

# Function

## 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})
```

Đầu tiên, hệ thống sẽ cài đặt địa chỉ để lưu những file mà học sinh tải lên hệ thống và tên của chúng khi lưu tại địa chỉ đó. Sau đó, việc upload file của học sinh sẽ phải trải qua những bước kiểm tra sau với điều kiện: có thể upload only 1 file word và 1 file ảnh (optional).

Hệ thống sẽ kiểm tra số lượng file mà học sinh tải lên. Với trường hợp tải 1 file lên hệ thống, hệ thống sẽ bắt buộc đó phải là file word. Sau khi kiểm tra, nếu đó không phải là file word thì hệ thống sẽ báo lỗi và gửi thông báo. Nếu vượt qua các bước kiểm tra, hệ thống sẽ tiến hành cài đặt tên, địa chỉ lưu file và lưu những thông tin đó vào database.

- Case upload 1 file: 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('png') || x.endsWith('jpg') || x.endsWith('gif') || x.endsWith('docx')) {
```

```
      if (x.endsWith('png') || x.endsWith('jpg') || x.endsWith('gif')) {
```

```
        res.send('<script>alert("Only file formats docx, img, png, gif can be uploaded. You must upload at least 1 docx file and 1</script>");
```

```
      } else {
```

```
        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);
```

```
          }
```

```

        console.log('result'+result);

    });

    let email = req.cookies.email

    var temp = new fileModel({

        filePathdoc: xdoc,

        filePath:filePath,

        nameFile : x,

        studentemail: email,

        slug: req.cookies.slug,

    })

    temp.save((err,data)=>{

        if(err){

            console.log(err)

        }

    })

```

- Case upload 2 file: 1 docx and 1 images are acceptable

Đối với trường hợp tải 2 file lên hệ thống. Đầu tiên, hệ thống sẽ tiến hành kiểm tra các file. Nếu học sinh tải lên các file không phải là các file word (**docx**), image (**png,jpg,gif**) thì hệ thống sẽ gửi thông báo lỗi tới sinh viên kèm theo loại file có thể tải lên. Sau khi đã kiểm tra xong thể loại, hệ thống sẽ tiến hành kiểm tra xem 2 file đó có phải là 2 file có cùng là file word hay file ảnh hay không. Nếu 2 file có cùng thuộc tính thì sẽ báo lỗi. Còn không thì sẽ tiến hành cài đặt tên, địa chỉ lưu file và tiến hành lưu thông tin vào database.

```

y = req.files[1].originalname

//check type of file 1 and file 2

if(x.endsWith('png')||x.endsWith('jpg')||x.endsWith('gif')||x.endsWith('docx')||y.endsWith('png')||y.endsWith('jpg')||y.endsWith('gif')||y.endsWith('docx')){

    if(x.substr(x.length - 4,x.length) !== y.substr(y.length-4,y.length)){

        if(x.endsWith('png')||x.endsWith('jpg')||x.endsWith('gif')){

            res.send('<script>alert("Only file formats docx, img, png, gif can be uploaded. You must upload at least 1 docx file and 1 image file")</script>');

        } else{

            for(var i = 0;i<2;i++){

                if(req.files[i].originalname.endsWith('png')||req.files[i].originalname.endsWith('jpg')||req.files[i].originalname.endsWith('gif')){

                    imgpath = 'uploads/'+ req.files[i].originalname

                    console.log(imgpath)

```

```

    }

    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);

    }

    console.log('result'+result);

});

let email = req.cookies.email

var temp = new fileModel({

    filePathdoc: x,

    filePath:filePath,

    nameFile : y,

    studentemail: email,

    slug: req.cookies.slug,

    filePathAnh:imgpath,

})

temp.save((err,data)=>{

    if(err){

        console.log(err)

    }

}

```

**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})
        }
    })
}
}

```

Từ front-end, người dùng sẽ ấn vào và đọc bài viết học quan tâm đến, phía back-end sẽ lấy thông tin là **id** của bài viết đó và lấy địa chỉ của bài viết đó trong database thông qua **id** và hiển thị bài viết thông qua thẻ `<iframe>` ở phía front-end.

## Receive email notification after student posting articles to the system.

```


// set up mail sever
var transporter = nodemailer.createTransport({
  host: 'smtp.gmail.com',
  port: 465,
  secure: true,
  auth: {
    user: 'nguyenminhsonhandsome@gmail.com',
    pass: 'minhson123a'
  },
  tls: {
    rejectUnauthorized: false
  }
});

```

Đầu tiên, ta cần tiến hành cài đặt mail cho sever thông qua thư viện nodemailer. Bên cạnh đó, tài khoản mail của sever cũng cần thiết lập 1 số cài đặt trên google như là: cho phép các ứng dụng bảo mật thấp truy cập. Việc này sẽ giúp cho sever có thể truy cập vào mail này mà không bị chặn lại bởi bảo mật của google. Ngoài ra, ta cần phải bật IMAP để truy xuất thư email từ máy chủ thư qua kết nối TCP/IP.

### Less secure app access

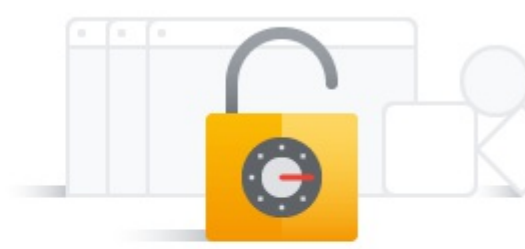
Your account is vulnerable because you allow apps and devices that use less secure sign-in technology to access your account. To keep your account secure, Google will automatically turn this setting OFF if it's not being used.

 On

[Turn off access \(recommended\)](#)

**IMAP access:**  
 (access Gmail from other clients using IMAP)  
[Learn more](#)

**Status: IMAP is enabled**  
☒ Enable IMAP  
☐ Disable IMAP



- Gửi mail cho student

```

let email = req.cookies.email
var content = 'Bạn vừa upload 1 bài báo lên hệ thống. Name: ' + x;
var mainOptions = {
  from: 'NQH-Test nodemailer',
  to: email,
  subject: 'Test Nodemailer',
  text: content
}
transporter.sendMail(mainOptions, function(err, info){
  if (err) {
    console.log(err);
  }
});

```

Để gửi mail cho sinh viên mỗi khi đăng bài lên hệ thống, đầu tiên cần phải viết nội dung mail. Sau đó tiến hành cài đặt địa chỉ gửi mail và chủ đề. Sau khi đã soạn mail xong thì sẽ tiến hành gửi mail thông qua hàm transporter.sendMail.

- Gửi mail cho giáo viên

```

let slug = req.cookies.slug
AccountModel.findOne({
  role: "teacher",
  slug: slug
},function(err, result){
  var content = email + 'vừa upload 1 bài báo lên hệ thống. Name: ' + x;
  var mainOptions2 = {
    from: 'NQH-Test nodemailer',
    to: result.email,
    subject: 'bài đăng mới',
    text: content
  }
  transporter.sendMail(mainOptions2, function(err, info){
    if (err) {
      console.log(err);
    }
  });
});

```

Sau khi sinh viên tải bài lên hệ thống, giáo viên cũng sẽ nhận được thông báo về bài đăng đó. Dựa vào mã khoa học của sinh viên (**slug**) và role của giáo viên, hệ thống sẽ tìm trong database và lấy về mail của giáo viên đó. Sau khi đã lấy được mail của giáo viên thì sẽ tiến hành soạn thư và gửi cho giáo viên.

## 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('faculty/danhgia.ejs',{data:data})
    }
    else{
      res.render('faculty/danhgia.ejs',{data:data})
    }
  })
}

```

Đầu tiên hệ thống sẽ lấy id của bài viết được chọn để đánh giá và từ **id** sẽ lấy thông tin của bài viết đó và gửi lên trang giao diện để tiến hành đánh giá. Tất cả thông tin (status,comment,etc) của file đó cũng sẽ được in ra trong trang giao diện.

Thực hiện lưu đánh giá

```

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('/faculty/allDocument/' + studentemail)
    })
  })
}

danhgiabaibao2nd(req,res){

```

Sau khi mà đọc và đưa ra các bình luận và trạng thái của bài viết (Pass or Fail), hệ thống sẽ lấy **id** và các thông tin bình luận, trạng thái bài viết và lưu lên database thông qua **id** của bài viết.

## Agree to Terms and Conditions before student submit.

Để kiểm tra xem học sinh có đồng ý với điều khoản trước khi upload file lên hệ thống hay không, hệ thống sẽ tạo 1 form pop-up để kiểm tra. Nếu học sinh không đồng ý thì gửi lên thông báo alert rằng phải đồng ý với điều khoản. Nếu học sinh đã đồng ý với điều khoản thì sẽ tiến hành kiểm tra thời gian mà học sinh được phép tải bài lên hệ thống. Đầu tiên, hệ thống sẽ lấy ngày giờ hiện tại và so sánh với ngày giờ mà Markerting Manager đã đưa ra (var **dealine**). Nếu quá giờ thì sẽ gửi thông báo rằng hết hạn. Hệ thống sẽ tự động chuyển đến trang uploadfile cho học sinh nếu vẫn còn thời hạn nộp bài.

```

<div id="myModal" class="modal">
  <!-- Modal content -->
  <div class="modal-content">
    <span class="close">&times;</span>
    <iframe src="../../uploads/dieukhoan.txt" class="dk"></iframe>
    <br>
    <p>Do you agree with the above terms?</p>
    <input type="radio" id="male" name="gender" value="Yes"> <label for="male">Yes</label>
    <input type="radio" id="female" name="gender" value="No"> <label for="female">No</label><br><br>
    <input type="submit" id="btn1" value="Next">
  </div>
</div>

```

```

document.getElementById("btn1").onclick = function ()
{
    let ts = Date.now();
    let date_ob = new Date(ts);
    let date = date_ob.getDate().toString().padStart(2, "0");
    let month = (date_ob.getMonth() + 1).toString().padStart(2, "0");
    let hour = date_ob.getHours().toString().padStart(2, "0");
    let minutes = date_ob.getMinutes().toString().padStart(2, "0");
    let year = date_ob.getFullYear();
    dl = year + "-" + month + "-" + date + " " + hour + ":" + minutes;
    var checkbox = document.getElementsByName("gender");
    for (var i = 0; i < checkbox.length; i++){
        if (checkbox[i].checked === true){
            if(checkbox[i].value === 'No'){
                alert("You must agree to continue")
            }else{
                var deadline = document.getElementById("del1").innerHTML ;
                if(dl < deadline){
                    location.href = "/file"
                }else{
                    alert("Out of date to upload file")
                }
            }
        }
    }
}
};

```

## Set time for upload file managine

```

settime(req, res, next) {
    let date = req.body.date;
    let time = req.body.time;
    let deadline1 = date + " " + time;
    var someDate = new Date(date);
    someDate.setDate(someDate.getDate() + 14);
    var dateFormatted = someDate.toISOString().substr(0, 10);
    let deadline2 = dateFormatted + " " + time;
    FacultyModel.updateMany({}, { deadline: deadline1, deadline2: deadline2 }, function (err, data) {
        if (err) {
            res.json("")
        }
        FacultyModel.find({}, function (err, data) {
            if (err) {
                res.json("")
            }
            res.jsonp({success : true})
        })
    })
}

```

Để tiến hành set deadline cho học sinh, hệ thống sẽ lấy thông tin về ngày giờ mà manager muốn cài đặt và lưu vào database. Sau khi lấy được hạn nộp (deadline 1), hệ thống sẽ tự động cộng thêm 14 ngày cho hạn sửa (deadline 2) và tiến hành lưu thông tin hạn nộp và sửa vào database. Nếu mà lưu thành công, hệ thống sẽ trả về thông báo.

Download zip:



```
var file_system = require('fs');
var archiver = require('archiver');
fileRouter.post('/abc',(req,res)=>{
  var output = file_system.createWriteStream('public/nameslug.zip');
  var archive = archiver('zip');
  var a = req.body.hobby
  output.on('close', function () {
    console.log(archive.pointer() + ' total bytes');
    console.log('archiver has been finalized and the output file descriptor has closed.');
```

Hệ thống sẽ dùng thư viện archiver để thực hiện việc nén các file docx thành zip.Đầu tiên, hệ thống sẽ tạo lên file zip: **nameslug.zip** và lưu ở thư mục **public**. Sau khi tạo file zip, hệ thống sẽ lấy các địa chỉ của bài viết qua check box ở bên front-end và tiến hành nén chúng vào trong file zip qua function archive.append() rồi tiến hành kết thúc quá trình nén tệp. Tiếp đó hệ thống sẽ chuyển đến Router mới và tiến hành tải file đó.

# Sprint Backlog & Burndown Chart

Sprint 1:

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

Sprint 2:



Task name	volunteer	estimate time (hours)	Day 1 (03/03)	Day 2	Day 3	Day 4	Day 5
code interface sends notice to students before posting.	Tuấn	3	1	0			
Code to check if the student agrees to the term or not.	Sơn	3	2	1	0		
Create database for upload file and image	Sơn,minh	6	4	2	0		
Code front-end for upload file and images	Tuấn	3	3	2	1	0	
Code back-end for upload file	Sơn,minh	5	5	4	3	1	
Test upload file (docx)	hướng khang	3	3	3	3	2	
Code back-end for upload images	Sơn,minh	5	5	5	3	1	
Test upload image	hướng khang	3	3	3	3	2	
Code back-end for the Marketing Coordinator receives email notifications every time a student posts on the system.	Sơn	4	4	4	4	4	
Test Marketing Coordinator receives email notifications every time a student posts on the system.	hướng khang	2	2	2	2	2	
Code frontend for the Marketing Coordinator reviews the student's articles	Tuấn	4	4	4	4	4	
Code backend for the Marketing Coordinator reviews the student's articles	Sơn	6	6	6	6	6	
Test function: Marketing Coordinator reviews the student's articles	hướng khang	4	4	4	4	4	
	Daily progress	51	46	40	33	26	
	Ideal progress	51	45,3	39,7	34,0	28,3	

Sprint 3:

Task name	volunteer	estimate time (hours)	Day 1 (12/03)	Day 2	Day 3	Day 4
code interface for customers to read the selected articles	Tuấn	3	1	0		
code back-end for customers to read the selected articles	Sơn	4	1	0		
Test customers read the selected articles	hướng khang	2	2	0		
code interface for manager to read the selected articles	Tuấn	3	3	2		
code back-end for manager to read the selected articles	Sơn	4	4	3		
Test manager read the selected articles	hướng khang	2	2	2		
Code interface for the manager to choose which posts to download	Tuấn	4	4	4		
Code back-end for the manager to choose which posts to download	Sơn	6	6	6		
Test download selected posts	hướng khang	4	4	4		
Create database for dashboard	Sơn,minh	3	3	3		
Code back-end to statistic data and save to database (dashboard)	Sơn,minh	6	6	6		
Code interface for displaying statistical data (pie chart).	Sơn	4	4	4		
Test statistic data and displaying	hướng khang	3	3	3		
Code interface for the manager to set deadline for submission	Tuấn	2	2	2		
Code back-end for the manager to set deadline for submission	Sơn	3	3	3		
Test set time	hướng khang	1	1	1		
	Daily progress	54	49	43		
	Ideal progress	54	48	42		

Sprint 4:

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

# Meetings

## Sprint meeting:

The first meeting at the start of the project was held on 21/2/2021 at 2:00 pm 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:

Back-end Nodejs

Front-end Mongo

Database Html-Css

Sever Heroku

## Daily Meeting:

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

We set 2 timelines for sprint 1 to monitor the situation.

The first timeline on February 25, 2021, we met together and divided the work as follows:

Bởi vì Sơn và Minh đã có nghiên cứu và thực hành với các công nghệ từ các khóa học trước nên việc thao tác với database và phân quyền được triển khai nhanh chóng. Trong nửa đầu của sprint 1, chúng tôi đã hoàn thành những công việc sau:

- Sơn , Minh: Đã tạo xong dabase để lưu trữ thông tin tài khoản.
- Tuấn: Đã thiết kế xong giao diện cho phần đăng nhập.
- Minh: Đã hoàn thành việc phân quyền cho các tài khoản dựa theo các role.
- Hương khang: Đã hoàn thành việc test phân quyền và không có rủi ro nào xảy ra.

Công việc cần hoàn thành tiếp theo khi Sprint 1 kết thúc (March 01, 2021) : Hoàn thành các chức năng cho Admin:

- Thêm sửa xóa các guest account
- Thêm sửa xóa các student account
- Thêm sửa xóa các marketing coordinator account
- Thêm sửa xóa các khoa marketing manager account

Member	Tasks	Status	Note
Son, minh	Create database for account	done	
Tuấn	Code front-end for login page	done	
Minh	Code backend for login and authorization functionality	done	
hướng khang	Test the login function and authorize the accounts (Guest,student,Marketing Coordinator,Marketing manager,admin).	done	
Tuấn	Code interface for functions: add, edit, delete accounts(Guest,student,Marketing Coordinator,Marketing manager,admin).	Not done	
Son, minh	Code backend for functions: add, edit, delete accounts(Guest,student,Marketing Coordinator,Marketing manager,admin).	Not done	
hướng khang	Test functions: add, edit, delete accounts(Guest,student, Marketing Coordinator,Marketing manager,admin).	Not done	

The second timeline on March 01, 2021 (the end of Sprint 1), we have released the results of our work during the 1st milestone of operation and report the work progress will be made in the 2nd timeline:

Son, Minh: 2 coder đã thống nhất chia công việc code back-end và đã hoàn thành công việc đúng thời gian và kịp thời giao cho hướng và khang test.

Tuấn: Các chức năng có cấu trúc giao diện tương tự nhau nên việc thiết kế cũng không mất quá nhiều thời gian, Tuấn đã hoàn thành công việc thiết kế giao diện cho các chức năng thêm sửa xóa các tài khoản ( 4 roles).

Khang, Hướng: Trong quá trình test, hướng và khang không phát hiện ra lỗi trong việc thêm sửa xóa các tài khoản nhưng có 1 số vấn đề về validate cần giải quyết khi tạo tài khoản như là: Tên có thể nhập số, điện thoại có thể nhập chữ, nhập email sai,etc.

- Note: Việc validate thông tin tài khoản sẽ được thực hiện ở sprint 4.

Member	Tasks	Status	Note
Son, minh	Create database for account	done	
Tuấn	Code front-end for login page	done	
Minh	Code backend for login and authorization functionality	done	
hướng khang	Test the login function and authorize the accounts (Guest,student,Marketing Coordinator,Marketing manager,admin).	done	
Tuấn	Code interface for functions: add, edit, delete accounts(Guest,student,Marketing Coordinator,Marketing manager,admin).	done	
Son, minh	Code backend for functions: add, edit, delete accounts(Guest,student,Marketing Coordinator,Marketing manager,admin).	done	
hướng khang	Test functions: add, edit, delete accounts(Guest,student, Marketing Coordinator,Marketing manager,admin).	done	

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

Trong sprint 1, cả đội đã hoàn thành các chức năng của admin. Trong sprint 2, cả đội sẽ tiến hành thực hành xây dựng các chức năng cho học sinh và 1 số chức năng cho Marketing Coordinator.

We set 2 timelines for sprint 2 to monitor the situation.

The first timeline on March 6, 2021, we met together and divided the work as follows:

Trong nửa đầu sprint 2, chúng tôi đã hoàn thành những công việc sau trong bản kế hoạch:

Tuấn: Đã thiết kế xong giao diện cho phần hiển thị ra các điều khoản của nhà trường khi mà học sinh quyết định đăng tải bài viết



lên hệ thống.

Son: Đã hoàn thành công việc viết code để kiểm tra xem học sinh có đồng ý với điều khoản không. Để thực hiện việc này, tuần đã giúp son hiểu rõ hơn về Pop-up ở phần giao diện. Từ đó, Son đã tìm hiểu việc sử dụng thẻ <script> ở Html để kiểm tra ngay tại front-end.

Son, Minh: Cả 2 đã cùng thực hiện tìm hiểu về những thuộc tính cần thiết của file vào database và từ đó đã hoàn thành công việc tạo database.

Tuấn: Sau khi thiết kế hiển thị điều khoản, tuần tiếp tục thiết kế giao diện cho phần học sinh tải bài viết lên hệ thống và đã hoàn thành.

Công việc cần hoàn thành tiếp theo khi Sprint 2 kết thúc (March 11, 2021) : Hoàn thành chức năng cho student và 1 số chức năng của marketing coordinator:

- Student: Upload file (docx), img, nhận mail khi đăng tải 1 bài viết lên hệ thống.
- Marketing coordinator: nhận mail khi học sinh của khoa đăng tải 1 bài viết lên hệ thống, đánh giá bài viết của học sinh.

Meeting	06-Thg3	
Member	Tasks	Status
Tuấn	code interface sends notice to students before posting.	done
Son	Code to check if the student agrees to the term or not.	done
Son,minh	Create database for upload file and image	done
Tuấn	Code front-end for upload file and images	done
Son,minh	Code back-end for upload file	Not done
hướng khang	Test upload file (docx)	Not done
Son,minh	Code back-end for upload images	Not done
hướng khang	Test upload image	Not done
Son	Code back-end for the Marketing Coordinator receives email notifications every time a student posts on the system.	Not done
hướng khang	Test Marketing Coordinator receives email notifications every time a student posts on the system.	Not done
Tuấn	Code frontend for the Marketing Coordinator reviews the student's articles	Not done
Son	Code backend for the Marketing Coordinator reviews the student's articles	Not done
hướng khang	Test function: Marketing Coordinator reviews the student's articles	Not done

The second timeline on March 11, 2021, we have released the results of our work during the 1st milestone of operation and report the work progress will be made in the 2nd timeline:

Son, Minh: Sau khi tuần thiết kế xong giao diện upload file and images, son và minh đã cùng nhau tìm hiểu về các thư viện của Nodejs để thực hiện lưu file vào hệ thống. Qua nhiều chọn lựa đã quyết định sử dụng thư viện **multer** và thực hiện công việc file và images thành công.

Khang, hướng: Trong quá trình test upload file, hệ thống có lỗi khi gặp các trường hợp như tải lên file không đúng định dạng docx,png,jpg.

Son: Sau khi thực hiện lưu file, Son đã tìm hiểu về thư viện để giúp hệ thống gửi mail và đã tìm ra thư viện **nodemailer**. Qua các hướng dẫn sử dụng, Son đã hoàn thành 2 chức năng nhận mail cho giáo viên và học sinh.

Khang, hướng: Trong quá trình test nhận mail, không có lỗi nào được phát hiện

Tuấn: Sau khi thiết kế trang upload file cho học sinh , Tuấn đã thiết kế xong giao diện để hiển thị bài viết và giao diện giúp cho Marketing coordinator đánh giá các bài viết của học sinh.

Son: Sau khi bàn luận với Tuấn, cả 2 đã quyết định hiển thị bài viết qua thẻ <iframe> với định dạng là PDF. Sau khi đã chốt, Son đã hoàn thành công việc code back-end cho phần đánh giá bài viết.

Hương khang: Hương và khang đã thực hiện kiểm tra phần đánh giá bài viết và không có lỗi nào xảy ra.

Note: Công việc sprint 2 đã kết thúc như trong kế hoạch.

Meeting	11-Thg3		
Member	Tasks	Status	Not
Tuấn	code interface sends notice to students before posting.	done	
Son	Code to check if the student agrees to the term or not.	done	
Son,minh	Create database for upload file and image	done	
Tuấn	Code front-end for upload file and images	done	
Son,minh	Code back-end for upload file	done	
hương khang	Test upload file (docx)	done	
Son,minh	Code back-end for upload images	done	
hương khang	Test upload image	done	
Son	Code back-end for the Marketing Coordinator receives email notifications every time a student posts on the system	done	
hương khang	Test Marketing Coordinator receives email notifications every time a student posts on the system.	done	
Tuấn	Code frontend for the Marketing Coordinator reviews the student's articles	done	
Son	Code backend for the Marketing Coordinator reviews the student's articles	done	
hương khang	Test function: Marketing Coordinator reviews the student's articles	done	

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

We set 2 timelines for sprint 1 to monitor the situation.

The first timeline on March 15, 2021, we met together and divided the work as follows:

Meeting	15-Thg3	
Member	Tasks	Status
Tuấn	code interface for customers to read the selected articles	done
Son	code back-end for customers to read the selected articles	done
hương khang	Test customers read the selected articles	done
Tuấn	code interface for manager to read the selected articles	done
Son	code back-end for manager to read the selected articles	done
hương khang	Test manager read the selected articles	done
Tuấn	Code interface for the manager to choose which posts to download	Not don
Son	Code back-end for the manager to choose which posts to download	Not don
hương khang	Test download selected posts	Not don
Son,minh	Create database for dashboard	Not don
Son,minh	Code back-end to statistic data and save to database (dashboard)	Not don
Son	Code interface for displaying statistical data (pie chart).	Not don
hương khang	Test statistic data and displaying	Not don
Tuấn	Code interface for the manager to set deadline for submission	Not don
Son	Code back-end for the manager to set deadline for submission	Not don
hương khang	Test set time	Not don

The second timeline on March 20, 2021, we have released the results of our work during the 1st milestone of operation and report the work progress will be made in the 2nd timeline:

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

We set 2 timelines for sprint 1 to monitor the situation.

The first timeline on March 24, 2021, we met together and divided the work as follows:

Son:

Tuấn:

Minh:

Khang:

Hương:

The second timeline on March 28, 2021, we have released the results of our work during the 1st milestone of operation and report the work progress will be made in the 2nd timeline:

Son:

Tuấn:

Minh:

Khang:

Hương: