Extracting MRZ (Machine Readable Zone) Data from Passport using Python

The MRZ of a Passport: In a passport booklet the MRZ (Machine Readable Zone) is on the bottom of the passport and contains two rows and each row has 44 characters.
Characters used are alphabets A-Z, numbers 0-9 and a filler character '<'

Image Courtesy: info.viselio.com/machine-readable-passport/

1. Letter 'P' stands for passport
2. This space is provided for a character indicating a passport's type. If the country issuing the passport doesn't have any passports' types, symbol '<' is used instead
3. Next three characters represent the country that issued the passport in ISO 3166-1 alpha-3 codes
4. Next 39 characters are reserved for surname and first name. First is the surname which is divided from the given name with two filler characters '<'. If there are two surnames or two given names, they are between themselves divided with a single filler character '<'.
5. These 9 characters are the passport's number
6. This character is a check digit. It is calculated with a help of an algorithm and it is based on the passport's number
7. Next three characters represent the nationality of the passport holder
8. These characters are the date of birth in the YYMMDD format
9. Check digit based on the date of birth
10. Sex of the passport holder. 'M' for male, 'F' for female, and '<' for unspecified
11. The expiration date of the passport in the YYMMDD format
12. Check digit based on the passport's expiration date
13. Personal number
14. Check digit based on the personal number
15. This is a check digit for positions 1 to 10, 14 to 20, and 22 to 43 on the second line

How the Data Quality measure works:

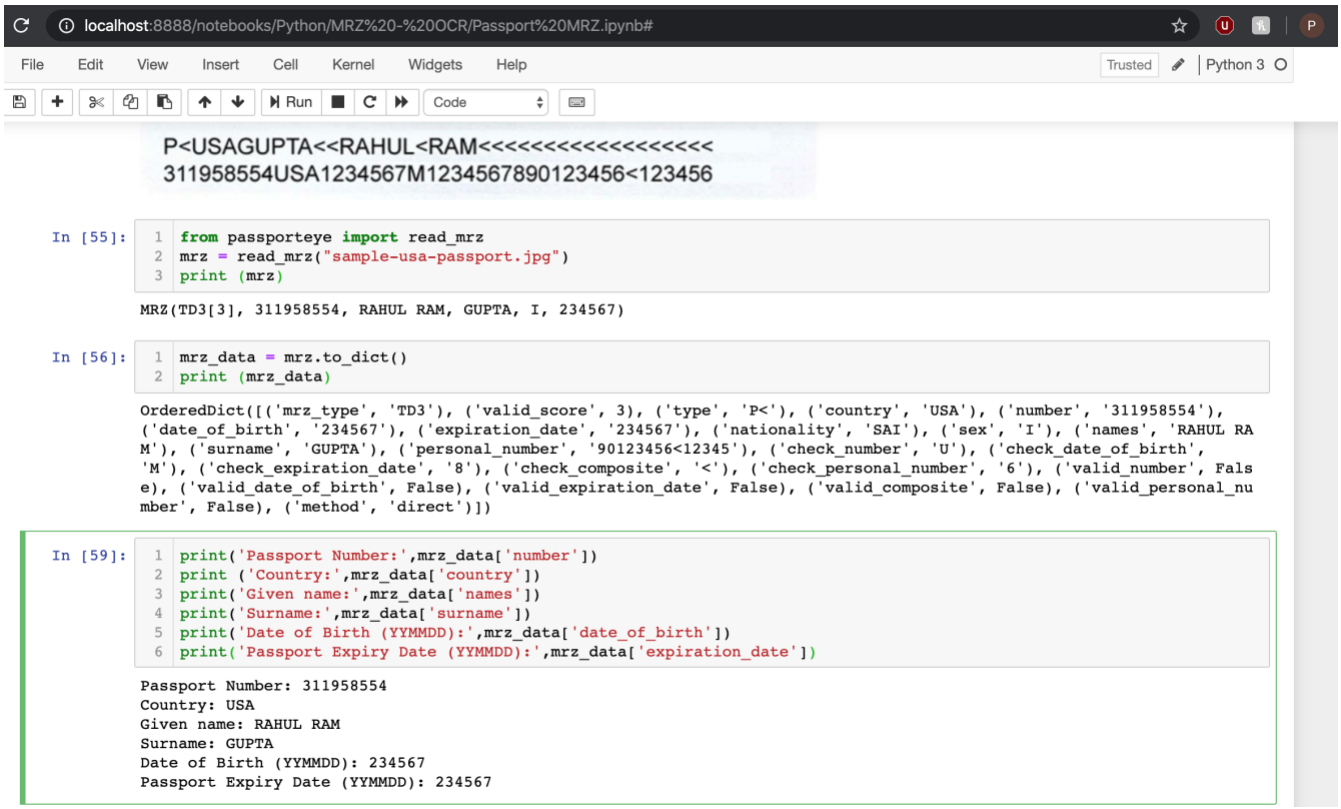
The data entry of a new tourist into the database at the first point of entry (first hotel of stay) has to happen perfectly so that subsequent search and use of the data with the unique key (passport number) by other hotels can happen seamlessly with accurate data. To make sure the data entry into the database at the first point is correct, I looked into the possibility of utilising the MRZ (Machine - Readable Zone) in the passport *(as explained above)* which is the bottom two lines on the information page, so that it can be used as a second key identifier for the person via the passport scan that needs to be uploaded. This can be used to compare PII with manually inputted data and generate an error if they don't match while registering tourist initially into the database.

I used the PassportEye package for Python from PyPi (Python package index); utilising the numpy, scipy, matplotlib and scikit-image libraries, to crop out the MRZ from passport and using this along with Tesseract OCR (Optical Character Recognition) engine developed by Google, the MRZ data can be extracted into string or json format. I used the package along with the Tesseract OCR engine on Python to extract MRZ data from sample passports (attaching a pdf document with examples along with a UML Class diagram of the database designed using LucidChart). Next, by comparing some of the PII extracted from the MRZ (Date of Birth, Country Code, Name, Gender, Passport Number, Passport Expiry Date) with the manually entered data, the data quality can be verified at the first point of entry like a two factor authentication. I tested the Python code using my personal passport as well, it works well in extracting the MRZ data from jpg, pdf and png file formats of good resolution.

Sample USA Passport:

Image Courtesy: immihelp.com/docs/sample-usa-american-passport.html

Python: PassportEye package; scipy, numpy, matplotlib and scikit-image libraries & Tesseract OCR Engine. Output of MRZ data from python; extracted from 'sample USA passport' above:

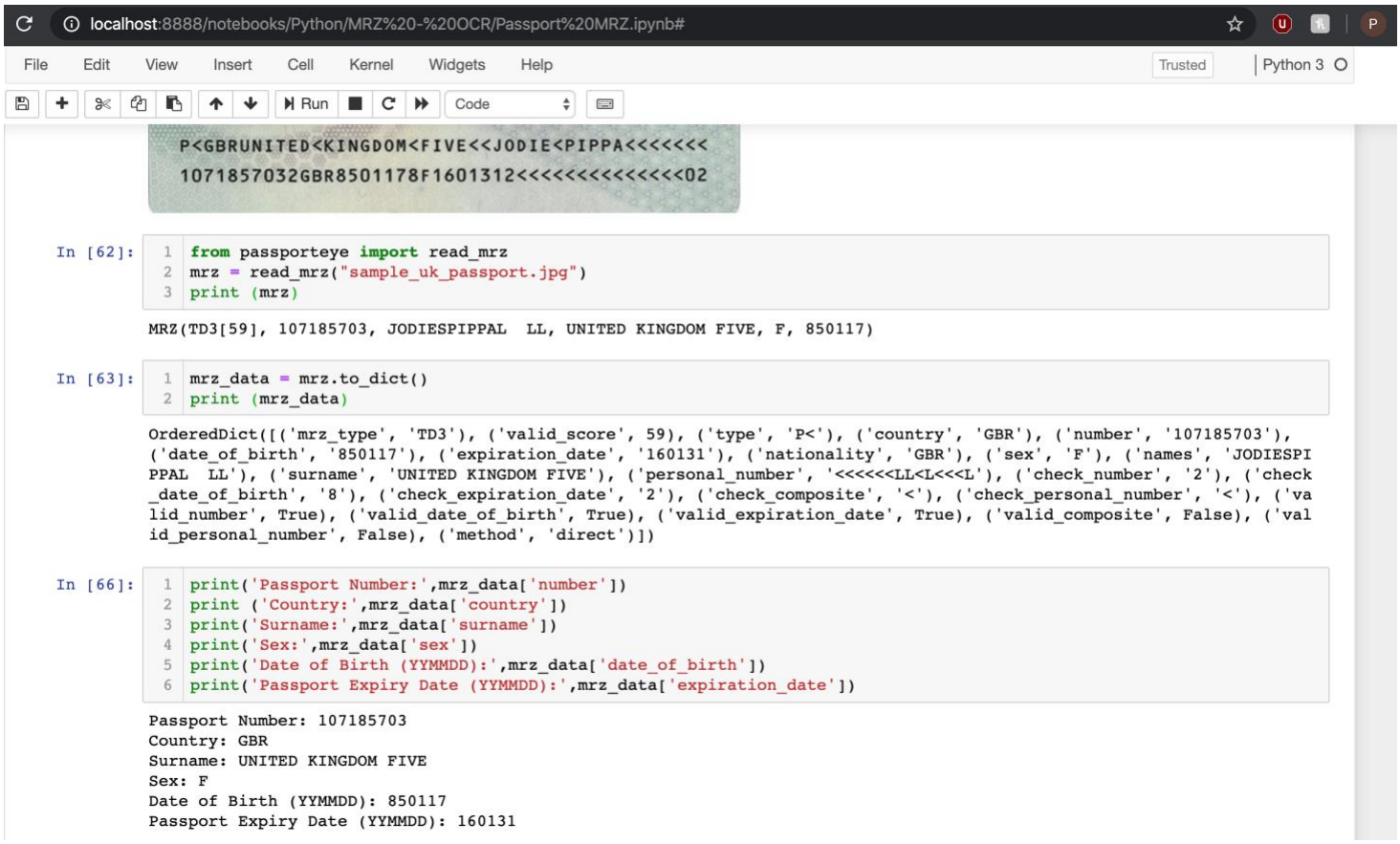


** In this particular example the Date of Birth & Passport Expiry Date is incorrect as the sample passport has a randomly generated MRZ. Only name and passport number from MRZ match the information zone of passport.*

Sample UK Passport:

Image Courtesy: onfido.zendesk.com/hc/en-us/articles/360003550639-Right-to-Work-Image-photo-requirements-of-your-document

Python Output:



UML Class Diagram of Database: