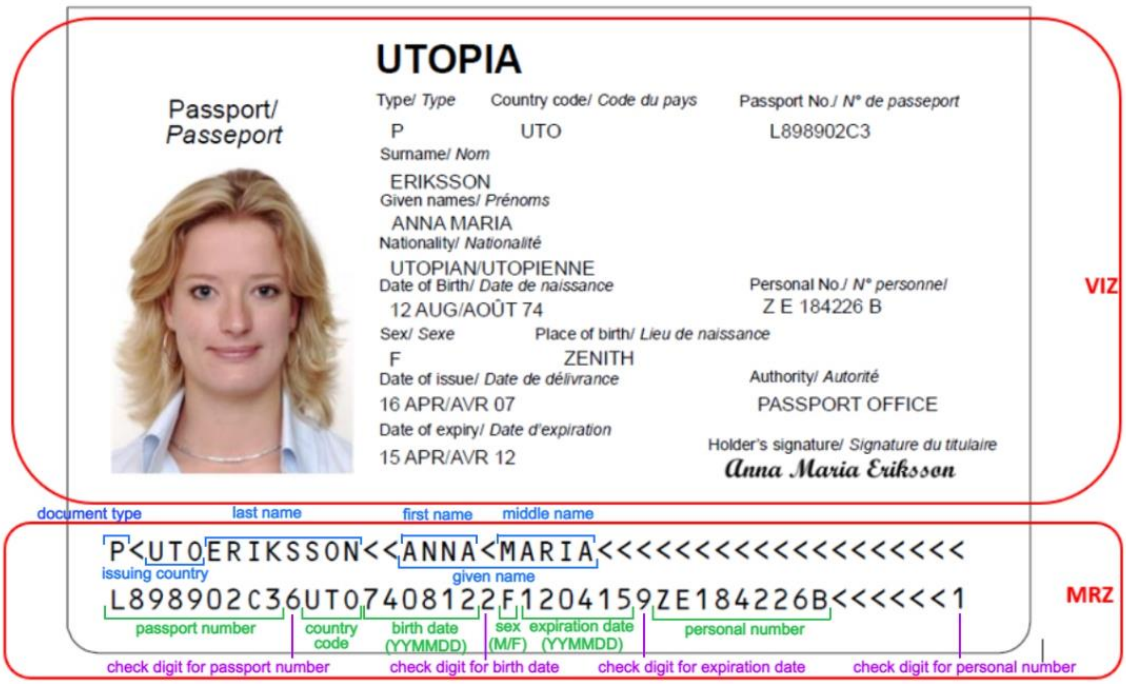
Requirement Testing of Final Project of SSW-567 A

Background Information

Description
A machine-readable travel document (MRTD) should present information necessary for global interoperability using visual inspection and machine-readable (optical character recognition) means. The following figure is an example. It is composed of two parts: a visual inspection zone (VIZ) and a machine-readable zone (MRZ).

The MRZ contains two lines. The first line specifies the Type of passport, the issuing country, and the name of the holder. The second line specifies the passport number, country code, birth date, gender, expiration date, and personal number. In addition to these information fields, there are four check digits inserted in between and at the end of the information fields. In the above example, they are “6”, “2”, “9”, and “1”. The check digit serves for checking the correctness of the information fields.
Following is an example illustrating the algorithm for calculating the check code. Assume that the calculation method for composite check digits is the same for all MRTDs.

Example 2 — Application of check digit to document number field

Using the number AB2134 as an example for coding a 9-character, fixed-length field (e.g. passport number), the calculation will be:

Sample data element:	A	B	2	1	3	4	<	<	<
Assigned numeric values:	10	11	2	1	3	4	0	0	0
Weighting:	7	3	1	7	3	1	7	3	1
Step 1 (multiplication) Products:	70	33	2	7	9	4	0	0	0

Step 2 (sum of products) 70 + 33 + 2 + 7 + 9 + 4 + 0 + 0 + 0 = 125

Step 3 (division by modulus) 125 = 12, remainder 5
10

Step 4. Check digit is the remainder, 5. The number and its check digit shall consequently be written as AB2134<<<5.

Encoding Steps	
1.	Multiply the numeric values of each digit with a weighting sequence. Note that “A” maps to 10, “B” maps to 11, and thereafter. Special symbols, such as “<”, always map to 0. You should always use the same weighting sequence of 7, 3, and 1 as shown in the above example in the scope of this project.
2.	Add up all the products from the previous step.
3.	Divide by a modulus of 10.
4.	The remainder will be the check digit, which is the final output of this algorithm.

Original Requirements

Suppose you are a developer for the project to implement a system that can read the MRZ of a travel document, process and obtain its fields, and check the fields against the check digits. Following are some requirements and specifications of your system

1.	The system shall be able to scan the MRZ of a travel document using a hardware device scanner and get the information in MRZ as two strings (line 1 and line 2 from the above Figure). Note that you do not need to worry about the implementation of the hardware device. But you need to define this method for the software part. This means that you define an empty method for this function.
2.	The system shall be able to decode the two strings from specification #1 into their respective fields and identify the respective check digits for the fields, following the same format in the above example.
3.	The system shall be able to encode travel document information fields queried from a database into the two strings for the MRZ in a travel document. This is the opposite process compared to specification #2. Assume that the database function is not ready. But for testing purposes, you need to define a method for database interaction and leave it empty.
4.	The system shall be able to report a mismatch between certain information fields and the check digit. The system shall report where the miss match happened, i.e., which information field does not match its respective check digit.

Result of Checklist Reviewing

Abbreviation Rules: Requirement #01 == R01			
Criteria	Y / N / NA	Exceptions	Notes
Prioritized			
Tradeoffs between requirements are clear.	N	R01 does not mention the system response by message if the scan is not successful.	R01 shall be divided into 2 requirements, each one must be clear.
		R02's description is unclear.	R02 does not clearly and detailly describe how two lines of info will be processed as string objects to be stored as data fields and then convert to integer objects by the algorithm.
		R03 misses important details of handling the data between the program and the database.	R03 does not elucidate what the dataset in the table of the target database looks like.
		R04 misses critical solutions that the system needs to process in functionality.	R04 does not expound what the error message likes like. And also, it does not describe if the passed and failed check will be recorded as a log item.
Multiple dimensions have been considered, such as cost, customer value, and development risk.	N	Missing development risks.	Each data scanned from the travel doc shall be encrypted and thence stored in the database for security concerns.
All product stakeholders have provided input to the prioritization process.	N	Some security concerns are not mentioned in the requirement even though it seems kind of irrelevant.	The custom board officers (scan operators) should not access the product's database except for admins.
The requirements are realistically distributed among the priority levels.	NA		There is no such priority-level notion, all requirements shall be considered and implemented fully.
Unambiguous			
Each requirement is clear to the intended audience, possessing a single interpretation.	NA		Requirement items have been added and refined, so this is now irrelevant.
Terms are defined where necessary	Y		

and used consistently.			
The requirements are devoid of weak words (easy, fast, etc.) and unbounded lists (such as, including, ...).	Y		
Diagrams, algorithms, use cases, tables, or other devices are used to reduce ambiguity where appropriate.	N	Missing critical concerns.	See the above yellow color highlighted to understand.
Verifiable			
Each requirement is unambiguous.	N	Missing critical concerns.	See the above yellow color highlighted to understand.
The implementation of each requirement can be clearly and effectively established via demonstration, inspection, or testing.	N	Missing critical concerns.	See the above yellow color highlighted to understand.
Non-functional requirements (performance, reliability, etc.) are quantified using an appropriate scale of measure.	N	Missing most of the system response mechanism of hinting to the operators when the error happens.	
Consistent			
Each requirement is represented only once in a specification and referenced where needed.	N	Missing critical concerns.	See the above yellow color highlighted to understand.
Each requirement is internally consistent with other product requirements at its level.	Y		
Each requirement is externally consistent with requirements at other levels (product, business, market, etc.).	NA		
Traceable			
Each requirement is uniquely and persistently identified.	Y		
Each requirement is written as concisely and simply as possible.	N	Missing critical concerns.	See the above yellow color highlighted to understand.
Each requirement expresses only one function or idea.	N	Missing critical concerns.	See the above yellow color highlighted to understand.

Revised and Refined Requirements

1. The system shall be able to scan the MRZ of a travel document using a hardware device scanner and get the information in MRZ.
2. The system's application shall be implemented in object-oriented programming methodology, which means the program should be encapsulated as a class.
3. The system's application program's class shall own setter and getter methods.
4. The system's application program can be coded in a programming language of Python or Java.
5. The system's application shall use a database of SQLite or MySQL.
6. When the system malfunctions, it shall hint to the operator with the displaying message.
7. When the system cannot scan or cannot identify the informational code of MRZ, the error message shall show to the operator.
8. After the system successfully scanned the information, two lines of string shall be assigned to a private member data field.
9. String line one and line two should be assigned to two different private member data fields in the class.
10. Dividing the gotten string line two into respective substrings as their respective meaning, e.g., passport string, birthday string, etc.
11. After getting each substring of two strings, convert any single digit to an integer primitive type then process the algorithm.
12. The database shall be always with the application console in a network connection, if disconnected, the system should display the message of error or failure.
13. During the testing phase, the developer shall mock the function of the database, and mimic the processing and storing behavior.
14. The mocking module for mimicking a database can be programmed as another class.
15. The dataset in the table of the database shall match the data field of the programmed class in the programming language.
16. Each scanned information from each of the travel documents of passengers should be recorded in the database as a row of the table. Using two dimensions arrays, array lists, or linked lists to mock the dataset.
17. Before storing each passenger's information in the database, each piece of information which means two lines of string shall be encrypted and then written into the table. If other systems like to view the records, decrypt these from the database. This mechanism is for security concerns.
18. The system should be able to pop the message "Check Passed!" if the check digit does match after the algorithm process.
19. The system should be able to pop the message "Check Digit Mismatches!" if the check digit does not match the digit after the algorithm processes the gotten string's each substring.
20. The scanning operators cannot access the database to retrieve the records and logs for security concerns.
21. The system admin can access the database for maintenance.