**Task Report**

**Python Validation Model**

Contents

[**1.** **Version history** 4](#_Toc101816912)

[**2.** **Introduction** 4](#_Toc101816913)

[2.1. Overview 4](#_Toc101816914)

[2.2. Source code 4](#_Toc101816915)

[**3.** **Contributors** 4](#_Toc101816916)

[3.1. List of contributors 4](#_Toc101816917)

[3.2. Initial task division 4](#_Toc101816918)

[**4.** **Challenges** 5](#_Toc101816919)

[4.1. Solved 5](#_Toc101816920)

[4.1.1. Python version issue 5](#_Toc101816921)

[4.1.2. Modeling upwards vs downwards 5](#_Toc101816922)

[4.1.3. Choice of database type 5](#_Toc101816923)

[4.2. Unsolved 5](#_Toc101816924)

[4.2.1. Records stored in csv without dedicated out-of-box database manager 5](#_Toc101816925)

[4.2.2. Object-oriented vs Procedural approach 5](#_Toc101816926)

[4.2.3. Storing and delimiting many-to-many relations 5](#_Toc101816927)

[4.2.4. Incomplete Hangman testing scope 5](#_Toc101816928)

[4.2.5. Searching by relations constrained by related IDs stored in only one record 5](#_Toc101816929)

[**5.** **Sources** 6](#_Toc101816930)

[5.1. Python documentation for base python functions 6](#_Toc101816931)

[5.2. Pandas user guide 6](#_Toc101816932)

[5.3. Stackoverflow threads with helpful information 6](#_Toc101816933)

[5.4. Other useful links 6](#_Toc101816934)

[5.5. Data sets for Hangman application 6](#_Toc101816935)

[**6.** **Ideas for future development** 6](#_Toc101816936)

[6.1. Add relations between Test Case and Test Cycle 6](#_Toc101816937)

[6.2. More data sets and balance update in Hangman game 6](#_Toc101816938)

[6.3. Overhaul to utilize existing, more flexible database solution 7](#_Toc101816939)

[6.4. Graphical User Interface 7](#_Toc101816940)

[**7.** **Completion report** 7](#_Toc101816941)

[7.1. Validation model application 7](#_Toc101816942)

[7.2. Hangman – Python application to be tested 7](#_Toc101816943)

[7.3. Final task division 7](#_Toc101816944)

[**8.** **Retrospection** 7](#_Toc101816945)

[8.1. What went well 7](#_Toc101816946)

[8.2. Difficulties during project 8](#_Toc101816947)

### **Version history**

|  |  |  |
| --- | --- | --- |
| **Revision** | **Changes** | **Date** |
| 0.1 | First draft published | 24-04-2022 |
| 0.2 | Footer cleanup, contributors listed | 24-04-2022 |
| 0.3 | Added sources, Validation App completion report, idea for future improvement | 25-04-2022 |
| 1.0 | Release version | 25-04-2022 |

### **Introduction**

### Overview

This document serves as the completion report for two-part task, which consisted of creating:

1. validation model and CLI application for managing test records, with proper documentation
2. Python application, with validation planned using above model and application

### Source code

All files created for the purpose of this task can be found at <https://github.com/dlukaszx/Project>

### **Contributors**

### List of contributors

* Dawid Łukaszewski
* Marta Czaja
* Piotr Skierkowski

### Initial task division

|  |  |  |  |
| --- | --- | --- | --- |
| Task | Dawid | Marta | Piotr |
| Validation model |  | X |  |
| CLI app for the model (all) |  |  | X |
| Model documentation |  | X |  |
| Python app to be tested | X |  |  |
| Testing scope | X | X | X |
| Test execution | X | X | X |
| Completion report |  | X |  |

### **Challenges**

### Solved

### Python version issue

Development of apps was initially split between Python version 3.6 and 3.10. As task clearly listed 3.6 version, we opted to downgrade to unify the code and ensure compatibility.

### Modeling upwards vs downwards

One of the things considered was keeping child data in parent record (e.g., all related Test Cases listed in Test Case Description). However, since TC has to be related to single TCD it was more efficient to ask for parent TCD ID upon TC creation and store the value in TC record. Same rule applies to TC-TR and Cycle-TR: they are all one-to-many relations.

### Choice of database type

The group decided to use pandas module for interaction with csv files. We have heard of this module before, and this task was as opportunity to learn more about pandas.

### Unsolved

### Records stored in csv without dedicated out-of-box database manager

Validation App in current state is not able to rely on SQL queries to search for particular records and relations. To make the database easier to use it would have to be ported to existing database manager, or all missing relations would have to be created manually.

### Object-oriented vs Procedural approach

The initial version of both apps has been procedurally written to gain better understanding of python syntax. However due to time limit we were unable to create object equivalents. In the long run editing procedural code will generate many repetitions and will not as easily reusable.

### Storing and delimiting many-to-many relations

Currently implemented version of validation app stores TCD-TP relation in TCD database. All related TPs IDs are stored in one field and delimited with "+" character. There should be another database file for storing relations, but due to lack of time it was the fastest method for this.

### Incomplete Hangman testing scope

While it is widely understood that it is impossible to test an application in 100%, we feel that our testing scope for this app could cover more functionalities over time.

### Searching by relations constrained by related IDs stored in only one record

Keeping relations with only one of two records necessitated creating custom multi-leveled queries per type of relation. It is not flexible solution in case new records were to be added.

### **Sources**

### Python documentation for base python functions

* <https://docs.python.org/3/tutorial/>

### Pandas user guide

* <https://pandas.pydata.org/docs/user_guide/index.html#user-guide>

### Stackoverflow threads with helpful information

* <https://stackoverflow.com/questions/517970/how-to-clear-the-interpreter-console>
* <https://stackoverflow.com/questions/3207219/how-do-i-list-all-files-of-a-directory>
* <https://stackoverflow.com/questions/31536835/extract-value-from-single-row-of-pandas-dataframe>
* <https://stackoverflow.com/questions/56436024/python-3-count-number-of-rows-in-a-csv>
* <https://stackoverflow.com/questions/26640129/search-for-string-in-all-pandas-dataframe-columns-and-filter>

### Other useful links

* <https://pandas-xlsxwriter-charts.readthedocs.io/chart_pie.html>
* <https://www.shanelynn.ie/pandas-iloc-loc-select-rows-and-columns-dataframe/>
* <https://peps.python.org/pep-0008/>

### Data sets for Hangman application

* <https://pypi.org/project/english-words/>
* <https://www.wordexample.com/list/proverbs-in-english>

### **Ideas for future development**

### Add relations between Test Case and Test Cycle

Current version of application only has Test Cycle related to Test Result, what means, that Test Case scope for each Test Cycle is the same. If there is new functionality implemented in tested application, all new Test Cases will be included in reports generated for older Test Cycles and marked as ‘NOT RUN’. The same situation will occur for new test Cycles if functionality is removed from tested application.

### More data sets and balance update in Hangman game

Available version of application contains nearly 70000 unique passwords, but most of them are single words. New data sets can be easily added in the future. Game also might need balance update. Six lives may be enough for proverbs, where it is easier to guess a correct letter, but there can be a problem with single words.

### Overhaul to utilize existing, more flexible database solution

If done early during development, this approach will provide long-term solution with proper support.

### Graphical User Interface

To facilitate better user experience.

### **Completion report**

### Validation model application

* Add new TP, TCD, TC, TR, Cycle – DONE
* Add relations between elements – PARTIALLY DONE
  + Add TCD – TP (many to many) relation – DONE
  + Cycle-TR, TR-TC, TC-TCD (one to many) relations – DONE
  + Cycle-TC – TO DO
* Display all TC for selected TP – PARTIALLY DONE
* Display all TR for selected TP – TO DO
* Display all TR for selected Cycle – PARTIALLY DONE
* Export validation report for specific Cycle as Excel file – DONE

### Hangman – Python application to be tested

* Choose game mode – DONE
* Guess letters and draw hangman – DONE
* End game and restart option – DONE
* Test application – DONE

### Final task division

|  |  |  |  |
| --- | --- | --- | --- |
| Task | Dawid | Marta | Piotr |
| Validation model |  | X |  |
| CLI app (framework) |  |  | X |
| CLI app (excel integration) |  |  | X |
| CLI app (adding relations) |  |  | X |
| CLI app (filtered display) |  | X |  |
| Model documentation |  | X |  |
| Python app to be tested | X |  |  |
| Testing scope | X | X | X |
| Test execution | X | X | X |
| Completion report | X | X | X |

### **Retrospection**

### What went well

* Efficient task division
* Achieved functional app/game to be tested
* Achieved partial validation database functionality with record submission and result export
* Opportunity to grow Python skills
* Review responsibility shared with teammates responsible for given segment

### Difficulties during project

* Different time availability for each team member
* Infrequent communication due to varied worktime & dedication to task distribution
* Work under time constrains
* Insufficient research on Python versatility before task distribution