User Guide

# About

The purpose of this app is to fill in three PDF forms with information from an input spreadsheet and its corresponding office spreadsheet and then to combine the three by household account (which are distinguished by their 18-character identifier).

This needs a folder of the three template files. If their names need to be changed, you can change two of them yourself in a way similar to what you have done before, but changing the third-party authorization form may require a little bit of digging, or you may request my assistance. If this is something that needs to be changed often, please submit a request to change the input to be more flexible.

# Requirements

The requirements are Python 3, pandas 2.0.3, and pdfrw 0.4.

# Definitions

Template PDFs are the PDFs that you wish to fill. As of November 2023, they were defined as a client contract conflict disclosure form, a client suitability form, and a third-party authorization form. There must be a folder called “template pdfs” in the same directory as `app.py` that at least contains the third-party authorization form, exactly “JBS DPL - Client 3rd Party Auth 2023-09-7.pdf”. The other template PDFs can be saved elsewhere, and their location would be indicated by the variable `template\_pdf\_folder`. The PDF files of the contract conflict disclosure form and the suitability form would be saved in the variables, `template\_pdf\_contract\_conflict\_disclosure` and `template\_pdf\_suitability\_form`, respectively.

The inputs that contain the data to use to fill the PDFs should be on their own single Excel sheets. These should both be saved in a directory that you can indicate with `excel\_folder`. The Excel sheet that will fill the contract conflict disclosure form and the suitability form can be indicated with `input\_excel\_name`, and the Excel sheet that contains the corresponding office’s information will be indicated by `office\_excel\_name`. To control which office for the app to expect, use `office\_name`.

The final output folder of the filled and merged PDF documents will be indicated by the variable `output\_folder`. There are in-between folders that will also contain the non-merged PDF files, which will be in `output\_pdfs\_split1` for the client suitability form, `output\_pdfs\_split2` for the third-party authorization form, and `output\_pdfs\_split3` for the client contract conflict disclosure form.

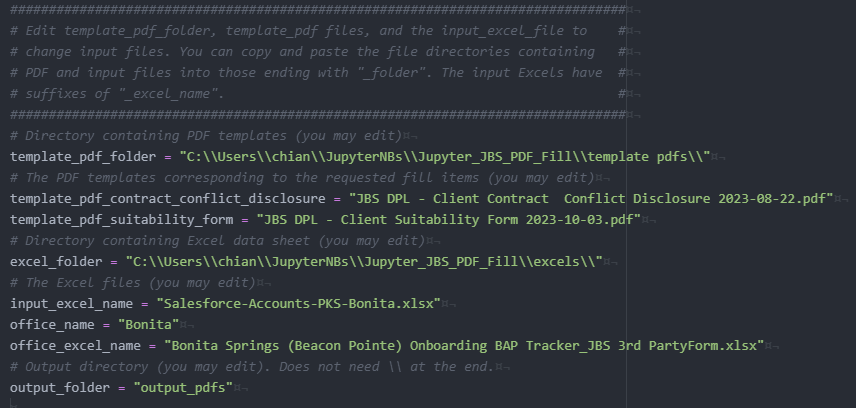
# Usage Instructions

## Quick Overview

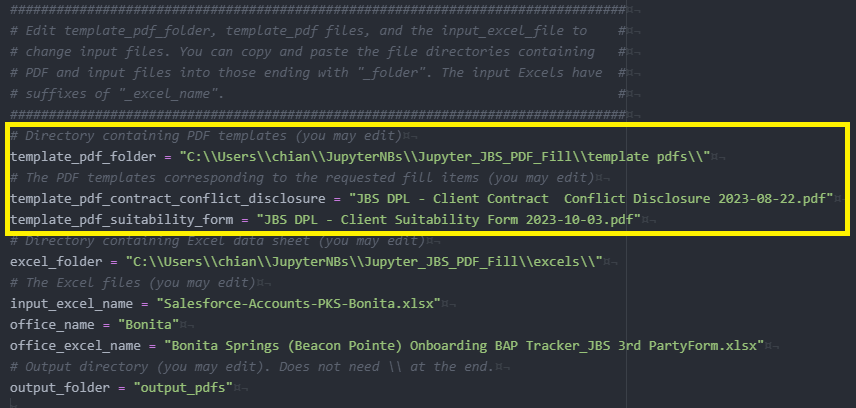
Open `app.py` in Visual Studio Code and run the program.

## Modifying File Names

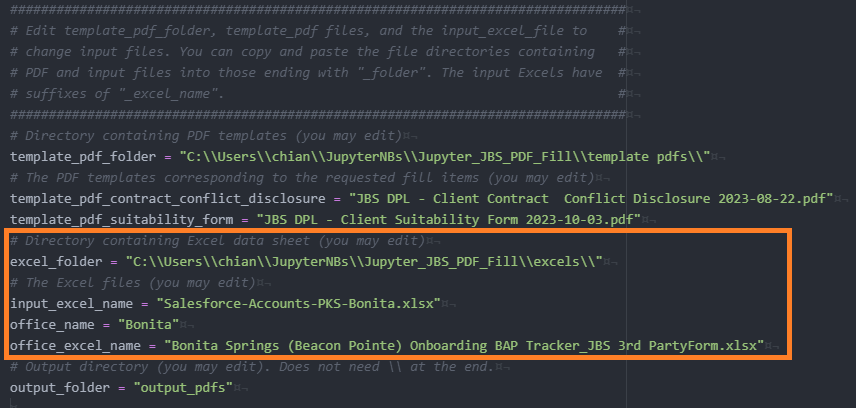
To change the file names or file paths, modify the strings near the top of the Python file `app.py`. They will be located under a big comment box, which begins on line 9.



In the image below, the variables enclosed in the yellow box are related to the template PDF files.



In the image below, the variables enclosed in the orange box are related to the input Excel sheets.



Note that the `template\_pdf\_folder` and `excel\_folder` need to have “\\” throughout the string and at the end of the string, whereas the `output\_folder` is just the name and does not require a “\\” at the end. The folder of outputs will be located in the same directory as `app.py`, and the data it contains **will be overwritten if it already exists** or created anew if it does not exist.

## Column Configuration

I have decided to do away with the JSON. Instead, column mappings are saved in variables prefixed with “cmap”. The column mappings for the contract conflict disclosure are saved in the variable `cmap\_ccd`, and the column mappings for the suitability form are saved in the variable `cmap\_suitability` because these are generic. The column mappings for the third-party authorization form are saved in the `pdf\_writer\_office\_bonita.py` file, located in the `src` folder because these are specific to the office of origin.

You can adjust the column mappings yourself, but I would recommend asking for assistance. The definition is similar to other scripts you have requested. Each input Excel column maps to a PDF field. If it maps to a list, then the data is duplicated into the new columns in the list. (For example, this mapping, `"Primary Contact: Full Name": ["Client Name", "Client Printed Name"]`, will make two new columns, “Client Name” and “Client Printed Name”, both containing the data from “Primary Contact: Full Name”.

Almost every column mapping can be edited, but make sure that the left-side key has a column in the input Excel sheet and the right-side values have fields in the template PDFs. Do not change the mapping `"AccountID-18Char": "Household ID"` because this controls how the data will be grouped.

## Valid Offices, How to Configure the Office, and How They Work

As of November 2023, only one office has been requested.

To change the office functionality for the third-party authorization form, this will be controlled by the variable near the top of the file, the `office\_name` (line 24).

The table below describes the rules and unique characteristics of the office.

|  |  |
| --- | --- |
| Office | Rule(s) and Characteristic(s) |
| Bonita | Bonita’s global office information is that the “IIA Address” is “9240 Bonita Beach Road, Suite 2211, Bonita Springs, FL 34135” and the “IIA Telephone” is “239-676-5676”.  Excel sheets from the Bonita office have rename columns (cmap) of 'Contract / Policy #': "Investment Account", 'IAR': "Investment Advisor", 'IAR Email': "IIA Email", and 'AccountID-18Char': 'Household ID'. All of these except for the final one can be changed.  The “Investment Description” is the string concatenation of (‘Source/Carrier’ | ‘If Other, Please Specify’) with ‘Product Name’. The choice for the first part of the string depends on if the data in ‘Source/Carrier’ is equal to “other”. |

# Troubleshooting

I did not write any warning messages this time, but the most likely errors to occur are the `FileNotFoundError`, which means that your filenames or file directories are probably incorrect, or a `KeyError`, which means that a PDP field name, Excel column name, or column mapping was written incorrectly.

# Warnings, Issues, and Considerations

Warnings

* Strict Usage
  + Every string must be *exactly* correct. There is no interpretation for typos or nearness. For example, the string “Dolly” is not the same as “Doll”, even if that was just a typo and they mean the same thing. If your input Excel has leading 0’s somewhere yet the office Excel does not have leading 0’s for the same information and these data are both relevant, then they will be treated as totally different strings.
* Special Characters
  + Do not use fancy character encodings or very special characters. It might be possible to use special characters for the input spreadsheet’s data because these are just being copied, but column headers and PDF fields depend on exact string matches.
* File Safety
  + Be careful about overwriting files. Back up your data somewhere else and work in clean, empty directories.

Issues

* There are rare occasions where some relevant data is missing, or unique households exist. When these are found, they are printed out to console.
  + The app will catch and report if household ID or account numbers are missing from the office Excel sheet, and it will catch and report if a household ID is missing from the input Excel sheet, because these are all seriously relevant data. It **will not write** PDF files for these rows.
  + The app will catch and report if there are more than three account owners (unique names) from the office Excel sheet. It **will write** PDF files for these rows, but the PDF form will not fill in the signature part. This is a rare occasion because as far as I have been seen and told, most household accounts have just the owner and the joint owner. Sometimes there are typos.

Further Considerations

* If renaming the template PDFs is a very common thing, then please request me to add the third-party authorization form name variable to `app.py` so that you have more convenient access.