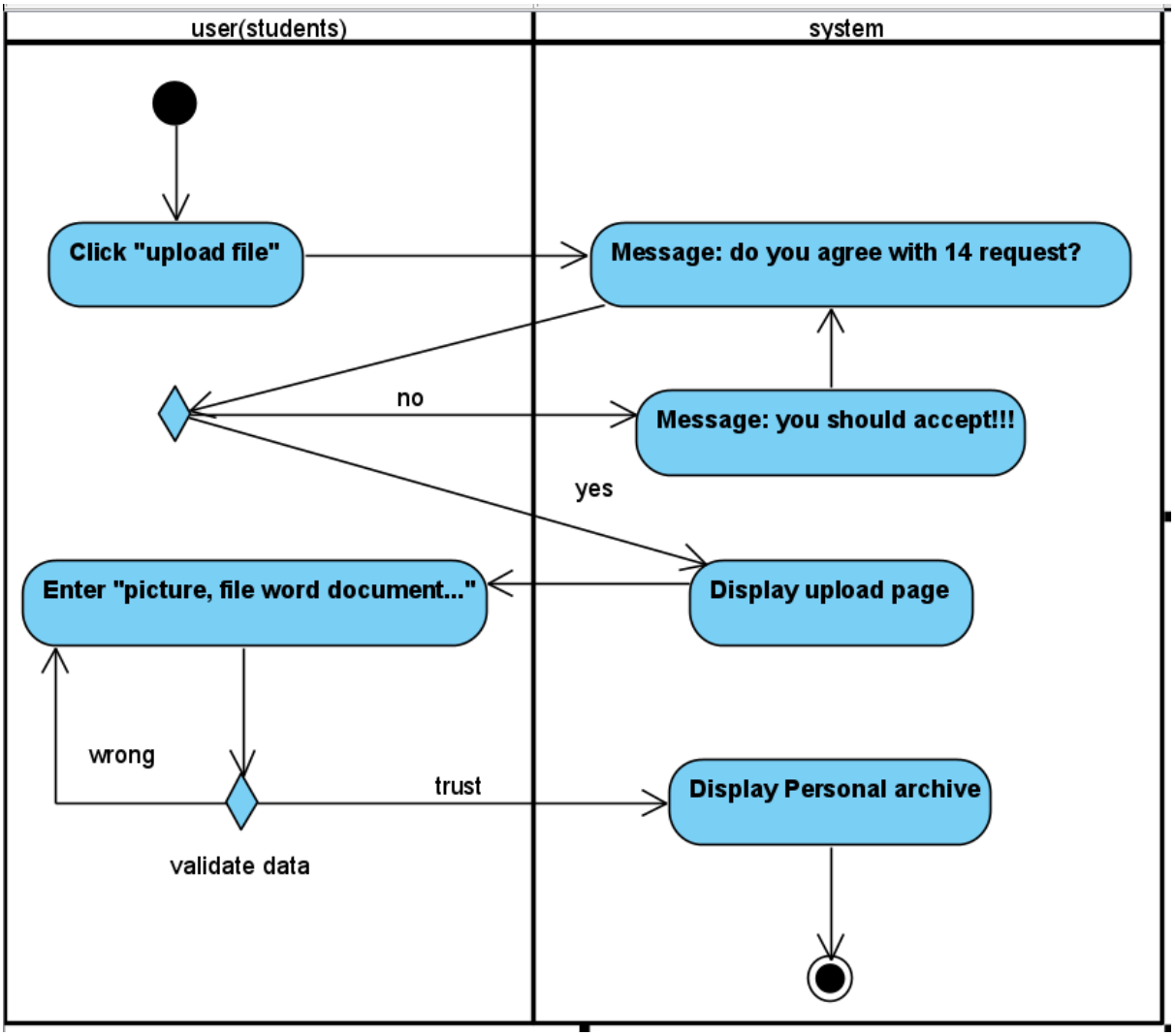


Figure 58 Upload activity diagram

Activity upload file

In this project, only students can upload files to submit their articles. Before they can upload their submission, they must agree to the term of the web; if they choose yes, the system will bring them to the submit page. To upload a file, the file must follow some rules of the web, and after the file bypasses them all, the file will be up on the web, and the system will show their personal archive.

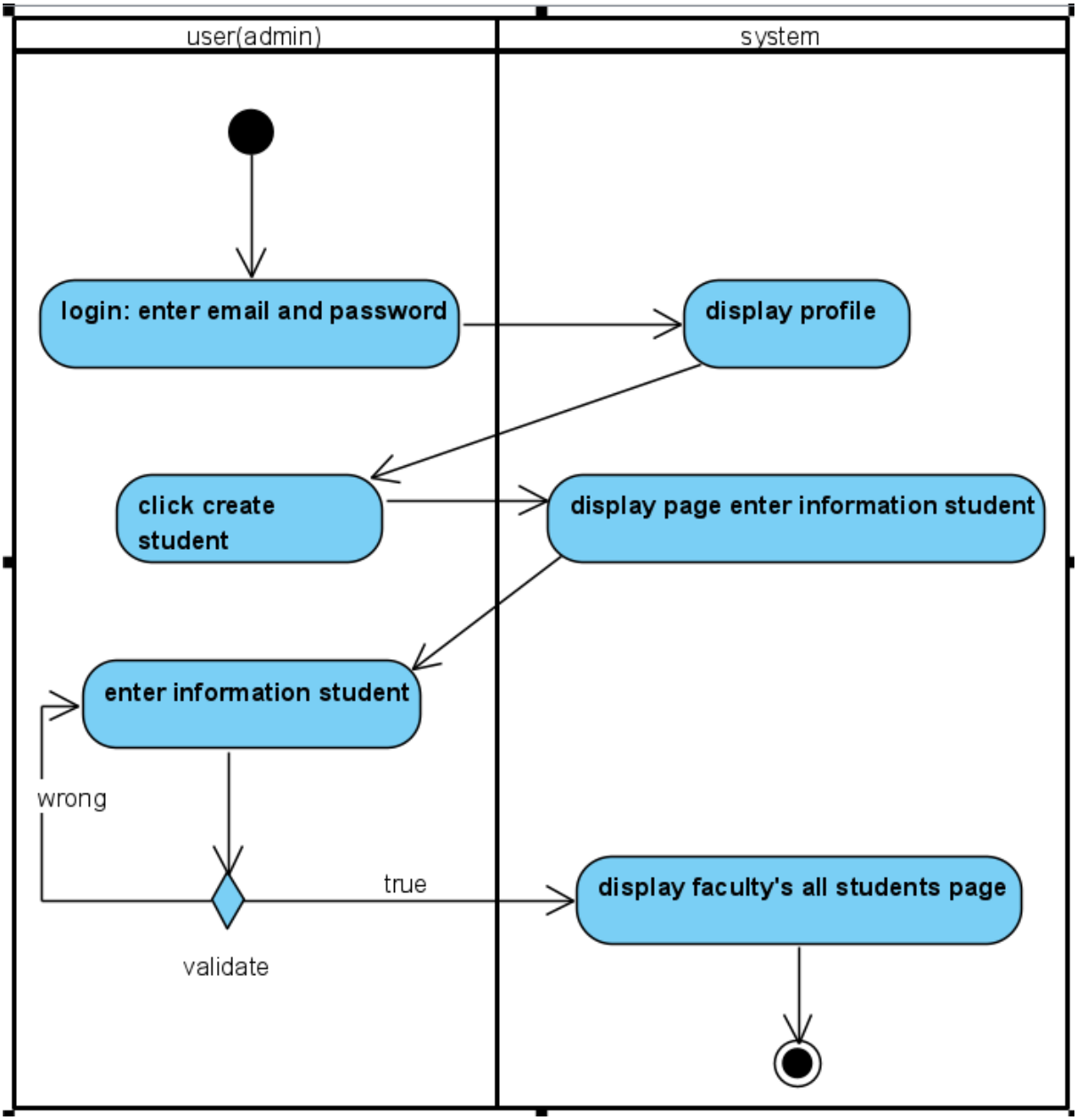


Figure 59 Add student activity diagram

Activity add the student to faculty

Except for the administrator, all other role accounts can be created by the admin, including the student. To do that, the user must be logged in by an admin account then click on “create student” After being brought to the create student page, they can enter student’s information and click on create, when the system finishes validating all the information standard, the account will be created and jump straight to the faculty’s all students page.

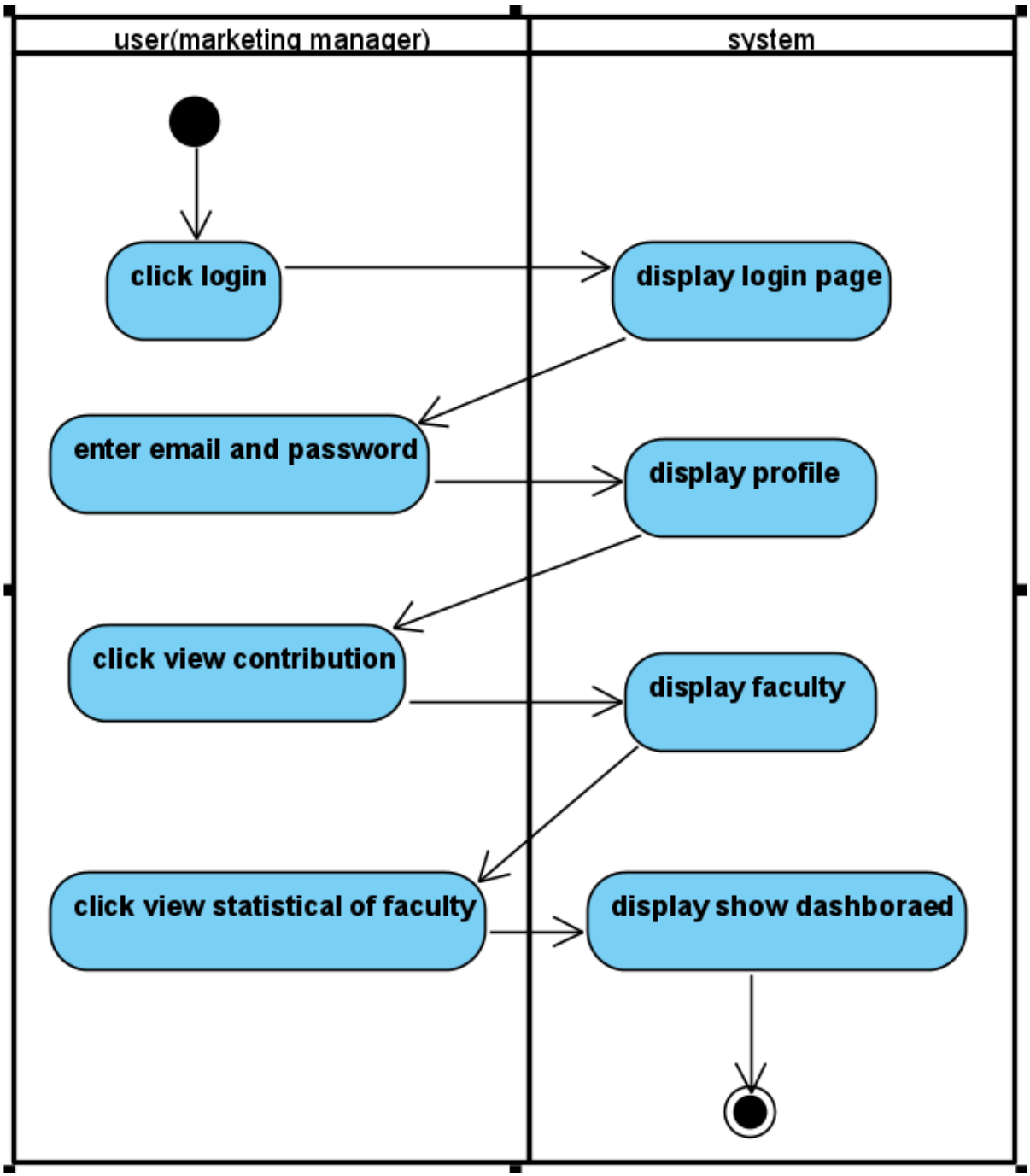


Figure 60 Dashboard activity diagram

Activity Dashboard

The dashboard is one of the functional requirements of the marketing manager role. After they logged in, they can click on view contribution, and the system will bring up all faculty for them to choose from. When they click on the button " view statistical of faculty," the system will display that faculty dashboard for the user.

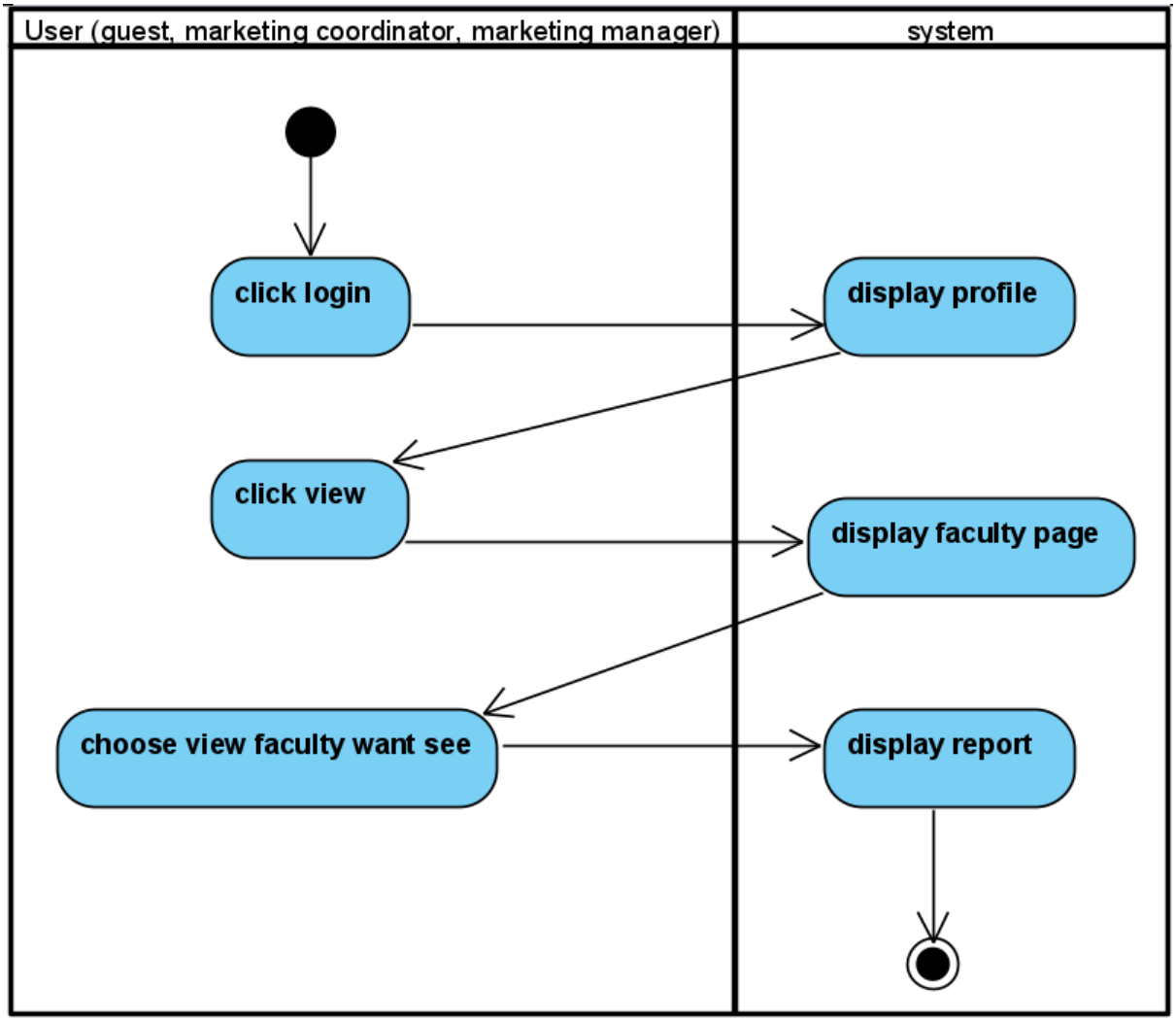


Figure 61 Download activity diagram

Activity download the selected reports

In order to download the selected report, the user must be a Marketing manager, marketing coordinator, or guest. For Guests and marketing coordinator, they must click on view article so the download button can show up. But with the marketing manager, you can click on the "download contribution" button. The system will bring you all the faculty. From that, the manager can click on view and selected an article to download without having to read it.

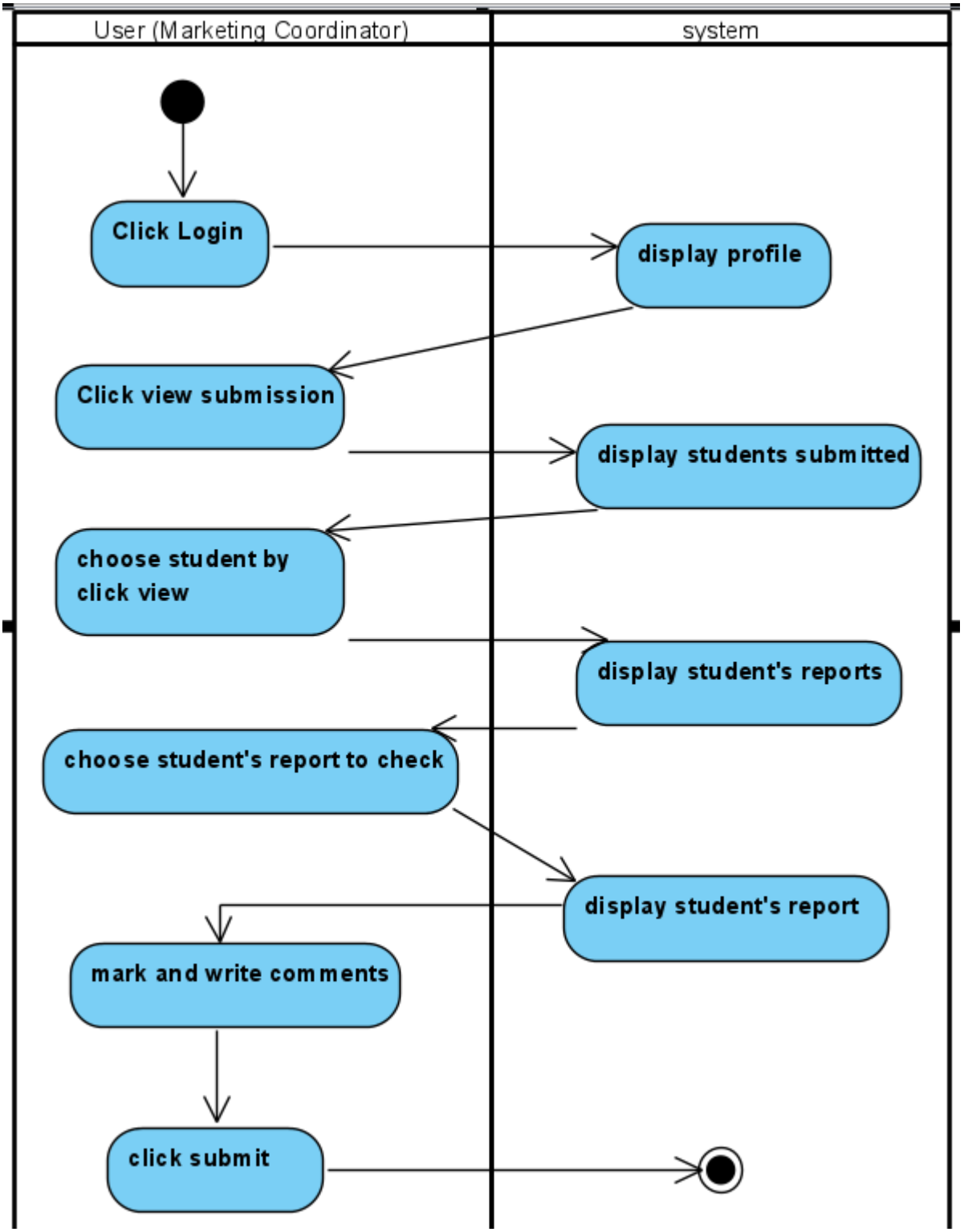


Figure 62 Comment activity diagram

Activity give comments

The comments function is only for the marketing coordinator to work on after they logged in and when they are marking a submission. After they click on student, every submission of that student in that faculty will show up; the coordinator can choose one of them to mark. The last activity is when they finish grading, write their comment on the comment section of the article and press submit.

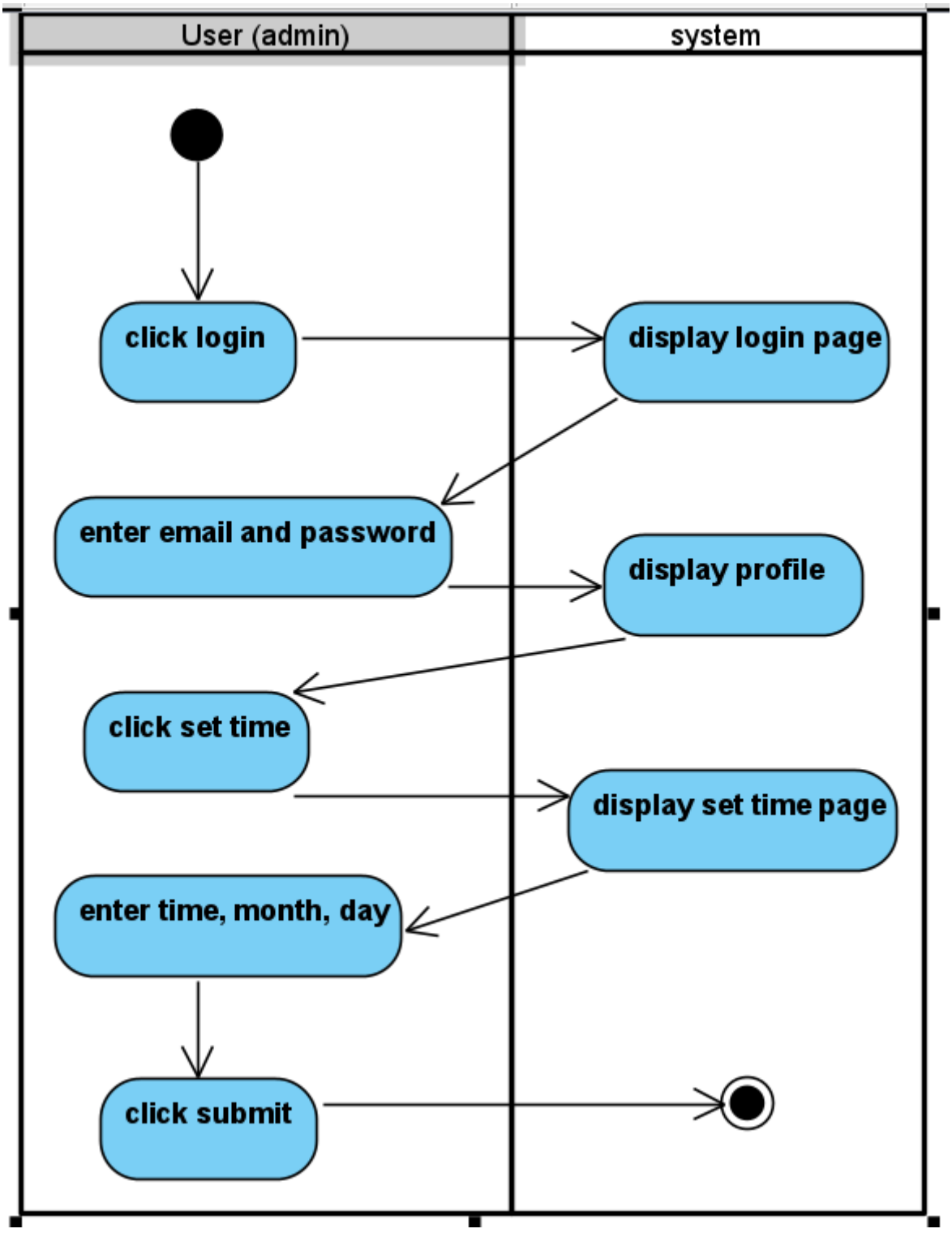


Figure 63 Set time activity diagram

Activity set time

Set time is the activity where only the admin can do. When they finished logging in, they can click onset time, and the system will display a Date and Timer for them to set. Entering time, month, day, and year, they can click on submit and set that time for all students.

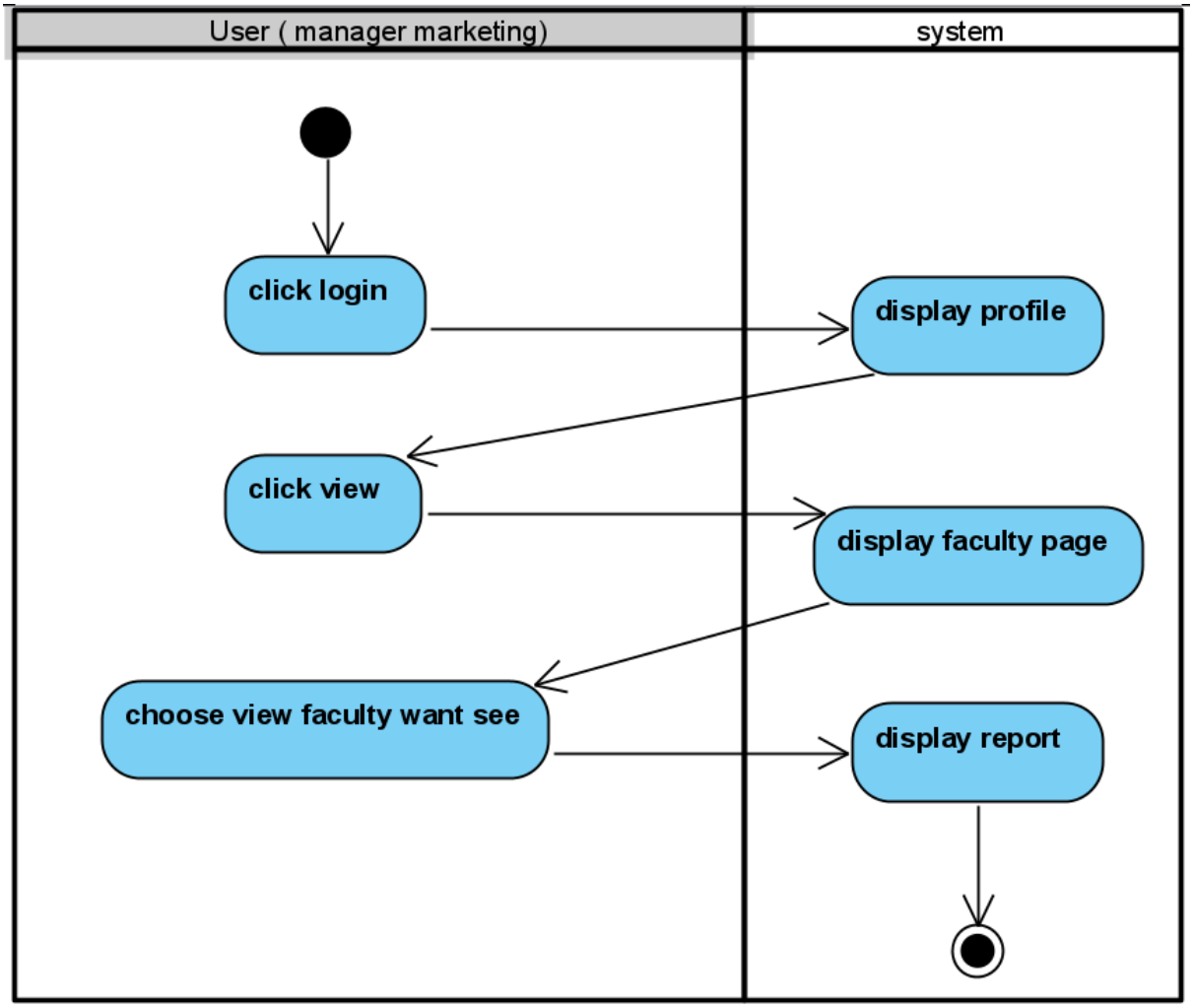


Figure 64 View article activity diagram

Activity view the selected reports

Viewing articles can be performed by the Marketing Manager, marketing coordinator, and guest. Just like every other activity, they must log in first. After clicking on view, the system will lead them to the page where it displays all faculty. The only thing they have to do left is to choose the faculty and click on the article they want to view.

3. Function

3.1 Data security

To secure the user's data, we used bcrypt and salt to encrypt information and authenticate. We use the hash () function to convert a value into another. The reason we use the hashing algorithm is that it cannot be decoded or converted back to its value.

The hash () function takes two parameter values, which are the data to be encoded and "salt." Salt is added to the hash head to protect the password in case of an attack. The value of "Salt" denotes the complexity of the hash function; the higher its value, the longer the password has to be hashed.

```
docreateAccount(req,res ){
    var password = req.body.password
    var role = req.body.Role
    const salt = bcrypt.genSaltSync(saltRounds);
    const hash = bcrypt.hashSync(password, salt);
    let facultyID = req.body.facultyID

    let newAccount = AccountModel({
        username: req.body.username,
        password :hash,
        email: req.body.email,
        facultyID: facultyID,
        role : role,
        phone:req.body.phone,
        birthday:req.body.birthday,
        address:req.body.address
    })
    newAccount.save(function(err,data){
```

Figure 47: Data security

Upon login, we have turned user information into a token including "_id." Server-side will look under _id for authentication, login, and authorization. As soon as the token is signed, we use cookies to store the token. Storing information in cookies for too long can present many security risks. That's why we have set the "maxAge" of cookies to last for one day to avoid hacker attacks

```
let token = jwt.sign({_id : req.user._id},'minh',{expiresIn :'1d'})
res.cookie("token",token,{maxAge: 24*60*60*1000, httpOnly: true });
```

Figure 48: Data security

To ensure cookies are only sent to URLs with HTTPS addresses, we have used "httpOnly: true" to protect against XSS attacks.

We use middleware to check the information the user enters upon login. By decompiling the token stored in the cookie to get the user's "_id" for comparison in the database. If the user's account already exists, the server will continue; if not, the server will disallow it and return to the login page.

```
let checkAuth = async (req,res,next)=>{
    try {
        var token = req.cookies.token || req.body.token
        let decodeAccount = jwt.verify(token,'minh')
        let user = await getUserById(decodeAccount._id)
        if(user){
            req.userLocal = user;
            next();
        }else{
            return res.status(400).json({
                message : "tk k ton tai",
                status: 400,
                error : true,
            })
        }
    } catch (error) {
        res.status(500).json({
            status: 500,
            error : true
        },
        res.redirect('/'))
```

Figure 49: Data security

To prevent a user of this role from being able to access another role's site, we have used an additional middleware to check information on the user cookies.

```
let checkAdmin = (req,res,next)=>{
    if (req.userLocal.role === "admin"){
        next()
    }else{
        res.status(400).json({
            message : "no permission",
            status: 400,
            error : true,
        })
    }
}
```

Figure 50: Data security

3.2 Data validation

In order to have actual data and avoid errors in the process of running the project, the system has validated some information of user accounts and information of departments.

Regarding email registration: Admin, when registering an account for a user, will have to enter the correct format for the mail. If not, it will report errors and not allow registration.

Notification: email must contain @ gmail.com, etc

```
<p>Email</p>
<input type="email" name = "username" required>
```

Figure 51: Validate email

About username: The system will check the user name by checking the characters in the name; if the admin enters a username with special characters or numbers, the system will send an error and prompt the user to enter it correctly. Form of the name.

Notification: The username shouldn't contain a number. e.g., Smith

```
<p>Name</p>
<input type="text" name = "name" pattern="[a-zA-
Z]+" title="Username shouldn't contain number. e.g. john" required >
```

Figure 52: Validate the name

The date of birth of the account: The system will display a calendar for the admin to choose when registering a report for the user. Using the calendar is more accurate and avoids minor errors like stamp, date format, etc

```
<p>Date of birth</p>
<input type="date" name = "birthday" required>
```

Figure 53: Validate birthday

About phone number information: Phone numbers are formatted as starting with the area code +84 (Vietnam) and having at least eight numbers. If the user enters text or special characters, the system will issue a format error and force the user to re-enter.

Notification: The phone number must not contain letters or any other characters. Phone numbers must be between 8-10 numbers.

```
<p>Phone</p>
<input type= "text" name = "Phone"  pattern="(\+84|0)\d{9,10}" minlength="8" m
axlength="10" title="Phone number must not contain letters or any other charac
ters.Phone number must be between 8-10 numbers." required>
```

Figure 54: Validate phone

About the faculty name: When the admin creates the faculty name, the faculty name will also have the same format as the username. If the user enters numbers or special characters in the faculty name, the system will also report an error.

Notification: Faculty shouldn't contain a number. e.g., Electronic technology

```
<p>Faculty</p>
<input type="text" name = "faculty" pattern="[a-zA-
Z]+" title="Faculty shouldn't contain number. e.g. Electronic technology" requ
ired >
```

Figure 55: Validate Faculty name

About password: In order for a password to be secure, a password needs to have at least eight characters, and next to it does not contain any special characters. Having special characters can cause difficulty logging in.

Notification:
Password needs to be at least eight characters without special characters (/,?, Etc)

```
<p>password</p>
<input type="password" name="password"title="Password needs to be at least 8 c
haracters without special characters (/,?, Etc)" minlength="8" pattern="[a-zA-
Z0-9]+"  required>
```

Figure 56: Validate password

3.3 The main function in requirement

In the event that there is data in the database, the server will return the user's token in the cookie and examine that person's role and return it to the homepage with each different role. Since the token can be decoded, to avoid the disclosure of important information, we only store "_id" in the token. When a signed token has one day, it should not be too long to avoid being hacked.

```
let loginController = function(req,res){
    bcrypt.compare(req.body.password, req.user.password, function(err,result){
        if(err){
            return res.status(500).json({
                message : "loi sever",
                status: 500,
                error : true
            })
        }
        if(result){
            let token = jwt.sign({_id : req.user._id},'minh',{expiresIn :'1d'})
            res.cookie("token",token,{maxAge: 24*60*60*10000});
            let user = req.user
            res.cookie('email',user.email, { maxAge: 90000000, httpOnly: true });
            res.cookie('id',user._id, { maxAge: 90000000, httpOnly: true });
            res.cookie('facultyID',user.facultyID, { maxAge: 90000000, httpOnly: true });
            if(user.role === "admin"){
                res.redirect("./indexAdmin")
            }
            if(user.role === "student"){
                res.redirect("./indexStudent")
            }
            if(user.role === "coordinator"){
                res.redirect("./indexCoordinator")
            }
            if(user.role === "guest"){
                res.redirect("./indexGuest")
            }
            if(user.role === "manager"){
                res.redirect("./indexManager")
            }
            }else{
                var message= "Username or password is invalid"
                res.render("login",{message:message})
            }
        }
    )
}
```

Figure 57: Authorize code

With the following logins, the server-side will decode the token again to verify that it is them. To be able to decode the token, it is necessary to have the correct secret code generated in the server. After decoding the token, the server will check the role and direct the users to the home page, depending on their role

```
let checkAuth = async (req,res,next)=>{
    try {
        var token = req.cookies.token || req.body.token
        let decodeAccount = jwt.verify(token,'minh')
        let user = await getUserById(decodeAccount._id)
        if(user){
            req.userLocal = user;
            next();
        }
    } catch (error) {
        res.status(500).json({
            message : "error sever",
            status: 500,
            error : true
        },
        res.redirect('/'))
    }
}
```

Figure 58: check Auth code

```
router.get('/indexAdmin',checkAuth ,checkAdmin, indexAdmin)
router.get('/indexTeacher',checkAuth ,checkTeacher, indexTeacher)
router.get('/indexStudent',checkAuth ,checkStudent, indexStudent)
router.get('/indexGuest',checkAuth,checkGuest,indexGuest)
router.get('/indexManager',checkAuth,checkManager,indexManager)
```

Figure 59: Home account

Upload file and images

```
var storage = multer.diskStorage({
    destination:function(req,file,cb){
        cb(null,'./public/uploads')
    },
    filename:function(req,file,cb){
        var namefile = file.originalname
        cb(null,file.originalname)
    }
})
var upload = multer({storage:storage})
```

Figure 60: Upload file and images

First, the system will set a location to store the files that students upload to the system and their names when saved at that location. After that, the uploading of the student's file will have to go through the following tests with the condition: can upload an only one-word file and one image file (optional).

The system will check the number of files uploaded by the student. In case of uploading a file to the system, the system will force it to be a word file. After checking, if it is not a word file, the system will report an error and send a notice. If the test passes, the system will convert the file to PPF and start setting the file name and location (both docx and PDF) and save that information to the database.

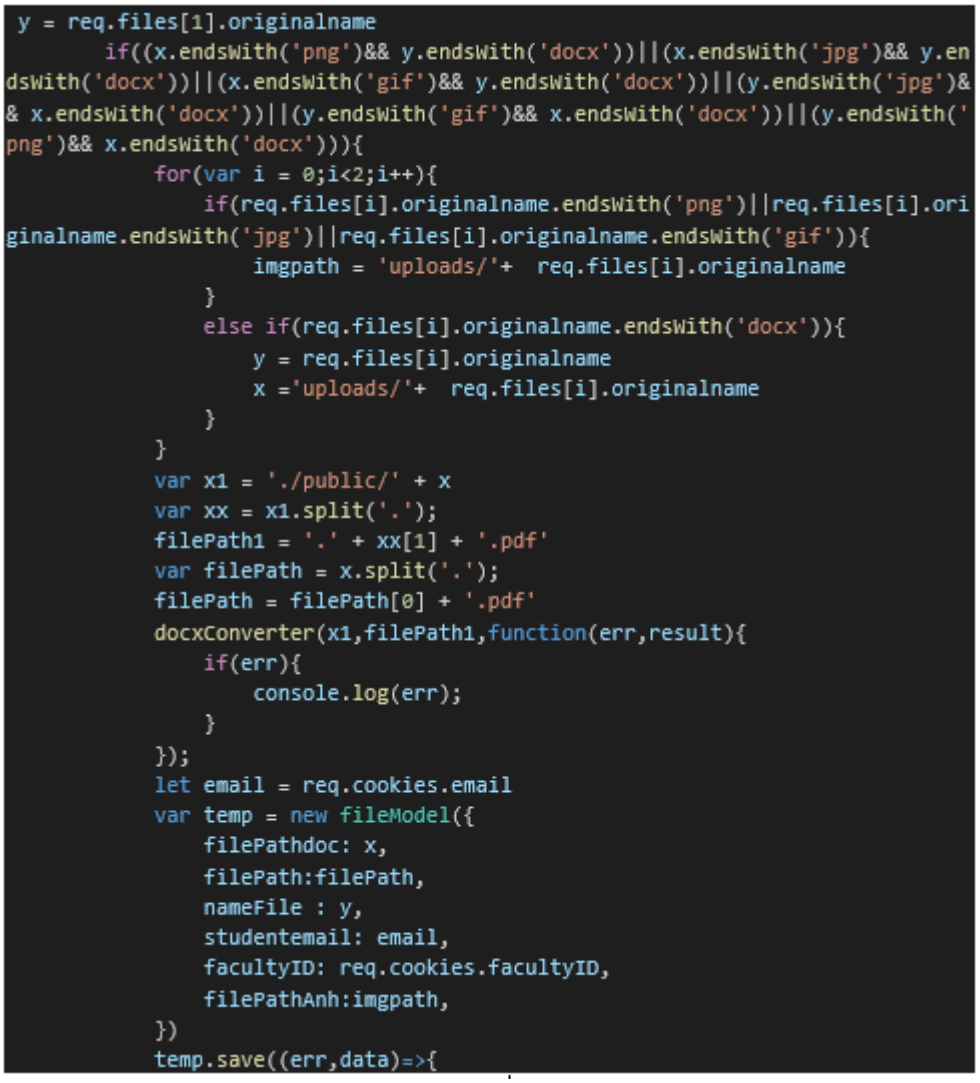
- Case upload one file: The only docx is acceptable

```
fileRouter.post('/upload',upload.array('filePath',2),(req,res)=>{
    x = req.files[0].originalname
    if(req.files.length == 1){
        if(x.endsWith('docx')){
            xdoc ='uploads/'+  req.files[0].originalname
            var x1 = './public/' + xdoc
            var xx = x1.split('.');
            filePath1 = '.' + xx[1] + '.pdf'
            var filePath = xdoc.split('.');
            filePath = filePath[0] + '.pdf'
            docxConverter(x1,filePath1,function(err,result){
                if(err){
                  console.log(err);
                }
            });
            let email = req.cookies.email
            var temp = new fileModel({
                filePathdoc: xdoc,
                filePath:filePath,
                nameFile : x,
                studentemail: email,
                facultyID: req.cookies.facultyID,
            })
            temp.save((err,data)=>{
```

Figure 61:Case upload one file code

- Case upload two files: One Docx and one image are acceptable

First, the system will proceed to check the files. If a student uploads files other than a word (Docx), image (png, jpg, gif) files, the system will send an error message to the student with an uploadable file type. After the list is checked, the system will conduct to check whether the two files are word files or image files. If two files have the same properties, the system will send a notification. Otherwise, the system will convert the file to PPF and start setting the name, file location (both docx and PDF), image address and save that information in the database.



```
 y = req.files[1].originalname
        if((x.endsWith('png')&& y.endsWith('docx'))||(x.endsWith('jpg')&& y.en
dsWith('docx'))||(x.endsWith('gif')&& y.endsWith('docx'))||(y.endsWith('jpg')&
& x.endsWith('docx'))||(y.endsWith('gif')&& x.endsWith('docx'))||(y.endsWith('
png')&& x.endsWith('docx'))){
            for(var i = 0;i<2;i++){
                if(req.files[i].originalname.endsWith('png')||req.files[i].ori
ginalname.endsWith('jpg')||req.files[i].originalname.endsWith('gif')){
                    imgpath = 'uploads/'+  req.files[i].originalname
                }
                else if(req.files[i].originalname.endsWith('docx')){
                    y = req.files[i].originalname
                    x ='uploads/'+  req.files[i].originalname
                }
            }
            var x1 = './public/' + x
            var xx = x1.split('.');
            filePath1 = '.' + xx[1] + '.pdf'
            var filePath = x.split('.');
            filePath = filePath[0] + '.pdf'
            docxConverter(x1,filePath1,function(err,result){
                if(err){
                    console.log(err);
                }
            });
            let email = req.cookies.email
            var temp = new fileModel({
                filePathdoc: x,
                filePath:filePath,
                nameFile : y,
                studentemail: email,
                facultyID: req.cookies.facultyID,
                filePathAnh:imgpath,
            })
            temp.save((err,data)=>{
```

Figure 62:Case upload two file code

View the selected reports.

```
readcontribution(req,res){
        let id = req.params.id
        fileModel.find({_id:id},(err,data)=>{
            if(err){
                console.log(err)
            }
            else if(data.length>0){
                res.render('guest/danhgia.ejs',{data:data})
            }
            else{
                res.render('guest/danhgia.ejs',{data:data})
            }
        })
}
```

Figure 63: View the selected report code.

From the front-end, the user will click and read the articles they are interested in; the back-end will get the information's id of that article and get the address of that article in the database through the id and display the post through the <iframe> tag on the front-end.

Receive email notifications after students posting articles to the system.

```
// set up mail sever
var transporter =  nodemailer.createTransport({
    host: 'smtp.gmail.com',
    port: 465,
    secure: true,
    auth: {
        user: 'fptedunotification@gmail.com',
        pass: 'son@1234'
    },
    tls: {
        rejectUnauthorized: false
    }
});
```

Figure 64: Set mail sever

First, we need to install mail for the server through the nodemailer library. Besides, the server's mail account also needs to set up some settings on google, such as: allowing low-security applications to access. This will allow the server to access this mail without being intercepted by Google's security. In addition, you need to enable IMAP to retrieve email messages from the mail server over a TCP / IP connection.

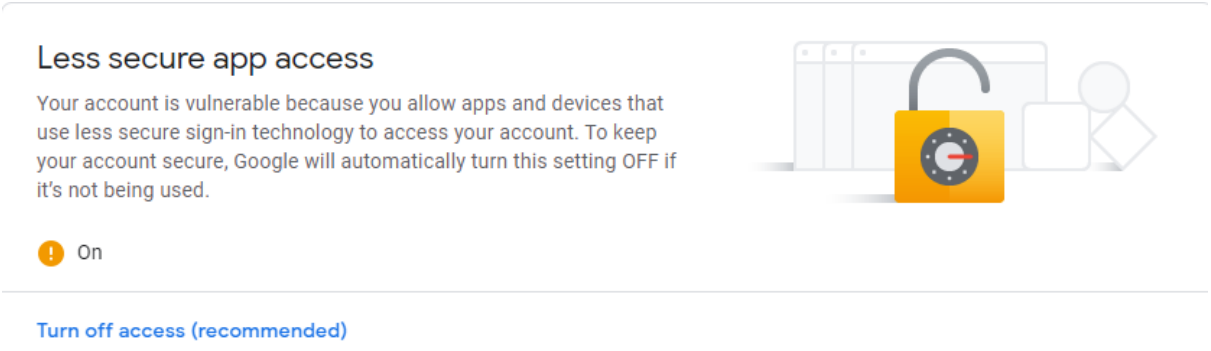


Figure 65: Turn on less secure app access

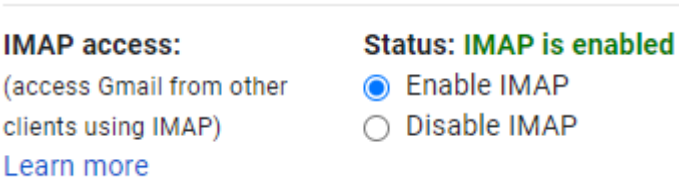


Figure 66:Enable IMAP

• Send email to the student

```
let email = req.cookies.email
var content = 'You have just uploaded an article to the system. Name: ' + x;
var mainOptions = {
    from: 'fptedunotification@gmail.com',
    to: email,
    subject: 'Notification',
    text: content
}
transporter.sendMail(mainOptions, function(err, info){
    if (err) {
        console.log(err);
    }
```

Figure 67: Send email to student code

To email students every time they post their file, the system will need to write the content of the mail. Then proceed to set up your mailing address and subject. After composing the mail, it will proceed to send the mail through the transporter.SendMail function. This operation will happen after saving the file information to the database.

- Send email to Marketing coordinator

```
let facultyID = req.cookies.facultyID
AccountModel.findOne({
    role: "coordinator",
    facultyID: facultyID
},function(err, result){
    var content = email + 'just uploaded an article to the system. Name: ' + x;
    var mainOptions2 = {
    from: 'fptedunotification@gmail.com',
    to: result.email,
    subject: 'new post',
    text: content
}
transporter.sendMail(mainOptions2, function(err, info){
    if (err) {
        console.log(err);
    }
});
```

Figure 68: Send email to Marketing coordinator code

After students upload a post to the system, the Marketing coordinator will also receive a notification about that post. Based on the student's scientific code (facultyID) and role as Marketing Coordinator, the system will look in the database and retrieve the mail of that Marketing coordinator. After obtaining the Marketing coordinator's mail, compose the letter and send it to the Marketing coordinator. This operation will happen after saving the file information to the database.

Make a comment

```
danhgiabaibao(req,res){
    let id = req.params.id
    fileModel.find({_id:id},(err,data)=>{
    if(err){
        console.log(err)
    }
    else if(data.length>0){
        res.render('faculity/danhgia.ejs',{data:data})
    }
    else{
        res.render('faculity/danhgia.ejs',{data:data})
    }
    })
}
```

Figure 69: Make a comment code

First, the system will get the id of the selected article, and from the id will get the information of that article and send it to the interface page for evaluation. All

information (status, comment, etc.) of that file will also be printed out on the interface page.

```
dodanhgiabaibao(req,res){
    let id = req.params.id
    let status = req.body.status
    let comment = req.body.comment
    fileModel.findById({_id :id},function(err,data){
        let studentemail = data.studentemail
        console.log(studentemail)
    fileModel.updateOne(
        { _id: id },   // Query parameter
        {                       // Replacement document
            status: status,
            comment: comment
        })
        .then(()=>{
            res.redirect('/faculity/allDocument/' + studentemail)
        })
    })
}
```

Figure 70: Make a comment code

After reading and giving comments and post status (Pass or Fail), the system will get the id and comment information, post status, and save it to the database via the post id.

Download zip:

```
var file_system = require('fs');
var archiver = require('archiver');
fileRouter.post('/abc',(req,res)=>{
    var facultyID = "public/"+  req.body.facultyID + ".zip"
    var name = req.body.facultyID + ".zip"
    console.log("sssssssssssss:",name)
    var output = file_system.createWriteStream(facultyID);
    var archive = archiver('zip');
    var a = req.body.hobby
    output.on('close', function () {

    });
    archive.on('error', function(err){
        throw err;
    });
    archive.pipe(output);
        for(var n = 1; n <a.length; n++ ){
            file = "public/" +  a[n]
            console.log("file name là:", file)
            archive.append(file_system.createReadStream(file), { name: file })
        }
    archive.finalize();
        res.redirect('./abc1/'+ name)
})

fileRouter.get('/abc1/:name',(req,res)=>{
            var name = req.params.name
            var x = __dirname.replace('routes','public/') + name
            res.download(x)
        }
)
```

Figure 71: Download article as zip code

The system will use the archive library to compress Docx files to zip. First, the system will create a zip file with the name facultyID and save it in the public folder. After completing the zip file, the system will get the article's address through the check box on the front-end and proceed to insert them into the zip file through the archive. Append () function and then proceed to finish the compression process. Then the system will switch to the new router and download the file.

Chatbox

The chat function will be in the details of the marketing coordinator or student, depending on the user's role. If you are a student, there is only one coordinator, but if it is the coordinator, you can choose from a list of students.

If the coordinator and the student do not have any chatbox before, the interface will have a button to add chat. When you click this button, the server-side will create a collection of "chat," including the receiver and the sender. All conversation content will be saved in this collection. After adding chat, the next time, the user does not need to add a chatbox anymore but can click the message button.

Server-side will query in the database according to sender and receiver email. New messages recorded by the user will be pushed into the collection

```
socket.on("new_user_message", (data) => {
    socket.join(`${data.cookiesemail}and${data.user}`);
    socket.join(`${data.user}and${data.cookiesemail}`);
    var query1 = {
        userSend: data.cookiesemail,
        userReceive: data.user,
    }
    var query2 = {
        userSend: data.user,
        userReceive: data.cookiesemail,
    }
```

Figure 72: Chatbox code

```
    let dbo = db.db("test");
    dbo.collection("chats").updateOne(query1, {
        "$push": {
            message: {
                check: 1,
                Mes: `${data.mes}`,
                index_time:new Date().valueOf(),
                date: `${new Date().getHours()}:${new Date().getMinutes()}-${new Date().getDate()}/

            }
        }
    });
    dbo.collection("chats").updateOne(query2, {
        "$push": {
            message: {
                check: 0,
                Mes: `${data.mes}`,
                index_time:new Date().valueOf(),
                date: `${new Date().getHours()}:${new Date().getMinutes()}-${new Date().getDate()}/

            }
        }
    });
```

Figure 73: Chatbox code

VI. Testing:

In order to make sure the project is working properly, we need to make sure every function of it works as it should be. For that purpose, just like almost every other project, we decided to create a session that focuses on testing the project. In this session, we will create a plan to test the service, and after that, we will show the test result in this report.

1. Test plan

1.1 Function requirement

- Front-end: login, Download submission by ZIP format, Chatbox, Statistic, report, and submission, give and read a comment on submission, accept or reject the submission, see submit deadline, and edit deadline.
- Back-end: Statistic management, notification email auto sending, exceptional report managing, comment management, chatbox generator, account creator, setting deadline and maintaining system, password encode generator.

1.2 Targeted users

Our system is created for a university magazine, so our user is 99 percent university students and school's staffs.

1.3 Purpose of the test

Determine the results of how the system work, make sure the system works perfectly as it should be, and help on the developing phase because the test starts right after a function is finished coding.

- Environment requirement
  - Server test: Heroku.
  - Device: laptop
  - Browser: Chrome, CocCoc, Microsoft Edge, Mozilla Firefox.
  - Test plan: Microsoft word
- Test case: Microsoft word
  - Image capture tool: Snipping tool, Window capture
  - Set up test report: Microsoft Word
  - Tool Support Test Interface: Chrome

1.4 Testing implementation plan (Test strategy)

- Strategy for building test case

- The test case must follow the interface of manuscript functions, and all functions of Front-end and Back-end must be included in the test case.
- Defining risk by risks then solve the problem that may occur
- Testing function as it finishes before jump into the next function

➢ Testing tool

- Integration test: Testing using Google Chrome and Microsoft Edge on PC and laptop, Safari for phone
- System test: Test on PC and mobile phone.

➢ Project testing recourses

- Nguyen Trung Doan Khang: Responsible for Implementation of project test
- Bui Chi Huong: Tester

➢ Test order (Test sprint)

❖ **System test: 27/03/2021**

- Test Role Admin in sever: 27/03/2021
- Test Role Marketing Manager in server: 27/03/2021
- Test Role Marketing Coordinator in sever: 28/03/2021
- Test Role student in sever: 28/03/2021
- Test Role Guest in sever: 29/03/2021
- Test chat box in sever: 29/03/2021

❖ **Test login/logout function: 25/03/2021**

- Login: 25/03/2021
- Logout: 25/03/2021

❖ **Test add/update/delete faculty and account: 02/03/2021**

❖ **Test upload file: 08/03/2021**

- Upload file by docx format: 07/03/2021
- Upload file by images: 08/03/2021

❖ **Marketing Coordinator function testing: 11/03/2021**

- Testing notification function: 10/03/2021
- Test function: Marketing Coordinator reviews the student's articles: 11/03/2021

❖ **Testing guest function: 13/03/2021**

❖ **Testing Marketing manager function: 20/03/2021**

- Test manager read the selected articles: 14/03/2021
- Test download selected report: 17/03/2021

- Test statistic data and displaying: 20/03/2021
- Test set time: 20/03/2021

1.5 Potential risks

Risk	Solution
inexperienced tester	This is one of the most potential risks; the solution is easy, we can hire a well-recommended tester or let them train our staff for a greater course.
The infrastructure may not fulfill the requirement to test	Continuously checking on the machines does the guarantee work on time.
Lacking interactions between tester, developer, and users	This may lead the project to a dead-end because of misunderstood; we should create a daily or weekly meeting to get more information from each other
Everyone may get difficult in the job because of time strict	We can get more people to directly or indirectly help us with the job. Another solution is managing everyone work every day so we can make sure that they finish the job on time or sooner than we need
A test case may not include every error and may occur when we did not expect it	The only thing we can do is test every case that we can think of. If they're an error, we may fix that in the future

2. Test case

a. Administrator

Test case	Input	Expected result	Actual result	Test condition	Test log
1 Create account for Marketing Manager	Create a marketing manager account with email: manager1@gmail.com And password: 123	Create successfully	Create successfully and go to the "All manager" page	Pass	17:16-02/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
1.2 Create an account for the Marketing Coordinator	Create a marketing Coordinator account with email: teacher1@gmail.com And password: 123	Create successfully	Create successfully and go to the "All Teacher" page	Pass	17:20-02/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
1.3 Create an account for student	Create a Student account with email: student1@gmail.com And password: 123	Create successfully	Create successfully and go to the "All Student" page	Pass	17:33-02/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
1.4 Create an account for guest	Create a Guest account with email: guest1@gmail.com And password: 123	Create successfully	Create successfully and go to the "All Guest" page	Pass	17:36-02/03/2021 https://localhost:3000/ Test exercutor: Khang

1.5 Create an account that contains a number of special characters	Create a Guest account with user name: guest1+	Create fail	Create successfully and go to the "All Guest" page	Fail	17:37-02/03/2021 User name can still contain a number and special character Test on URL: https://localhost:3000/ Assigned for Son to fix this problem. Test exercutor: Khang
1.6 Create an account	Create a Marketing Manager account with email: Manager2	Create fail	Create successfully and go to the "All manager" page	Fail	17:37-02/03/2021 Email can be created without @gmail.com Test on URL: https://localhost:3000/ Assigned for Son to fix this problem.

					Test exercutor: Khang
1.7 Create an account	Create a Guest account with password: +++	Create fail	Create successfully and go to the "All guest" page	Fail	17:37-02/03/2021 Password can contain special characters. Test on URL: https://localhost:3000/ Assigned for Son to fix this problem. Test exercutor: Khang
2 Update account for Marketing Manager	Click on update and update information	Update successfully	Update successful and go to the "All manager" page	Pass	17:37-02/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
2.1 Update account for Marketing Coordinator	Click on update and update information	Update successfully	Update successful and go to the "All Teacher" page	Pass	17:40-02/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
2.2 Update account for student	Click on update and update information	Update successfully	Update successful and go to the "All Student" page	Pass	17:42-02/03/2021 Test on URL: https://localhost:3000/

					Test exercutor: Khang
2.3 Update account for guest	Click on update and update information	Update successfully	Update successful and go to the “All Guest” page	Pass	17:45-02/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
3.1 Delete account for Marketing Manager	Click on Delete	Delete successfully	Delete successful and go to the “All manager” page	Pass	17:47-02/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
3.2 Delete account for Marketing Coordinator	Click on Delete	Delete successfully	Delete successful and go to the “All Teacher” page	Pass	17:52-02/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
3.3 Delete account for student	Click on Delete	Delete successfully	Delete successful and go to the “All Student” page	Pass	17:58-02/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
3.4 Delete account for guest	Click on Delete	Delete successfully	Delete successful and go to the “All Guest” page	Pass	18:00-02/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang

4.1 Create a new Faculty	Create a new faculty with Faculty Name: Python	Create successfully	Create successfully and go to the “Faculty” page	Pass	18:10-02/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
4.2 Create a new Faculty without filling the form	Create a faculty without filling the Name and class ID of that faculty	Can not create	Create successfully and go to the “Faculty” page	Fail	18:11-02/03/2021 Test on URL: https://localhost:3000/ Assigned for Minh to fix this problem. Test exercutor: Khang
4.3 Update a faculty	Click on detail, then update on the “Faculty page.”	Update successfully	Update successful and go to the “Faculty” page	Pass	18:12-02/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
4.4 Delete a faculty	Click on detail, then Delete on “Faculty page.”	Delete successfully	Delete successful and go to the “Faculty” page	Pass	18:13-02/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
5.1 Login	Log in as an administrator with email: admin@gmail.com	Login successfully.	Login successful and Home page is admin’s Personal information	Pass	17:05-25/03/2021

	Password: 123				Test on URL: https://localhost:3000/ Test exercutor: Khang
5.2 Login	Log in as an administrator with email: admin@gmail.com Password: 1	I cannot log in	Cannot log in and inform username or password is invalid	Pass	17:08-25/02/2021 Wrong password Test on URL: https://localhost:3000/ Test exercutor: Khang
5.3 Logout	Press the X button on the interface	Return to Login screen	Return to Login screen	Pass	17:11-25/02/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
6 Set deadline for student	Click on “set time.”	Time settled	The deadline was all set for student	Pass	13:13-20/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang

b. Marketing Manager

Test case	Input	Expected result	Actual result	Test condition	Test log

1.1 Login	Login as Marketing manager with email: manager1@gmail.com Password: 123	Login successfully.	Login successful and Home page is manager's Personal information	Pass	13:05-25/02/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
1.2 Login	Login as Marketing Manager with email: admin@gmail.com Password: 1	I cannot log in	Cannot log in and inform username or password is invalid	Pass	17:08-25/02/2021 Wrong password Test on URL: https://localhost:3000/ Test exercutor: Khang
2 View Statistic	Click on View contribution, then view statistical	View successful	When you click on it, the pie chart of that faculty showed up	Pass	13:08-20/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
3 View all submission by faculty	Click on View contribution, then view contributions again on a faculty	View successful	All contribution of that faculty showed up	Pass	13:10-14/03/2021 Test on URL: https://localhost:3000/

					Test exercutor: Khang
4 Download submission into a ZIP	Click on download contribution	Download successful	The contribution was downloaded in ZIP format	Pass	13:12-17/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang

c. Marketing Coordinator

Test case	Input	Expected result	Actual result	Test condition	Test log
1.1 Login	Login as Marketing Coordinator with email: teacher1@gmail.com Password: 123	Login successfully.	Login successful and Home page is teacher's Personal information	Pass	13:20-25/02/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
1.2 Login	Login as Marketing Manager with email: admin@gmail.com Password: 1	I cannot log in	Cannot log in and inform username or password is invalid	Pass	13:22-25/02/2021 Wrong password Test on URL: https://localhost:3000/ Test exercutor: Khang

2 View submissions by faculty	Click on view submissions	All submission showed up	All submission and students submitted showed up	Pass	13:25-11/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
3 Receive notification	An email automatically sent when a student submits	An email automatically sent when a student submits	An email automatically sent when a student submits	Pass	13:28-10/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
4 Give comment before the due date	Give comment 14 days after submission	A timer will show around the comment box show that if it passes 14 days, the submission can not receive a comment	Submission can be a comment on anytime	Fail	13:29-11/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
5.1 Accept submission	Choose “Pass” in submission Status	Status of submission showed as Pass	Status of submission showed as Pass	Pass	13:35-11/03/2021

					Test on URL: https://localhost:3000/ Test exercutor: Khang
5.2 Reject submission	Choose “Fail” in submission Status	Status of submission showed as fail	Status of submission showed as fail	Pass	13:40-11/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
6 Use chatbox	Add a student to box chat, then use it to communicate	The student added and received the teacher message	The student added and received the teacher message	Pass	14:00-29/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang

d. Student

Test case	Input	Expected result	Actual result	Test condition	Test log
1.1 Login	Log in as a student with email: student1@gmail.com Password: 123	Login successfully.	Login successful and Home page is student’s Personal information	Pass	14:20-25/02/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang

1.2 Login	Login as Student with email: admin@gmail.com Password: 1	I cannot log in	Cannot log in and inform username or password is invalid	Pass	14:22-25/02/2021 Wrong password Test on URL: https://localhost:3000/ Test exercutor: Khang
2.1 submit an article as word or pdf	Click on Upload file, accept the term, then choose a file to upload	File uploaded	File uploaded and returned to the Summited file	Pass	14:30-07/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
2.2 submit an article as images/photos	Click on Upload file, accept the term, then choose a file to upload	File uploaded	File uploaded and returned to the Summited file	Pass	14:30-08/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
2.3 Edit submission	Click on the Update file	File updated	File updated then returned to the Summited file	Pass	14:36-29/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang

3 Read comment for own submission	Click on the file submitted	View comment of the submitted file	Comment showed on the Submitted file page	Pass	15:18-29/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
4 Use chatbox	Click on the chat box	The chatbox showed up	Chatbox which student was added showed up, and after clicked on one, the chatbox between teacher and student showed up	Pass	15:40-29/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang

e. Guest

Test case	Input	Expected result	Actual result	Test condition	Test log
1.1 Login	Login as guest with email: guest1@gmail.com Password: 123	Login successfully.	Login successful and Home page is guest's Personal information	Pass	15:50-25/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang
1.2 Login	Login as guest with email: admin@gmail.com Password: 1	I cannot log in	Cannot log in and inform username or password is invalid	Pass	15:56-25/02/2021 Wrong password

					Test on URL: https://localhost:3000/ Test exercutor: Khang
2 View accepted the submission by faculty	Click on the view button of the submission in that faculty	All accepted submission of that faculty showed	All accepted submission of that faculty showed an can be view detailed after clicking on the view button	Pass	16:00-13/03/2021 Test on URL: https://localhost:3000/ Test exercutor: Khang

f. System

Test case	Input	Expected result	Actual result	Test condition	Test log
1 Test Role Admin in sever	Log in and test all admin function	Admin function work perfectly	Admin function work perfectly	Pass	13:50-27/03/2021 Test on URL: https://appcuatoi123.herokuapp.com/ Test exercutor: Huong
2 Test Role Marketing Manager in sever	Log in and test all Marketing Manager function	The Marketing Manager function work perfectly	The Marketing Manager function work perfectly	Pass	16:03-27/03/2021 Test on URL: https://appcuatoi123.herokuapp.com/

					Test exercutor: Huong
3 Test Role Marketing Coordinator in sever	Log in and test all Marketing Coordinator function	The marketing Coordinator function work perfectly	The marketing Coordinator function work perfectly	Pass	11:28-28/03/2021 Test on URL: https://appcuatoi123.herokuapp.com/ Test exercutor: Huong
4 Test Role Student in sever	Log in and test all Student function	Student function work perfectly	Student function work perfectly	Pass	11:28-28/03/2021 Test on URL: https://appcuatoi123.herokuapp.com/ Test exercutor: Huong
5 Test Role Guest in sever	Log in and test all Guest function	Guest function work perfectly	Guest function work perfectly	Pass	09:10-29/03/2021 Test on URL: https://appcuatoi123.herokuapp.com/

					Test exercutor: Huong
6 Test chat box in sever	Use Marketing Coordinator account to add a student to the chatbox and interact between 2 accounts	The chatbox work perfectly	Only added student can interact in the chatbox with the teacher who added him/her	Pass	20:11-29/03/2021 Test on URL: https://appcuatoi123.herokuapp.com/ Test exercutor: Huong

3. Test evident

To increase the reliability of the test, we decided to create a section contain test evidence. In this section, we will present a screenshot of the testing process

a. Login/Logout

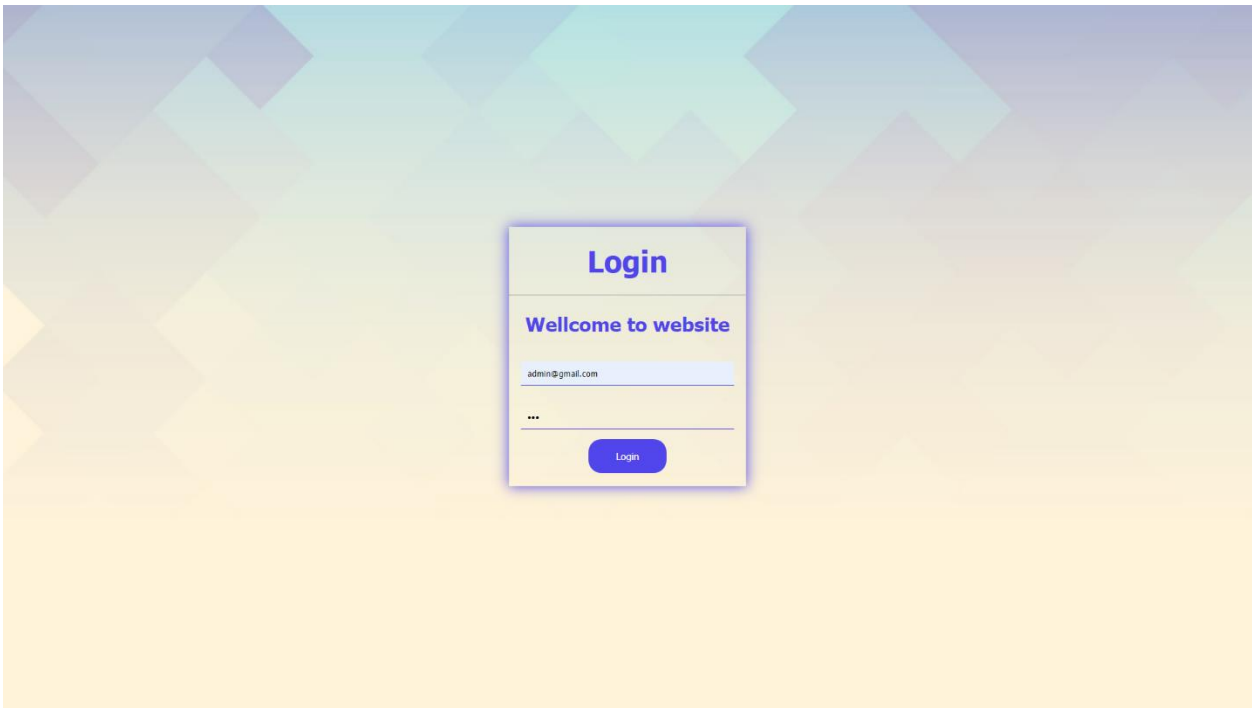


Figure 74:Login as admin

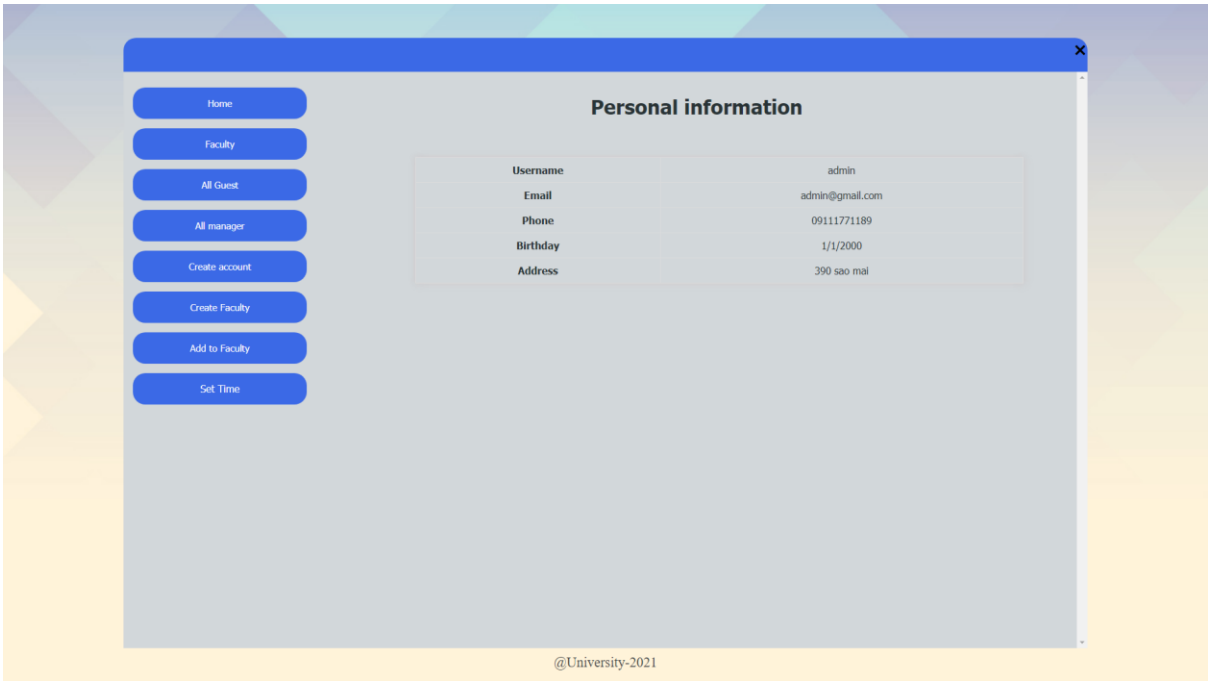


Figure 75: Login successful

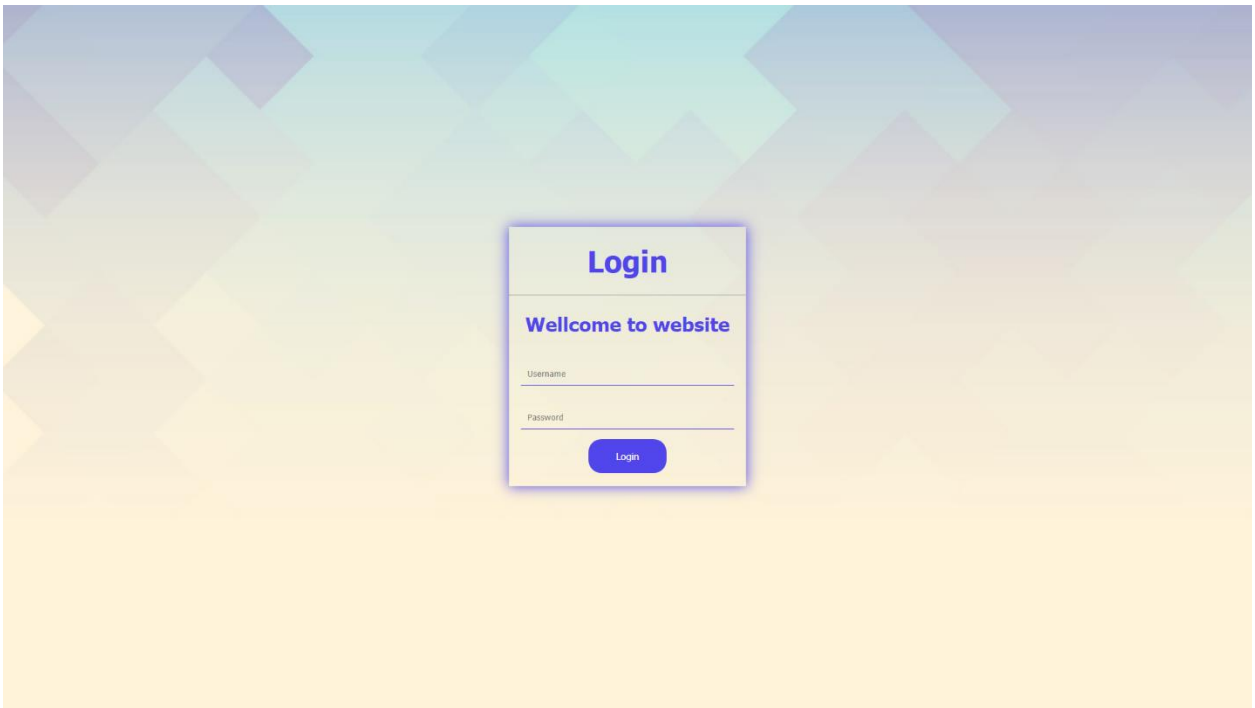


Figure 76:Logout by pressing the “X” button

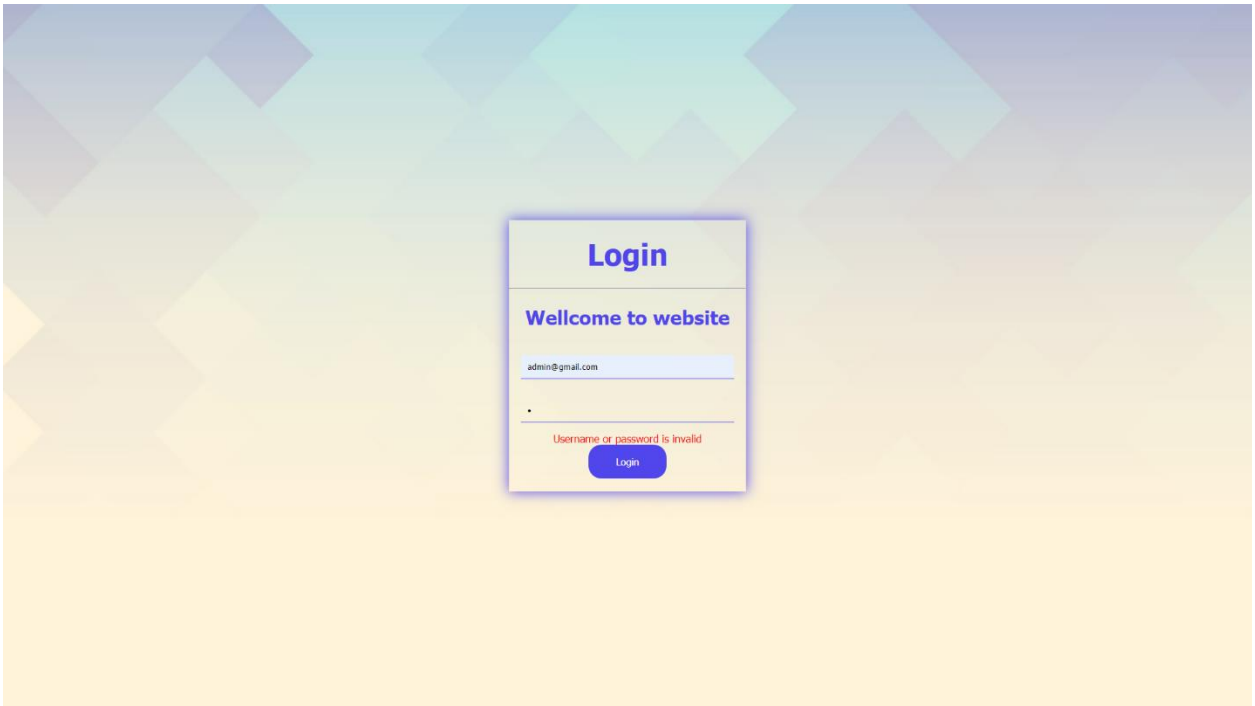


Figure 77:Login fail (wrong password)

b. Create, Update and Delete faculty

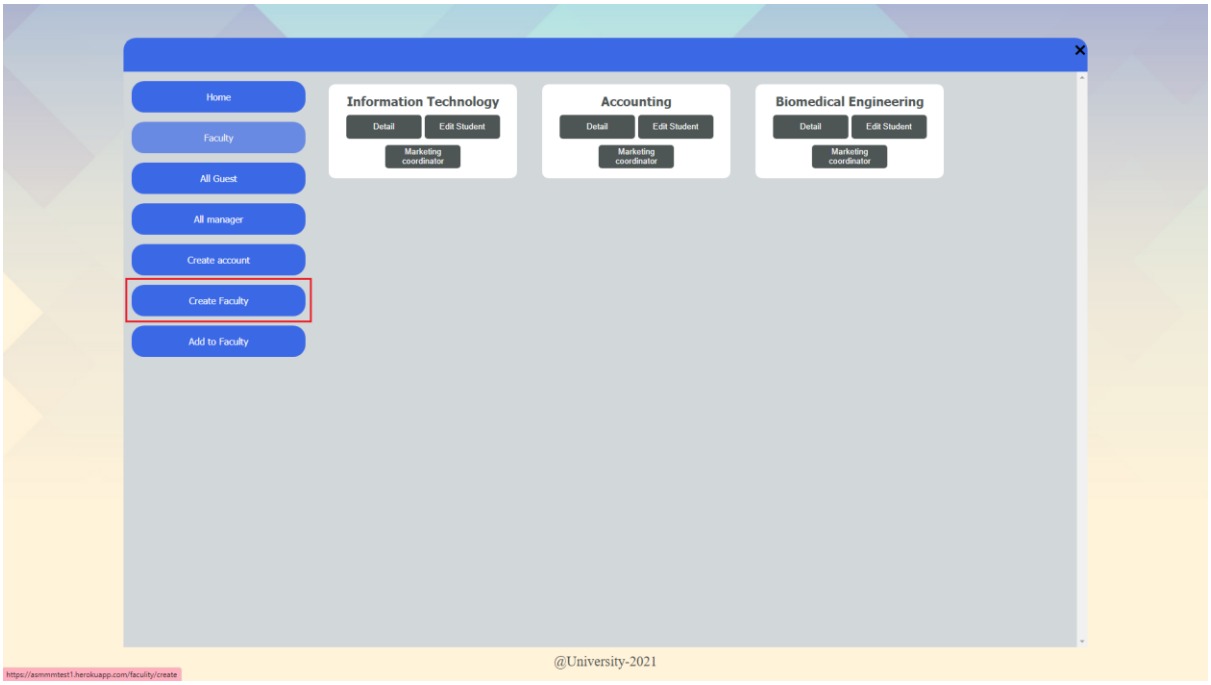


Figure 78:Faculty list before creating a new one

Figure 79: Creating a new faculty

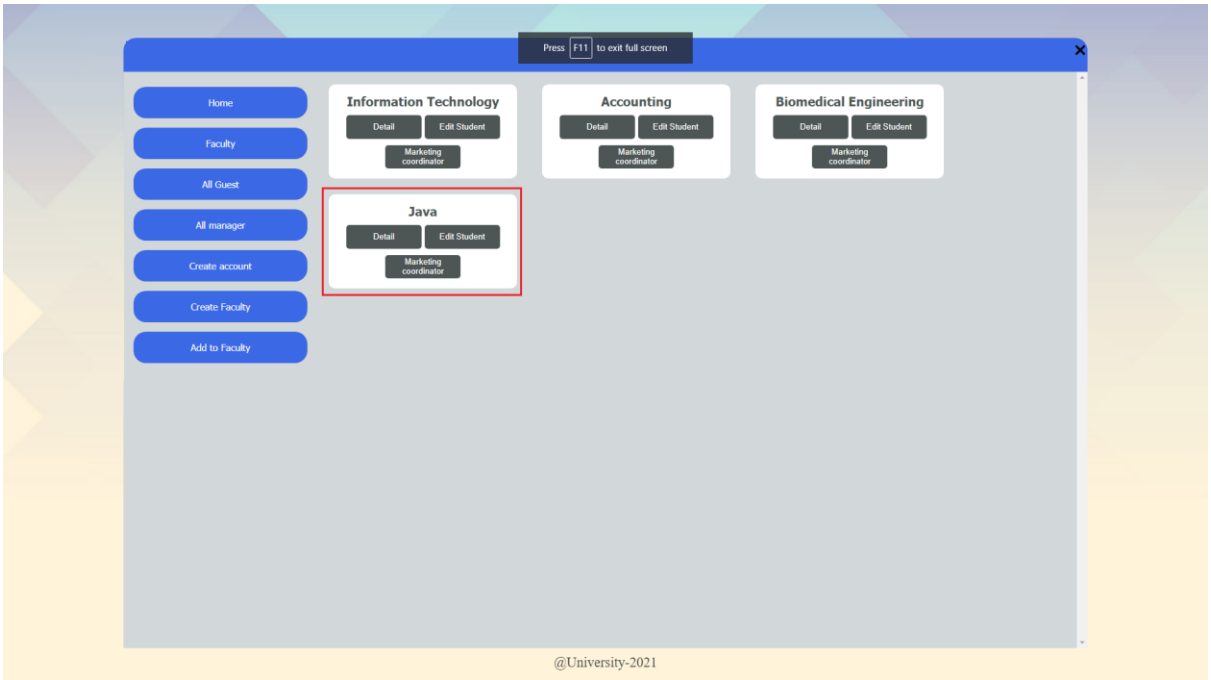


Figure 80: Create faculty successfully

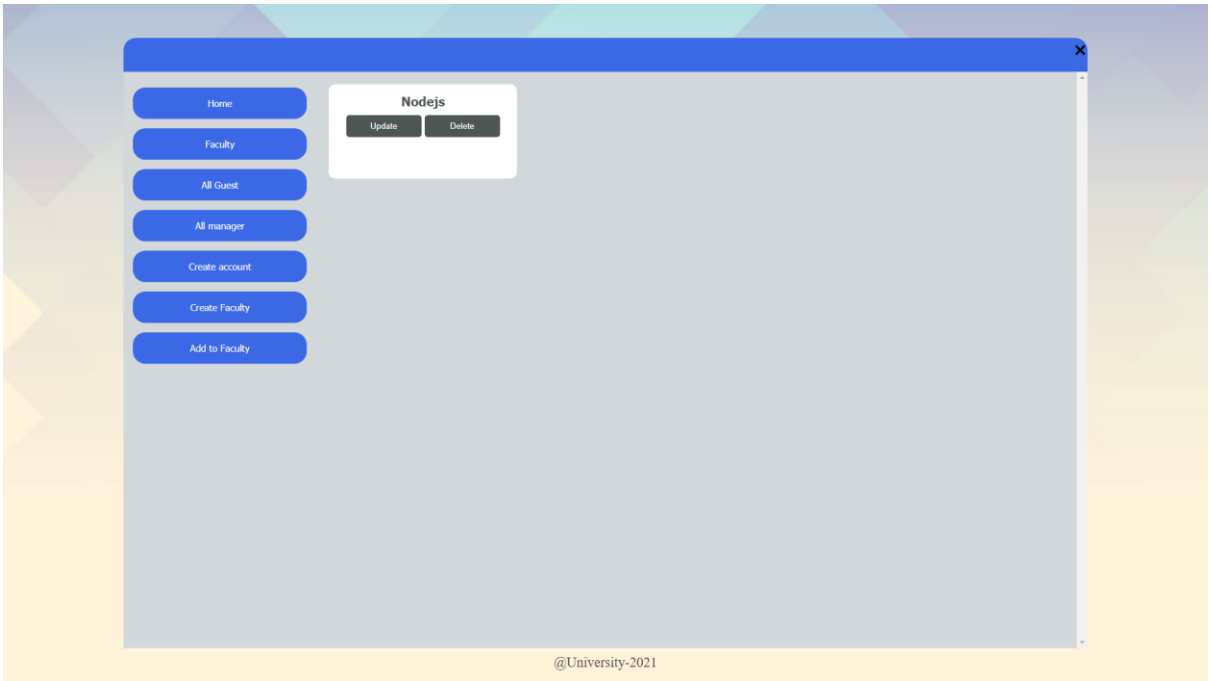


Figure 81: Update or delete button of faculty

Figure 82: Changing the subject name to Python

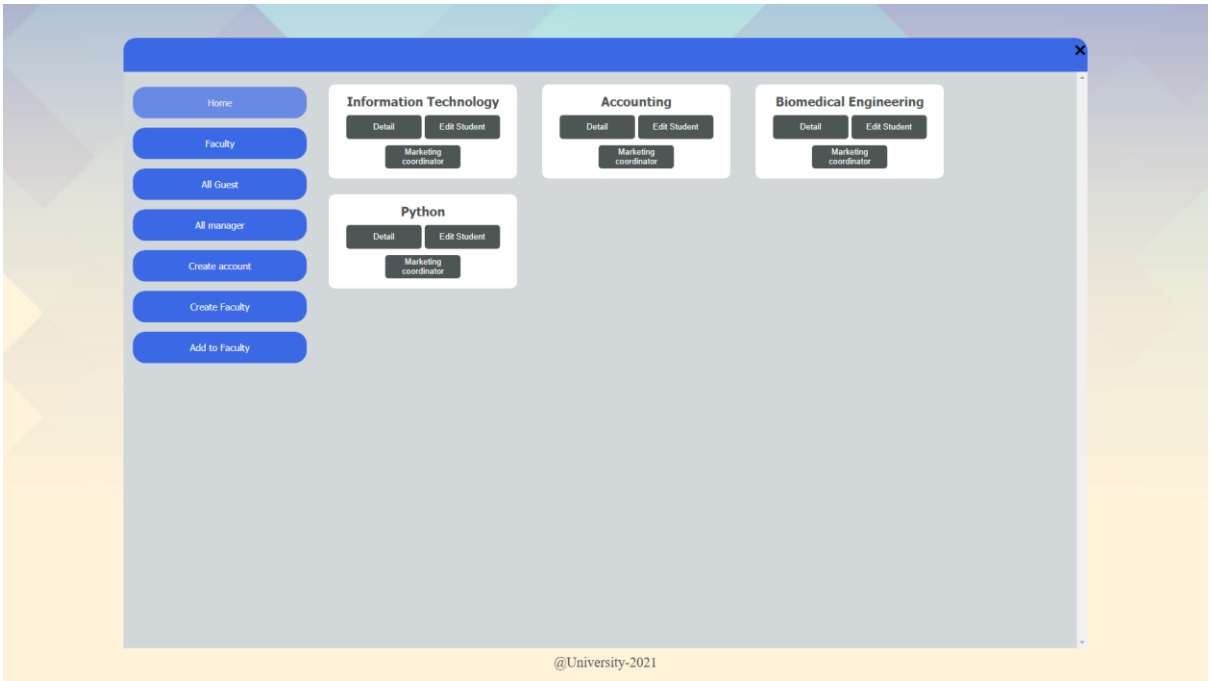


Figure 83:Faculty display after update

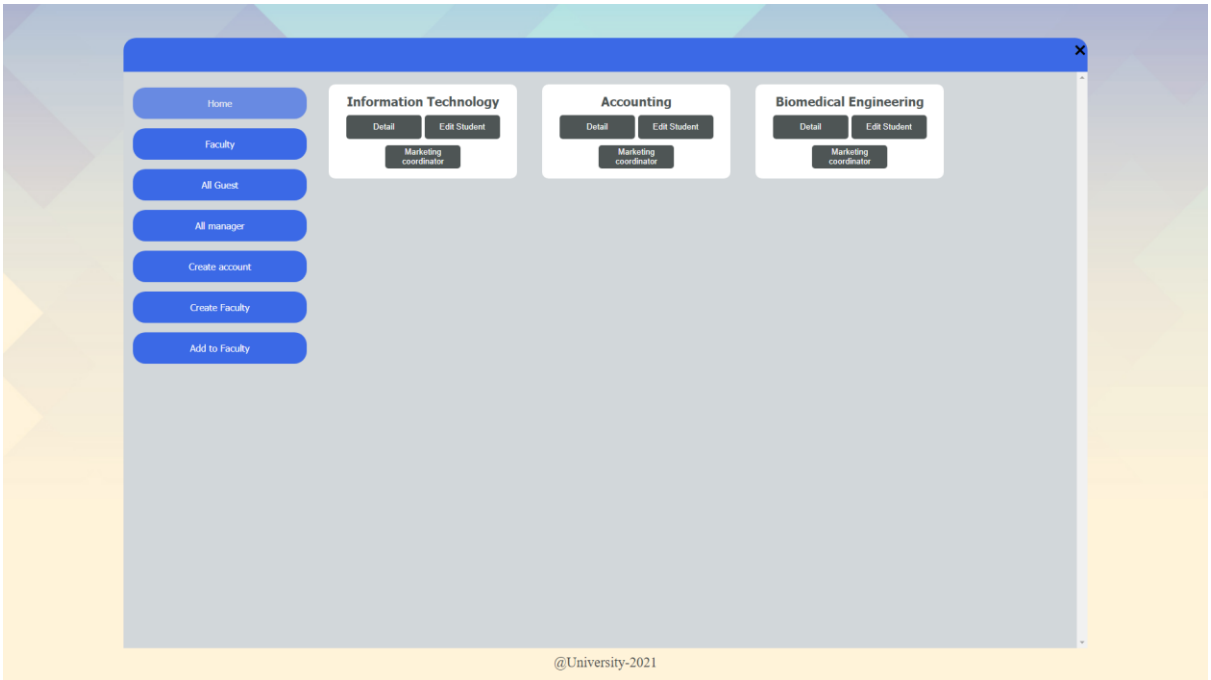


Figure 84: Faculty display after delete

c. Create, Update and Delete account

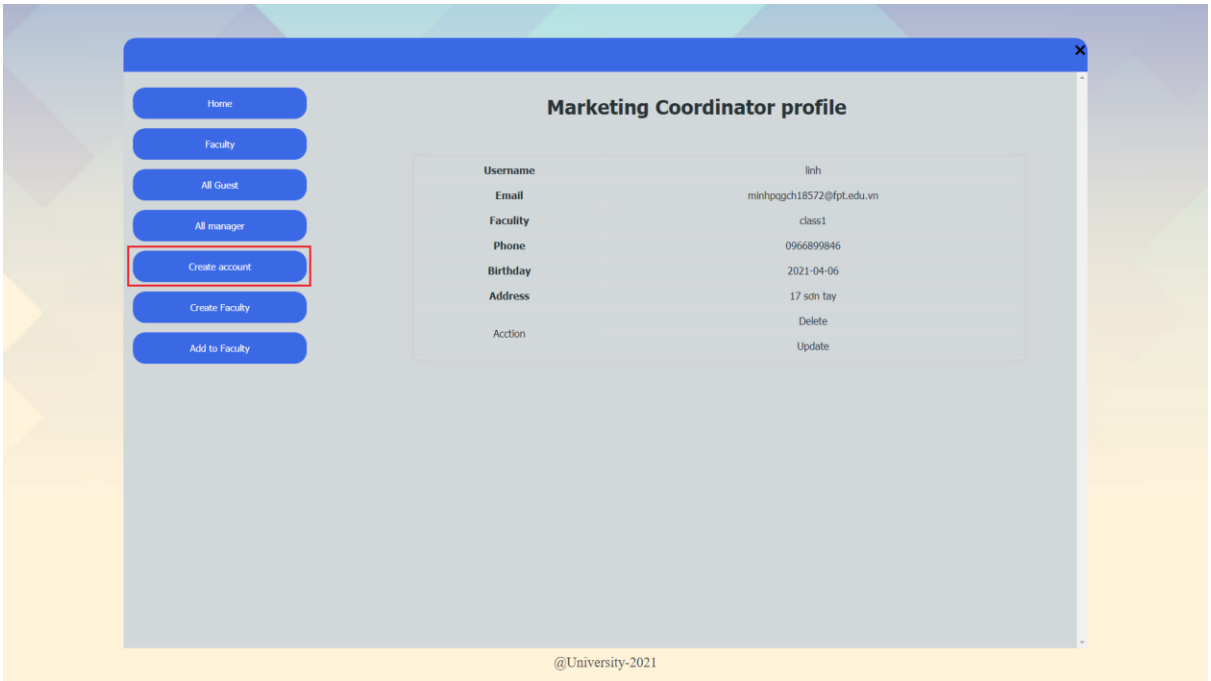


Figure 85: Marketing coordinator list before create a new one

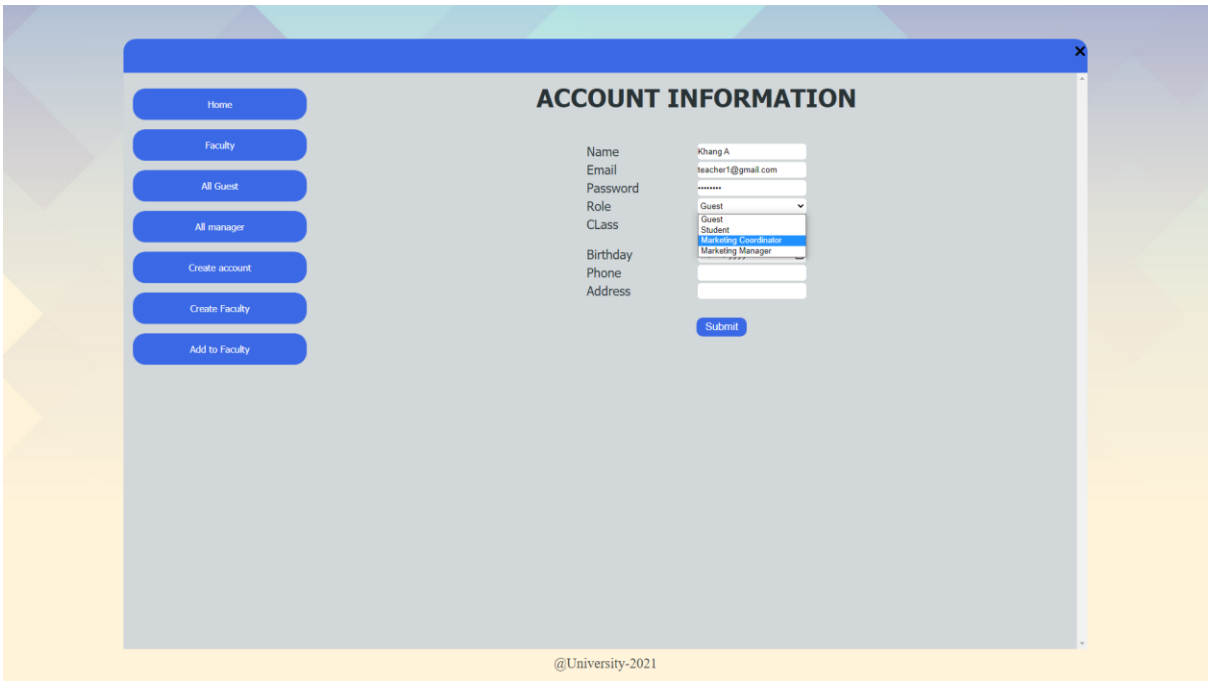


Figure 86: Creating a new Marketing coordinator

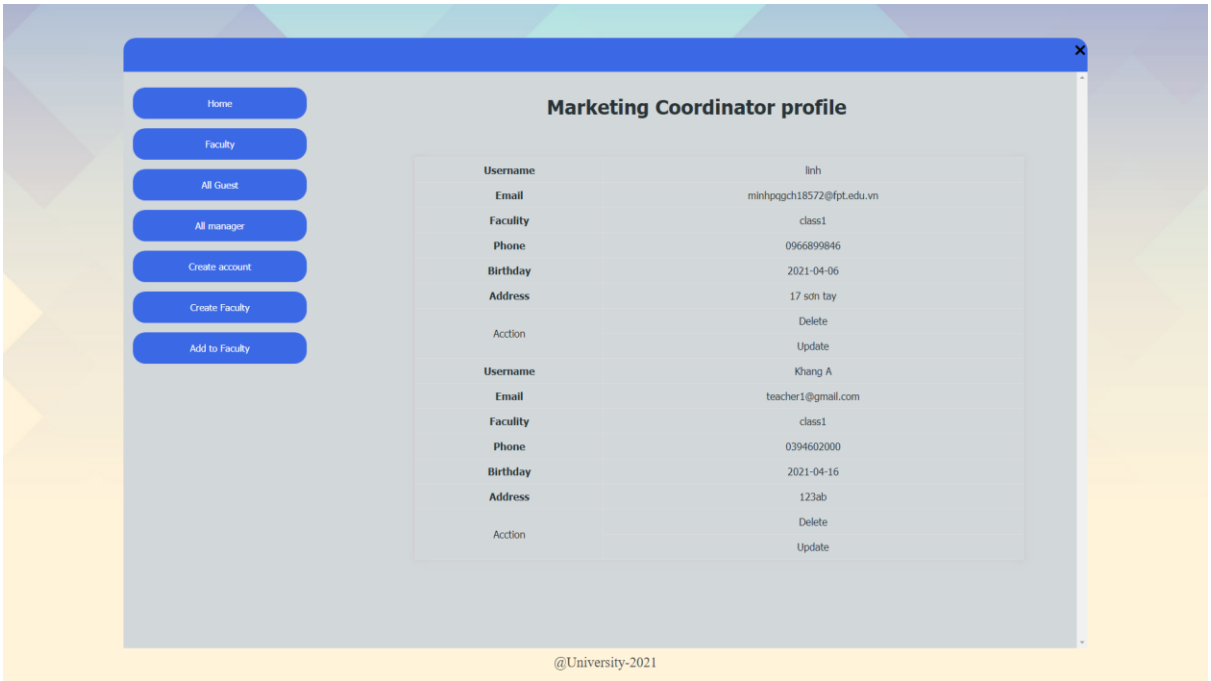


Figure 87: Create a new marketing coordinator successful

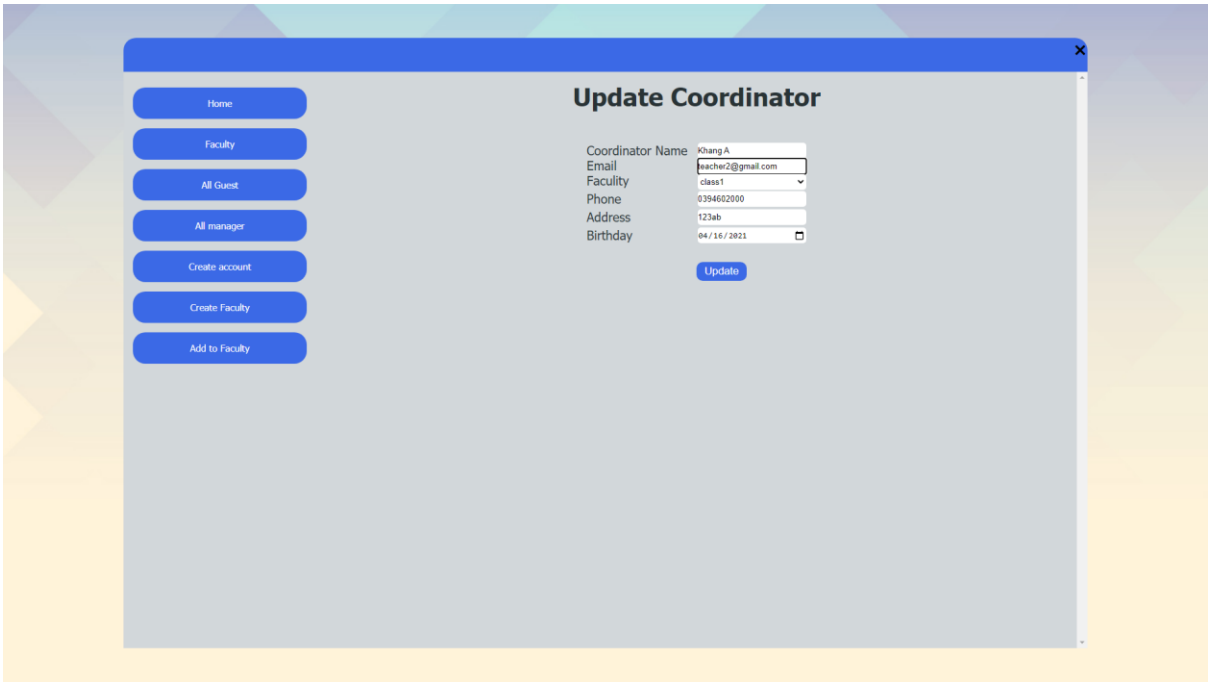


Figure 88: Update teacher email to teacher2@gmail.com

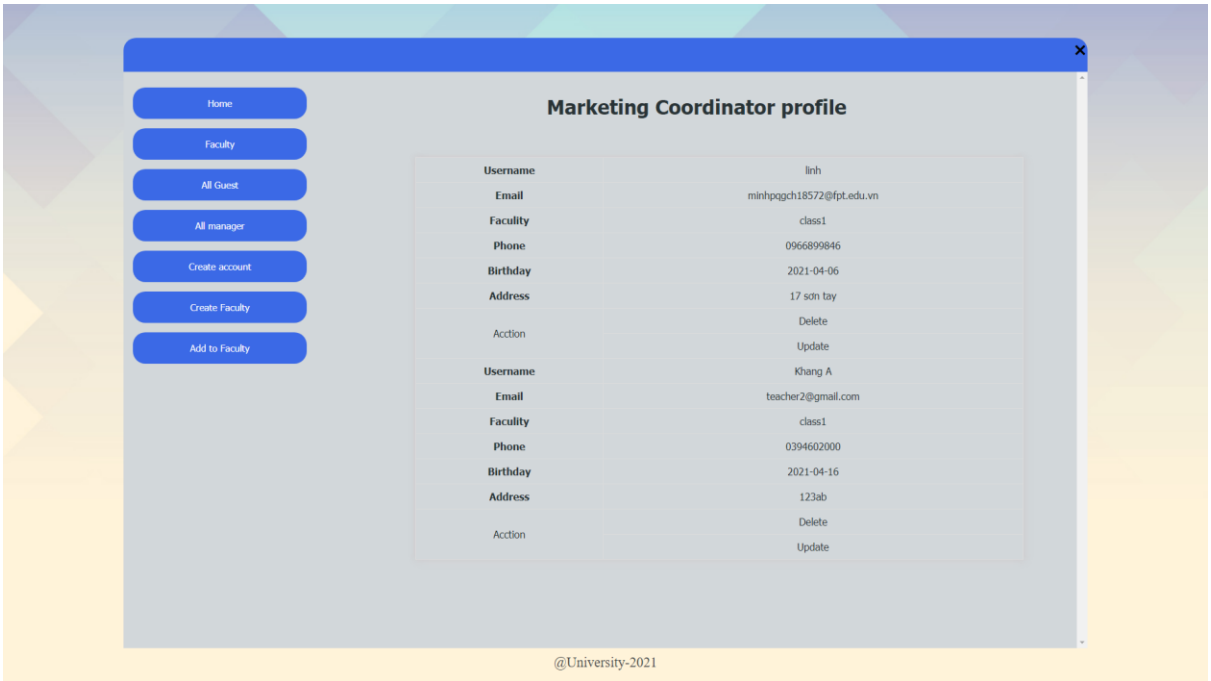


Figure 89: Update marketing coordinator successful

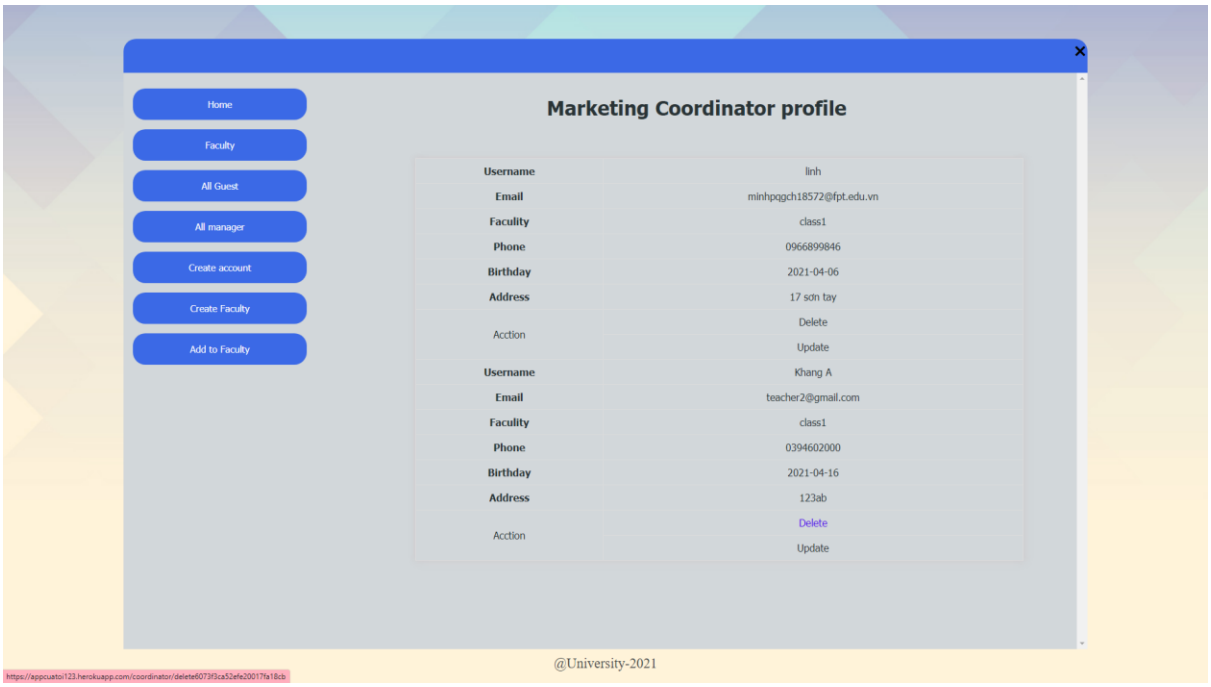


Figure 90: Delete marketing coordinator button

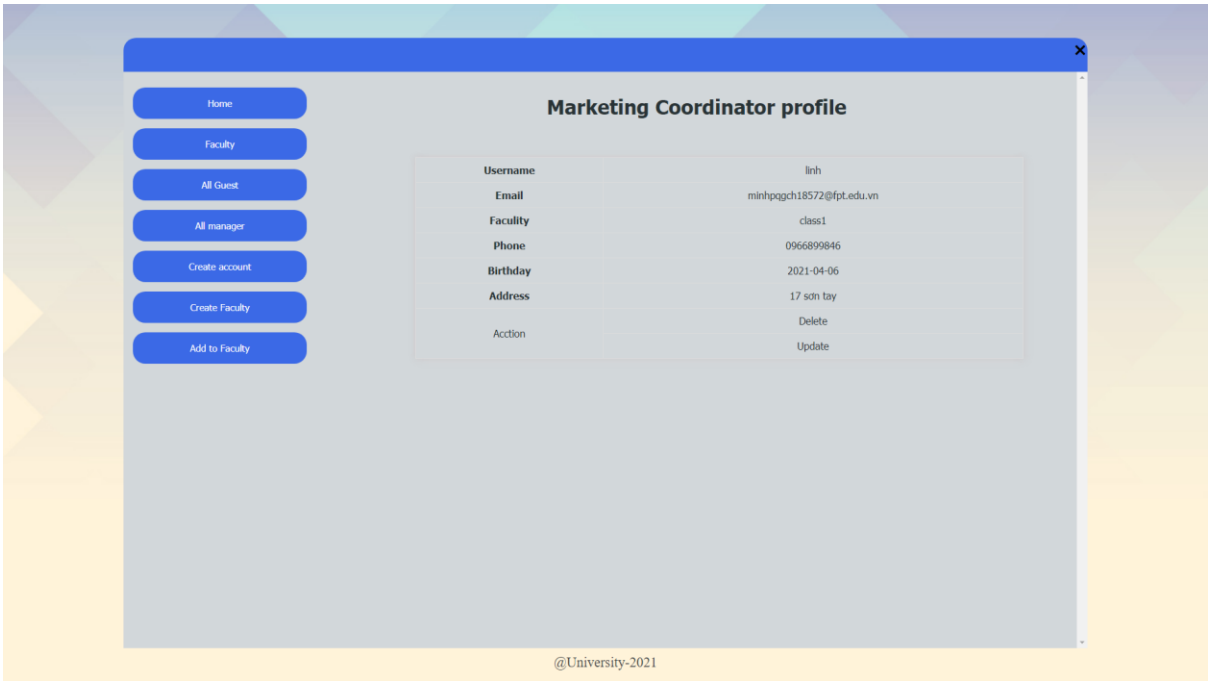


Figure 91: Delete successful

d. View statistical

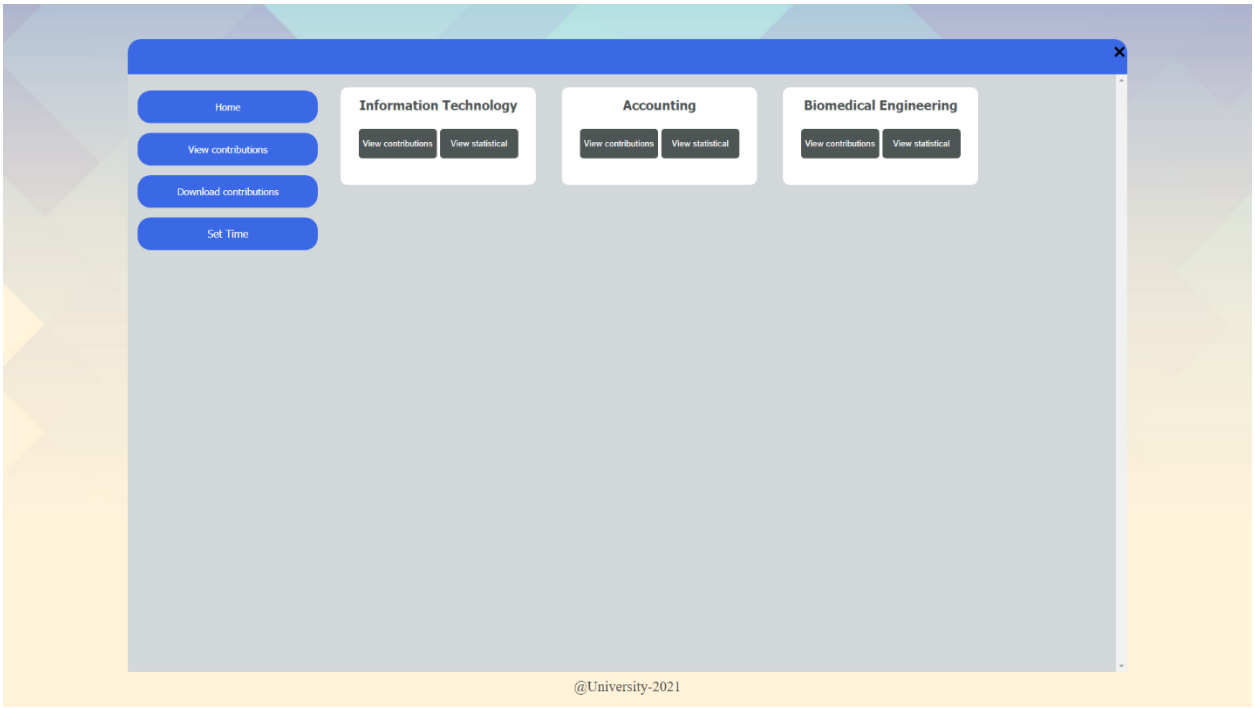


Figure 92: Display faculty list and view statistic button

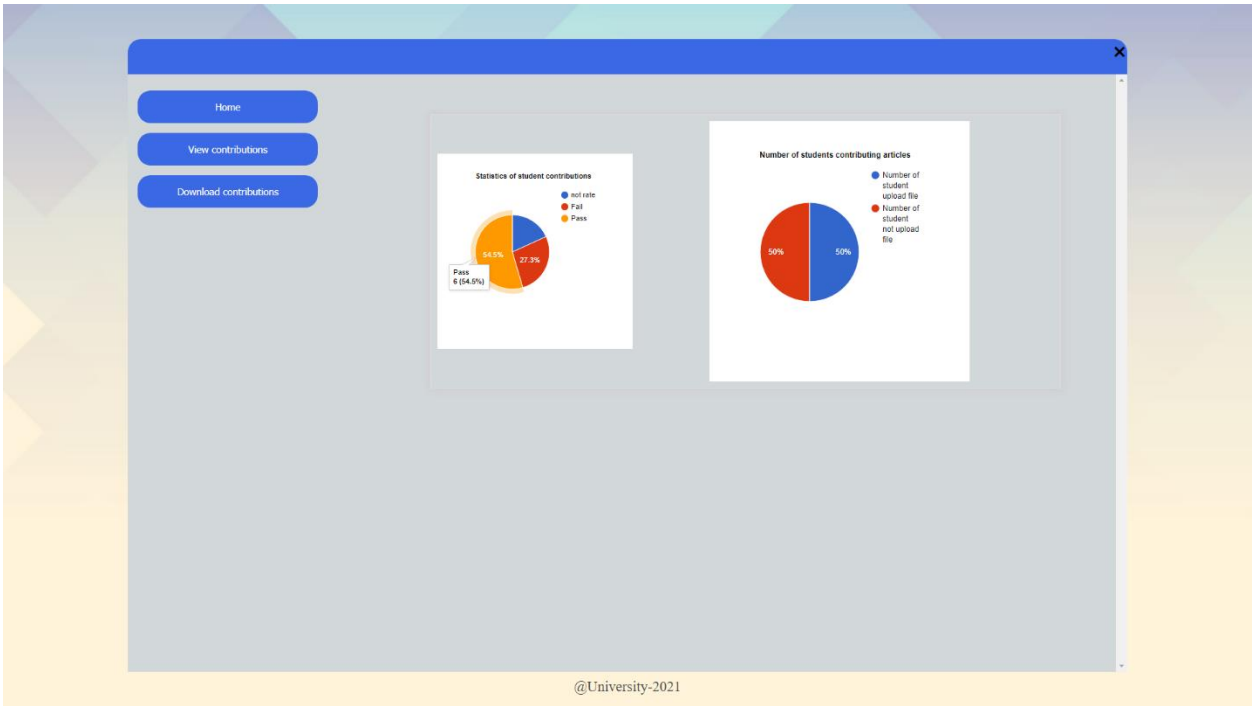


Figure 93: Statistic data

e. View contribution

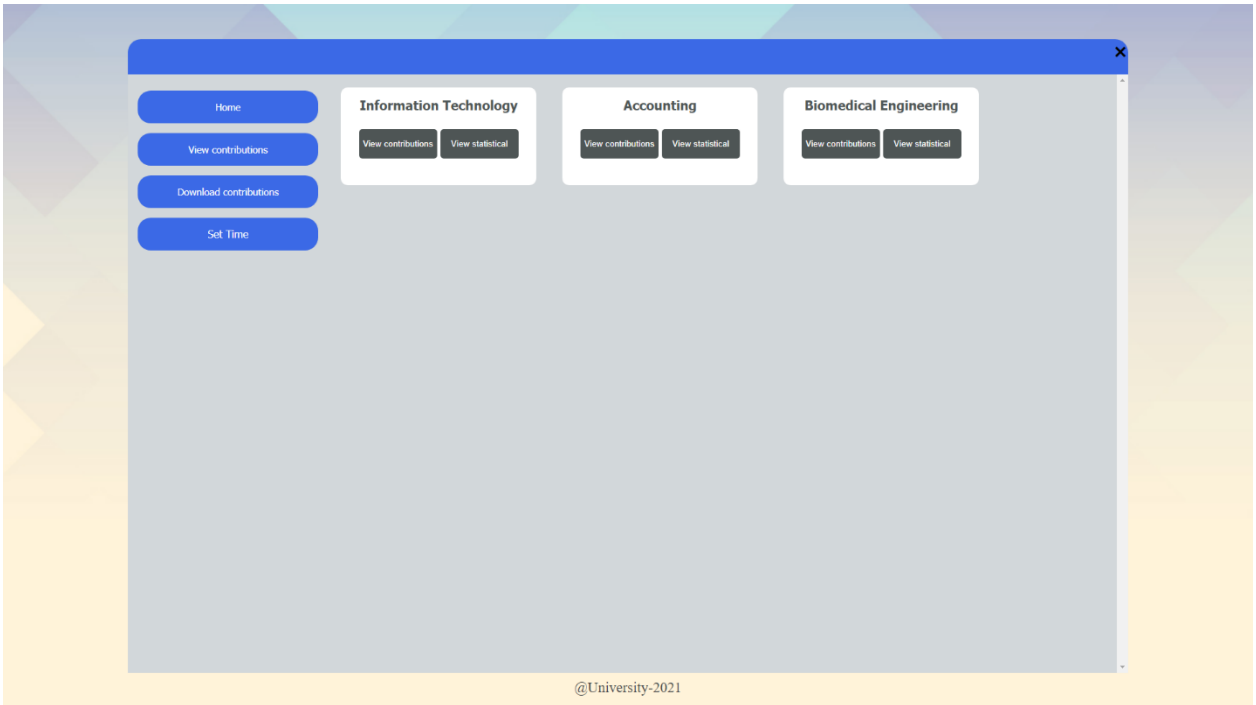


Figure 94: Display faculty list and view contribution button

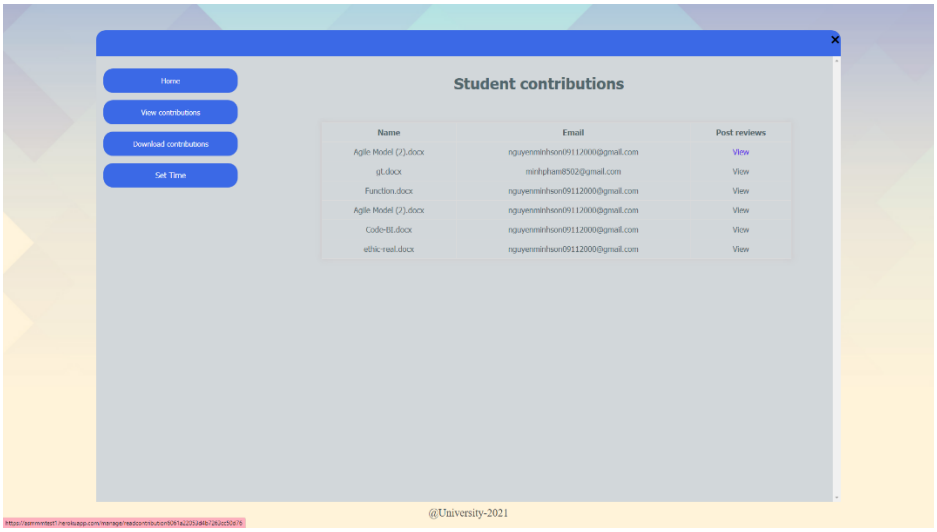


Figure 95: Submission list of that faculty

f. Notification email

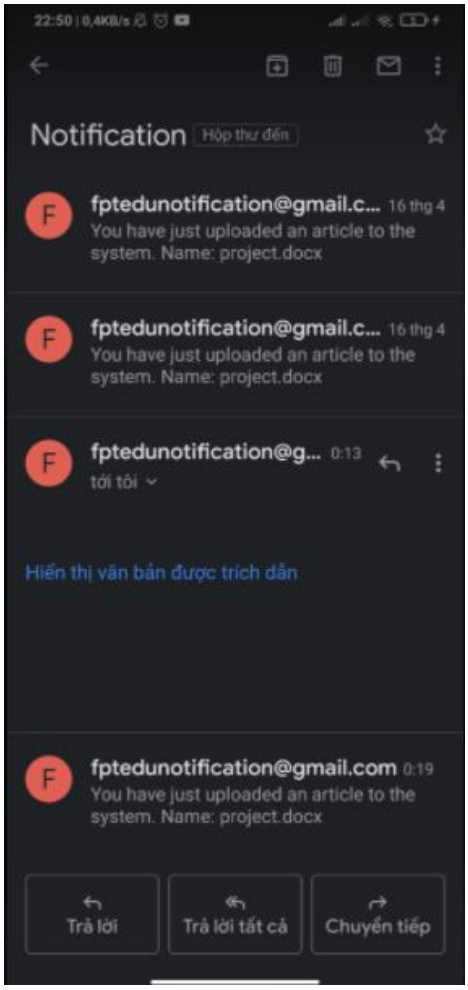


Figure 96: Mail notification

g. Download contribution

Figure 97: Click on All faculty and see the faculty list

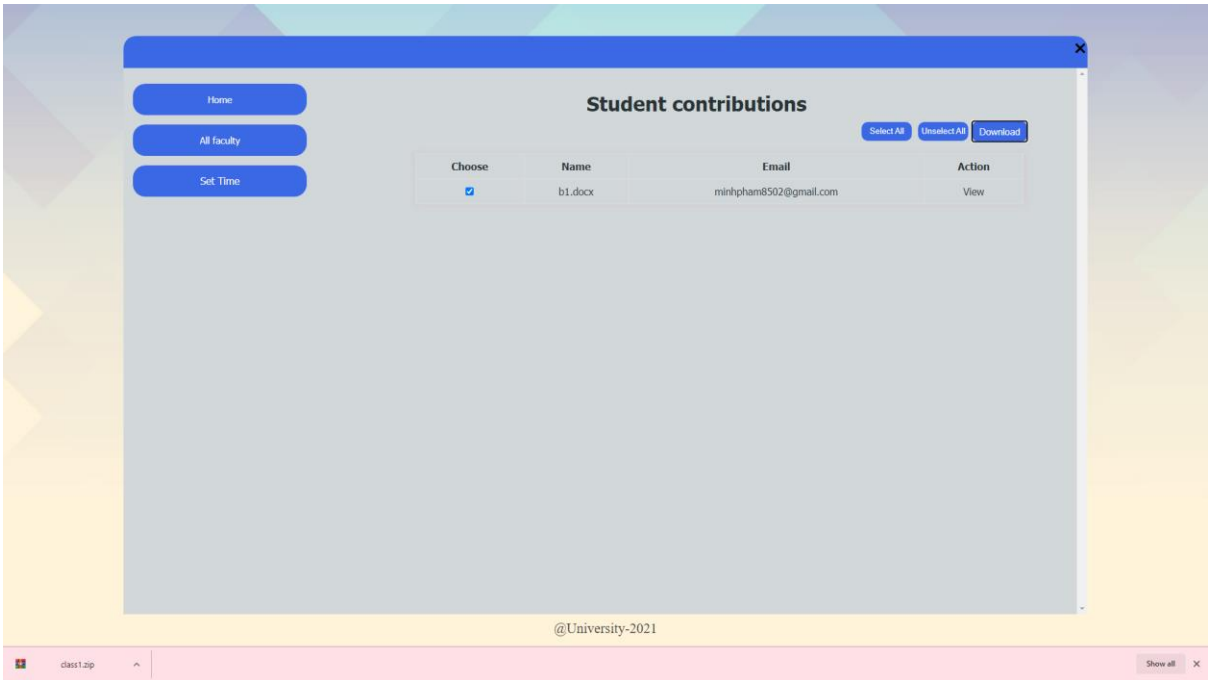


Figure 98: Select submission and download successful

h. Set a deadline for student

Figure 99: Click on the set time button

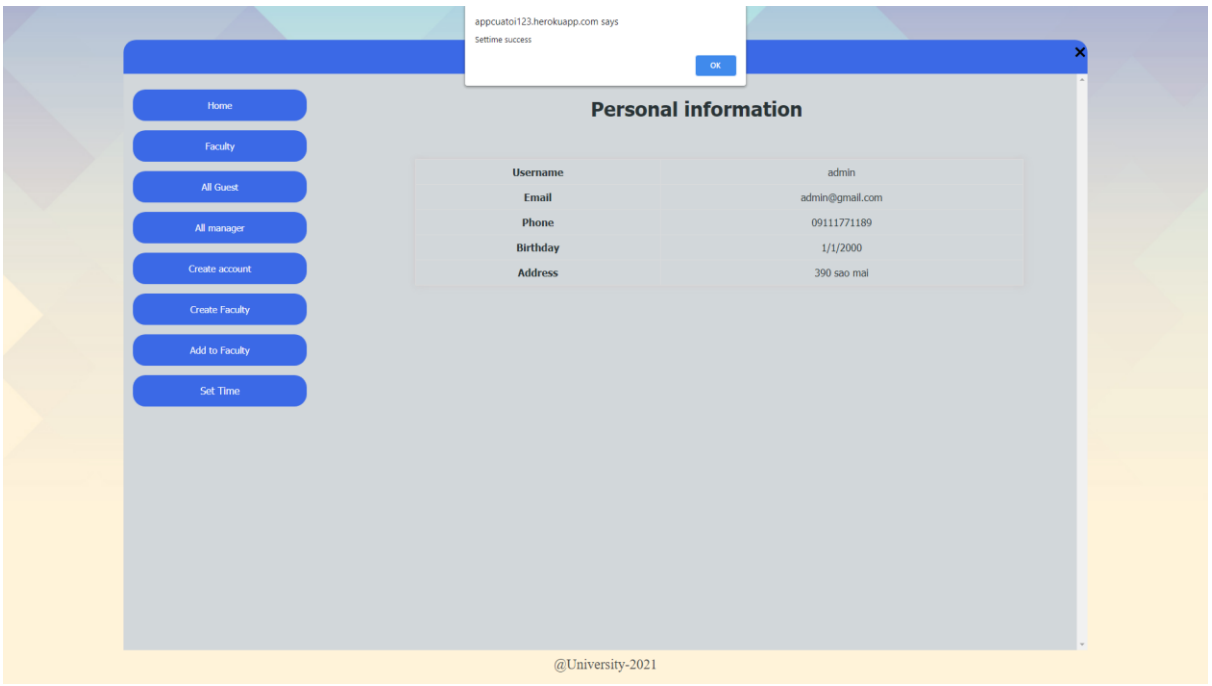


Figure 100: Set time successful

i. Comment and grading

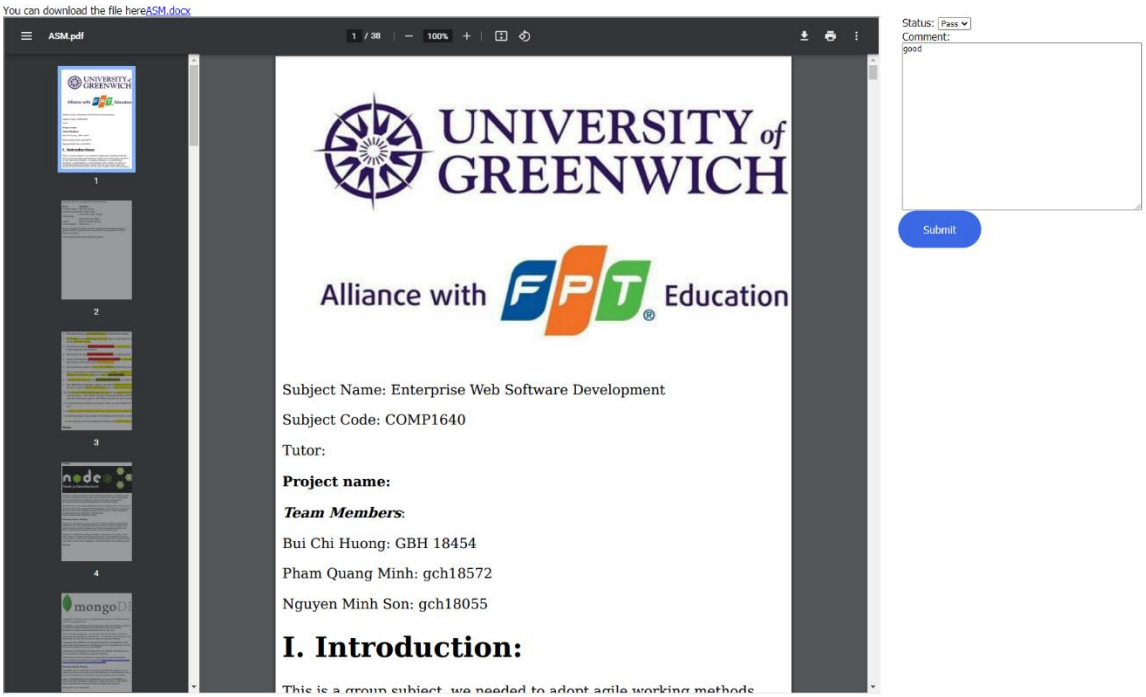


Figure 101: Grading and comment box after a click on view submission

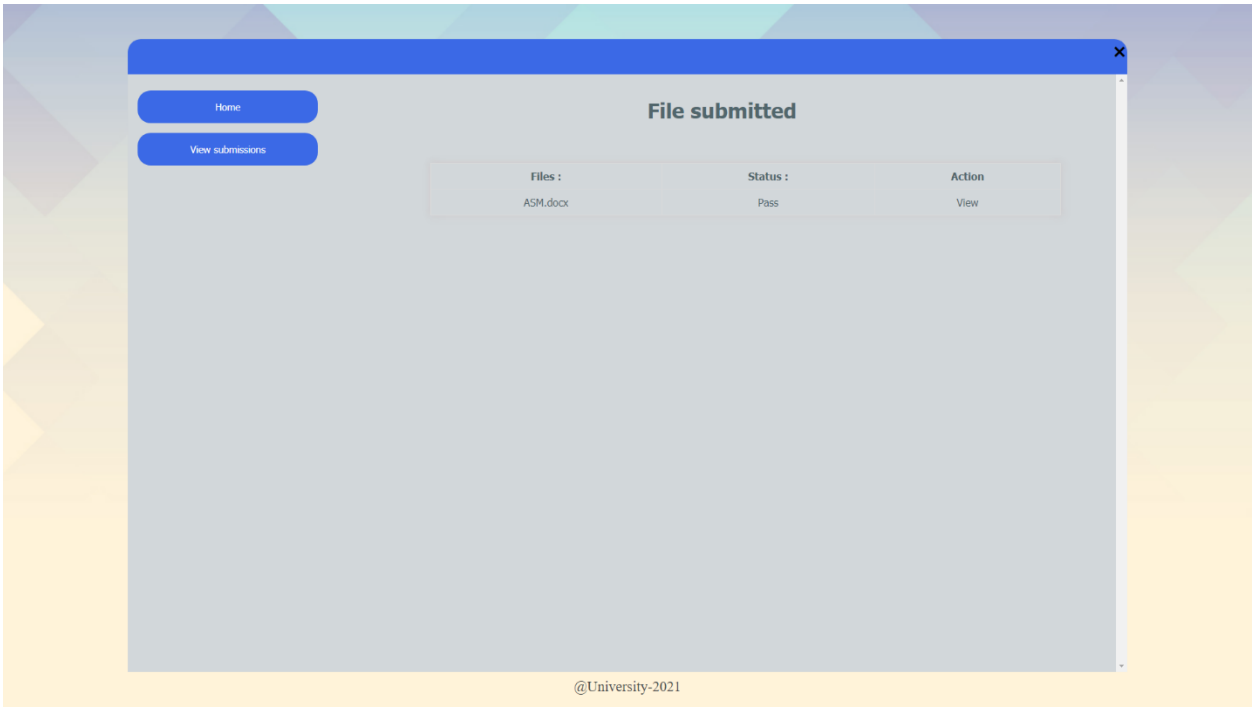


Figure 102: Grading successful

j. Chatbox

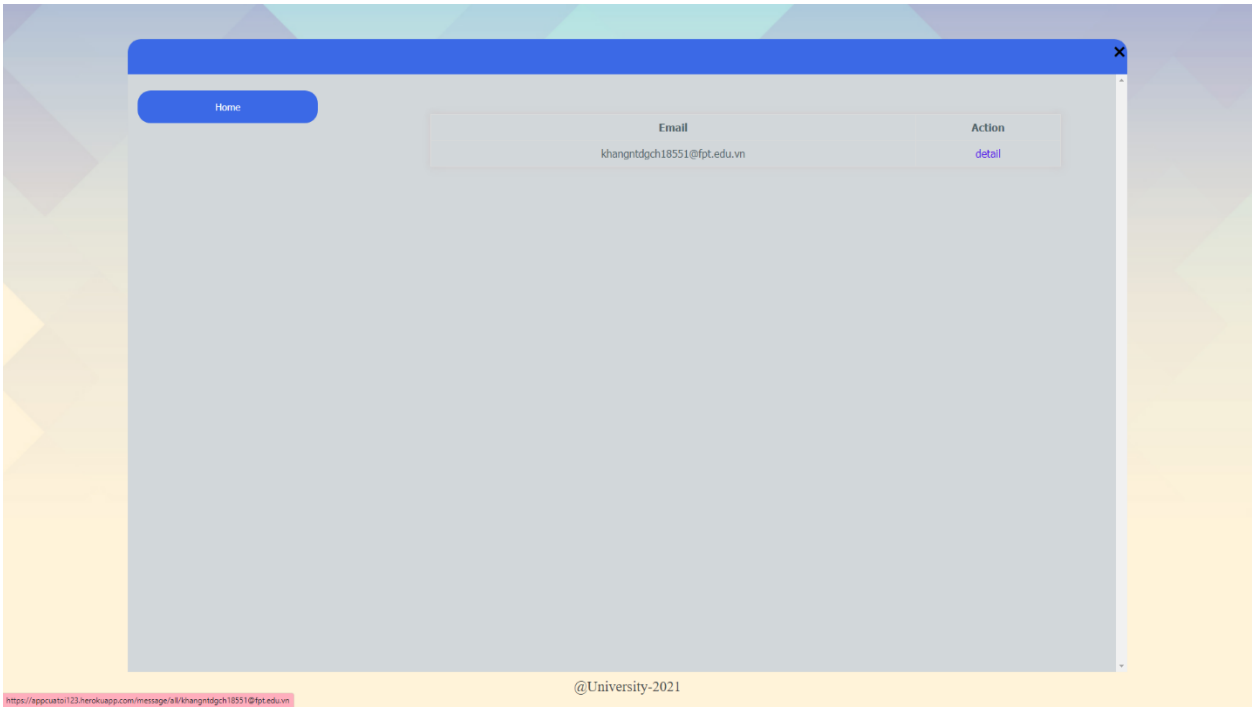


Figure 103: Adding people to the chat box

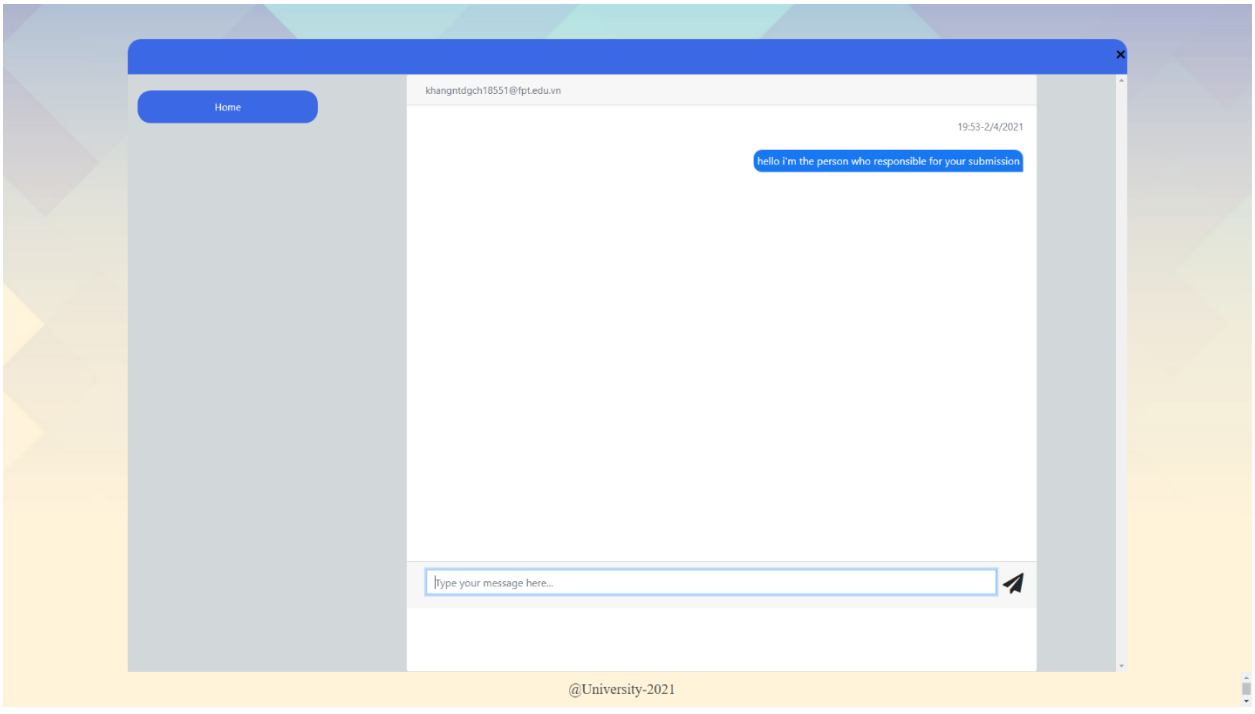


Figure 104: Using the chat box

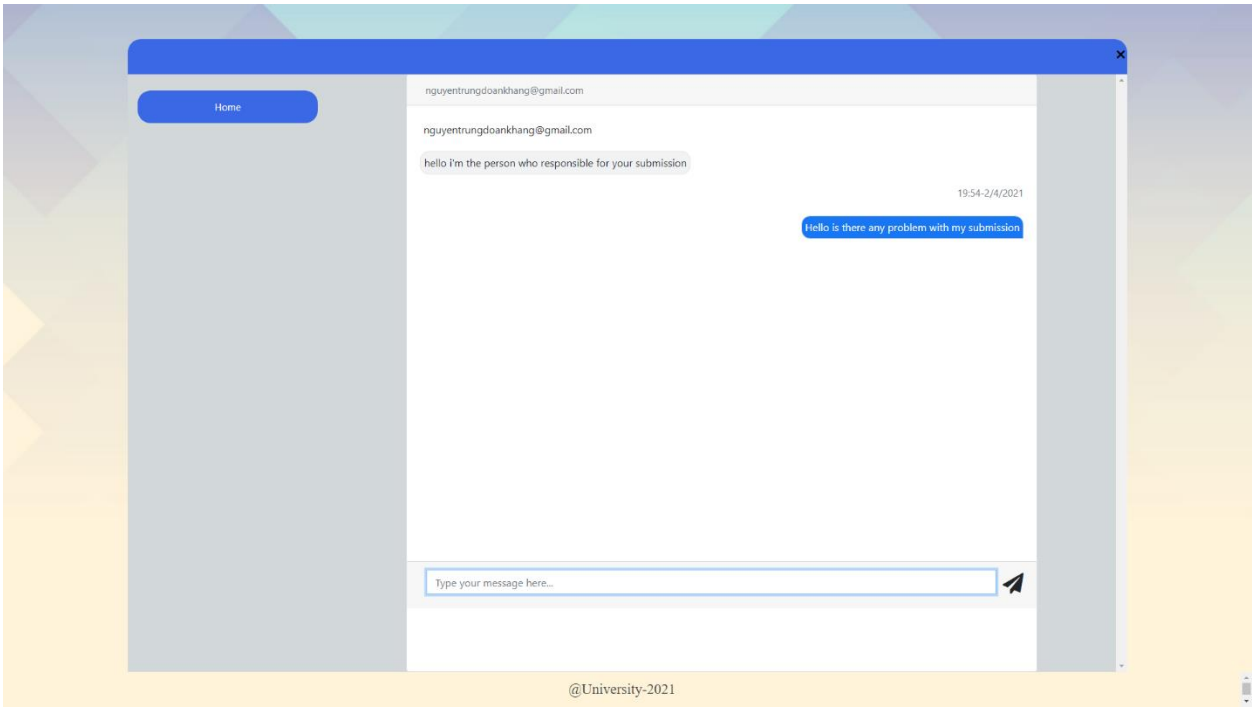


Figure 105: Interaction in chatbox between Coordinator and Student

k. Upload/ Update and view comment

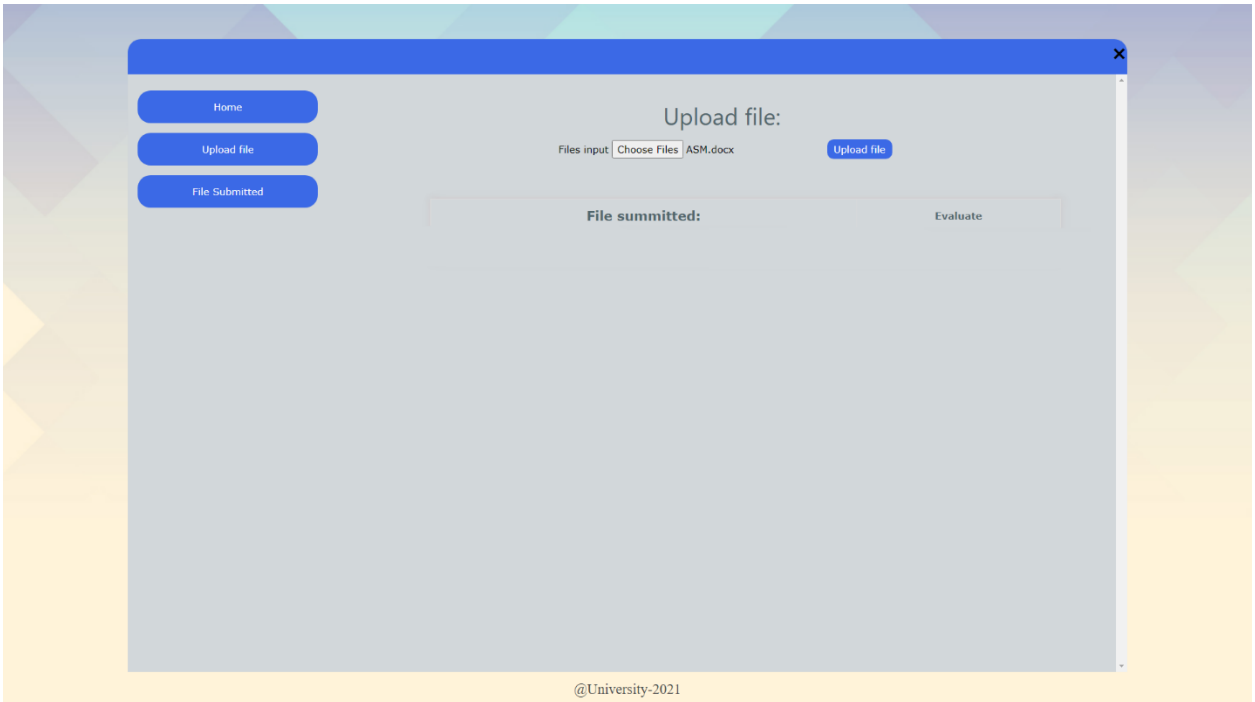


Figure 106: Uploading file

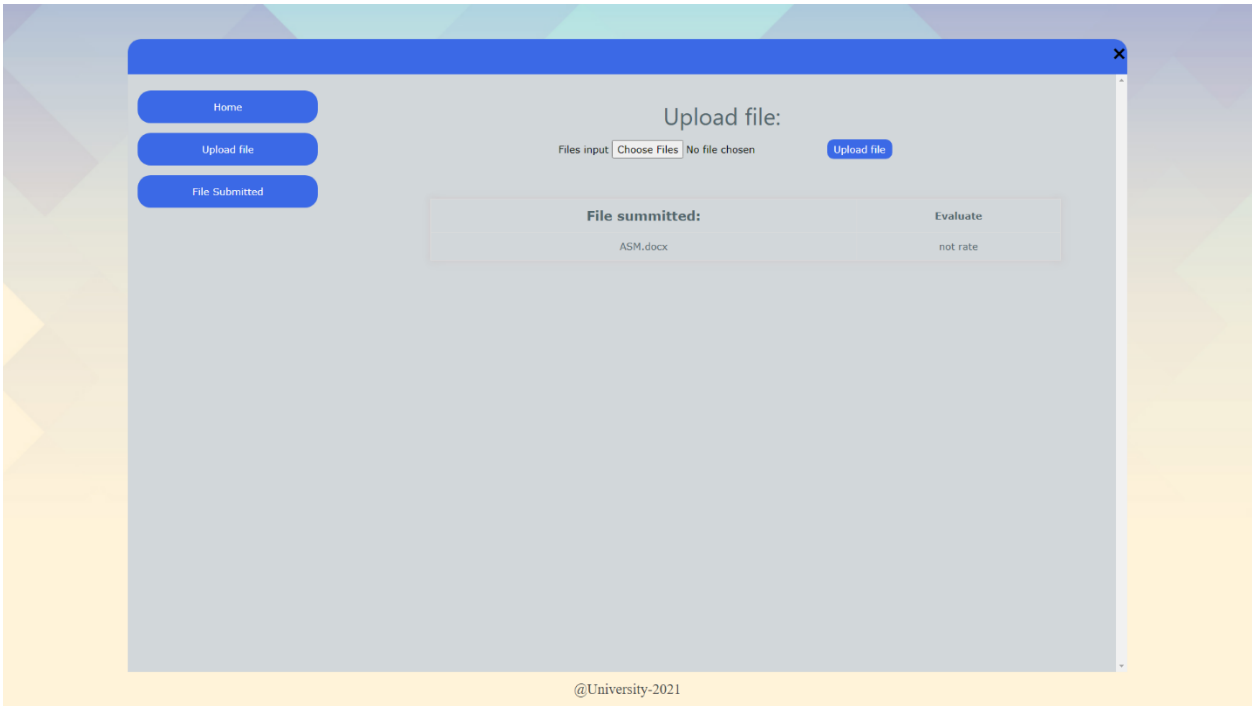


Figure 107: Uploading successful

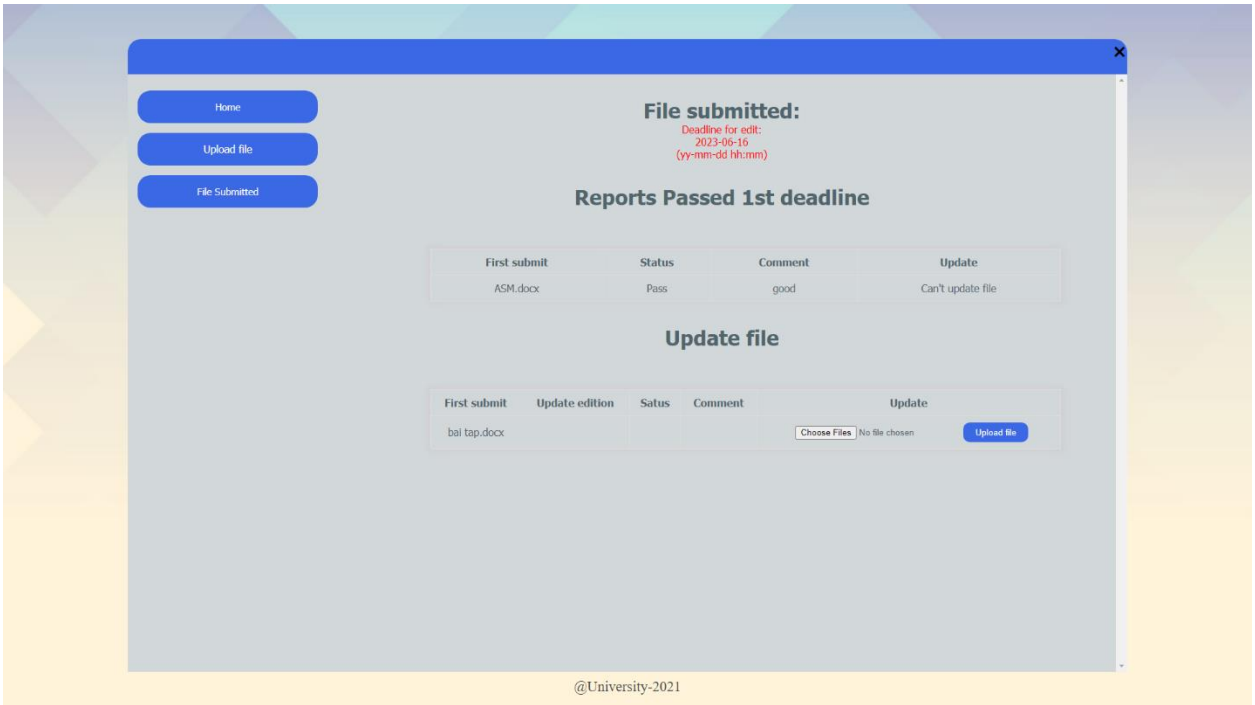


Figure 108: Display submission graded with comment

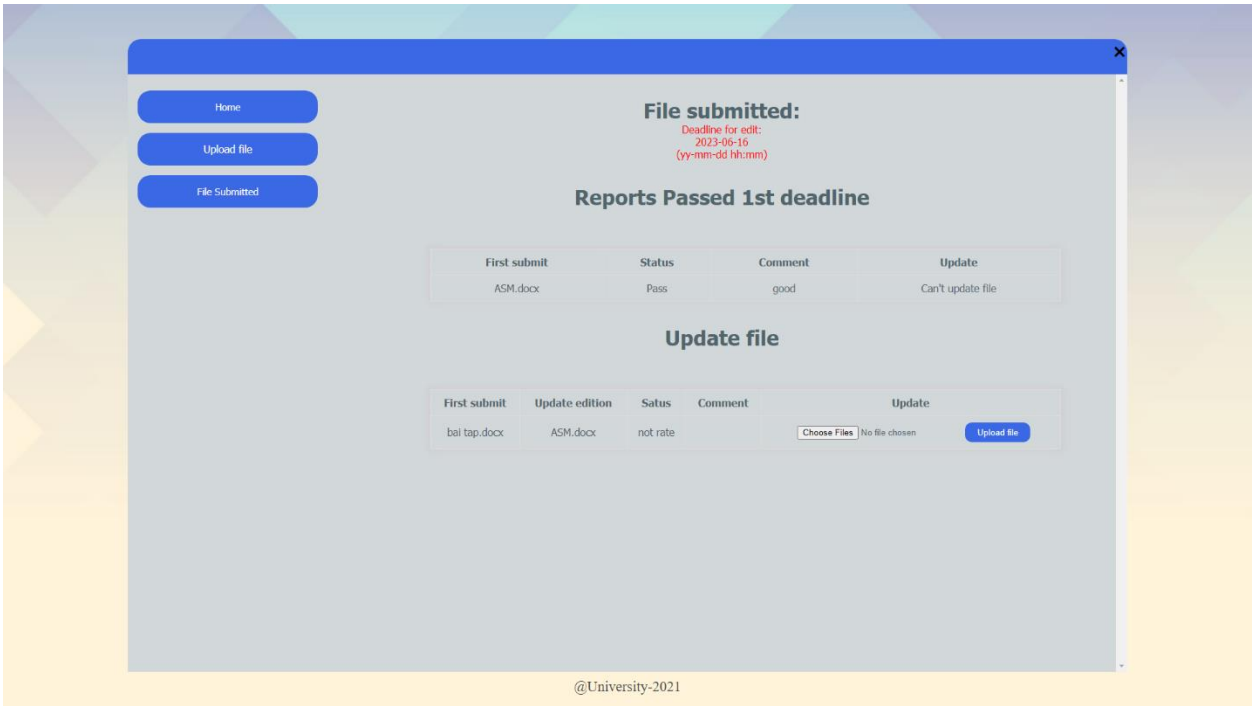


Figure 109: Update file success

l. View submission

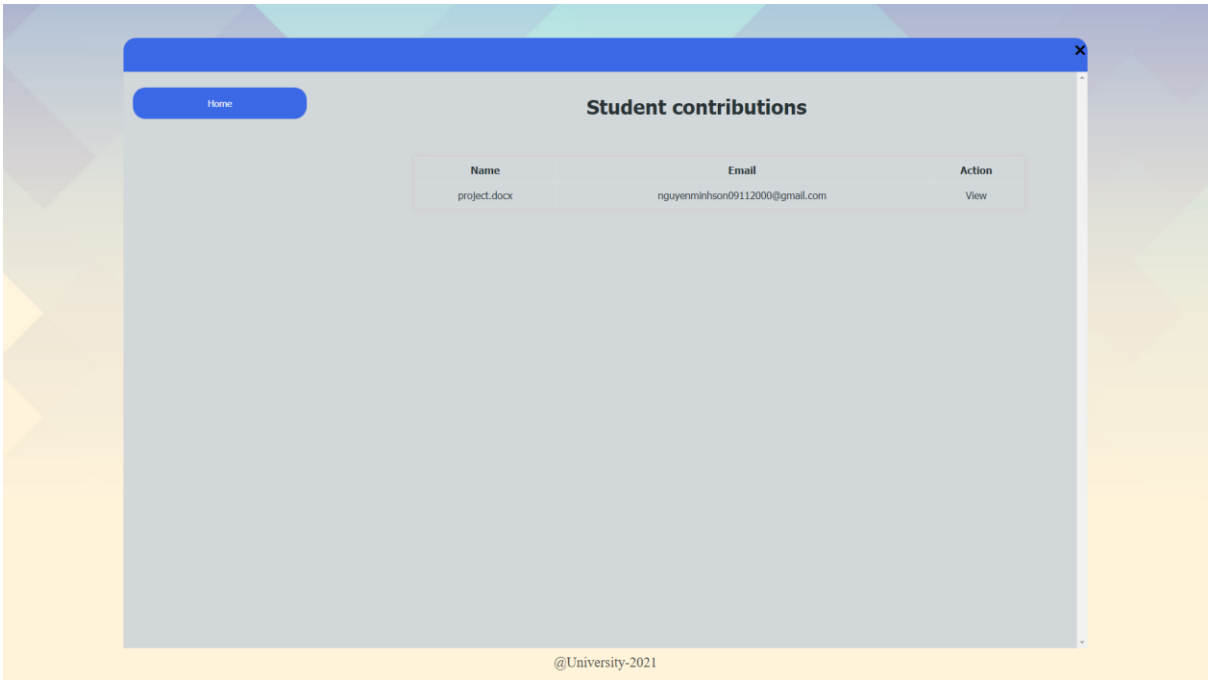


Figure 110: All submission in selected faculty

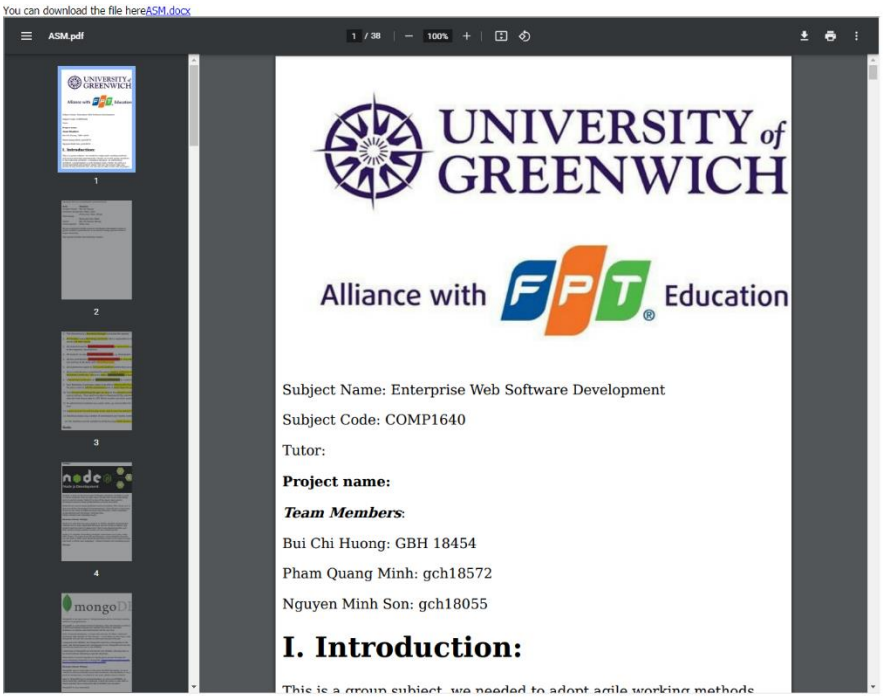


Figure 111: Submission being viewed by guest

m. Creating requirement

Figure 112 Faculty Name requirement

Figure 113 Class ID requirement

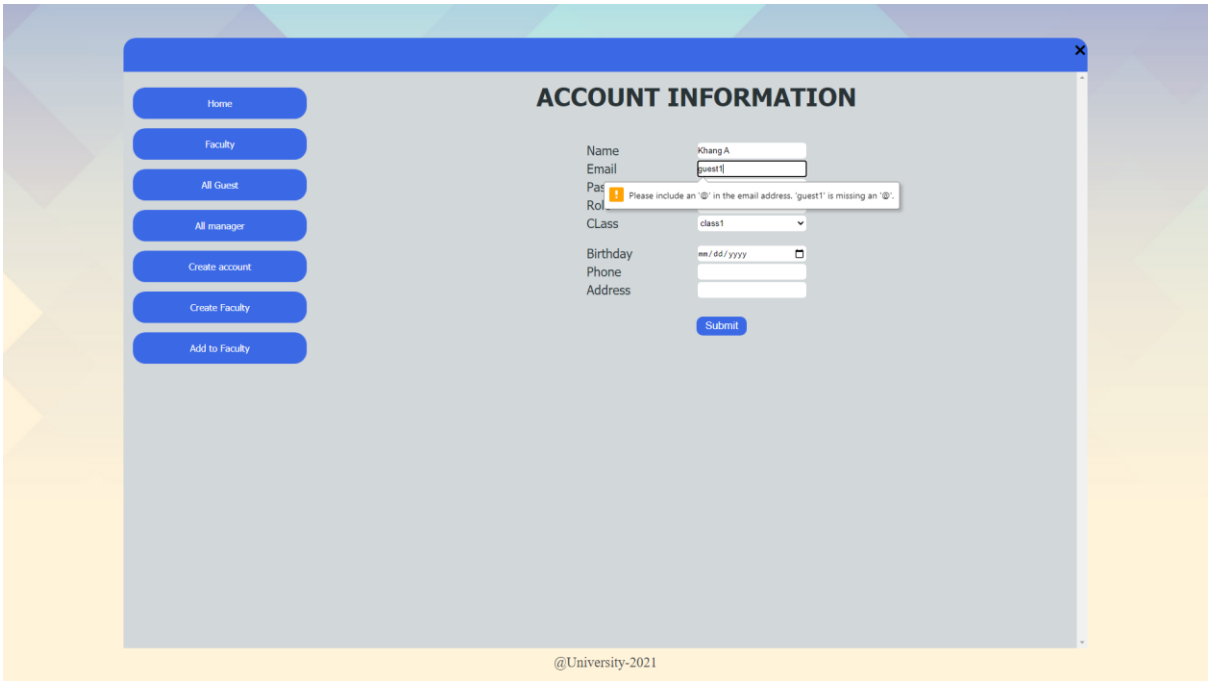


Figure 114 Email requirement

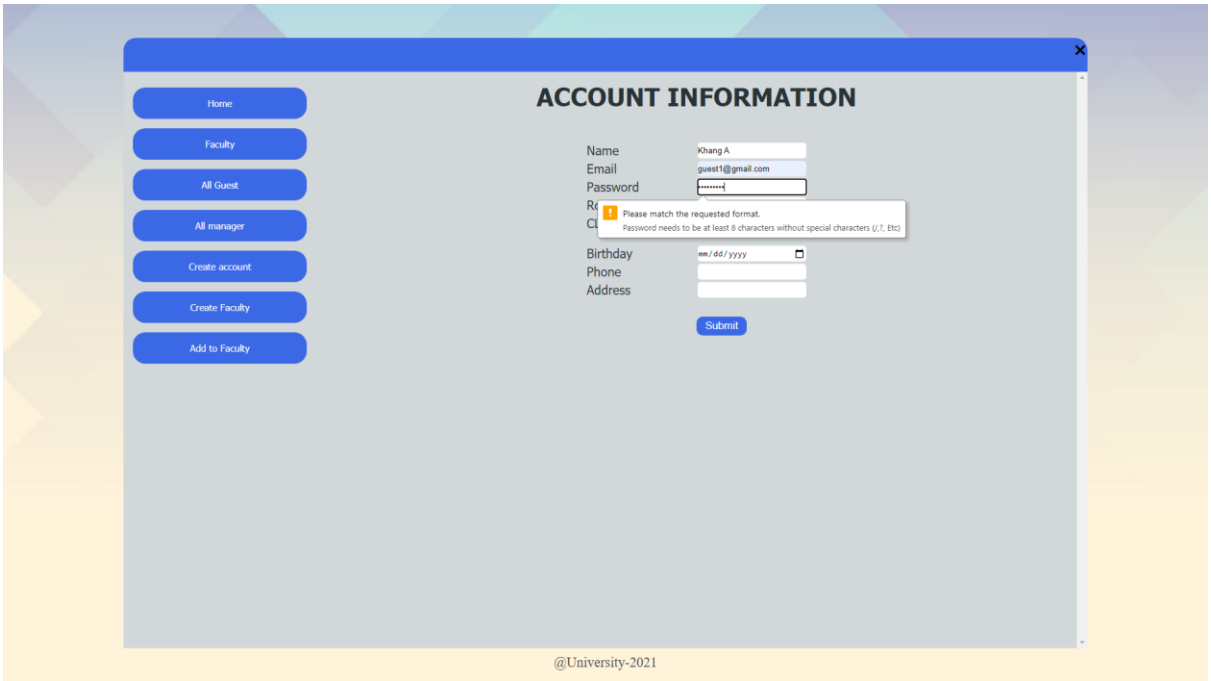


Figure 115 Password requirement

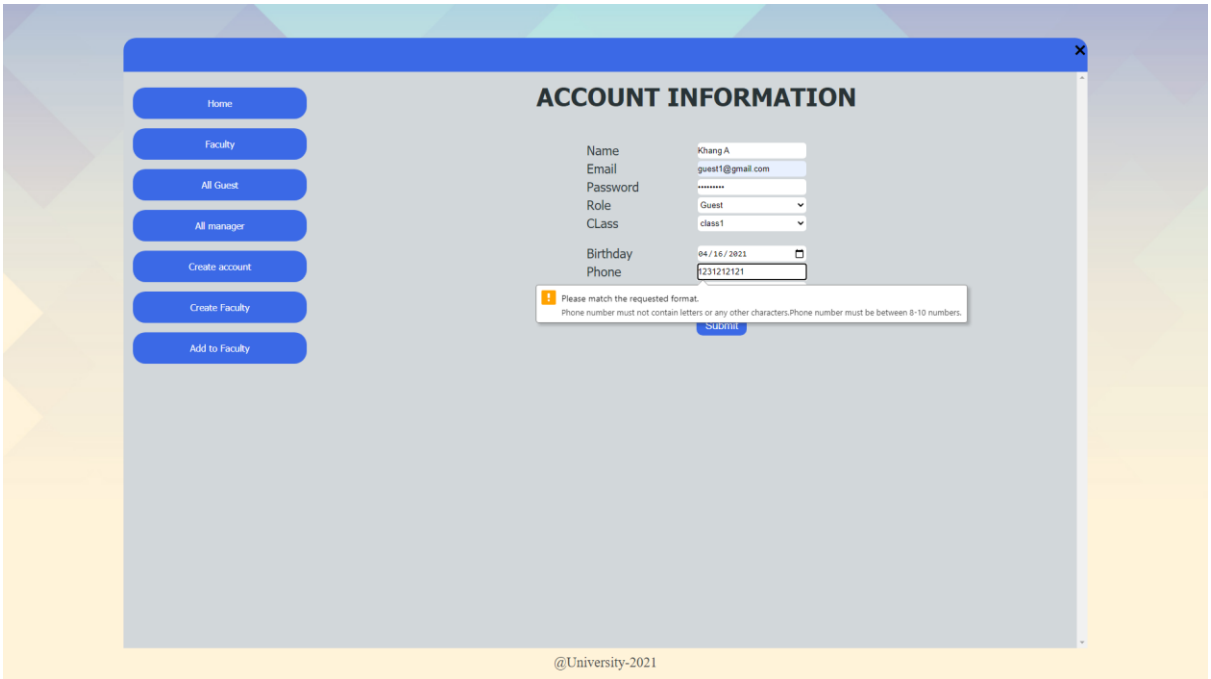


Figure 116 Number requirement

VII. Agile:

The first meeting at the start of the project was held on 21/2/2021 from 2:00 p.m. to 5:00 p.m.

The meeting was held to discuss the following issues:

- Discuss each person's competencies according to strengths and weaknesses and then assign work to roles (Scrum master, product owner, coder, design, etc.) in project development.
- Discuss and make a decision about the technology to be used for the project development.
- Analyze project requirements and evaluate how important they are to the project and how difficult it is to carry out those requirements (creating a product backlog).
- Divide the work by weeks based on how important the project is and how difficult it is to do it.

Technical:

Technical	
Back-end	Nodejs
Front-end	Html-Css-JavaScript
Database	MongoDB
Sever	Heroku

Figure 117:Technical

1. Product backlog

Story ID	As a/an	Story	priority	added in sprint	story point (10/10)
1	guest	I want to login my Account so that I can do my job	must have	Sprint 1	5
2		I want to view the selected reports.	must have	Sprint 3	4
3		I want to the interface must be suitable for all devices	could have	Sprint 4	3
4	student	I want to login my Account so that I can do my job	must have	Sprint 1	5
5		I want to submit one or more articles as Word documents to the magazine.	must have	Sprint 2	6
6		I want to upload high quality images,e.g. photographs.	must have	Sprint 2	3
7		I have to agree to Terms and Conditions before they can submit.	should have	Sprint 2	2
8		I want to the interface must be suitable for all devices	could have	Sprint 4	3
9	Marketing Coordinator	I want to login my Account so that I can do my job	must have	Sprint 1	5
10		I want to receive email notification after student posting articles to the system	could have	Sprint 2	5
11		I have to make a comment within 14 days.	should have	Sprint 2	4
12		I want to interact with students in their Faculty to edit and to select those for publication.	Want to have	Sprint 4	8
13		I want to the interface must be suitable for all devices	could have	Sprint 4	3
14	Marketing Manager	I want to login my Account so that I can do my job	must have	Sprint 1	5
15		I want to view all the selected contributions but cannot edit any	should have	Sprint 3	4
16		I want to download all the selected contributions after the final closure date in a ZIP file.	must have	Sprint 3	6
17		I want to statistical analysis number of contributions per Faculty	should have	Sprint 4	3
18		I want to the interface must be suitable for all devices	could have	Sprint 3	5
20	administrator	I want to login my Account so that I can do my job	must have	Sprint 1	5
21		CRUD faculty, Account (Guest, Student, Marketing Coordinator, Marketing Manager)	must have	Sprint 1	5
22		I want to the interface must be suitable for all devices	could have	Sprint 4	3
23		I want to set time for upload file managine	should have	Sprint 4	3

Figure 118:Product backlog

2. Sprints, burndown chart, meeting

2.1 Sprint 1 (22/2/2021- 2/3/2021)

Task name	volunteer	estimate time (hours)	Day 1 (22/02)	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9 (02/03)	Status
Create database for account	Sơn, minh	5	3	2	0							Done
Code front-end for login page	Tuấn	5	3	1	0							Done
Code backend for login and authorization functionality	Minh	8	6	4	2	0						Done
Test the login function and authorize the accounts (Guest,student, Marketing Coordinator,Marketing manager,admin).	hướng khang	4	4	3	2	0						Done
Code interface for functions: add, edit, delete accounts(Guest,student, Marketing Coordinator,Marketing manager,admin).	Tuấn	10	10	10	10	8	6	4	2	0		Done
Code backend for functions: add, edit, delete faculty and accounts(Guest,student, Marketing Coordinator,Marketing manager,admin).	Sơn, minh	13	13	13	13	13	11	10	7	3	0	Done
Test functions: add, edit, delete faculty and accounts(Guest,student, Marketing Coordinator,Marketing manager,admin).	hướng khang	6	6	6	6	6	6	4	3	2	0	Done
	Actual progress	51	45	39	33	27	23	18	12	5	0	
	Ideal progress	51	45,3	39,7	34,0	28,3	22,7	17,0	11,3	5,7	0E+00	

Figure 119: Sprint 1

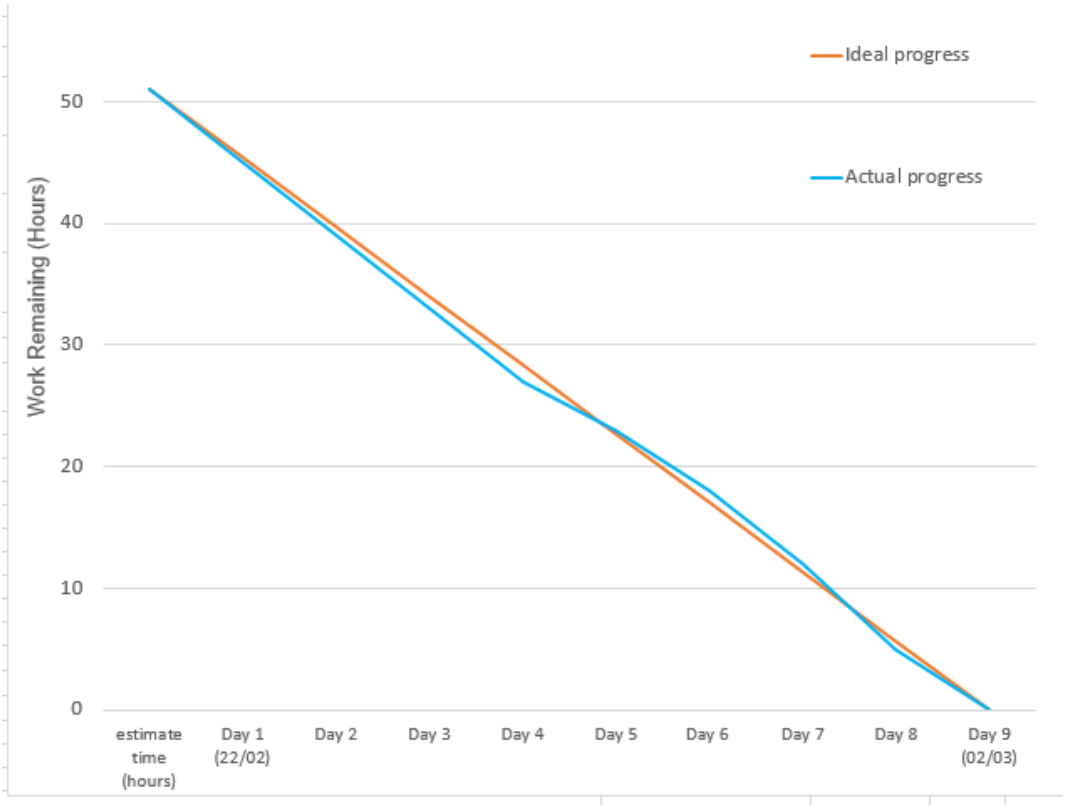


Figure 120: Burndown chart Sprint 1

The work schedule of sprint one is completed according to the set plan. Since Son and Minh are familiar with the technology, the implementation and construction of the functions took place as planned.

The first timeline of sprint 1 is on February 25, 2021:

Because Son and Minh have researched and practiced with technologies from previous courses, database manipulation and decentralization are implemented quickly. In the first half of sprint 1, we accomplished the following:

- Sơn, Minh: Database has been created to store account information.
- Tuan: The user interface has been designed for the login.
- Minh: performs the function of logging in and encrypts personal data of users
- Hướng, Khang: The decentralization test has been completed, and there is no risk.

Next work to be completed when Sprint 1 ends (March 02, 2021): Complete functions for admin:

• Add, delete, and update guest account

• Add, edit and delete a student account

• Add, delete, and update marketing coordinator account

• Add, edit and delete marketing manager account

• Add, editing and deleting faculties

The second timeline of sprint 1 is on March 02, 2021 (the end of Sprint 1):

The work has been completed

- Son, Minh: 2 coders agreed to divide the work of the back-end code for the work of implementing the functions of the admin and completed the job on time and in a timely manner, assigned to the direction and testing.
- Minh: Decentralized the accounts based on roles
- Tuan: The functions have the same interface structure, so the design doesn't take too much time; Tuan has finished designing the interface for the functions to add and delete the accounts (4 roles) and for faculty.
- Khang, Huong: During the test, the direction and direction did not detect errors in adding and deleting accounts, but there are some validate problems that need to be solved when creating accounts such as name can enter numbers, electricity phone may enter text, enter the wrong email, etc.

Remaining work: The account information validation will be a move to sprint 4.

2.2 Sprint 2 (3/3/2021- 11/3/2021)

Task name	volunteer	Estimate time (hours)	Day 1 (03/03)	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9 (11/03)	Status
code interface sends notice to students before posting.	Tuấn	3	1	0								Done
Code to check if the student agrees to the term or not.	Sơn	3	2	1	0							Done
Create database for upload file and image	Sơn,minh	6	4	2	0							Done
Code front-end for upload file and images	Tuấn	3	3	2	1	0						Done
Code back-end for upload file	Sơn,minh	5	5	4	3	1	0					Done
Test upload file (docx)	hướng khang	3	3	3	3	2	0					Done
Code back-end for upload images	Sơn,minh	5	5	5	3	1	0					Done
Test upload image	hướng khang	3	3	3	3	2	1	0				Done
Code back-end for the Marketing Coordinator receives email notifications every time a student posts on the system.	Sơn	4	4	4	4	4	4	2	0			Done
Test Marketing Coordinator receives email notifications every time a student posts on the system.	hướng khang	2	2	2	2	2	2	2	2	0		Done
Code frontend for the Marketing Coordinator reviews the student's articles	Tuấn	4	4	4	4	4	4	4	3	2	0	Done
Code backend for the Marketing Coordinator reviews the student's articles	Sơn	5	5	5	5	5	5	5	3	2	0	Done
Test function: Marketing Coordinator reviews the student's articles	hướng khang	4	4	4	4	4	4	4	4	2	0	Done
	Actual progress	50	45	39	32	25	20	17	12	6	0	
	Ideal progress	50	44,4	38,9	33,3	27,8	22,2	16,7	11,1	5,6	0	

Figure 121: Sprint 2

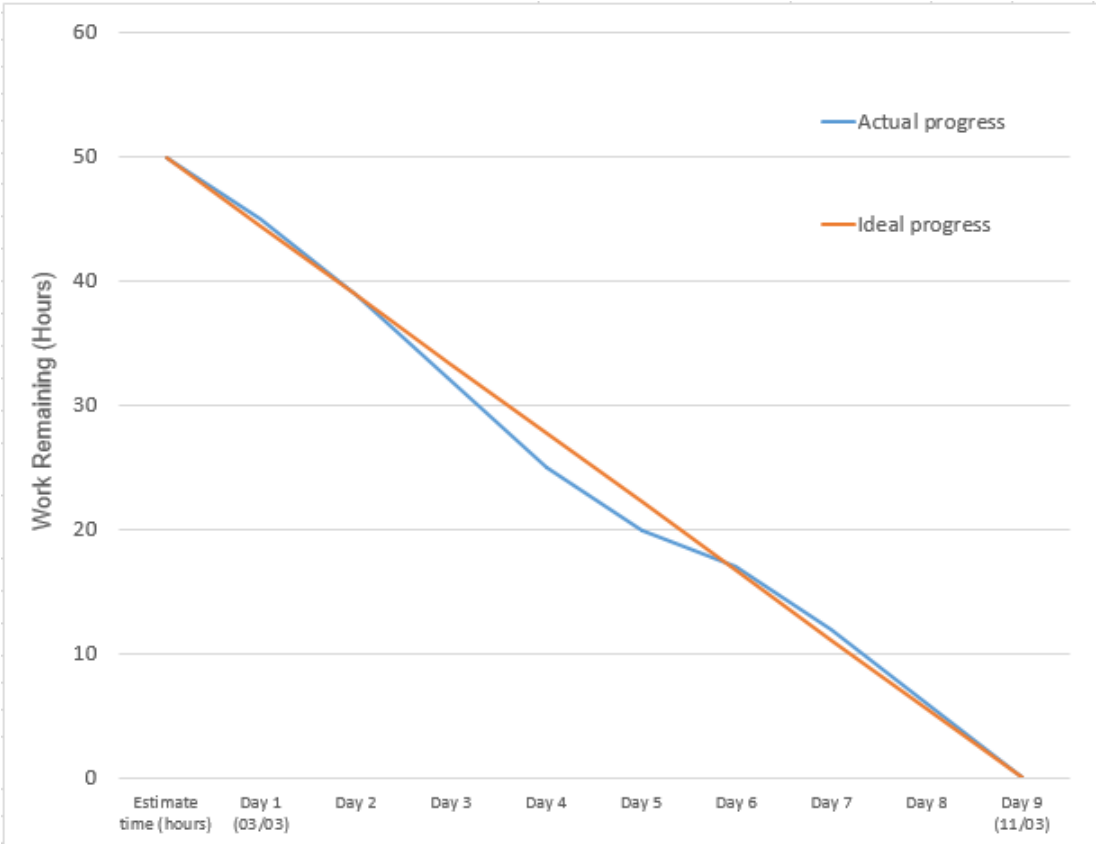


Figure 122: Burndown chart Sprint 2

In sprint 2, we also got the job done on time and were a bit over-scheduled in the midst of sprint. In the middle of sprint 2, due to work being similar in nature and functionality, the

implementation took place faster than intended. (upload file, upload image, student and marketing coordinator receive mail).

In sprint 1, the team completed the admin functions. In sprint 2, the whole team will practice building functions for students and some functions for the Marketing Coordinator. In sprint 2, we set 2 timelines for sprint 2 to monitor the situation.

The first timeline of sprint 2 is on March 6, 2021

During the first half of sprint 2, we completed the following work in the plan:

- Tuan: The interface has been designed for the display of the school's terms when students decide to post their articles on the system.
- Son: Code work completed to check if the student agrees with the term. To do this, Tuan has helped paint better understand Pop-up in the interface. From there, Sơn learned to use the <script> tag in Html to test it right at the front-end.
- Sơn, Minh: Both of them carried out to find out about the necessary attributes of the file to the database and have since completed the database creation.
- Tuan: After designing and displaying the terms, Tuan continues to design the interface for students who upload their articles to the system and have finished.

The next work needs to be completed when Sprint 2 ends (March 11, 2021): Complete functions for students and some functions of marketing coordinator.

- Student: Upload file (Docx), img, receive mail when uploading an article to the system.
- Marketing coordinator: receives an email when a student of the department posts an article on the system, evaluates the student's articles.

The second timeline of sprint 2 is on March 11, 2021 (the end of Sprint 2)

The work has been completed

- Sơn, Minh: After completing the design of the interface for uploading files and images, they painted and learned together about Nodejs libraries to save files to the system. Through many options, they decided to use the multi library and perform the file and images work successfully.
- Hướng, Khang: During the file upload test, the system crashes when there are cases such as uploading files in the incorrect format: Docx, png, jpg.

- Sơn, Minh: After Minh performed the function to save the file. Sơn learned about the library to help the system send mail and found the nodemailer library. Through the manuals, Son has completed two functions of receiving mail for teachers and students.
- Hướng, Khang: During the test of receiving mail, no errors are detected
- Tuan: After designing the page to upload files for students, Tuan has finished designing the interface to display the articles and the interface that helps the Marketing coordinator evaluate the students' articles.
- Sơn: After discussing with Tuấn, the two of you decided to display the post with the <iframe> tag in PDF format. After closing, Son has completed the back-end coding for the article evaluation.
- Hướng, Khang: Hướng and Khang have checked the article reviews and no errors have occurred.

Remaining work: None. The sprint two work has ended as planned.

2.3 Sprint 3 (12/3/2021- 20/3/2021)

Task name	volunteer	estimate time (hours)	Day 1 (12/03)	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9 (20/03)	Status
code interface for customers to read the selected articles	Tuấn	3	1	0								Done
code back-end for customers to read the selected articles	Sơn	4	1	0								Done
Test customers read the selected articles	hướng khang	2	2	0								Done
code interface for manager to read the selected articles	Tuấn	3	3	2	0							Done
code back-end for manager to read the selected articles	Sơn	4	4	3	0							Done
Test manager read the selected articles	hướng khang	2	2	2	0							Done
Code interface for the manager to choose which posts to download	Tuấn	4	4	4	4	2	0					Done
Code back-end for the manager to choose which posts to download	Sơn	6	6	6	6	3	0					Done
Test download selected posts	hướng khang	4	4	4	4	4	3	0				Done
Create database for dashboard	Sơn,minh	3	3	3	3	3	3	0				Done
Code back-end to statistic data and save to database (dashboard)	Sơn,minh	6	6	6	6	6	6	6	3	0		Done
Code interface for displaying statistical data (pie chart).	Sơn	4	4	4	4	4	4	4	2	0		Done
Test statistic data and displaying	hướng khang	3	3	3	3	3	3	3	3	1	0	Done
Code interface for the manager to set deadline for submission	Tuấn	2	2	2	2	2	2	2	2	2	0	Done
Code back-end for the manager to set deadline for submission	Sơn	3	3	3	3	3	3	3	3	3	0	Done
Test set time	hướng khang	1	1	1	1	1	1	1	1	1	0	Done
	Actual progress	54	49	43	36	31	25	19	14	7	0	Done
	Ideal progress	54	48	42	36	30	24	18	12	6	0	

Figure 123: Sprint 3

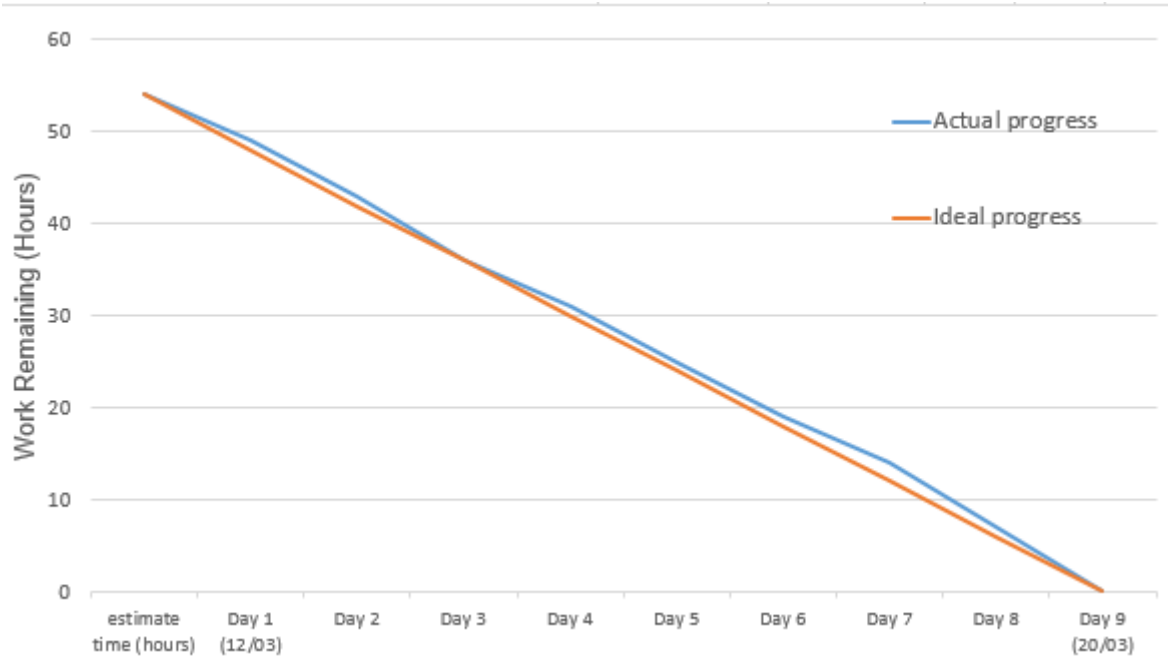


Figure 124: Burndown chart Sprint 3

In sprint 3, functionality is assessed to a moderate level (3-6). Therefore, the construction implementation is quite an accurate time estimate. In addition, the statistical viewing function is quite new but also quickly and completely explored. Therefore, the work was also completed as planned.

In Sprint 2, the student functions and Marketing coordinator have been completed. Following sprint 3, the job will be to build the Client and Marketing Manager functions.

The first timeline on March 15, 2021

In the first four days of sprint 3, the team has completed two functions of the Customer and Marketing Manager, which is the function of reading articles that have been evaluated and approved by the Marketing coordinator.

- Tuan: Create an interface for reading articles for the Guest and Marketing Manager.
- Sơn: Write back-end code to perform the process of getting post information and displaying the post on the front-end. And learn how to perform the function of downloading articles as zip files of Marketing Manager.
- Hướng, Khang: Do the test, and there is no problem during the test.

- Minh: Researching charts to perform statistical functions for Marketing Manage and created statistical charts as static data

Remaining work:

- The function of marketing manager: Set deadline for upload file, download selected article as zipping, view statistic chart.

The second timeline on March 20, 2021 (the end of Sprint 3)

The work has been completed

- Tuan: Design the interface for the Marketing Manager to choose the articles to download and the interface for the Marketing Manager to set a deadline for the student's article loading.
- Son, Minh: Created database for statistical data. Write back-end code to statistic data from the database.
- Sơn: The back-end code for the selection process, uploading students' articles to Marketing Manager and setting deadlines for students to submit. Create an interface to display statistics graphs.
- Hướng, Khang: Test the above functions, and no errors occur.

Remaining work: None. The sprint three work has ended as planned.

2.4 Sprint 4 (21/3/2021- 29/3/2021)

Task name	volunteer	estimate time (hours)	Day 1 (21/03)	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9 (29/03)	Status
Code validate data	Sơn	2	0									Done
Create database for chat box	Minh	8	5	2	0							Done
Code interface for chat box	Minh, Tuấn	8	8	6	4	2	0					Done
Code back-end for chat box	Minh	8	8	8	10	6	3	0				Done
Deplay code to sever	Sơn	3	3	3	3	3	3	0				Done
Test Role Admin in sever	Khang, Hướng	3	3	3	3	3	3	3	0			Done
Test Role Marketing Manager in sever	Khang, Hướng	3	3	3	3	3	3	3	0			Done
Test Role Marketing Coordinator in sever	Khang, Hướng	3	3	3	3	3	3	3	3	0		Done
Test Role student in sever	Khang, Hướng	3	3	3	3	3	3	3	3	0		Done
Test Role Guest in sever	Khang, Hướng	3	3	3	3	3	3	3	3	3	0	Done
Test chat box in sever	Khang, Hướng	3	3	3	3	3	3	3	3	3	0	Done
	Daily progress	47	42	37	35	29	24	18	12	6	0	
	Ideal progress	47	41,78	36,56	31,33	26,11	20,89	15,67	10,44	5,22	0	

Figure 125: Sprint 4

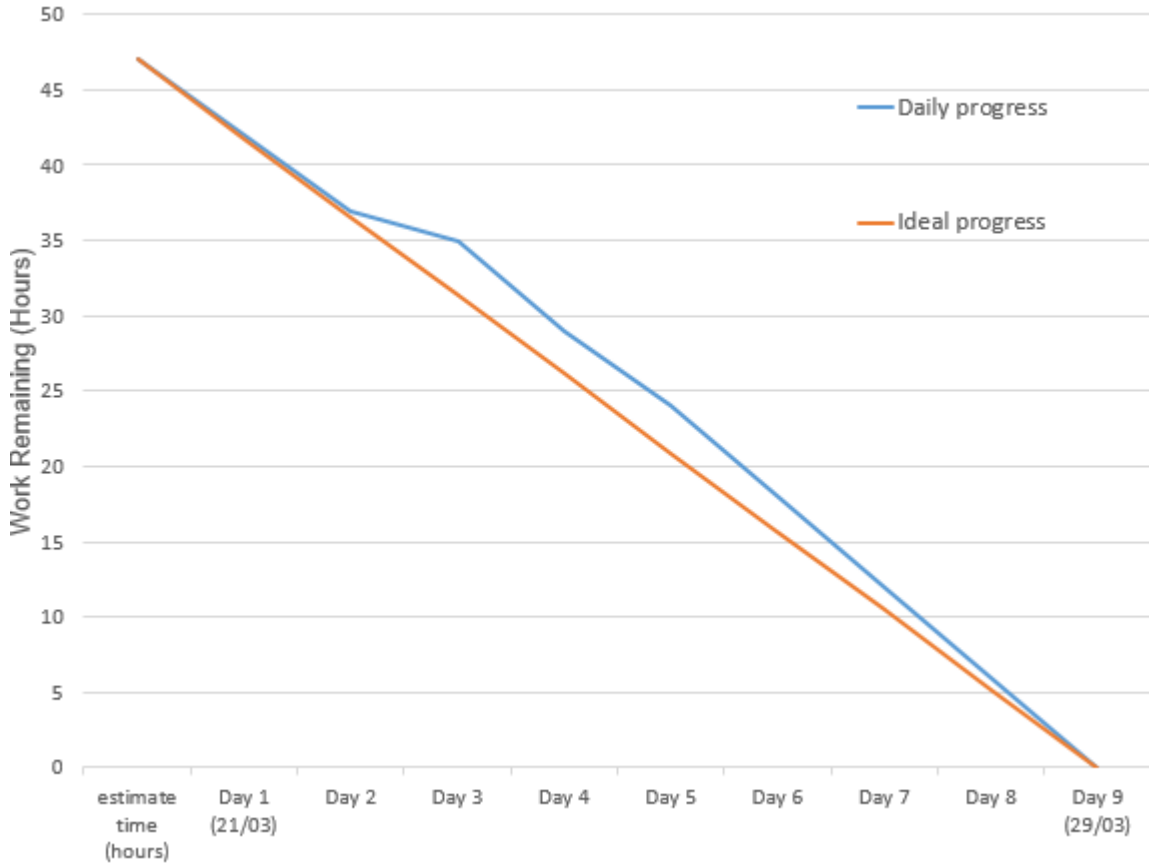


Figure 126: Burndown chart Sprint 4

At sprint 4, the implementation of building chat functions for students and the marketing coordinator took longer than expected. We have evaluated this functionality as difficult compared to the power of the coder (8/10 points), and the implementation of this function has slowed the project forward. In addition, we have to validate the account information left by sprint 1.When we finished the chat function, we quickly started pushing the project and testing it on the server, and there was no risk incurred. The project has also ended as scheduled.

After completing the functions of Guest and Marketing Manager in Sprint 3, at Sprint 4, we will proceed to handle the problems in the previous sprints and push the project to the server to progress to test the project. Run-on server.

The first timeline on March 24, 2021

During the first four days of sprint 4, we completed the following tasks:

- Sơn: Execute code completion for account information validation (work needs to be done from sprint 1)
- Minh: Designing database for student chat and Marketing Coordinator and researching back-end code for chat. Since the chat is considered to be the most difficult part, the database design takes a lot of time

Remaining work: Complete the student chat and Marketing Coordinator and test the project running on Heroku.

The second timeline on March 28, 2021

- Sơn: Pushes the project onto Heroku. During the push of the code, there were some errors such as Node version not specified in package.json, Node_modules checked into source control and promptly processed.
- Minh: Research socket io, design interfaces for chat functions between student and Marketing Coordinator.
- Tuan: edit the interface in some functions to make it reasonable
- Minh: Doing the back-end coding for the chat. He kept the chat content confidential by not allowing other users to see the content
- Khang, Hướng: Test the project running on Heroku, and the result runs as if running localhost and did not detect any new errors.

VIII. Conclusion

Through the process of studying and working, we have had experiences, accumulated new knowledge, and walked through difficulties together. There have been a lot of problems happening during the project being built, but thanks to teamwork, we bring complete software with extremely high practicality. Our program is not considered to be excellent yet, but we will improve the management program in the future and help it grow further. More suitable with the vision of life.