# PDF Generation and Email Service Implementation:

## **Project Overview:**

The project aims to implement a PDF generation service, an email service, and a task scheduler. The PDF generation service will generate PDF documents based on form input data, using data from “Buyer”, “Seller”, and “Attorney” forms, while the email service will send these generated PDFs using a smtp service. The task scheduler will periodically trigger the email service to check for new PDFs and send them to the specified email addresses.

### Project Services and Tasks:

1. PDF Generation Service

2. Email Service

3. Task Scheduler

# Task 1: PDF Generation Service

**Description**: Implement a service responsible for generating PDF documents based on form input data received from the frontend.

### Work Flow:

* Use current Add/Update Deal API to pass data to backend. Add options on Buyer, Seller, and Attorney forms to enable/disable sending of email.
* Based on option new service will be created in backend to receive API data from Seller, Buyer, and/or Attorney form.

### Methods:

1. **GeneratePdf(formData: FormData, templateID: int): byte[]**

- Description: Generates a PDF document based on the provided form data and the specific template.

- Steps:

- Load the PDF template.

- Map the form data to the placeholders in the PDF template.

- Inputs: formData (Form data object containing input field values)

- Outputs: byte[] (Generated PDF document)

**2. SavePDFtoFile(templateID: int, pdfData: Byte []) : String;**

* Description:
  + Saves the generated PDF to a file and returns file name/location of PDF file
* Steps:
  + Generate random file name based on template type, deal ID, and timestamp.
  + Save PDF to file location

**3. SavePDFtoDB(data: CreateTemplateRqDto, pdfData: Byte[] )**

* Description: Save the required data to the master table in the database with required metadata
  + Title
  + Firstname
  + Surname
  + Email Address
  + PDF File location
  + Status
  + Fail Count
  + CreatedOn
  + UpdatedOn
  + SentOn

**Time Estimation: 30 hours**

# Task 2: Email Service

Description: Develop a service responsible for sending PDF documents via email.

### Methods:

1. **SendPendingPdfs(): void**

* **Description:**
  + Sends pending PDF documents via email to the specified recipients. Need to handle failed result.
* **Steps:**
  + Retrieve pending PDF documents from the database.
  + Attach PDFs to email messages.
  + Send emails with attachments using an email client.
  + Inputs: None
  + Outputs: None

**Handling failed result:** If the email is unable to send the service will update the table field “failedCounter”. This counter will have a limit to filter out the email and stop sending (or mark status as 1 – deleted)

**Time Estimation: 20 hours**

# Task 3: Task Scheduler

**Description**: Implement a task scheduler to trigger the email service periodically.

**Methods:**

1. ExecuteAsync(CancellationToken): Task

- **Description**: Executes the email service to send pending PDFs at specified intervals.

- **Steps**:

- Periodically invoke the email service's SendPendingPdfs method.

- Handle cancellation token to stop execution when required.

- **Inputs**: CancellationToken (Used to signal cancellation)

- **Outputs**: Task (Asynchronous task representing the execution)

**Time Estimation: 12 hours**

**Other tasks:**

**-** Create DB tables

- Implement PDF templates with input fields

- Backend API business logic

- PDF generation will be based on a pre-designed template.

- Email service will utilize SMTP for sending emails.

- Connecting and testing SMTP Service.

- The task scheduler will be implemented as a background service.

- Input data format and PDF template structure are predefined.

**Total Estimated Time: 52 hours (Excluding Other tasks as above)**

|  |  |  |
| --- | --- | --- |
| Status | Date and Time | Details |
| Sent | 2024/04/16 09:05 | - |

|  |  |  |
| --- | --- | --- |
| Status | Date and Time | Details |
| Failed | - | Incorrect email address |