🎓 Certificate Sender Web App – Complete Documentation

📌 Project Overview

This Flask-based web application automates the process of sending **pre-generated certificate PDFs** to participants via email. It matches certificate files to names and emails listed in an uploaded Excel spreadsheet, attaches the appropriate certificate, and sends it with a custom thank-you message.

📁 Folder & File Structure

```
certificate_sender/
├── app.py                    # Main Flask app logic
├── config.py                 # Email configuration
├── Certificate_Names_Emails.xlsx # Excel file with names & emails
├── uploads/                  # Temporary folder for uploaded files (created at runtime)
└── templates/
    └── index.html            # HTML form page (not included)
```

⚙️ Features

- Upload Excel file (.xlsx) with Name and Email columns.
- Upload pre-generated PDF certificates.
- Input a custom thank-you message via the web form.
- Automatically matches and emails certificates to participants.

🚀 Installation and Setup

1. Clone or download the project.

2. Install required packages:

```
pip install flask pandas yagmail openpyxl
```

3. Set up your Gmail:

- Enable **2-step verification** in Gmail.
- Generate an **App Password** under Google Account → Security → App Passwords.

4. Edit `config.py`:

EMAIL_SENDER = 'your_email@gmail.com'

EMAIL_PASSWORD = 'your_app_password'

EMAIL_SUBJECT = 'Thank You for Participating'\

🧠 How It Works

Web Interface (app.py + HTML form)

1. **User uploads**:
    - An Excel file (.xlsx) with two columns: Name, Email.
    - One or more PDF certificates.
    - A custom thank-you message.
2. **Backend processing**:
    - Excel file is read using pandas.
    - Names are formatted (e.g., "Alice Smith" → alice_smith.pdf).
    - Matching certificate files are located.
    - Emails are sent using yagmail.

📄 config.py – Configuration File

python

CopyEdit

EMAIL_SENDER = 'your_email@gmail.com'

EMAIL_PASSWORD = 'your_app_password'

EMAIL_SUBJECT = 'Thank You for Participating'

🗒️ app.py – Core Logic

Key Libraries:

```
import os
import pandas as pd
import yagmail
from flask import Flask, render_template, request
from werkzeug.utils import secure_filename
```

File Upload & Processing:

- Files are saved to the uploads/ directory.

Certificate filenames must match names in the Excel file after formatting:

```
formatted_name = name.strip().lower().replace(' ', '_') + '.pdf'
```

-

Sending Email:

```
yag.send(
    to=email,
    subject=EMAIL_SUBJECT,
    contents=thank_you_message,
    attachments=certificate_path
)
```

Error Handling:

- Errors like missing attachments are caught and printed but don't stop the entire process.

📈 Excel File Format

Example Certificate_Names_Emails.xlsx:

Name	Email
Alice Smith	alice@gmail.com
Bob Johnson	bob@example.com

📌 Certificate Requirements

- Certificate files must be **PDF**.
- Filenames should follow:
  "First Last" → first_last.pdf

🌐 URL Endpoint

Run with:
python app.py

- Access via:
  http://127.0.0.1:5000/

🛠️ Troubleshooting

- Check that certificate filenames exactly match Excel names (lowercase, underscores).
- Make sure Excel uses correct headers: Name, Email.
- If using Gmail, ensure App Password is active.
- Check console logs for missing files or email errors.

👨‍💻 Developer Notes

- Temporary files are stored in uploads/.
- Uses yagmail for secure email via Gmail SMTP.
- Flask handles routing and rendering.
- Requires an HTML form template (index.html) with inputs:
    - Excel file
    - Certificate folder (as zip or individual files)
    - Thank-you message

✅ To-Do / Future Improvements

- Add email status feedback in UI.
- Allow dynamic certificate generation.
- Add multi-language support for emails.