Django Application: Excel File Handling and Email Summary

Overview

This Django application allows users to upload an Excel file, processes the data to generate a summary, emails the summary to a recipient, and displays it on a webpage with an option to download the summary as an Excel file.

Key Features

1. File Upload: Users upload an Excel file through a form.
2. Data Processing: The file is processed using `pandas`, and a summary is created by grouping data based on "Cust State" and "Cust Pin".
3. Email Summary: The summary is emailed to a pre-defined recipient with the summary attached as an Excel file.
4. Summary Display: The summary is displayed on a results page, with a downloadable link for the Excel summary.

Main Functions

1. handle_uploaded_file(file)
- Purpose: Reads and processes the uploaded Excel file to create a summary.
- Process:
  - Groups data by `Cust State` and `Cust Pin`, calculates the count (`DPD`).
  - Saves the summary as an Excel file (`customer_summary.xlsx`).
- Returns: The summary dataframe and the file path.

```python
def handle_uploaded_file(file):
    df = pd.read_excel(file)
    summary = df.groupby(['Cust State', 'Cust Pin']).size().reset_index(name='DPD')
    file_path = os.path.join(settings.BASE_DIR, 'customer_summary.xlsx')
    summary.to_excel(file_path, index=False)
    return summary, file_path
```

2. send_summary_via_email(summary, file_path, recipient_email)
- Purpose: Sends the summary as an email with the file attached.
- Process:
  - Converts the summary to text and includes it in the email body.
  - Attaches the Excel summary file to the email.

```python
def send_summary_via_email(summary, file_path, recipient_email):
    email = EmailMessage(
        'Python Assessment - Parth Nangroo',
        f'Please find the summary report:\n\n{summary.to_string(index=False)}',
        settings.DEFAULT_FROM_EMAIL, [recipient_email])
    email.attach_file(file_path)
    email.send()
```

3. main(request)
- Purpose: Handles the file upload and email sending.
- Process:
  - If `GET`, renders the upload form.
  - If `POST`, processes the uploaded file, sends the summary via email, and redirects to the results page.

4. result(request)
- Purpose: Displays the summary and download option for the Excel file.

5. download_file(request)
- Purpose: Allows the user to download the Excel summary file.

```python
def download_file(request):
    file_path = request.GET.get('file_path')
    if os.path.exists(file_path):
        with open(file_path, 'rb') as fh:
            response = HttpResponse(fh.read(),
content_type="application/vnd.openxmlformats-officedocument.spreadsheetml.sheet")
            response['Content-Disposition'] = f'attachment; filename={os.path.basename(file_path)}'
            return response
    return HttpResponse('File not found')
```

Deployment Notes
- Email Settings: Configure email backend in `settings.py` with SMTP settings.
- Session Handling: Ensure proper session management to store summary data between views.

This summarizes the key logic of the application, its main functions, and how it handles file uploads, processing, and emailing in a Django environment.