PDF Contract Alignment & Naming Fix (Test Project)

---

1. Background

In the Al Natour system, PDF contracts are generated dynamically (Arabic & English) using Python and PyMuPDF (`fitz`).
Currently, the logic works but there are two issues:

1. Some fields in the PDFs are not perfectly aligned with the template.
2. The generated file names are not following the required naming standard.

You are required to work on a **separate test project** (not production) to:

- Fix PDF field alignment.
- Update the file naming logic.
- Validate output using provided test data.

---

2. Objectives

Your objectives are:

1. **Fix field alignment** for all dynamic fields inside the PDF:
   - Arabic contract template.
   - English contract template.

Generate the PDF file with this **exact naming format**:
`Prelim Contract <Client Name>.pdf`

2. Examples:
   - `Prelim Contract أحمد بن خالد العتيبي.pdf`
   - `Prelim Contract Saud Al Otaibi.pdf`
3. Generate **four PDFs** using two datasets on both templates.

---

3. Scope

- You will NOT work on production code.

- Build a new **test-only Python project** (recommended) or a mini Django project.
- Use:
  - `arabic.pdf` as Arabic template.
  - `english.pdf` as English template.
  - Arabic font file for proper RTL rendering.
- Use **hardcoded test data** only.

---

4. Required Tools

Set up a Python environment with:

```
pip install pymupdf arabic-reshaper python-bidi
```

---

5. Test Data (Saudi Style)

You will create two datasets that simulate Saudi clients.

5.1 Arabic Dataset (Saudi)

```
arabic_fields = {
    "day": "الأحد",
    "date": "2025/01/01",
    "client_name": "أحمد بن خالد العتيبي",
    "id_number": "1023456789",
    "mobile_number": "0551234567",
    "alnatour_fees_page_1": "15000",
    "application_id_page_2": "12345",
    "application_id_page_3": "12345",
    "alnatour_fees_page_3": "15000",
    "alnatour_fees_page_3_2": "15000",
    "creation_date": "2024/12/15",
    "alnatour_fees_words": "خمسة عشر ألف ريال سعودي فقط لا غير",
    "due_date": "2025/02/01",
    "address": "الرياض، المملكة العربية السعودية",
    "client_name_page3": "أحمد بن خالد العتيبي",
    "client_national_id_page3": "1023456789",
    "client_location_page3": "الرياض، المملكة العربية السعودية",
}
```

5.2 English Dataset (Saudi)

```
english_fields = {
    "day": "Sunday",
    "date": "2025/01/01",
    "client_name": "Saud Al Otaibi",
    "id_number": "2034567890",
    "mobile_number": "+966551234567",
    "alnatour_fees_page_1": "15000",
    "application_id_page_2": "98765",
    "application_id_page_3": "98765",
    "alnatour_fees_page_3": "15000",
    "alnatour_fees_page_3_2": "15000",
    "creation_date": "2024/12/15",
    "alnatour_fees_words": "Fifteen thousand Saudi Riyals only",
    "due_date": "2025/02/01",
    "address": "Riyadh, Kingdom of Saudi Arabia",
    "client_name_page3": "Saud Al Otaibi",
    "client_national_id_page3": "2034567890",
    "client_location_page3": "Riyadh, Kingdom of Saudi Arabia",
}
```

---

6. Generate 4 PDFs (Mandatory)

You will generate four PDFs by testing every dataset against both templates:

Dataset	Template	Output
Arabic Data	Arabic Template	✅
Arabic Data	English Template	✅

English Data	Arabic Template	✅
English Data	English Template	✅

Test Execution Code

```
# Arabic dataset
generate_unsigned_alnatour_contract(arabic_fields, language="A")
generate_unsigned_alnatour_contract(arabic_fields, language="E")

# English dataset
generate_unsigned_alnatour_contract(english_fields, language="A")
generate_unsigned_alnatour_contract(english_fields, language="E")
```

---

7. Fix File Naming

Required Format

```
Prelim Contract <Client Name>.pdf
```

Implementation Example

```
import re

def build_pdf_name(client_name: str):
    client_name = client_name.strip()
    client_name = re.sub(r'[\\/:*?"<>|]', '', client_name)
    client_name = " ".join(client_name.split())
    return f"Prelim Contract {client_name}.pdf"
```

Inside the generator:

```
name = build_pdf_name(fields["client_name"])

output = io.BytesIO(pdf_bytes)
```

```
output.name = name
output.seek(0)
return output
```

---

8. Alignment Fixing (Your Main Task)

You will adjust the coordinates here:

```
field_definitions = {
    "client_name": {"page": 0, "x": 410, "y": 329},
    ...
}
```

Steps:

1. Run PDF generation.
2. Open the output PDF.
3. Inspect:
    - Client name
    - National ID
    - Mobile number
    - Fees
    - Date
    - Address
4. If misaligned:
    - Adjust `x` and `y`.
    - Repeat until correct.
5. Keep Arabic & English configurations **separate**.

---

9. Deliverables

✅ Code

- Python/Django project containing:
    - Script or command to generate PDFs.
    - All dependencies.

    - PDF templates.
    - Arabic font file.

✅ PDFs

You must submit four files:

- `Prelim Contract أحمد بن خالد العتيبي.pdf` (from Arabic template)
- `Prelim Contract أحمد بن خالد العتيبي.pdf` (from English template)
- `Prelim Contract Saud Al Otaibi.pdf` (from Arabic template)
- `Prelim Contract Saud Al Otaibi.pdf` (from English template)

✅ Documentation

A `README.md` containing:

- How to run the script.
- How to install dependencies.
- List of fields edited.
- Final coordinates used for each field (Arabic & English).

---

10. Acceptance Criteria

Task is complete if:

✔ All fields aligned correctly.
✔ RTL Arabic displayed correctly.
✔ File naming format is correct.
✔ No random digits used.
✔ All 4 PDFs generated.
✔ No production code touched.

---

11. Notes

- When uncertain about alignment, **visual correctness** is the priority.
- Document assumptions.

📎 APPENDIX A

Starter Code (Based on Existing Al Natour Logic)

✅ The following code already exists in **Al Natour production project**.
✅ You are provided these as a **starting point reference** to speed up your work.
❌ You must NOT modify production code.
✅ Copy this code into your test project and adapt safely.

A.1 `generator.py` (Main PDF Generator Function)

```
import io
import fitz
from datetime import datetime
from pathlib import Path
from arabic_reshaper import reshape
from bidi.algorithm import get_display


BASE_DIR = Path(__file__).resolve().parent


def generate_unsigned_alnatour_contract(fields, language='A'):
    """
    Existing function from Al Natour project (replicated for test
environment).

    You should:
    - keep the structure.
    - only adjust coordinates and filename logic.
    - NOT change production.
    """

    # Determine template
    pdf_path = BASE_DIR / "contracts" / ("arabic.pdf" if language ==
"A" else "english.pdf")
```

```
    if not pdf_path.exists():
        raise FileNotFoundError(f"PDF template not found: {pdf_path}")

    # Open document
    doc = fitz.open(str(pdf_path))

    # Load Arabic font
    arabic_font_path = BASE_DIR / "contracts" / "font" /
"alfont_com_arial-1.ttf"
    font_name = None

    if arabic_font_path.exists():
        font_buffer = arabic_font_path.read_bytes()
        font_name = "arabic-font"

        for page_num in range(doc.page_count):
            page = doc.load_page(page_num)
            page.insert_font(fontname=font_name,
fontbuffer=font_buffer)

    # Coordinate configuration (Same as production - initial baseline)
    if language == 'A':
        field_definitions = {
            "day": {"page": 0, "x": 445, "y": 212},
            "date": {"page": 0, "x": 345, "y": 212},
            "id_number": {"page": 0, "x": 144, "y": 327},
            "mobile_number": {"page": 0, "x": 389, "y": 357},
            "client_name": {"page": 0, "x": 410, "y": 329},
            "alnatour_fees_page_1": {"page": 1, "x": 129, "y": 241},
            "application_id_page_2": {"page": 1, "x": 133, "y": 286},
            "application_id_page_3": {"page": 2, "x": 275, "y": 166},
            "alnatour_fees_page_3": {"page": 2, "x": 394, "y": 230},
            "alnatour_fees_words": {"page": 2, "x": 381, "y": 246},
            "creation_date": {"page": 2, "x": 390, "y": 204},
            "alnatour_fees_page_3_2": {"page": 2, "x": 176, "y": 311},
            "client_name_page3": {"page": 2, "x": 428, "y": 480},
            "client_national_id_page3": {"page": 2, "x": 400, "y":
500},
            "client_location_page3": {"page": 2, "x": 400, "y": 550},
            "due_date": {"page": 2, "x": 415, "y": 355},
        }
    else:
        field_definitions = {
            "day": {"page": 0, "x": 149, "y": 207},
            "date": {"page": 0, "x": 256, "y": 207},
            "id_number": {"page": 0, "x": 248, "y": 321},
            "mobile_number": {"page": 0, "x": 360, "y": 321},
            "client_name": {"page": 0, "x": 131, "y": 319},
            "alnatour_fees_page_1": {"page": 0, "x": 291, "y": 605},
            "address": {"page": 0, "x": 455, "y": 321},
            "application_id_page_3": {"page": 2, "x": 148, "y": 158},
            "alnatour_fees_page_3": {"page": 2, "x": 130, "y": 191},
            "alnatour_fees_words": {"page": 2, "x": 162, "y": 208},
            "creation_date": {"page": 2, "x": 159, "y": 175},
            "due_date": {"page": 2, "x": 319, "y": 291},
            "alnatour_fees_page_3_2": {"page": 2, "x": 402, "y": 276},
            "client_name_page3": {"page": 2, "x": 167, "y": 410},
            "client_national_id_page3": {"page": 2, "x": 160, "y":
435},
            "client_location_page3": {"page": 2, "x": 112, "y": 458},
        }

    font_size = 9 if language == 'A' else 8

    for key, value in fields.items():
        if key not in field_definitions:
            continue

        text = str(value)

        # Arabic shaping if needed
        if any(ord(c) > 127 for c in text):
            try:
                text = get_display(reshape(text))
            except:
                pass
        config = field_definitions[key]
        page = doc.load_page(config["page"])

        if font_name:
            page.insert_text(
                (config["x"], config["y"]),
                text,
                fontname=font_name,
                fontsize=font_size,
                color=(0, 0, 0)
            )
        else:
            page.insert_text(
                (config["x"], config["y"]),
                text,
                fontsize=font_size,
                color=(0, 0, 0)
            )

    # Save as memory file
    pdf_bytes = doc.write()
    doc.close()

    output = io.BytesIO(pdf_bytes)
    return output
```

A.2 Filename Utility (Required Modification)

Add this function inside the test project (not production):

```
import re

def build_pdf_name(client_name: str):
    name = client_name.strip()
    name = re.sub(r'[\\/:*?"<>|]', '', name)
    name = " ".join(name.split())
    return f"Prelim Contract {name}.pdf"
```

Then apply it:

```
pdf = generate_unsigned_alnatour_contract(fields, language)
pdf.name = build_pdf_name(fields["client_name"])
```

A.3 Test Runner (`test_runner.py`)

```
from generator import generate_unsigned_alnatour_contract
from test_data import arabic_fields, english_fields
from filename import build_pdf_name


def save(pdf, name):
    with open(name, "wb") as f:
        f.write(pdf.read())
    print("✅ Generated:", name)


# Arabic dataset
pdf = generate_unsigned_alnatour_contract(arabic_fields, "A")
save(pdf, "AR_data_AR_template.pdf")

pdf = generate_unsigned_alnatour_contract(arabic_fields, "E")
save(pdf, "AR_data_EN_template.pdf")

# English dataset
pdf = generate_unsigned_alnatour_contract(english_fields, "A")
save(pdf, "EN_data_AR_template.pdf")

pdf = generate_unsigned_alnatour_contract(english_fields, "E")
save(pdf, "EN_data_EN_template.pdf")
```

A.4 Dependencies (`requirements.txt`)

```
Same as the used dependecines in alnatour project
```