Dharmsinh Desai University, Nadiad

Faculty of Technology

Department of Computer Engineering

B. Tech. CE Semester – V

Subject: Advanced technology

Project title: <u>Online Email System</u>

By:

1)Yash Shah, Roll no: CE141, Id: 19CEUEG051

2)Utsav Suthar, Roll no: CE150, Id: 19CEUBS025

3)Yash Vaghani, Roll no: CE167, Id: 19CEUOS078

Guided by: Prof. Prashant M. Jadav

DHARMSINH DESAI UNIVERSITY

NADIAD-387001, GUJARAT

CERTIFICATE

This is to certify that the project entitled as "**Online Email System**" is a bonafide report of the work carried out by

1) **Mr. Yash Shah**, Student ID No: **19CEUEG051**
2) **Mr. Utsav Suthar**, Student ID No: **19CEUBS025**
3) **Mr. Yash Vaghani,** Student ID No: **19CEUOS078**

of Department of Computer Engineering, semester V, under the guidance and supervision of Prof. Prashant M. Jadav for the subject Advanced Technology during the academic year 2021-2022.

Project Guide	Head of the Department
Assistance Professor	Dr. C.K. Bhensdadia
Prof. Prashant M. Jadav	Prof. & Head,
Department of Computer	Department of Computer
Engineering,	Engineering,
Faculty of Technology,	Faculty of Technology,
Dharmsinh Desai University,	Dharmsinh Desai University,
Nadiad	Nadiad
Date: 22/10/2021	

Contents:

Abstract

This system facilitates mailing among users. In the fast-growing world the information is needed as fast as possible. Quick passing of emails is not possible in manual way because an information is passed through a person. This leads to unnecessary delay in delivering information. So, we need quick and accurate system. This can be achieved by developing web based email system.

Introduction

In this fast-growing world, it is necessary that to develop quick and efficient system to achieve instant communication between a person. **Online Email System** cater the need of information sharing.

Online Email system allows the users to exchange their views thru mails and send electronic files thru attachments. It has also traditional inbox facility, from where user can see received emails. There should be another facility to see sent items.

Technologies/tools used:

- Framework Used	- Angular
- Database Application	- MongoDB
- Frontend	- Angular
- Backend	- Node.js
- Development Tool	- Visual Studio

Software Requirement Specification:

Online Email System

R.1: Login and sign up:

Description:

User have to first register him/her credential. User has unique email and password for logging in to the software. If entered email and password by user is correct then system allowed them to enter.

R.1.1: Sign up

Input: User details

Process: Validate details

Output: Redirect to email generation page

R.1.2: Email generation

Input: Enter email according to user's choice

Process: Validate email

Output: Redirect to login page

R.1.3: Login

Input: User credentials

Process: Validate credentials

Output: Redirect to home page

R.2: Mailing

Description:

In this module user can have **Compose** option that allows to compose new mail and sent it to other user. **Inbox** option shows all received messages and user can view and delete such messages. **Sent Messages** option allows us to view and delete message that have been sent. **Draft** option shows drafted mail.

R.2.1: Compose mail

Input: User selection

Output: Open editor

R.2.2: Inbox

Input: User selection

Output: List of received mails

R.2.3: Drafting mail

Input: User selection

Output: saved mails

R.2.4: Sent

Input: User selection

Output: List of sent mails

R.3: Attachment (extra tools)

Description:

In this module system will allow user to not only sending and receiving text messages, but it supports to attach file along with mail messages. User can use functionality like cc and bcc.

User can change font of text, bold the text, italic and Underline also. User can search the mail using email address, subject and substring of mail.

R.3.1: Compose with TO and FROM

Input: Attached Text with Image.

Output: Next operation

R.3.2: Styling text

Input: User selection

Output: changing text according to user selection

R.3.3: Image attachment

Input: Upload Image

Output: shown into editor

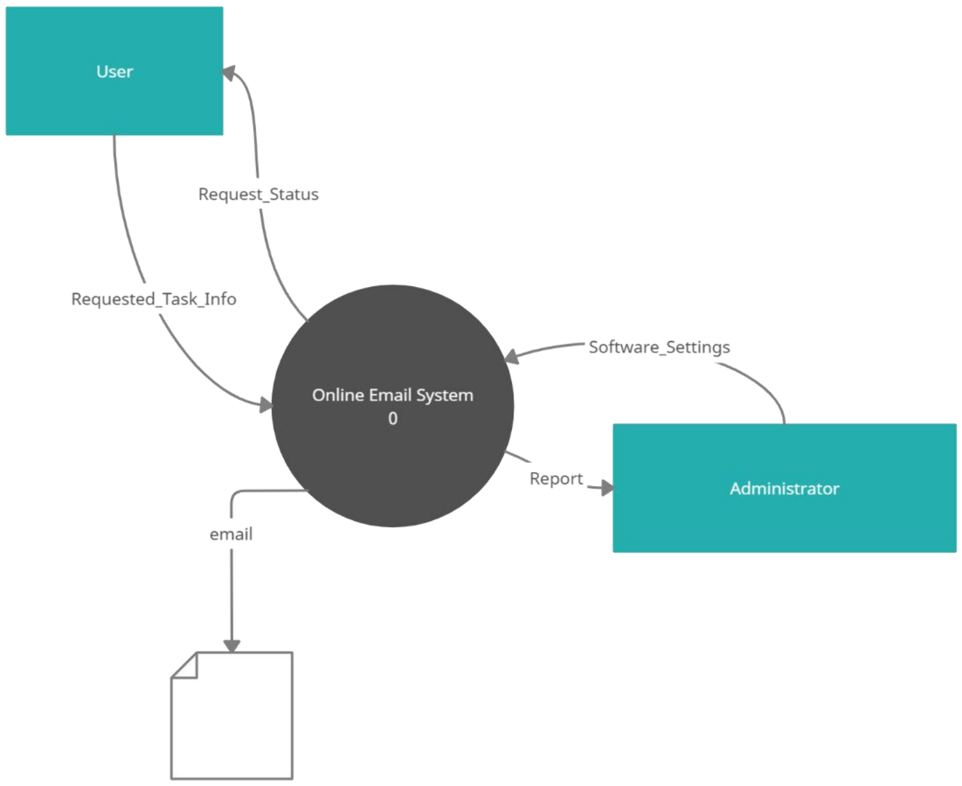
R.3.4: Searching mail

Input: enter string want to be searched

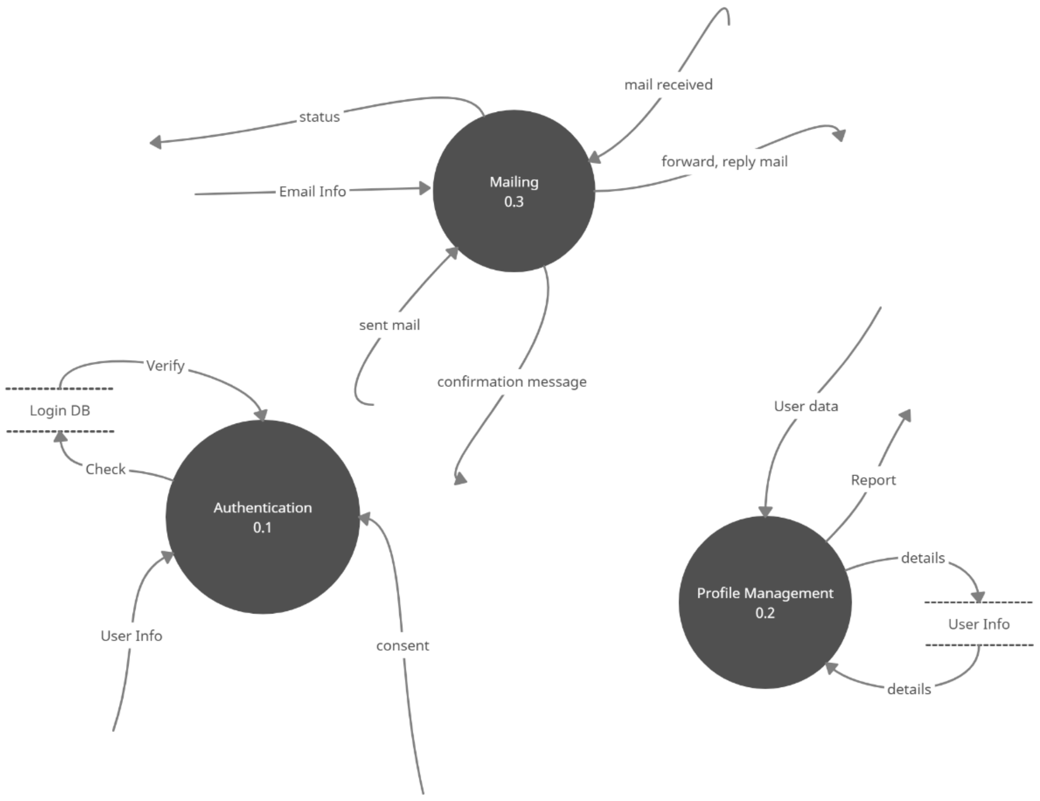
Output: list of mail according to search string

<u>**Design:**</u>

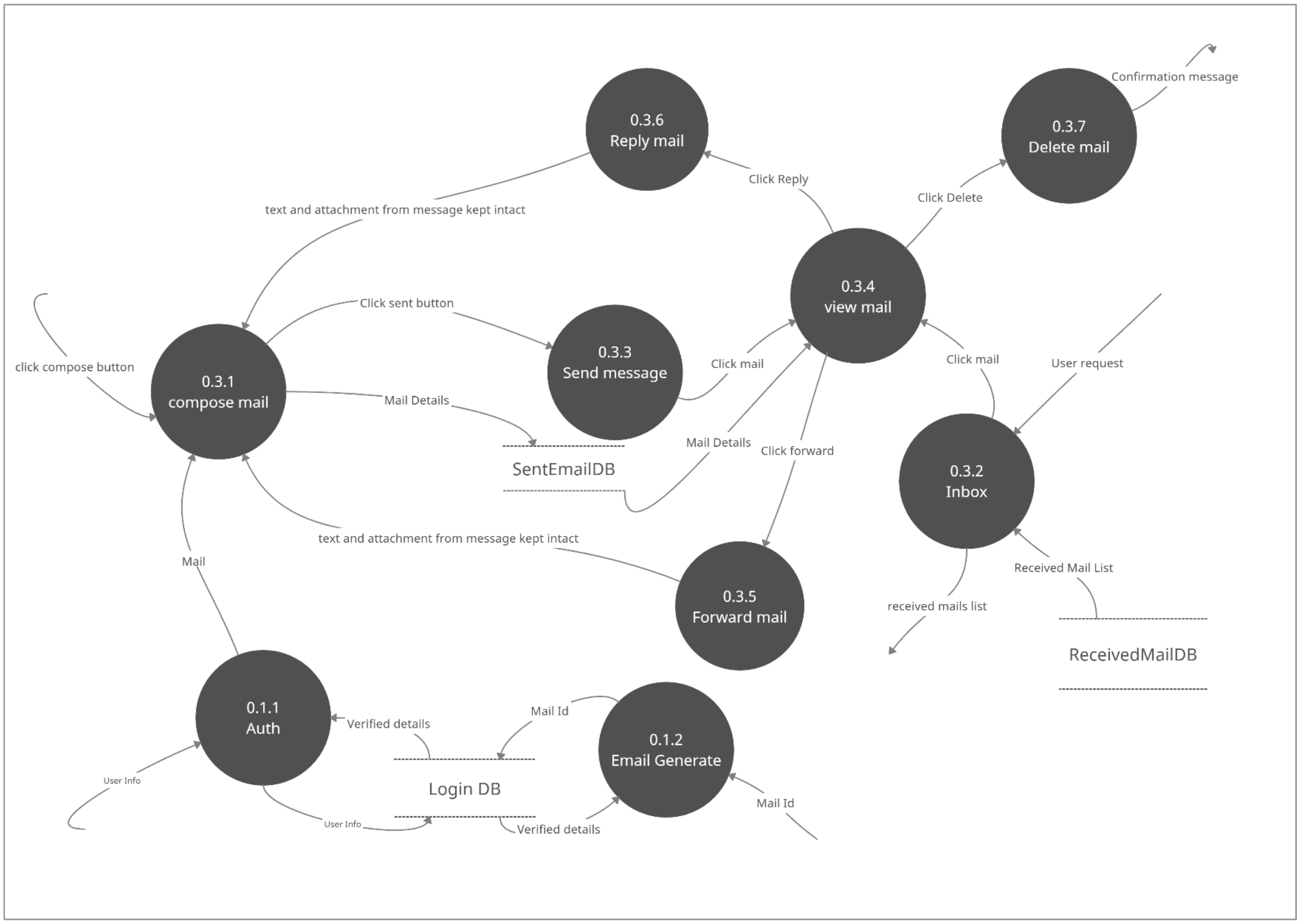
II) DFD diagram



[Level 0 diagram (context diagram)]

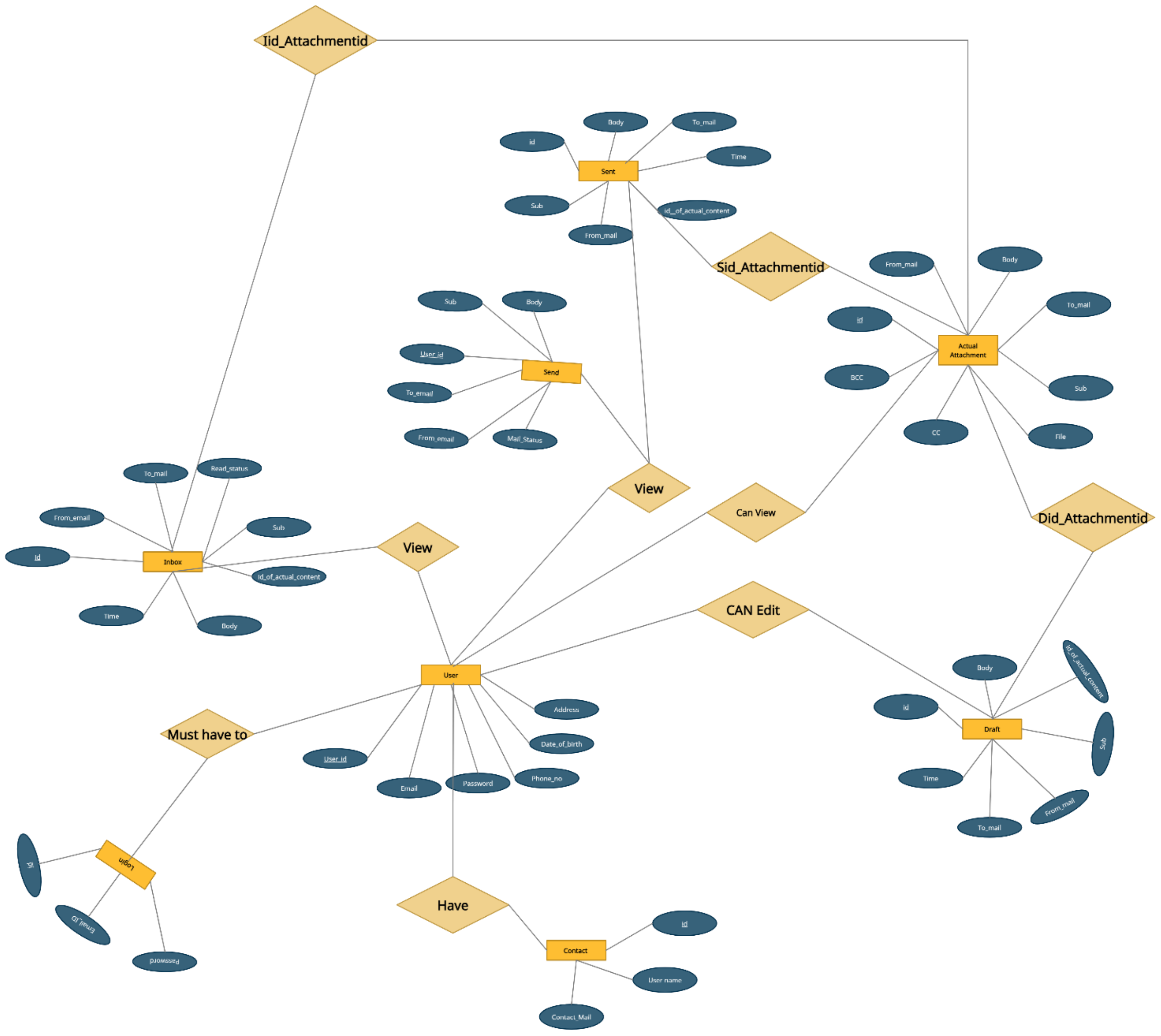


[Level 1 diagram]



[Level 2 diagram]

III) ER Diagram



Implementation Detail:

1. Modules:

In the following section a brief description of each module is given.

Login:

User have to first register him/her credential. An user has unique id and password for logging in to the software. If entered id and password by user is correct then system allowed them to enter.

Mailing:

In this module user can have **Compose** option that allows to compose new mail and sent it to another user. **Inbox** option shows all received messages and user can view and delete such messages. **Sent Messages** option allows us to view and delete message that have been sent. **Draft** option shows drafted mail.

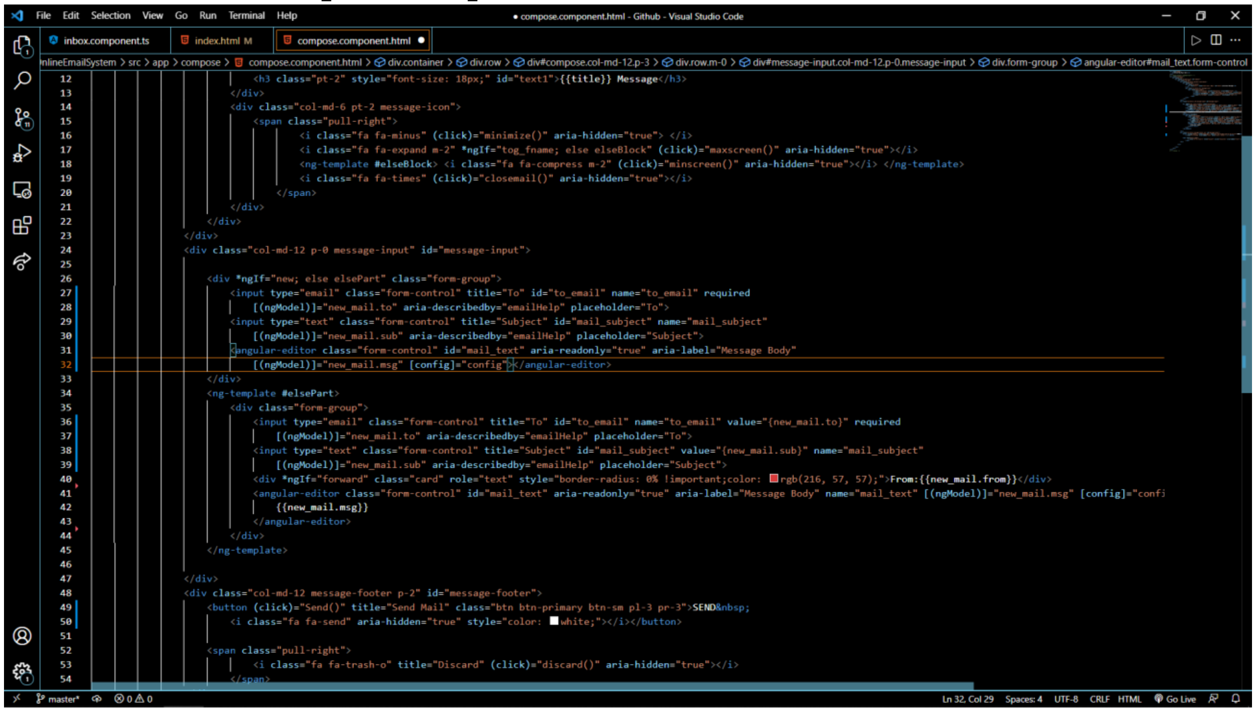
Attachments (extra tools):

In this module system will allow user to not only sending and receiving text messages, but it supports to attach image along with mail messages. User can change font of text, bold the text, italic and Underline also. User can search the mail using email address, subject and substring of mail.

2. Major Functions Prototype:

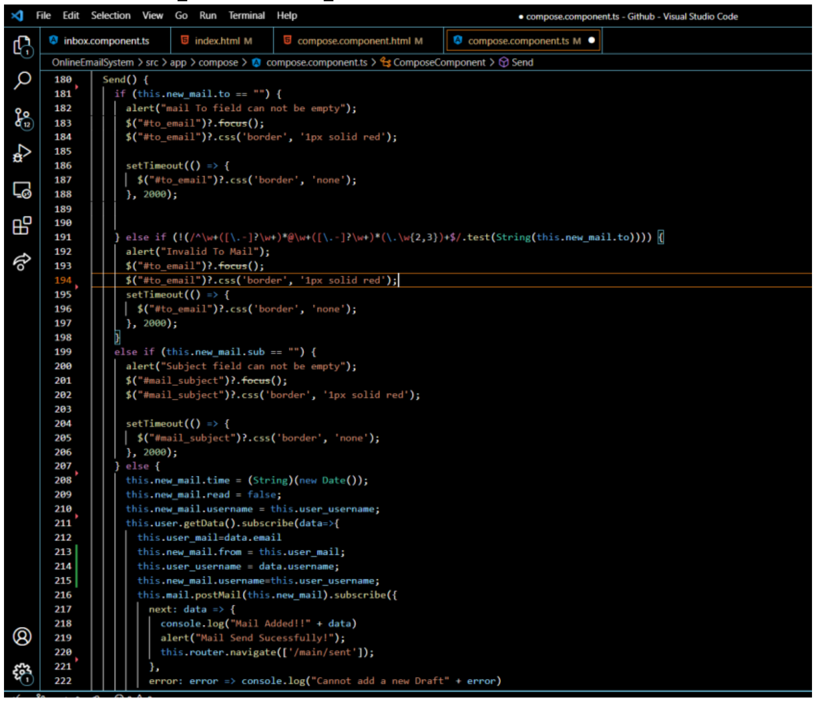
1. Compose New Mail

Compose.component.html



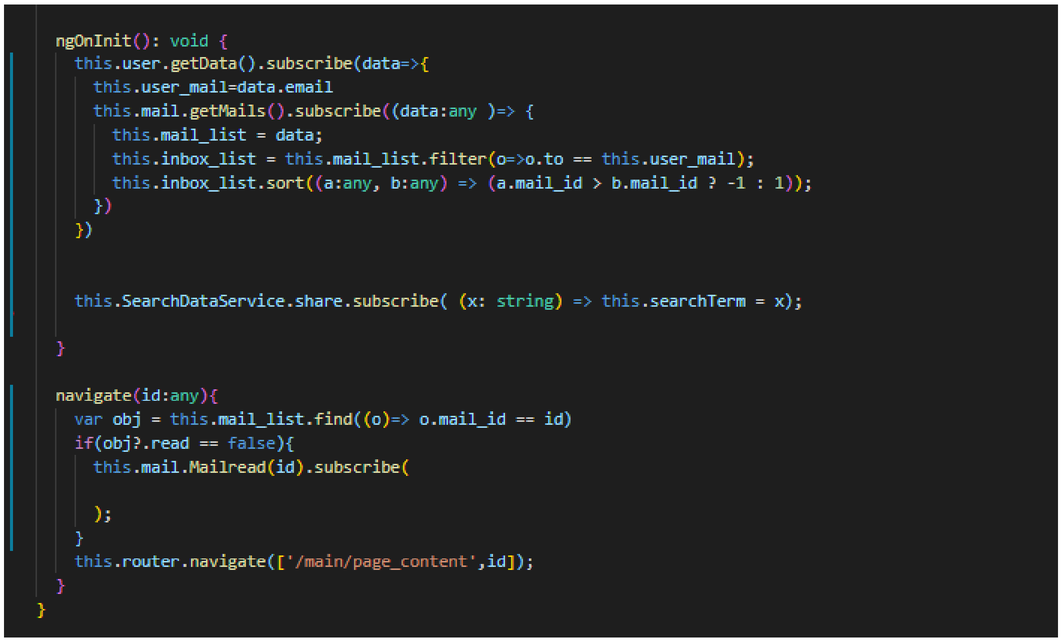
```html
                    <h3 class="pt-2" style="font-size: 18px;" id="text1">{{title}} Message</h3>
                </div>
                <div class="col-md-6 pt-2 message-icon">
                    <span class="pull-right">
                        <i class="fa fa-minus" (click)="minimize()" aria-hidden="true"> </i>
                        <i class="fa fa-expand m-2" *ngIf="tog_fname; else elseBlock" (click)="maxscreen()" aria-hidden="true"></i>
                        <ng-template #elseBlock> <i class="fa fa-compress m-2" (click)="minscreen()" aria-hidden="true"></i> </ng-template>
                        <i class="fa fa-times" (click)="closemail()" aria-hidden="true"></i>
                    </span>
                </div>
            </div>
        </div>
        <div class="col-md-12 p-0 message-input" id="message-input">

            <div *ngIf="new; else elsePart" class="form-group">
                <input type="email" class="form-control" title="To" id="to_email" name="to_email" required
                    [(ngModel)]="new_mail.to" aria-describedby="emailHelp" placeholder="To">
                <input type="text" class="form-control" title="Subject" id="mail_subject" name="mail_subject"
                    [(ngModel)]="new_mail.sub" aria-describedby="emailHelp" placeholder="Subject">
                <angular-editor class="form-control" id="mail_text" aria-readonly="true" aria-label="Message Body"
                    [(ngModel)]="new_mail.msg" [config]="config"></angular-editor>
            </div>
            <ng-template #elsePart>
                <div class="form-group">
                    <input type="email" class="form-control" title="To" id="to_email" name="to_email" value="{new_mail.to}" required
                        [(ngModel)]="new_mail.to" aria-describedby="emailHelp" placeholder="To">
                    <input type="text" class="form-control" title="Subject" id="mail_subject" value="{new_mail.sub}" name="mail_subject"
                        [(ngModel)]="new_mail.sub" aria-describedby="emailHelp" placeholder="Subject">
                    <div *ngIf="forward" class="card" role="text" style="border-radius: 0% !important;color: rgb(216, 57, 57);">From:{{new_mail.from}}</div>
                    <angular-editor class="form-control" id="mail_text" aria-readonly="true" aria-label="Message Body" name="mail_text" [(ngModel)]="new_mail.msg" [config]="confi
                        {{new_mail.msg}}
                    </angular-editor>
                </div>
            </ng-template>

        </div>
        <div class="col-md-12 message-footer p-2" id="message-footer">
            <button (click)="Send()" title="Send Mail" class="btn btn-primary btn-sm pl-3 pr-3">SEND&nbsp;
                <i class="fa fa-send" aria-hidden="true" style="color: white;"></i></button>

            <span class="pull-right">
                <i class="fa fa-trash-o" title="Discard" (click)="discard()" aria-hidden="true"></i>
            </span>
```

Compose.component.ts



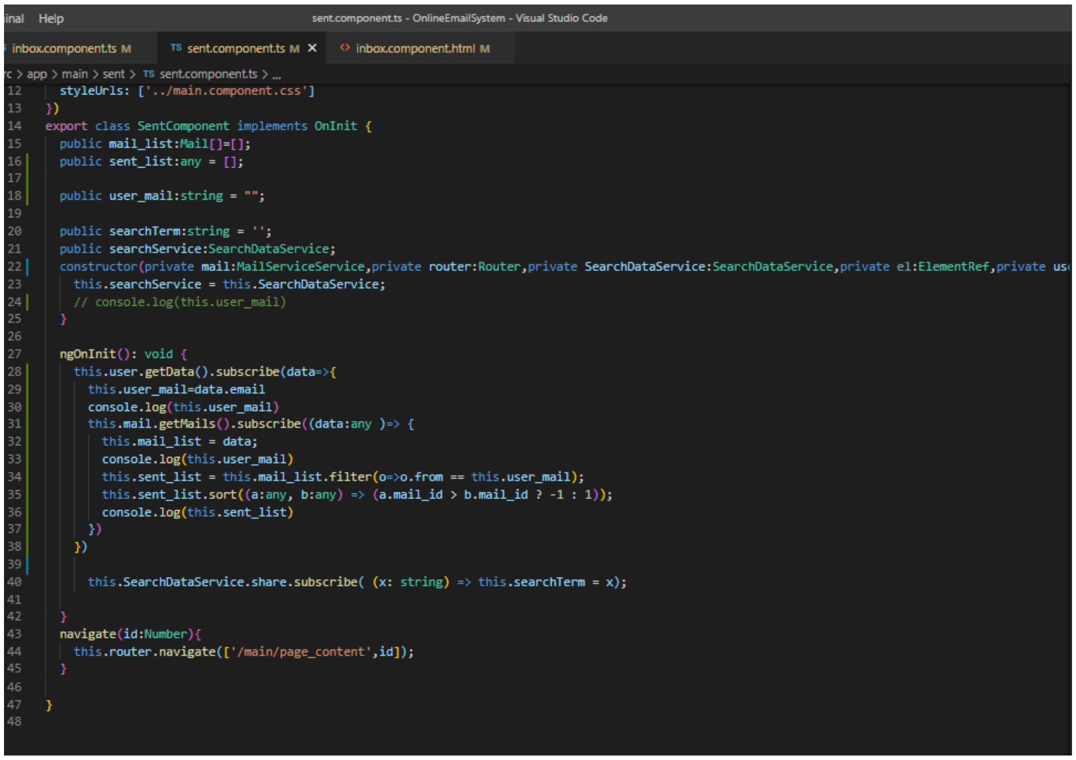
```typescript
  Send() {
    if (this.new_mail.to == "") {
      alert("mail To field can not be empty");
      $("#to_email")?.focus();
      $("#to_email")?.css('border', '1px solid red');

      setTimeout(() => {
        $("#to_email")?.css('border', 'none');
      }, 2000);


    } else if (!(/^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$/.test(String(this.new_mail.to)))) {
      alert("Invalid To Mail");
      $("#to_email")?.focus();
      $("#to_email")?.css('border', '1px solid red');
      setTimeout(() => {
        $("#to_email")?.css('border', 'none');
      }, 2000);
    }
    else if (this.new_mail.sub == "") {
      alert("Subject field can not be empty");
      $("#mail_subject")?.focus();
      $("#mail_subject")?.css('border', '1px solid red');

      setTimeout(() => {
        $("#mail_subject")?.css('border', 'none');
      }, 2000);
    } else {
      this.new_mail.time = (String)(new Date());
      this.new_mail.read = false;
      this.new_mail.username = this.user_username;
      this.user.getData().subscribe(data=>{
        this.user_mail=data.email
        this.new_mail.from = this.user_mail;
        this.user_username = data.username;
        this.new_mail.username=this.user_username;
        this.mail.postMail(this.new_mail).subscribe({
          next: data => {
            console.log("Mail Added!!" + data)
            alert("Mail Send Sucessfully!");
            this.router.navigate(['/main/sent']);
          },
          error: error => console.log("Cannot add a new Draft" + error)
```

2. Inbox Mails



```
  ngOnInit(): void {
    this.user.getData().subscribe(data=>{
      this.user_mail=data.email
      this.mail.getMails().subscribe((data:any )=> {
        this.mail_list = data;
        this.inbox_list = this.mail_list.filter(o=>o.to == this.user_mail);
        this.inbox_list.sort((a:any, b:any) => (a.mail_id > b.mail_id ? -1 : 1));
      })
    })


    this.SearchDataService.share.subscribe( (x: string) => this.searchTerm = x);

  }

  navigate(id:any){
    var obj = this.mail_list.find((o)=> o.mail_id == id)
    if(obj?.read == false){
      this.mail.Mailread(id).subscribe(

      );
    }
    this.router.navigate(['/main/page_content',id]);
  }
}
```

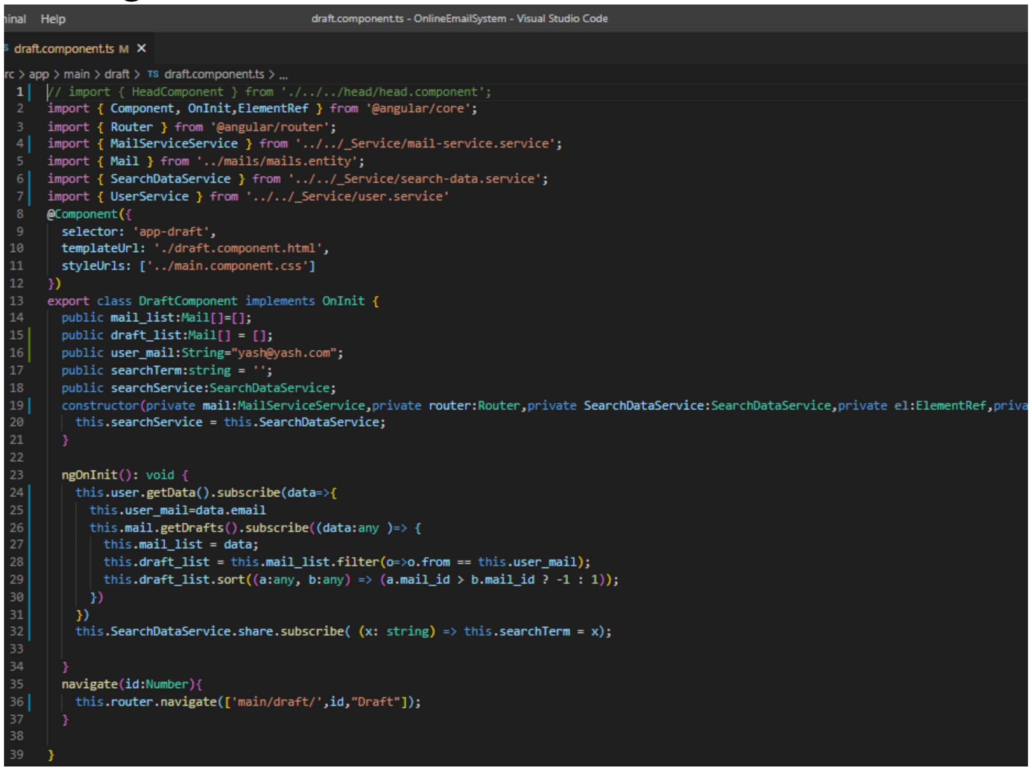
3. Sent Mails



inal Help
sent.component.ts - OnlineEmailSystem - Visual Studio Code
inbox.component.ts M
TS sent.component.ts M ×
inbox.component.html M
rc > app > main > sent > TS sent.component.ts > ...

```
  styleUrls: ['../main.component.css']
})
export class SentComponent implements OnInit {
  public mail_list:Mail[]=[];
  public sent_list:any = [];

  public user_mail:string = "";

  public searchTerm:string = '';
  public searchService:SearchDataService;
  constructor(private mail:MailServiceService,private router:Router,private SearchDataService:SearchDataService,private el:ElementRef,private us
    this.searchService = this.SearchDataService;
    // console.log(this.user_mail)
  }

  ngOnInit(): void {
    this.user.getData().subscribe(data=>{
      this.user_mail=data.email
      console.log(this.user_mail)
      this.mail.getMails().subscribe((data:any )=> {
        this.mail_list = data;
        console.log(this.user_mail)
        this.sent_list = this.mail_list.filter(o=>o.from == this.user_mail);
        this.sent_list.sort((a:any, b:any) => (a.mail_id > b.mail_id ? -1 : 1));
        console.log(this.sent_list)
      })
    })

      this.SearchDataService.share.subscribe( (x: string) => this.searchTerm = x);

  }
  navigate(id:Number){
    this.router.navigate(['/main/page_content',id]);
  }

}
```

4. Drafting Mails



ninal Help draft.component.ts - OnlineEmailSystem - Visual Studio Code

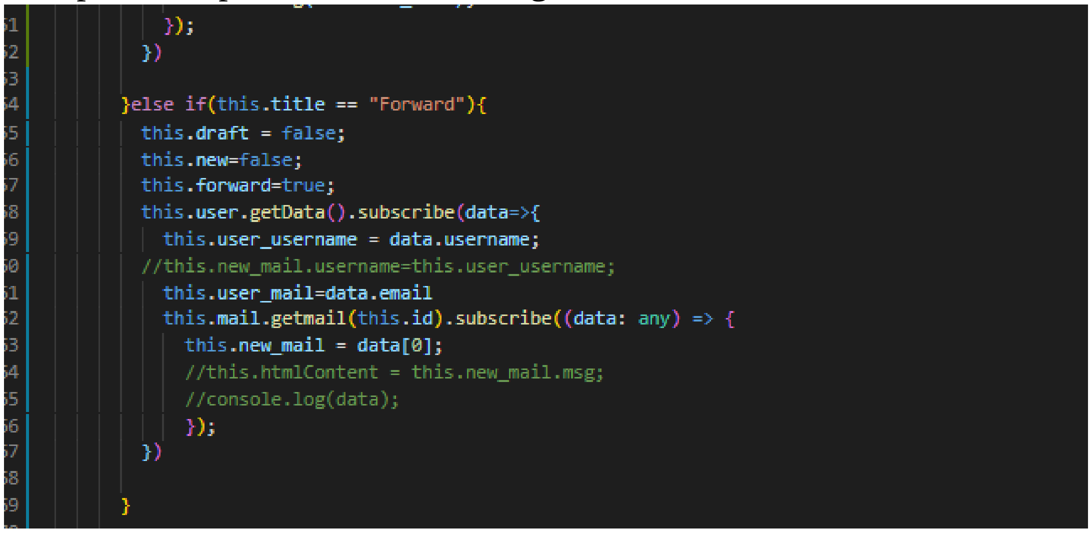
draft.component.ts M ×

rc > app > main > draft > TS draft.component.ts > ...

```
// import { HeadComponent } from './../../head/head.component';
import { Component, OnInit,ElementRef } from '@angular/core';
import { Router } from '@angular/router';
import { MailServiceService } from '../../_Service/mail-service.service';
import { Mail } from '../mails/mails.entity';
import { SearchDataService } from '../../_Service/search-data.service';
import { UserService } from '../../_Service/user.service'
@Component({
  selector: 'app-draft',
  templateUrl: './draft.component.html',
  styleUrls: ['../main.component.css']
})
export class DraftComponent implements OnInit {
  public mail_list:Mail[]=[];
  public draft_list:Mail[] = [];
  public user_mail:String="yash@yash.com";
  public searchTerm:string = '';
  public searchService:SearchDataService;
  constructor(private mail:MailServiceService,private router:Router,private SearchDataService:SearchDataService,private el:ElementRef,priva
    this.searchService = this.SearchDataService;
  }

  ngOnInit(): void {
    this.user.getData().subscribe(data=>{
      this.user_mail=data.email
      this.mail.getDrafts().subscribe((data:any )=> {
        this.mail_list = data;
        this.draft_list = this.mail_list.filter(o=>o.from == this.user_mail);
        this.draft_list.sort((a:any, b:any) => (a.mail_id > b.mail_id ? -1 : 1));
      })
    })
    this.SearchDataService.share.subscribe( (x: string) => this.searchTerm = x);

  }
  navigate(id:Number){
    this.router.navigate(['main/draft/',id,"Draft"]);
  }

}
```

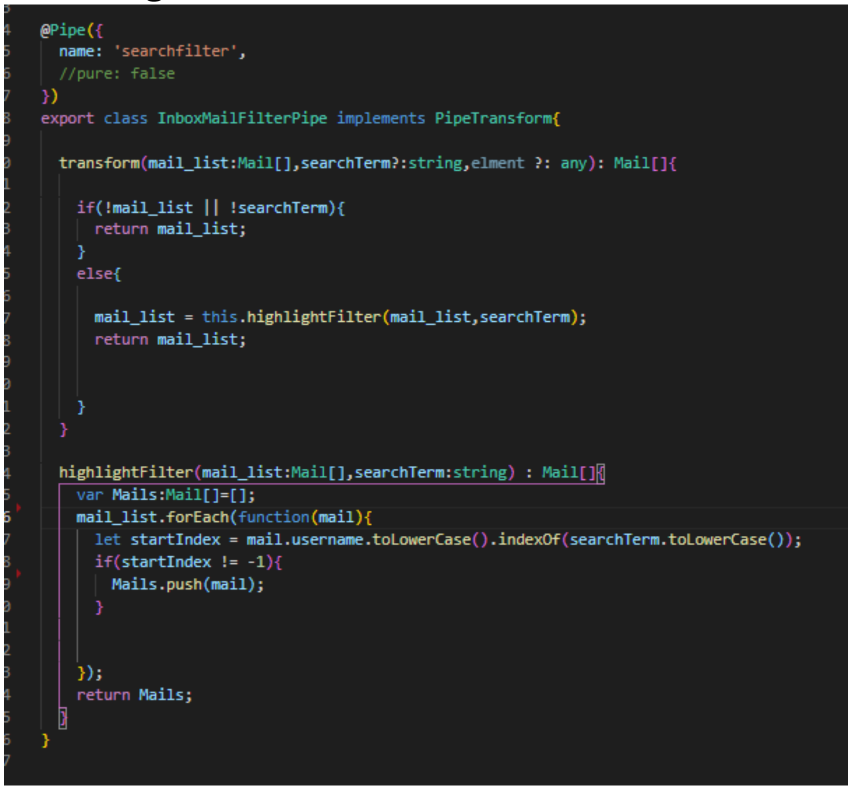
5. Forwarding Mails

Compose component for forwarding mail



```
            });
          })

        }else if(this.title == "Forward"){
          this.draft = false;
          this.new=false;
          this.forward=true;
          this.user.getData().subscribe(data=>{
            this.user_username = data.username;
          //this.new_mail.username=this.user_username;
            this.user_mail=data.email
            this.mail.getmail(this.id).subscribe((data: any) => {
              this.new_mail = data[0];
              //this.htmlContent = this.new_mail.msg;
              //console.log(data);
            });
          })

        }
```

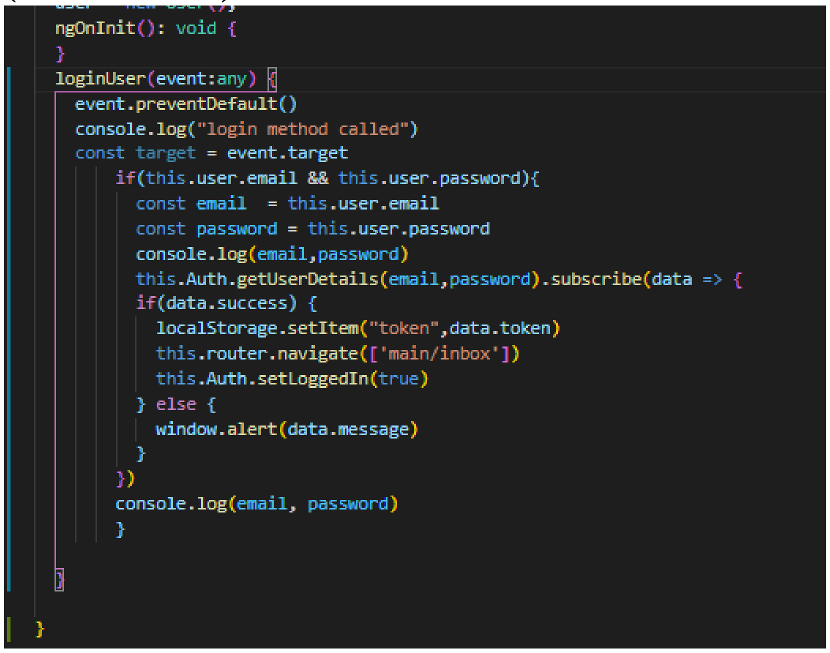
6. Searching



```
@Pipe({
  name: 'searchfilter',
  //pure: false
})
export class InboxMailFilterPipe implements PipeTransform{

  transform(mail_list:Mail[],searchTerm?:string,elment ?: any): Mail[]{

    if(!mail_list || !searchTerm){
      return mail_list;
    }
    else{

      mail_list = this.highlightFilter(mail_list,searchTerm);
      return mail_list;


    }
  }

  highlightFilter(mail_list:Mail[],searchTerm:string) : Mail[]{
    var Mails:Mail[]=[];
    mail_list.forEach(function(mail){
      let startIndex = mail.username.toLowerCase().indexOf(searchTerm.toLowerCase());
      if(startIndex != -1){
        Mails.push(mail);
      }


    });
    return Mails;
  }
}
```

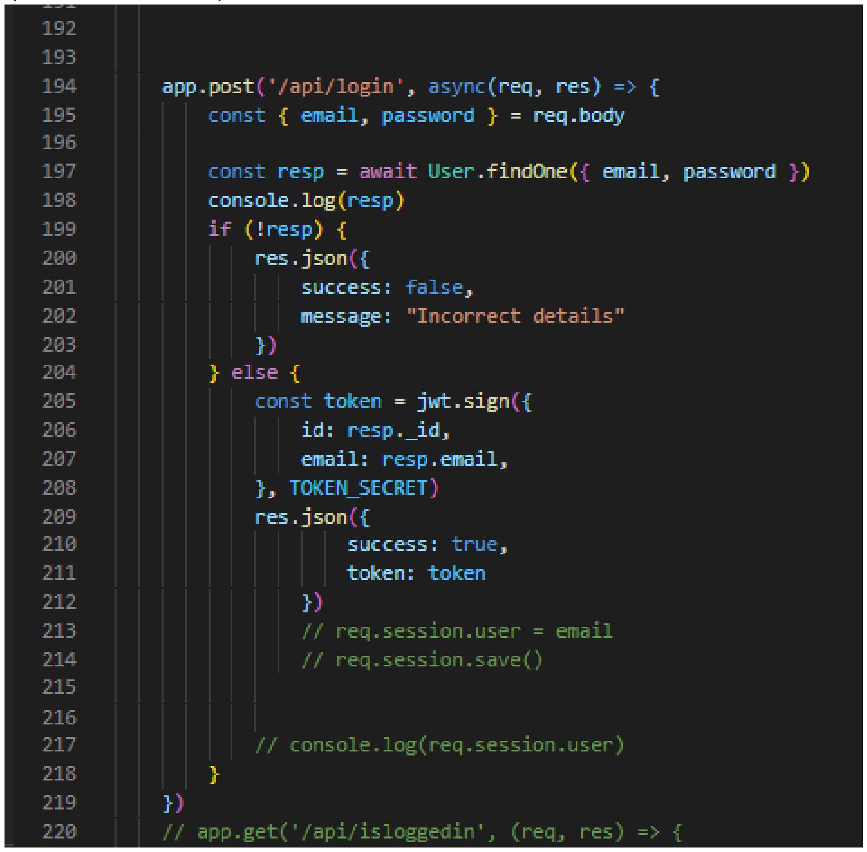
7. Login

(Frontend Part)



```
  ngOnInit(): void {
  }
  loginUser(event:any) {
    event.preventDefault()
    console.log("login method called")
    const target = event.target
      if(this.user.email && this.user.password){
        const email  = this.user.email
        const password = this.user.password
        console.log(email,password)
        this.Auth.getUserDetails(email,password).subscribe(data => {
        if(data.success) {
          localStorage.setItem("token",data.token)
          this.router.navigate(['main/inbox'])
          this.Auth.setLoggedIn(true)
        } else {
          window.alert(data.message)
        }
      })
      console.log(email, password)
      }

  }

}
```

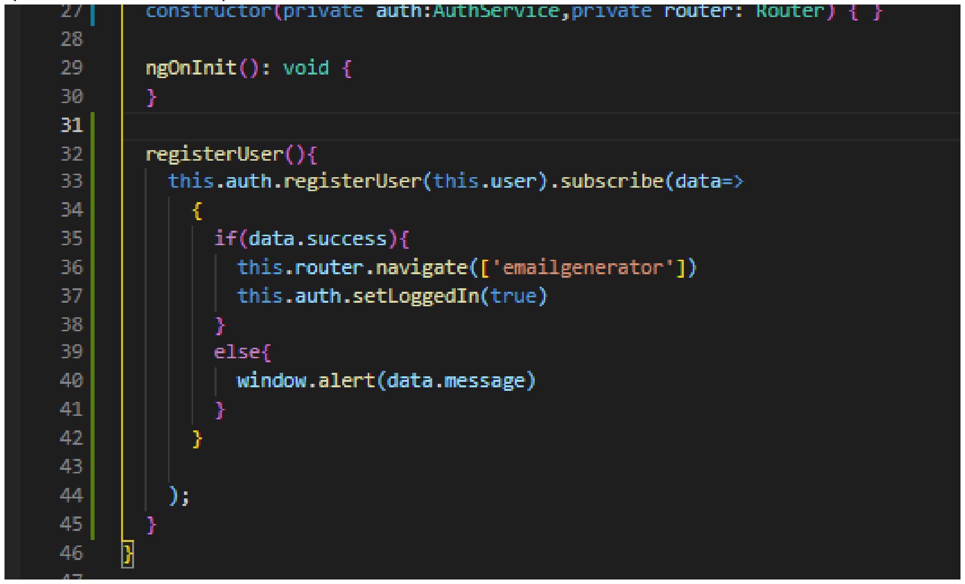
(Backend Part)



```
app.post('/api/login', async(req, res) => {
    const { email, password } = req.body

    const resp = await User.findOne({ email, password })
    console.log(resp)
    if (!resp) {
        res.json({
            success: false,
            message: "Incorrect details"
        })
    } else {
        const token = jwt.sign({
            id: resp._id,
            email: resp.email,
        }, TOKEN_SECRET)
        res.json({
                success: true,
                token: token
            })
            // req.session.user = email
            // req.session.save()


        // console.log(req.session.user)
    }
})
// app.get('/api/isloggedin', (req, res) => {
```

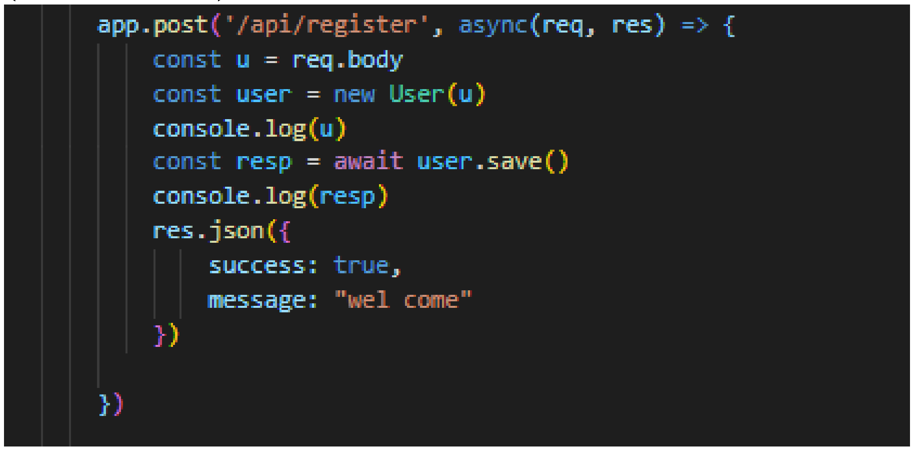
8. Register

(Frontend Part)



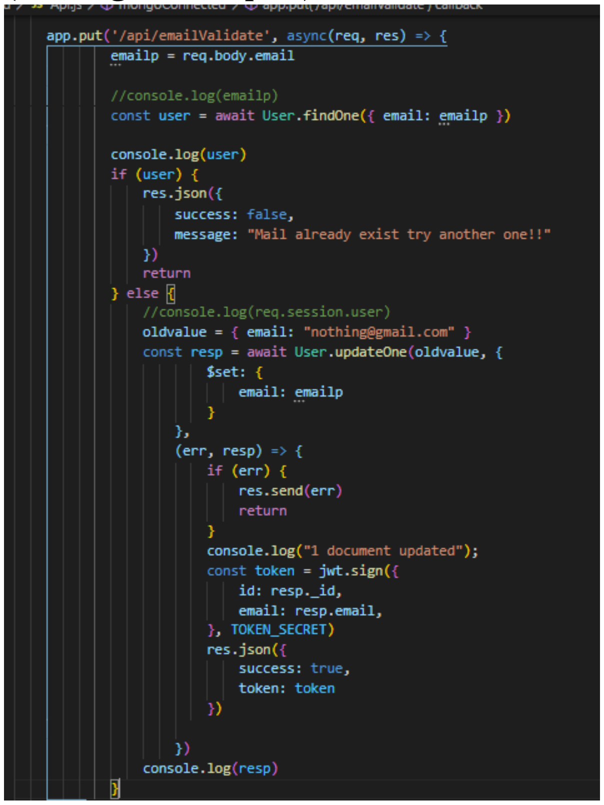
```
constructor(private auth:AuthService,private router: Router) { }

ngOnInit(): void {
}

registerUser(){
  this.auth.registerUser(this.user).subscribe(data=>
    {
      if(data.success){
        this.router.navigate(['emailgenerator'])
        this.auth.setLoggedIn(true)
      }
      else{
        window.alert(data.message)
      }
    }

  );
}
}
```

(Backend Part)



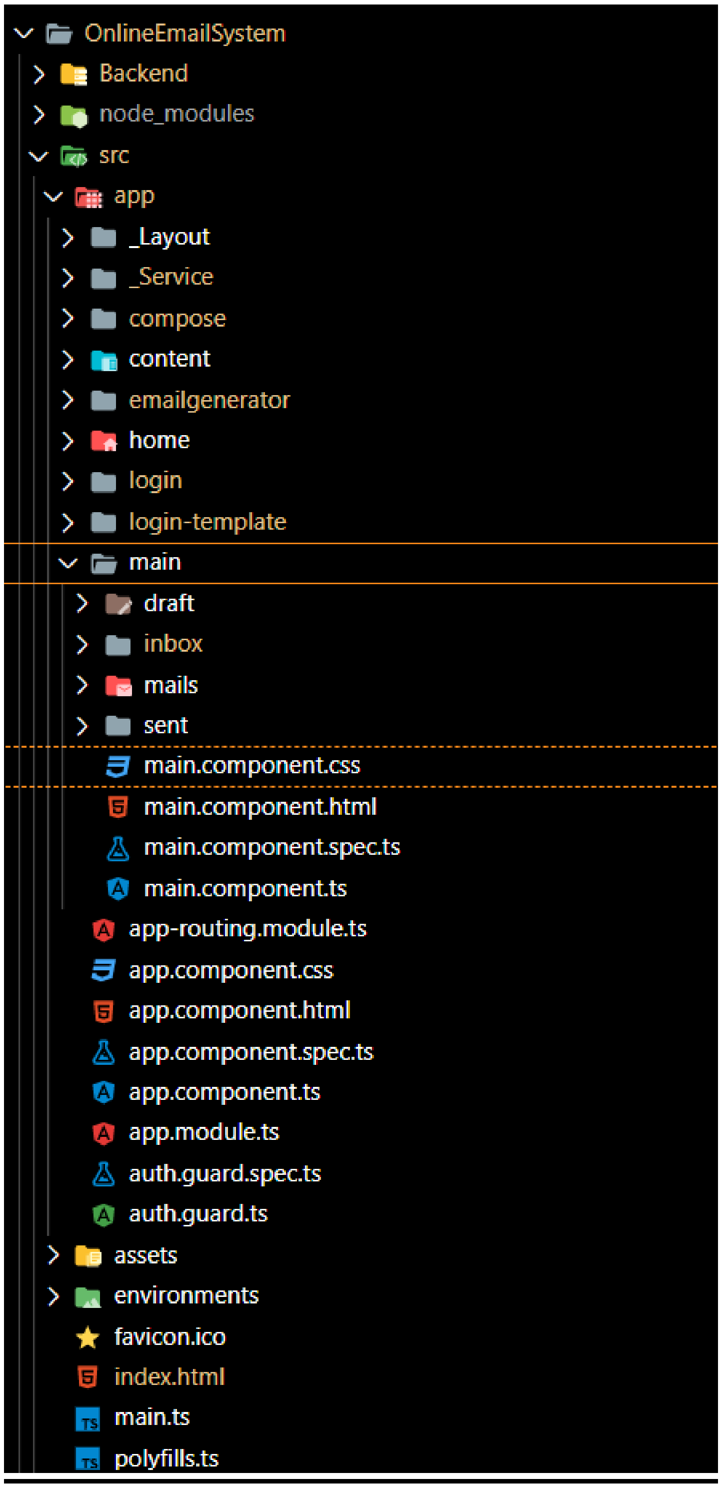
```
app.post('/api/register', async(req, res) => {
    const u = req.body
    const user = new User(u)
    console.log(u)
    const resp = await user.save()
    console.log(resp)
    res.json({
        success: true,
        message: "wel come"
    })

})
```

(email generate part)



```
app.put('/api/emailValidate', async(req, res) => {
            emailp = req.body.email

            //console.log(emailp)
            const user = await User.findOne({ email: emailp })

            console.log(user)
            if (user) {
                res.json({
                    success: false,
                    message: "Mail already exist try another one!!"
                })
                return
            } else {
                //console.log(req.session.user)
                oldvalue = { email: "nothing@gmail.com" }
                const resp = await User.updateOne(oldvalue, {
                        $set: {
                            email: emailp
                        }
                    },
                    (err, resp) => {
                        if (err) {
                            res.send(err)
                            return
                        }
                        console.log("1 document updated");
                        const token = jwt.sign({
                            id: resp._id,
                            email: resp.email,
                        }, TOKEN_SECRET)
                        res.json({
                            success: true,
                            token: token
                        })

                    })
                console.log(resp)
            }
```

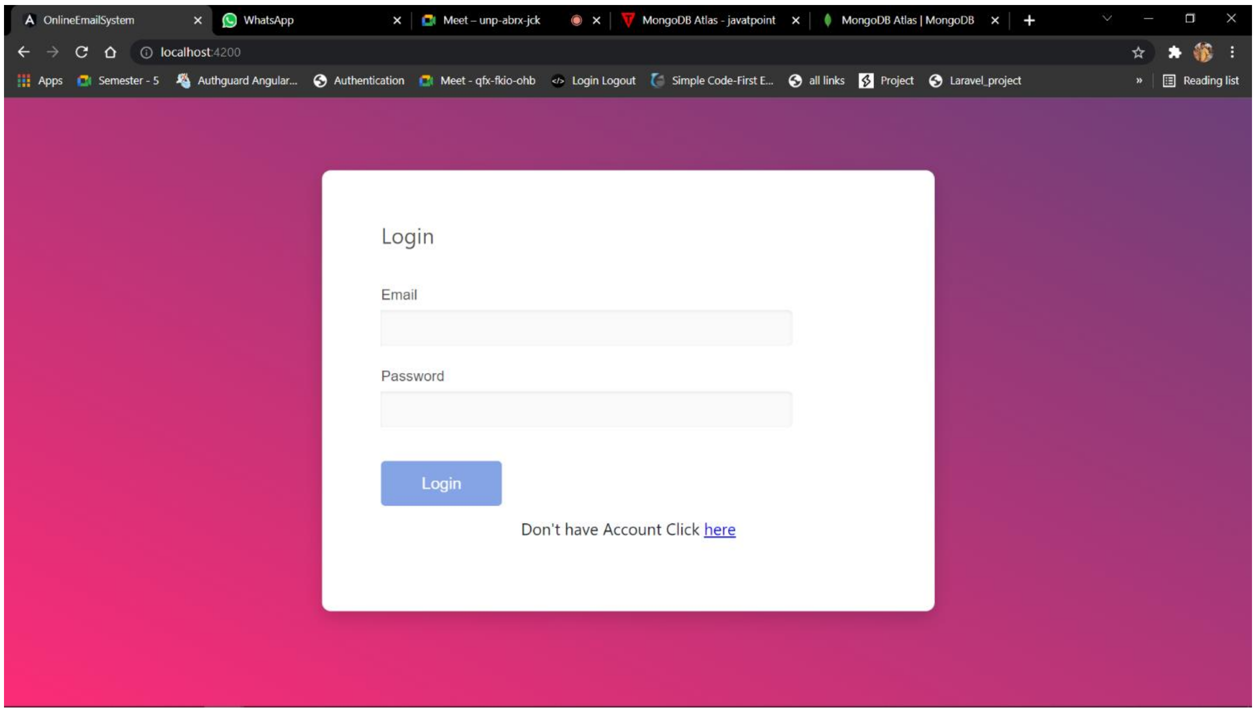
Project Structure:



```
OnlineEmailSystem
  Backend
  node_modules
  src
    app
      _Layout
      _Service
      compose
      content
      emailgenerator
      home
      login
      login-template
      main
        draft
        inbox
        mails
        sent
        main.component.css
        main.component.html
        main.component.spec.ts
        main.component.ts
      app-routing.module.ts
      app.component.css
      app.component.html
      app.component.spec.ts
      app.component.ts
      app.module.ts
      auth.guard.spec.ts
      auth.guard.ts
    assets
    environments
    favicon.ico
    index.html
    main.ts
    polyfills.ts
```

Screen-Short

Login:



Login

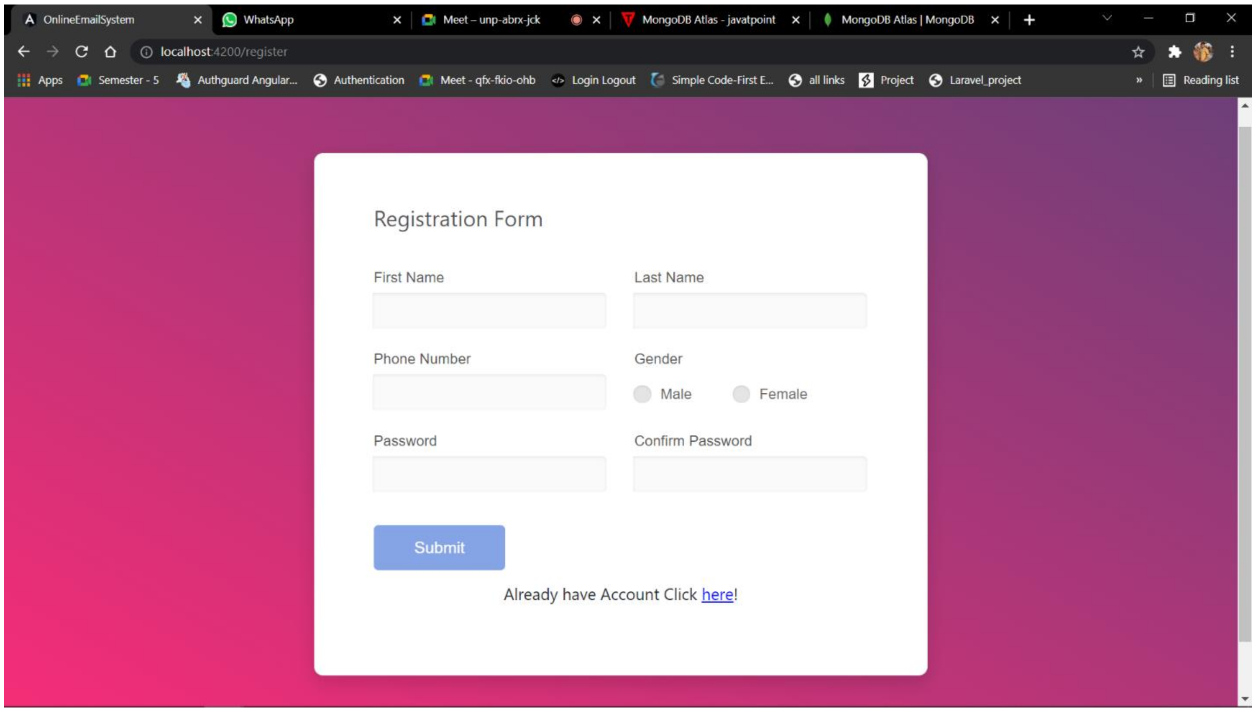
Email

Password

Login

Don't have Account Click here

Register:



Registration Form

First Name

Last Name

Phone Number

Gender

Male Female

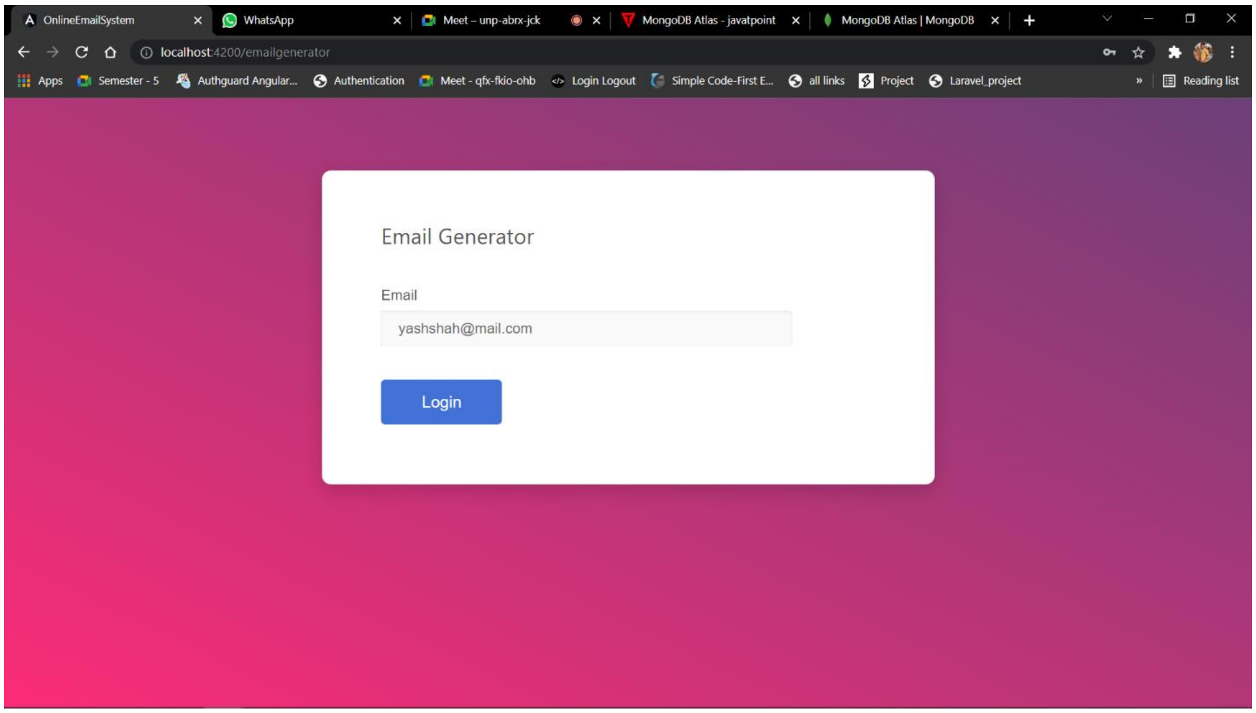
Password

Confirm Password

Submit

Already have Account Click here!

Email Generate:



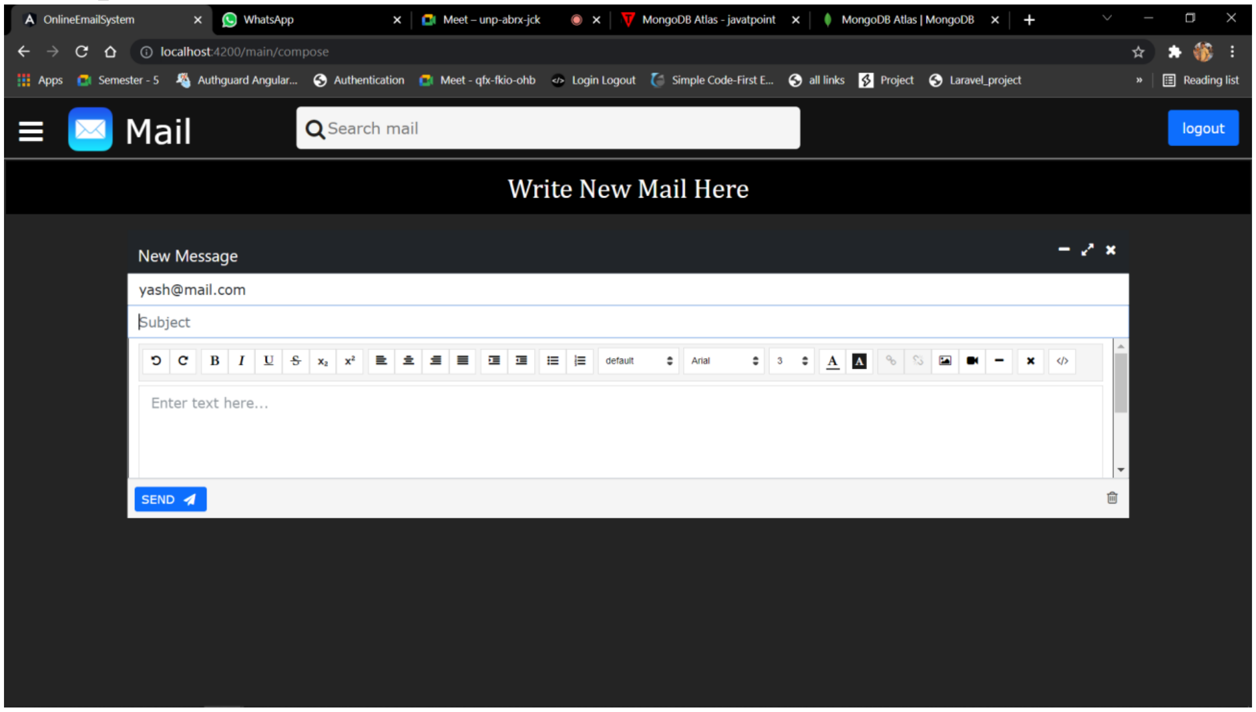
Email Generator

Email

yashshah@mail.com

Login

Compose:



Write New Mail Here

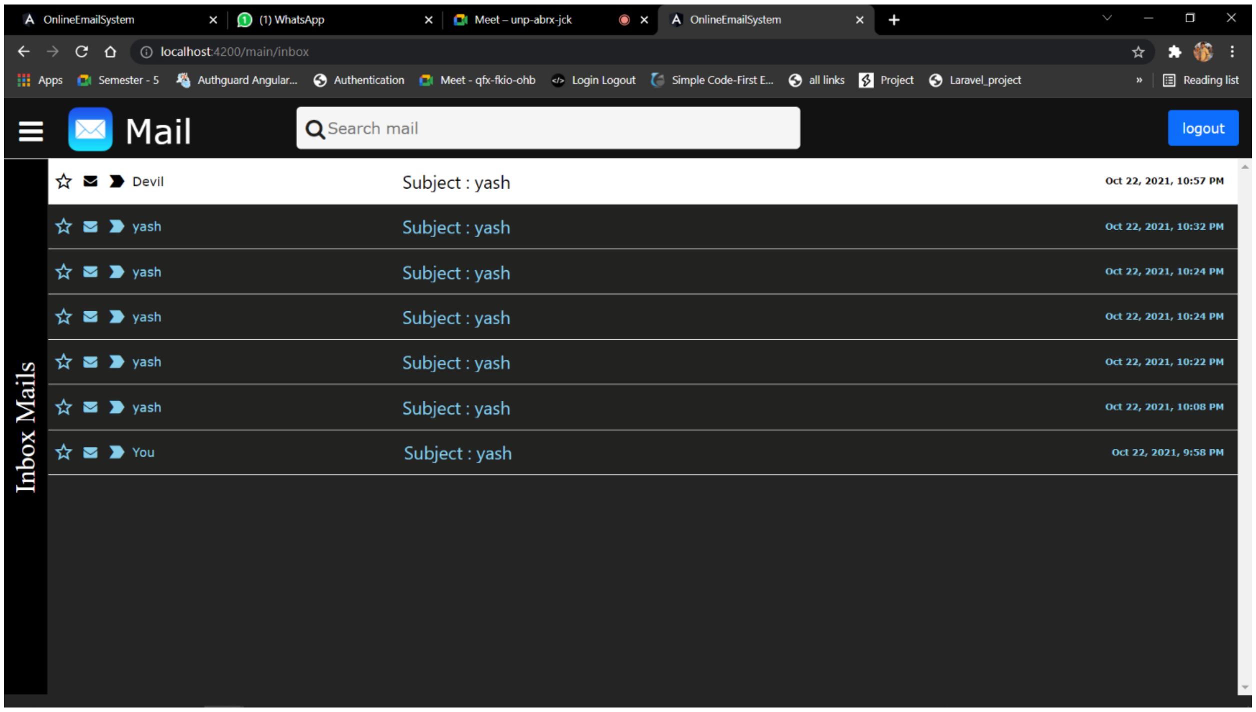
New Message

yash@mail.com

Subject

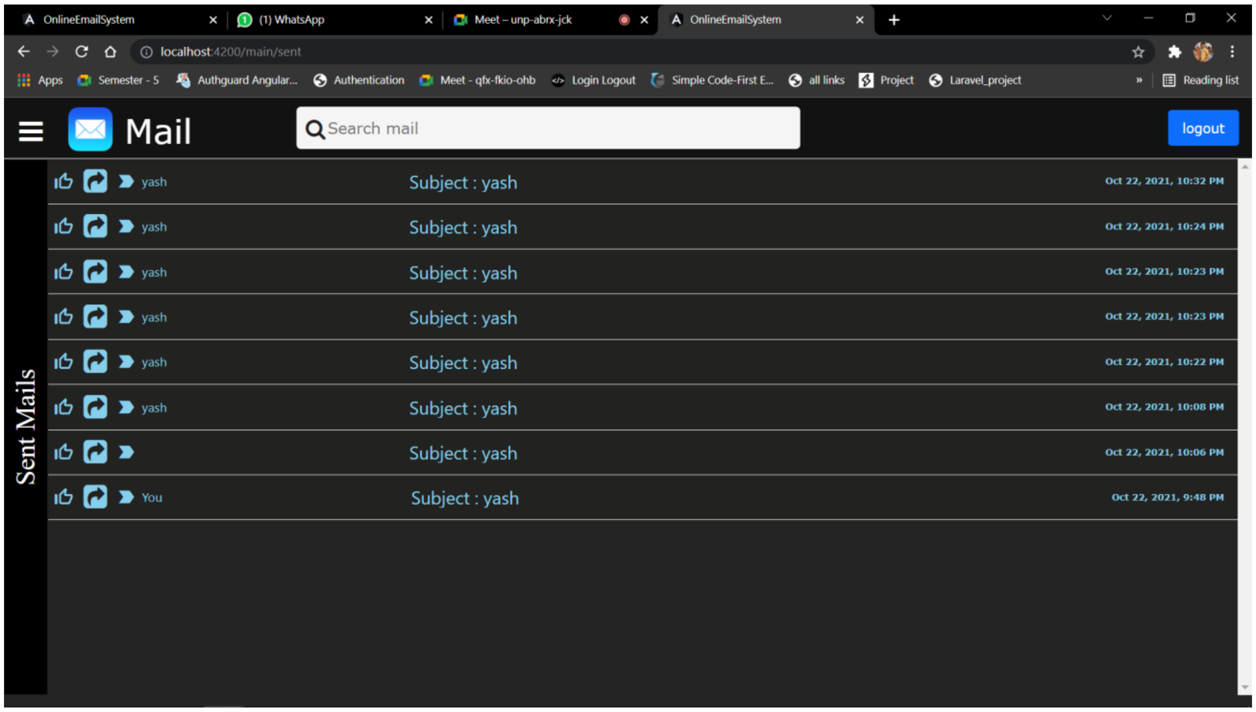
Enter text here...

Inbox:



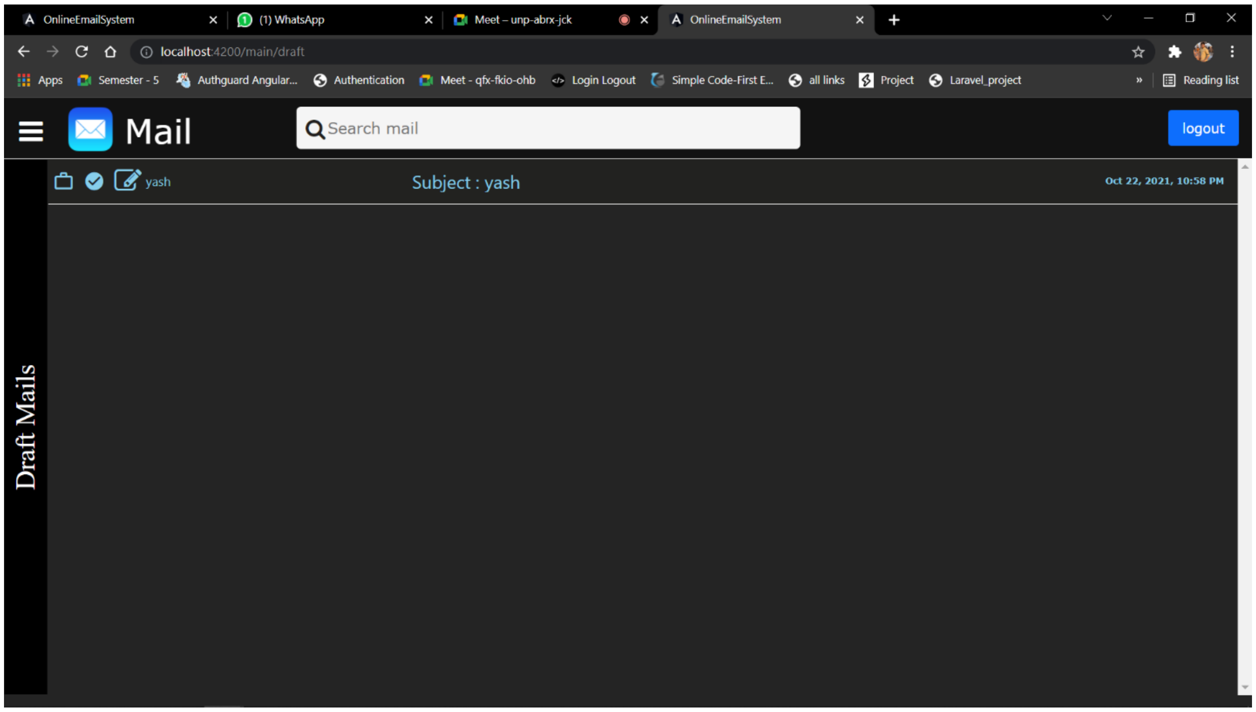
Devil	Subject : yash	Oct 22, 2021, 10:57 PM
yash	Subject : yash	Oct 22, 2021, 10:32 PM
yash	Subject : yash	Oct 22, 2021, 10:24 PM
yash	Subject : yash	Oct 22, 2021, 10:24 PM
yash	Subject : yash	Oct 22, 2021, 10:22 PM
yash	Subject : yash	Oct 22, 2021, 10:08 PM
You	Subject : yash	Oct 22, 2021, 9:58 PM

Sent:

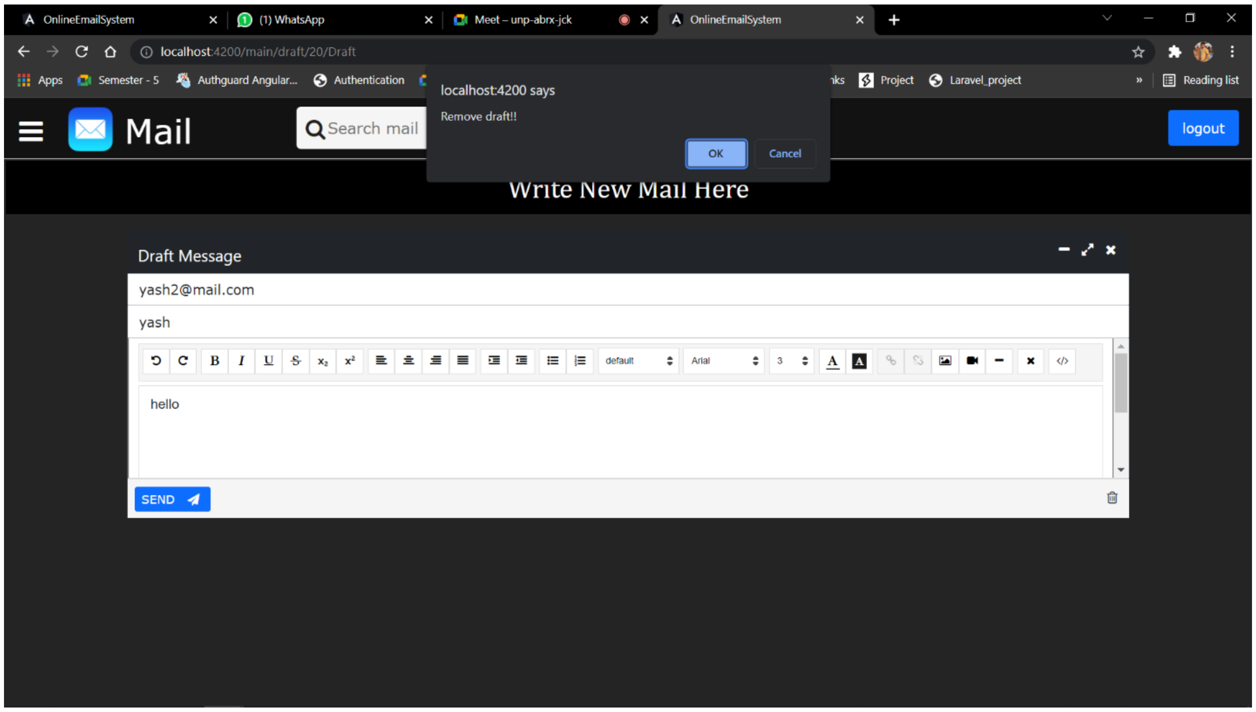


yash	Subject : yash	Oct 22, 2021, 10:32 PM
yash	Subject : yash	Oct 22, 2021, 10:24 PM
yash	Subject : yash	Oct 22, 2021, 10:23 PM
yash	Subject : yash	Oct 22, 2021, 10:23 PM
yash	Subject : yash	Oct 22, 2021, 10:22 PM
yash	Subject : yash	Oct 22, 2021, 10:08 PM
	Subject : yash	Oct 22, 2021, 10:06 PM
You	Subject : yash	Oct 22, 2021, 9:48 PM

Draft:



Remove Draft:



localhost:4200 says

Remove draft!!

Write New Mail Here

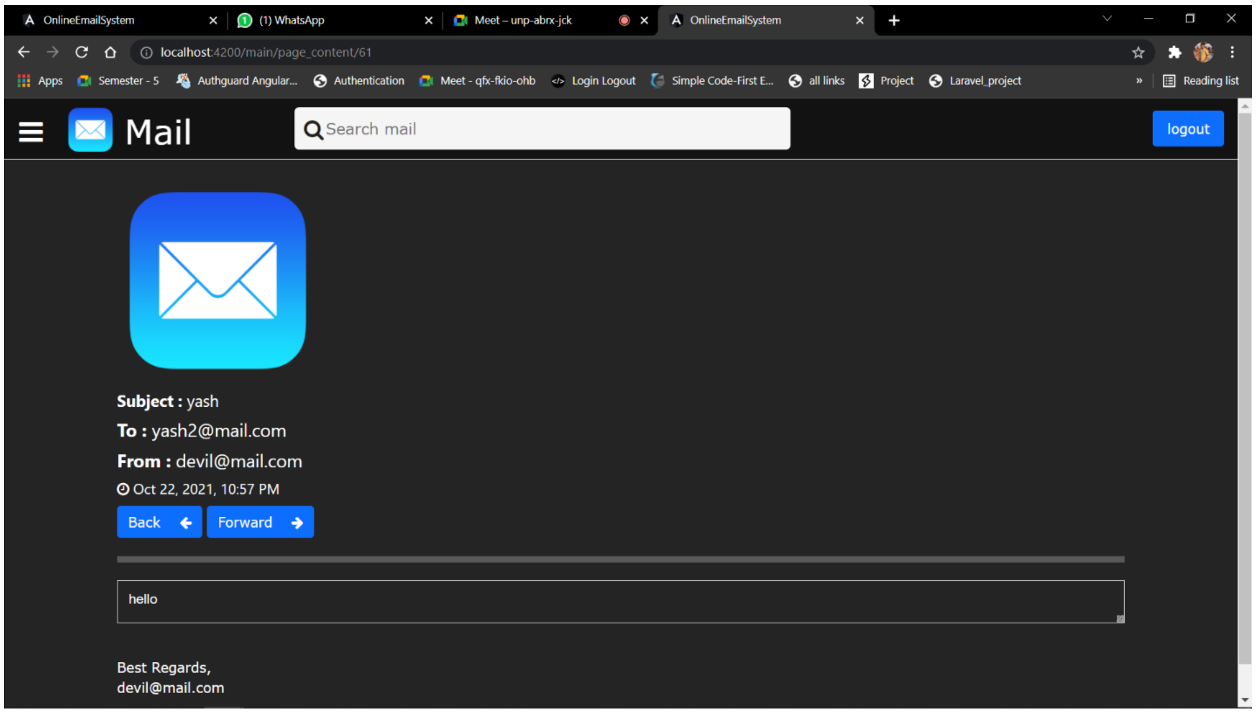
Draft Message

yash2@mail.com

yash

hello

Mail Content:

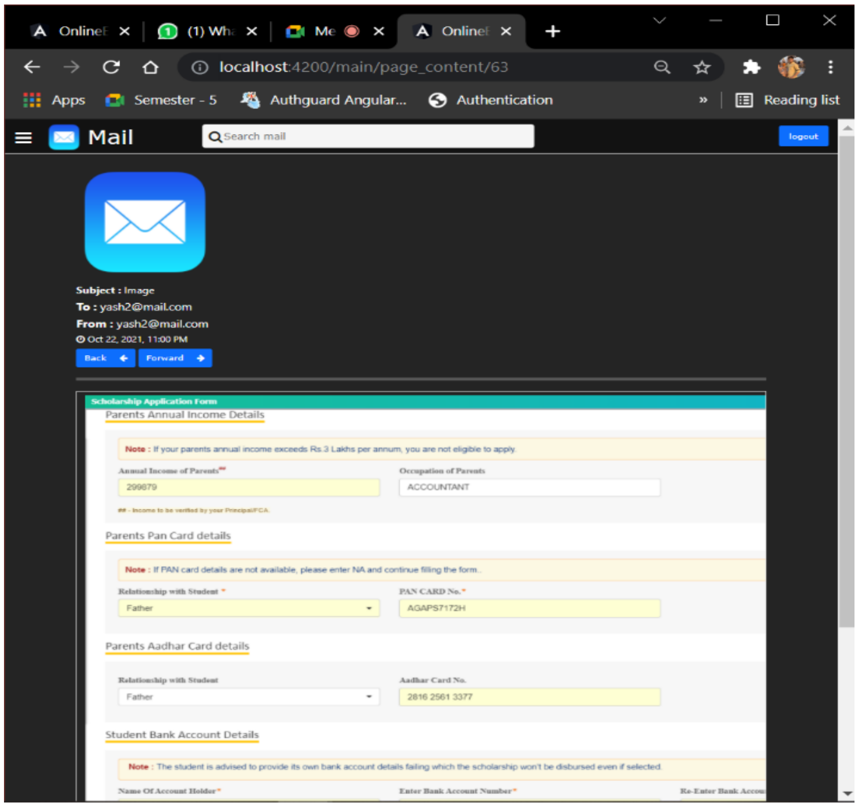


Subject : yash
To : yash2@mail.com
From : devil@mail.com
Oct 22, 2021, 10:57 PM

hello

Best Regards,
devil@mail.com

Image:



Subject : Image
To : yash2@mail.com
From : yash2@mail.com
Oct 22, 2021, 11:00 PM

Scholarship Application Form

Parents Annual Income Details

Note : If your parents annual income exceeds Rs.3 Lakhs per annum, you are not eligible to apply.

Annual Income of Parents## : 299879
Occupation of Parents : ACCOUNTANT

- Income to be verified by your Principal/FCA.

Parents Pan Card details

Note : If PAN card details are not available, please enter NA and continue filling the form..

Relationship with Student * : Father
PAN CARD No.* : AGAPS7172H

Parents Aadhar Card details

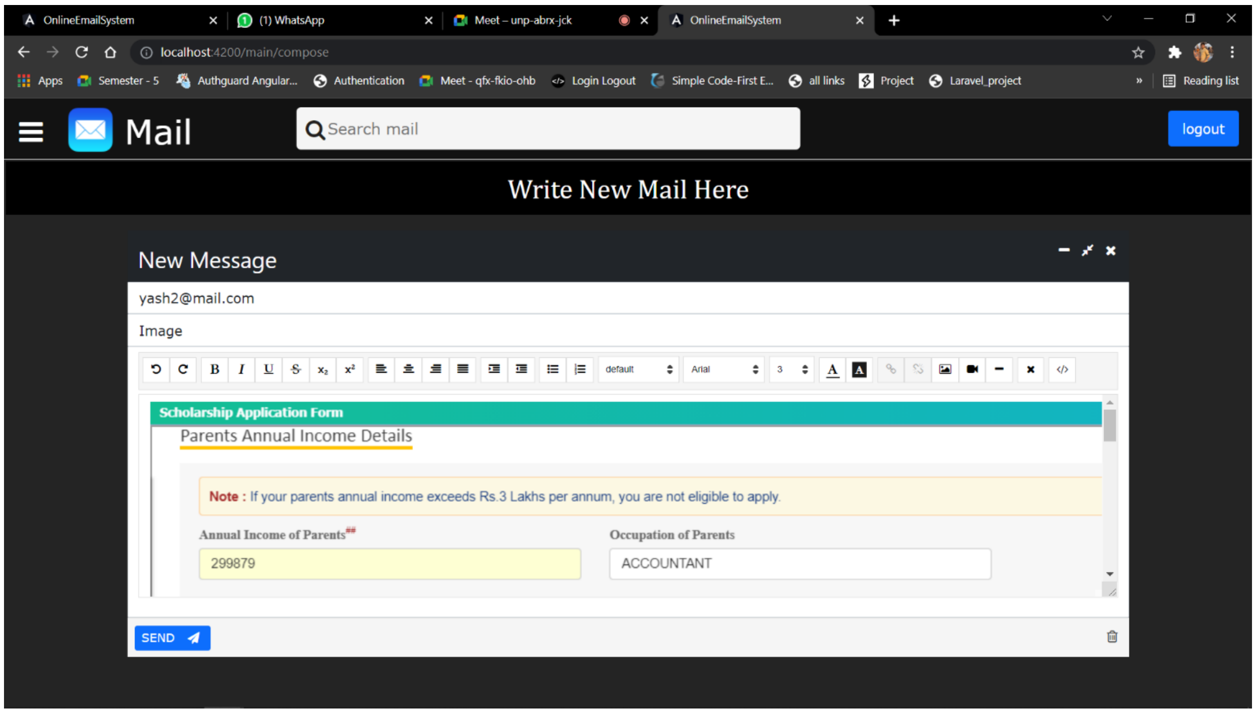
Relationship with Student : Father
Aadhar Card No. : 2816 2561 3377

Student Bank Account Details

Note : The student is advised to provide its own bank account details failing which the scholarship won't be disbursed even if selected.

Name Of Account Holder*
Enter Bank Account Number*
Re-Enter Bank Accou

Send Image:



yash2@mail.com

Image

Scholarship Application Form

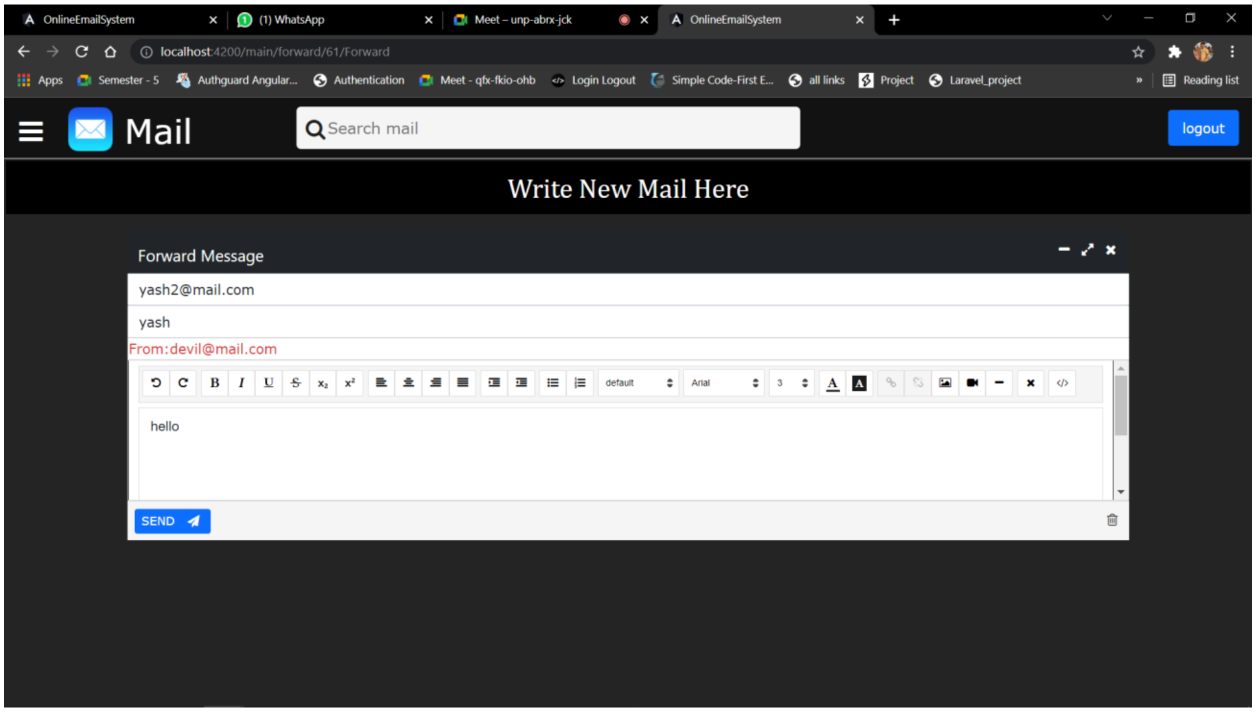
Parents Annual Income Details

Note : If your parents annual income exceeds Rs.3 Lakhs per annum, you are not eligible to apply.

Annual Income of Parents** 299879

Occupation of Parents ACCOUNTANT

Forward Mail:



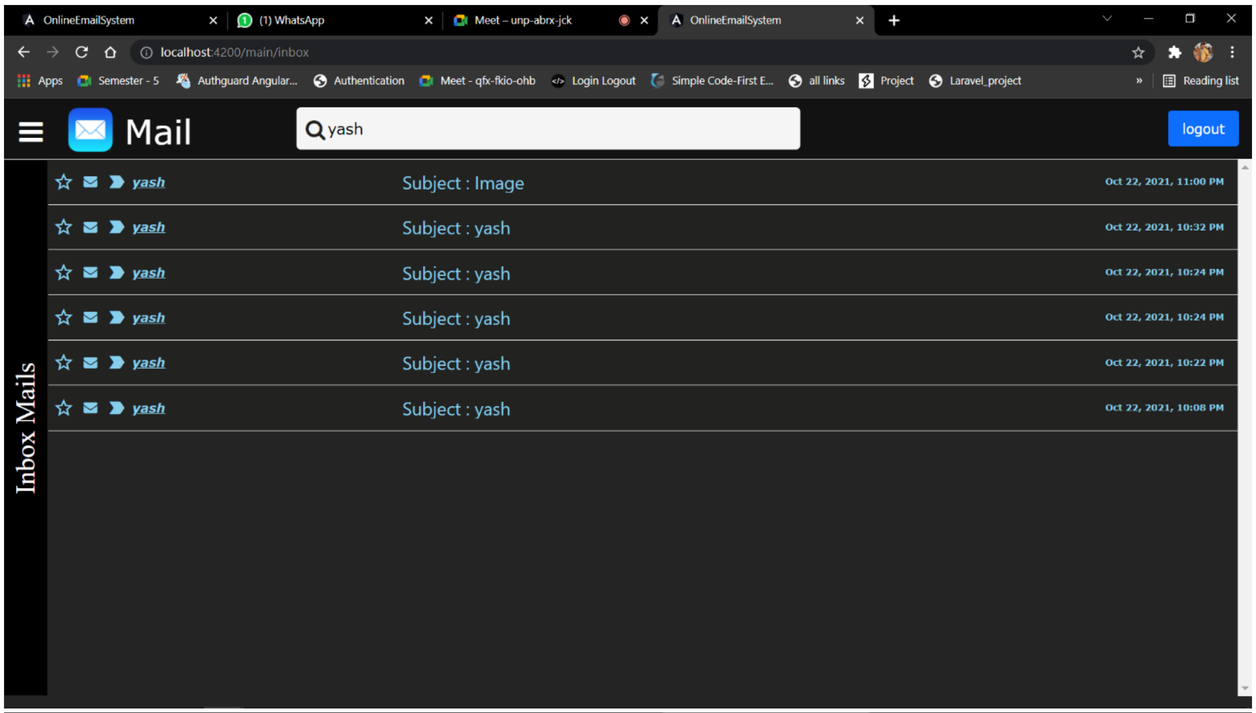
yash2@mail.com

yash

From:devil@mail.com

hello

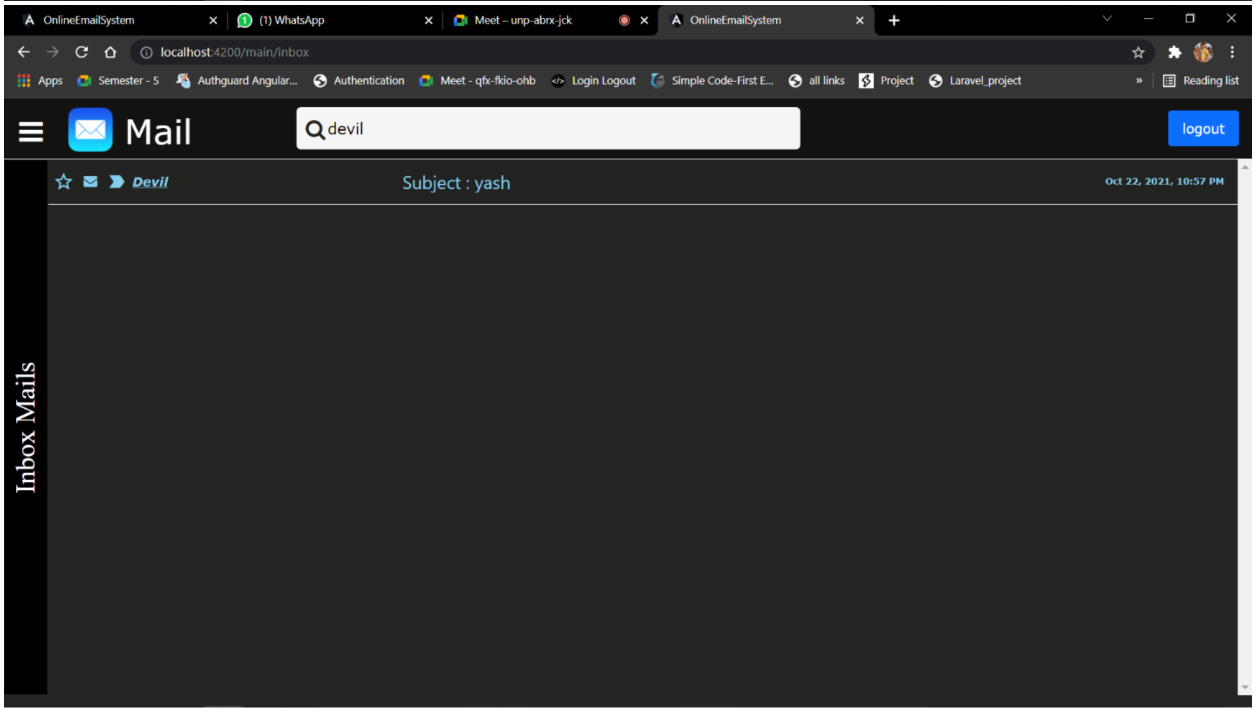
Search Mails with Username:



yash

Inbox Mails

yash	Subject : Image	Oct 22, 2021, 11:00 PM
yash	Subject : yash	Oct 22, 2021, 10:32 PM
yash	Subject : yash	Oct 22, 2021, 10:24 PM
yash	Subject : yash	Oct 22, 2021, 10:24 PM
yash	Subject : yash	Oct 22, 2021, 10:22 PM
yash	Subject : yash	Oct 22, 2021, 10:08 PM



devil

Inbox Mails

Devil	Subject : yash	Oct 22, 2021, 10:57 PM

Conclusion

Hence-forth in this project we have successfully implemented the Client-side functionality. User can register or login to project and they can mail another user. They can check their received mails and also able to reply those mails. Also, they are able to forward mails. Also possible to search mail by the username of the user. So Online Email System will be useful for official communication.

Limitations

- This project is implemented in small level. This project can be implemented in large level like (GMAIL, YAHOO etc.)
- Functionalities like bin, spam mails, schedules, snoozed, starred is remaining to be implemented.

Future Extensions

To take over the limitations we are planning this future extension in our system.

- Including security specific software in the project
- Include Zoom and Hangout like Gmail
- Include more dynamic feature in project.
- Include User’s Profile management by admin.

Bibliography

References/resources used for developing project:

- https://www.w3schools.com/nodejs/
- https://angular.io/docs
- https://www.npmjs.com/package/@kolkov/angular-editor