TAAS - Feature #355

Mail

08/28/2024 04:36 PM - Anil KV

Status:	Coding Done	**Start date:**	08/28/2024
Priority:	Normal	**Due date:**	
Assignee:	Junaid M	**% Done:**	60%
Category:		**Estimated time:**	0:00 hour
Target version:	22.12	**Spent time:**	0:00 hour
Owner(Agency):	Travvise	**Tested By:**	
Time Taken(HH):		**Code Reviewed By:**	
Module:	TAAS-General		

Description

Mail

History

#1 - 08/28/2024 05:15 PM - Anil KV

1) Travel Date near(1/2/3 Days back) then send message to customer and Staff - Setting base

#2 - 04/04/2025 11:40 AM - Anil KV

- Assignee changed from travvise Admin to Anil KV

#3 - 04/04/2025 03:41 PM - Anil KV

```
import * as pdfMake from "pdfmake/build/pdfmake";
import * as pdfFonts from "pdfmake/build/vfs_fonts";
import { HttpClient } from '@angular/common/http';
import { Component } from '@angular/core';

(pdfMake as any).vfs = pdfFonts.pdfMake.vfs;

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  constructor(private http: HttpClient) {}

  generateAndSendPdf() {
    const documentDefinition = {
      content: ['Hello, this is a test PDF generated using pdfmake!']
    };

    const pdfDocGenerator = pdfMake.createPdf(documentDefinition);

    pdfDocGenerator.getBlob((pdfBlob: Blob) => {
      const formData = new FormData();
      formData.append('file', pdfBlob, 'example.pdf');

      // Send to the .NET backend
      this.http.post('http://localhost:5000/api/send-email', formData)
        .subscribe(response => {
          console.log('PDF sent to server:', response);
        });
    });
  }
}
```

#4 - 04/04/2025 03:42 PM - Anil KV

dotnet add package MailKit
dotnet add package MimeKit

```
using Microsoft.AspNetCore.Mvc;
using System.IO;
using System.Net.Mail;
using System.Net;
using MimeKit;
using MailKit.Net.Smtp;

[ApiController]
[Route("api/[controller]")]
public class SendEmailController : ControllerBase
{
    [HttpPost]
    [Route("send-email")]
    public async Task<IActionResult> SendEmail(IFormFile file)
    {
        if (file == null || file.Length == 0)
        {
            return BadRequest("No file uploaded.");
        }

        // Save the uploaded file to a temporary location
        var tempFilePath = Path.GetTempFileName();
        using (var stream = new FileStream(tempFilePath, FileMode.Create))
        {
            await file.CopyToAsync(stream);
        }

        try
        {
            // Send the email with the attached PDF
            var message = new MimeMessage();
            message.From.Add(new MailboxAddress("Your Name", "your_email@example.com"));
            message.To.Add(new MailboxAddress("Recipient", "recipient@example.com"));
            message.Subject = "PDF Attachment";

            var body = new TextPart("plain")
            {
                Text = "Please find the attached PDF file."
            };

            var attachment = new MimePart("application/pdf")
            {
                Content = new MimeContent(File.OpenRead(tempFilePath)),
                ContentDisposition = new ContentDisposition(ContentDisposition.Attachment),
                ContentTransferEncoding = ContentEncoding.Base64,
                FileName = "example.pdf"
            };

            var multipart = new Multipart("mixed");
            multipart.Add(body);
            multipart.Add(attachment);
            message.Body = multipart;

            using (var client = new SmtpClient())
            {
                await client.ConnectAsync("smtp.yourserver.com", 587, false);
                await client.AuthenticateAsync("your_email@example.com", "your_password");
                await client.SendAsync(message);
                await client.DisconnectAsync(true);
            }

            return Ok("Email sent successfully!");
        }
        catch (Exception ex)
        {
            return StatusCode(500, $"Error sending email: {ex.Message}");
        }
        finally
        {
            // Clean up the temp file
            System.IO.File.Delete(tempFilePath);
        }
    }
}
```

#5 - 04/04/2025 03:53 PM - Anil KV

```
using System;
using System.IO;

class Program
{
    static void Main()
    {
        // Base64 string representing the PDF file
        string base64String = "yourBase64StringHere"; // Replace with your Base64 string

        // Decode the Base64 string to binary data
        byte[] pdfData = Convert.FromBase64String(base64String);

        // Specify the file path for the output PDF
        string outputFilePath = "output.pdf";

        // Write the binary data to the file
        File.WriteAllBytes(outputFilePath, pdfData);

        Console.WriteLine($"PDF file created at: {outputFilePath}");
    }
}
```

#6 - 04/04/2025 03:57 PM - Anil KV

Use This Logic

```
using System;

class Program
{
    static void Main()
    {
        // Example blob data (binary data as a byte array)
        byte[] blobData = new byte[] { 0x48, 0x65, 0x6C, 0x6C, 0x6F }; // Represents "Hello"

        // Convert the binary data to a Base64 string
        string base64String = Convert.ToBase64String(blobData);

        // Output the Base64 string
        Console.WriteLine($"Base64 String: {base64String}");
    }
}
```

#7 - 04/04/2025 04:05 PM - Anil KV

Set signatureImage(from Mail Setting)
if No Image then set
signature text from Mail Setting

```
using System;
using System.Net;
using System.Net.Mail;

class Program
{
    static void Main()
    {
        // Email settings
        string smtpServer = "smtp.yourserver.com";
        int smtpPort = 587;
        string smtpUser = "your_email@example.com";
        string smtpPass = "your_password";
        string senderEmail = "your_email@example.com";
        string recipientEmail = "recipient@example.com";

        // Path to the signature image
        string signatureImagePath = "signature.png"; // Replace with the actual file path

        // Create the email message
        MailMessage mail = new MailMessage();
        mail.From = new MailAddress(senderEmail);
```

```
        mail.To.Add(new MailAddress(recipientEmail));
        mail.Subject = "Email with Signature";
        mail.IsBodyHtml = true;

        // Embed the signature image in the email body
        string imageContentId = Guid.NewGuid().ToString(); // Generate unique Content ID for the image
        mail.Body = $@"
            <p>Dear Recipient,</p>
            <p>Please find my signature below:</p>
            <p><img src='cid:{imageContentId}' alt='Signature'></p>
            <p>Best Regards,</p>
        ";

        // Attach the image and set it to be inline
        Attachment signatureImage = new Attachment(signatureImagePath);
        signatureImage.ContentId = imageContentId;
        signatureImage.ContentDisposition.Inline = true; // Set the image to be inline
        signatureImage.ContentType.MediaType = "image/png";
        mail.Attachments.Add(signatureImage);

        // Send the email
        using (SmtpClient smtpClient = new SmtpClient(smtpServer, smtpPort))
        {
            smtpClient.Credentials = new NetworkCredential(smtpUser, smtpPass);
            smtpClient.EnableSsl = true;

            try
            {
                smtpClient.Send(mail);
                Console.WriteLine("Email sent successfully!");
            }
            catch (Exception ex)
            {
                Console.WriteLine($"Failed to send email: {ex.Message}");
            }
        }
    }
}
```

#8 - 04/05/2025 12:07 PM - Anil KV

- Status changed from New to Ready for Coding

- Assignee changed from Anil KV to Junaid M

- % Done changed from 0 to 20

#9 - 05/02/2025 04:23 PM - Junaid M

- Status changed from Ready for Coding to Coding Done

- % Done changed from 20 to 60