D&D Utilities

Presented by: Ethan Powell

Bachelor of Science in Cybersecurity & Computer Science

Presented to: Dr. Sean Hayes

Expected Graduation Date: May 2022

Table of Contents

Problem Statement

Tabletop roleplaying games can provide a great social experience for all ages and help young adults with social interactions, learning basic math, and developing problem solving strategies that can be used throughout their life in a fun environment. Dungeons and Dragons 5th edition provides an excellent entry point for those who wish to get into this style of learning; however, creating, storing, and editing a character can be one of the main issues that can prevent people from being able to involve themselves with such a format. This is a result of difficult to understand calculations and the intimidating number of items on a character sheet. A software-based assisted game tool can alleviate these intimidating factors and allow for anyone who can use a computer to make their way into tabletop gaming. With a software-assisted game tool, a player need only know the very basic values that need to be input, and the tool completes all the necessary calculations and performs the necessary tracking to alleviate a large portion of the load that can be placed on the player. This tool provides a simple to use, portable game tool for tracking character data, referencing values, and storing large amounts of this data without the need for hundreds of pieces of paper. The creation of a software-assisted game tool for Dungeons and Dragons 5th edition is necessary for allowing more people to have access to the game overall and prevent miscalculations that can result from human error.

Research and Background

Deciding a Language

The first issue a developer must face is the language in which they implement their software. With the ever-growing coding language eco-system, there are many options to choose from, each with their own benefits and detriments. The three languages that most drew my attention were C++, C#, and Python.

C++ - Using a sledgehammer to kill a fly

I originally started with a C++ implementation, as it was my most utilized language thus far in my coding career. It offered high-performance option in which I felt comfortable, in addition to having long time support; however, despite all the benefits that C++ offers, the amount of control was much more than I needed. After diving into C++ and looking at GUI implementation and file interaction, especially PDFs, the amount required to perform the actions would practically be a program by themselves.

C# - Getting better, but not quite there

C# was the next language I took a look at, and it was almost everything that I needed. C# offers the portability of a scripting language while having similar memory control as C++. It has many libraries available to it, and it runs on the .NET framework, providing portability and GUI control; however, when I began to delve into the actual GUI development and file interaction that C# offered, it still had similar issues as C++ in that the control it offers was much more than I needed as well as requiring the development of tools that would take a large portion of production time.

Python – Easy to use, lots of support.

Python was the final language I looked at, and it ended up being the best fit for the application I was attempting to develop. The main draw to python was the documentation and library availability catered towards beginner developers. I had not had much experience with python but coding an entire project in it was much easier than I anticipated. Python has quite the amount of overhead, but for a small application like mine, the need for speed and performance wasn't as necessary. Lastly, the documentation and support for python is very vast and geared towards younger developers, allowing it to become a strong fit for the language that I used.

GUI Implementation Prospects

C++ - Hardware not software

C++ offers significant hardware control but was never originally intended as a GUI based programming langued when used on its own. There are several GUI libraries that C++ offers, but in the overall availability of GUI libraries pales in comparison to other languages.

C# - Better for GUI, but still involved

C# offers much more support in developing a GUI, especially with the help of the .NET framework at its back; however, as I researched creating a GUI and dug through documentation, I found that the creation of a GUI in C# was still fairly involved and had few tutorials that were helpful to me.

Python – A good option that went underused

Python was the last language I looked at and researched, and it proved to be the most promising in terms of GUI support. In addition to having libraries that were useful for other actions I was performing (like reading from PDF files), it offered a fairly simple GUI system

through the PySimpleGui library, although I was not able to implement the GUI functionality in the end.

Library Availability

C++ - Robust

The C++ libraries are extensive and offer a robust series of pre-made functions for memory management and data structures, yet when it comes to interacting with files, I was unable to find specifically useful libraries for pulling form filled information from PDFs. Although, C++ does offer SQLite implementation for database usage and creation.

C# - Little to comment

I was unable to research C# thoroughly, as the main draw to C# was the ability to implement a GUI through the .NET framework. As such, I cannot offer much on the usefulness and extensiveness of C# libraries, although I am under the impression that C# has many similar capabilities to C++.

Python – Extensive

Python's extensive library list was my main draw in deciding to use it as the main language implementation for this project. It offered many diverse libraries, involving SQLite in a native implementation, PyPDF2 for form-filled information gathering, and finally PySimpleGUI for GUI implementation. This aspect resulted in Python becoming the main choice of implementation for this project.

Documentation

C++ - Difficult to understand unless practiced

C++ has a wealth of documentation associated with it; however, much of the documentation involved with C++ can be very technical due to the control required in order to

operate C++ effectively. When attempting to discern the documentation from a young-developer standpoint, I ended up with more questions than I had answers, causing the project to become bogged down with learning how to use a library rather than actually utilizing it.

C# - Good start, hard to follow

C# has a strong start to its documentation with Microsoft having an official website that helps teach the introduction of C#, yet when attempting to understand how to involve GUI and other libraries, I felt as though I would have spent more time learning the language than actually programming. This then led me to my final choice of Python.

Python – Easy to learn, hard to master

Python offered an extensive repository of documentation that was difficult to follow at first, but because of Python's simple syntax, it became much easier to use than anticipated. I feel as though Python offers an easy introduction to basic functions and library usages; however, one can spend large amounts of time truly diving into large libraries such as Pandas. PyPDF2 offered an easy introduction to exactly what I wanted to accomplish, making it my main choice for the project's language.

Project Languages

This project utilizes Python as its implementation, allowing for a write-once-run-anywhere implementation similar to Java. This language will also be supported by SQL statements in order to interact with a backend database file. Since computing power and overhead are minimal considerations in this project, the need for strong memory control and a lightweight run-time are discarded in favor of programmability and ease of development.

Project Software

This project utilizes Python3 in addition to PyPDF2 as its main software components. An optional SQL database viewer can provide the ability to read the database file created by the program but is not necessary for program execution. Any operating system capable of utilizing a python package manager and Python3 will be able to run this program. For development, Github and VSCode were utilized extensively for patch management and debugging/text editing respectively. The SQL database browser was also utilized to ensure database accuracy and integrity.

Project Hardware

Because of the lightweight operations involved with the program, very little hardware is needed. Anything capable of running Python3 and allows for at least 1 GB of secondary and main memory will be able to effectively run this program.

Project Requirements

Functional Requirements

Requirement ID: 01a
Requirement Type: PDF Scraping
Description: This product shall allow the input of a PDF file path after which the information contained within will be applied and saved to the database file.
Originator: Ethan Powell
Fit Criterion: The user shall be able to define the file path for an official D&D 5e character sheet, and the program will take the character name, current and maximum hp, ac, and base stat scores and apply them to a character object which will be temporarily saved in RAM.
Priority: 1

Requirement ID: 01b
Requirement Type: Character Sheet Viewing
Description: This product shall allow the viewing of a player's character sheet in an easy-to-read format.
Originator: Ethan Powell
Fit Criterion: The user shall be able to view a character's stats from a queried database in a character sheet format. This will involve displaying the character's name, base statistics, ac, current and maximum hp, modifiers, skills and skill modifiers, resistances, vulnerabilities, proficiencies, and expertise.
Priority: 3

Requirement ID: 01c
Requirement Type: Character Data Creation
Description: This product shall allow the creation of a character sheet through the program's interface without the need for external tools
Originator: Ethan Powell
Fit Criterion: The user shall be able to input a character's name, ac, current hp, maximum hp, strength score, dexterity score, constitution score, intelligence score, wisdom score, and charisma score, finally prompting with a confirmation screen in order to create a character entry in the database.
Priority: 1

Requirement ID: 01d
Requirement Type: Character Data Editing
Description: This product shall allow the editing of a character sheet through the program's interface without the need for external tools
Originator: Ethan Powell
Fit Criterion: The user shall be able to change a character's base statistic scores, skills, add and remove proficiencies, vulnerabilities, resistances, and expertise in the program's interface. This edited data should then be applied to the character's state in the database and reflect when viewing the character at a later time.
Priority: 1

Requirement ID: 01e
Requirement Type: Character Data Removal
Description: This product shall allow the removal of a character sheet through the program's interface without the need for external tools
Originator: Ethan Powell
Fit Criterion: The user shall be able to remove a character's entry from the database file by utilizing the "Remove Character" option in the program's interface. The user then shall be able to enter the character's name they wish to remove before the program removes the specified character's database entry. This should then reflect in future runs of the program by disallowing access to that character's data.
Priority: 1

Ease of Use Requirements

Requirement ID: 02a
Requirement Type: Usability
Description: This product shall be able to be used by people with a basic knowledge of computer software.
Originator: Ethan Powell
Fit Criterion: A user with a basic knowledge of a keyboard and mouse should be able to navigate all the interface and functions within 30 minutes of starting the program.
Priority: 4

Requirement ID: 02b
Requirement Type: Appearance
Description: This product shall provide a text-based user interface for interaction with the program
Originator: Ethan Powell
Fit Criterion: A user should be able to discern all the program's functions through the text-based interface. A user should not have to question how to navigate through the program's functions or require external help to understand specific program functionality.
Priority: 4

Performance Requirements

Requirement ID: 03a
Requirement Type: Speed and Latency
Description: All actions performed on entities in the program shall be completed in a timely manner.
Originator: Ethan Powell
Fit Criterion: A user shall experience a delay of no longer than 10 seconds when performing data creation, editing, importing, or destroying.
Priority: 4

Requirement ID: 03b
Requirement Type: Precision and Accuracy
Description: This product shall be accurate in its display of character data information.
Originator: Ethan Powell
Fit Criterion: A user shall be able to have the data values entered be what's displayed on screen. There should be no discrepancy between the user's expectations and the program's display when data is entered. Additionally, all database information must be stored with reasonable expectations of corruption and data loss.
Priority: 3

Requirement ID: 03c
Requirement Type: Reliability and Availability
Description: This product shall be available to the user until the user decides to terminate the program.
Originator: Ethan Powell
Fit Criterion: The product shall continue to run without interruptions until the user terminates the program by pressing "ENTER" on the home screen. The program shall not crash during operation and must handle input validation and exceptions that occur as a result of incorrect data input.
Priority: 2

Requirement ID: 03d
Requirement Type: Scalability and Capacity A
Description: This product shall be able to be modified by the developer such that new modules can be added to extend functionality without breaking previous functions of the product.
Originator: Ethan Powell
Fit Criterion: The product should have the capability to be patched at a later time, allowing for code updates and changes to be applied when necessary through the use of Github access.
Priority: 4

Requirement ID: 03e
Requirement Type: Scalability and Capacity B
Description: This product shall have the capacity to track at least 50 entities at one time.
Originator: Ethan Powell
Fit Criterion: The product shall track and modify at least 50 entities without termination or noticeable slowdown during program runtime with base hardware concerns.
Priority: 3

Maintainability and Scalability Requirements

Requirement ID: 04
Requirement Type: Maintenance
Description: This product shall be able to be patched by removing bugs and adding performance improvements as soon as they are available.
Originator: Ethan Powell
Fit Criterion: When a software bug is discovered, a time of no more than 1 month shall pass before the bug is fixed. Additionally, if a performance enhancement becomes available, the product owners should be notified of an update and have access to the performance enhancement as soon as it is available.
Priority: 3

Security Requirements

Requirement ID: 05a
Requirement Type: Access Requirements
Description: This product shall provide reference to all data belonging only to the program without blockage for the user in additional to preventing data modification outside the program.
Originator: Ethan Powell
Fit Criterion: A user shall be able to access database information without the need for root or special permissions. Additionally, the program should not change data outside of the program's operation functionality.
Priority: 5

Requirement ID: 05b
Requirement Type: Integrity Requirements
Description: This product shall provide accurate information and prevent harmful altering of data inside the database.
Originator: Ethan Powell
Fit Criterion: Input validation should occur for each attribute inputted into the database. SQL injection attacks should be prevented from altering the database through proper input sanitization.
Priority: 5

Project Description

This project focuses on providing a simple interface that any level of user can use without requiring extensive instructions or direction, aiming to help those who have never played in a tabletop RPG with one of the most difficult portions of the game. With the ability to input basic character information (or import information from an official form-filled pdf), players can hop right into the game without having to worry about incorrect modifier calculations and the rules behind proficiencies and expertise. This tool will also aid in storing large numbers of characters through a backend database in a quickly editable format so that players will not have the issue of carrying around many character sheets and maintaining those sheets in their paper form.

Project Implementation

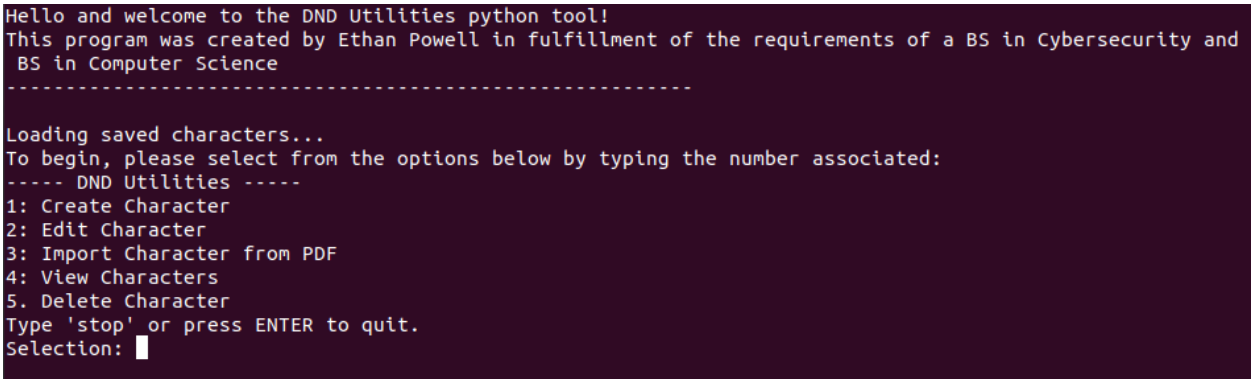

Figure 1- This displays the program's start screen when the main function "dndutitilites.py" is executed.

```
----- Create Character -----

Saved Characters:
Tom
Boblin the Goblin
Gwendolynn
Caoimhe
Artyom

Please type your new character's name or press ENTER to go back: Clementine
Please type Clementine's level: 10
Please type Clementine's current HP: 100
Please type Clementine's max HP: 100
Please type Clementine's armor class: a

Please enter a positive number with no alphabetic characters.
Please type Clementine's armor class: 16
Please type Clementine's strength score: 6
Please type Clementine's dexterity score: 16
Please type Clementine's constitution score: 14
Please type Clementine's intelligence score: 11
Please type Clementine's wisdom score: 18
Please type Clementine's charisma score: 16
----- Clementine's Statistics -----

Level: 10
HP: 100/100
Armor Class: 16
STR: 6 (-2)
DEX: 16 (+3)
CON: 14 (+2)
INT: 11 (+0)
WIS: 18 (+4)
CHA: 16 (+3)

 Does this look right (y/n)? y
```

Figure 2- This displays the first menu option of "Character Creation." Data entry is screened so that improper input cannot be used. Finally, a confirmation screen is available to ensure correct input.

```
----- Edit Character -----

Saved Characters:
Tom
Boblin the Goblin
Gwendolynn
Caoimhe
Artyom
Clementine

Please type the character's name you wish to edit or press ENTER to go back: alskdjf
Character does not exist. Please check spelling.
Please press ENTER to continue...
```

Figure 3 - This displays the second menu option of "Edit Character." An error is displayed to the user if a character that does not exist is entered.

```
----- Editing Character: Artyom -----
1. Base Stats
2. Saves
3. Skills
4. Proficiencies & Expertises
5. Resistances
6. Vulnerabilities
7. Enable/Disable Jack of All Trades
Please choose the section you wish to edit or press ENTER to go back:
```

Figure 4 - When a correct character name is entered, the user is brought to the edit screen.

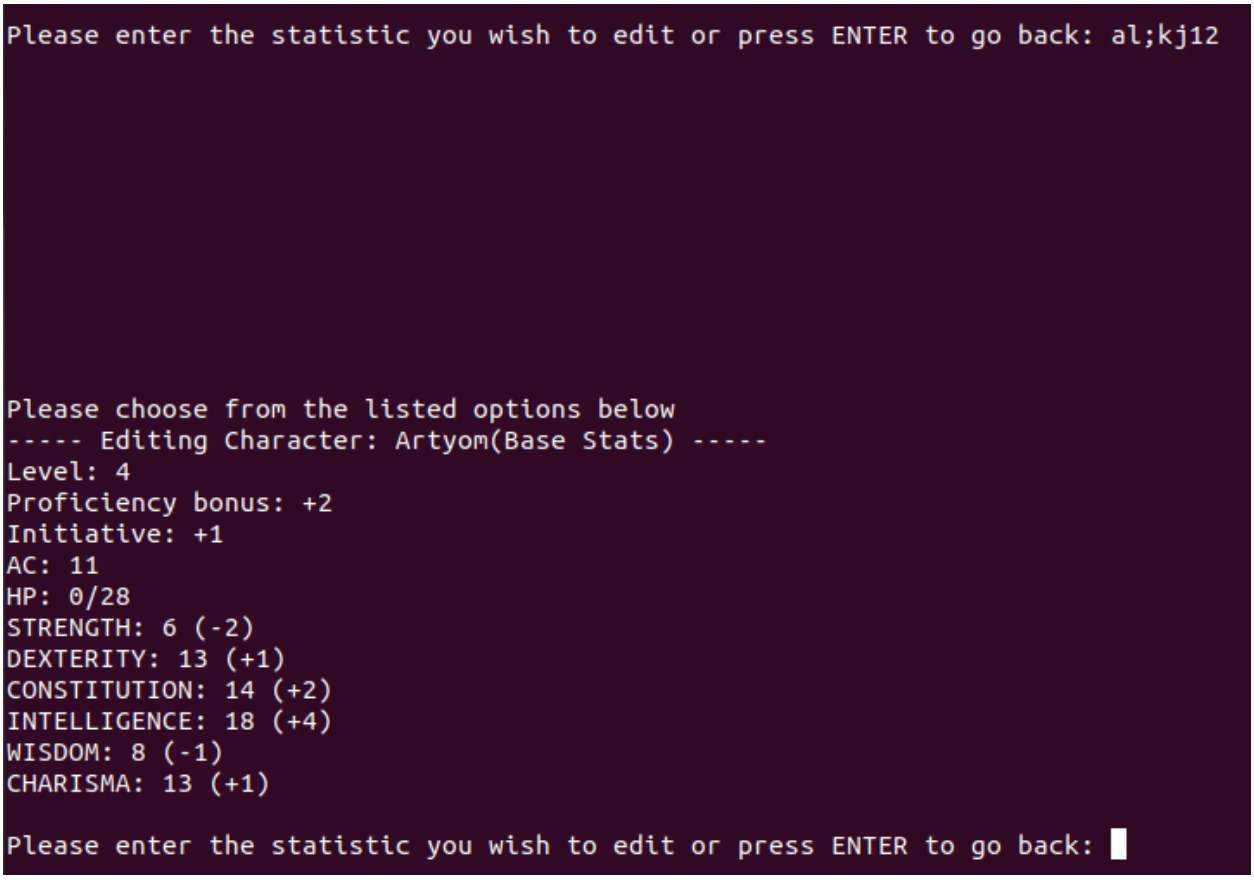

Figure 5 - This displays the "Base Stats" edit screen in which error validation takes for selections.

```
----- Editing Character: Artyom(Base Stats) -----
Level: 4
Proficiency bonus: +2
Initiative: +1
AC: 11
HP: 0/28
STRENGTH: 6 (-2)
DEXTERITY: 13 (+1)
CONSTITUTION: 14 (+2)
INTELLIGENCE: 18 (+4)
WISDOM: 8 (-1)
CHARISMA: 13 (+1)

Please enter the statistic you wish to edit or press ENTER to go back: Level
Artyom's new level: a
Please enter a positive number with no alphabetic characters.
Artyom's new level: 5
----- Editing Character: Artyom(Base Stats) -----
Level: 5
Proficiency bonus: +2
Initiative: +1
AC: 11
HP: 0/28
STRENGTH: 6 (-2)
DEXTERITY: 13 (+1)
CONSTITUTION: 14 (+2)
INTELLIGENCE: 18 (+4)
WISDOM: 8 (-1)
CHARISMA: 13 (+1)

Please enter the statistic you wish to edit or press ENTER to go back:
```

Figure 6 - This displays editing the character's level with input validation

```
Please enter the statistic you wish to edit or press ENTER to go back: Wisdom
Artyom's new wisdom score: 10
----- Editing Character: Artyom(Base Stats) -----
Level: 5
Proficiency bonus: +2
Initiative: +1
AC: 11
HP: 0/28
STRENGTH: 6 (-2)
DEXTERITY: 13 (+1)
CONSTITUTION: 14 (+2)
INTELLIGENCE: 18 (+4)
WISDOM: 10 (+0)
CHARISMA: 13 (+1)

Please enter the statistic you wish to edit or press ENTER to go back:
```

Figure 7 - This displays editing the character's wisdom score with an automatic modifier update

```
Please enter the statistic you wish to edit or press ENTER to go back: 12









Please choose from the listed options below
----- Editing Character: Artyom(Saves) -----
Strength Save: -2
Dexterity Save: +1
Constitution Save: +2
Intelligence Save: +4
Wisdom Save: -1
Charisma Save: +1

Please enter the statistic you wish to edit or press ENTER to go back: wis
Artyom's new wisdom save: 2
----- Editing Character: Artyom(Saves) -----
Strength Save: -2
Dexterity Save: +1
Constitution Save: +2
Intelligence Save: +4
Wisdom Save: +2
Charisma Save: +1

Please enter the statistic you wish to edit or press ENTER to go back:
```

Figure 8 - This displays editing the character's wisdom save with input validation

```
----- Editing Character: Artyom(Proficiencies & Expertises) -----

Proficiencies: ['None']
Expertises: ['None']

1. Add Proficiency
2. Add Expertise
3. Remove Proficiency
4. Remove Expertise
Please choose from the menu options above or press ENTER to go back:
```

Figure 9 - This displays option '4', showing current proficiencies and expertise as 'none'

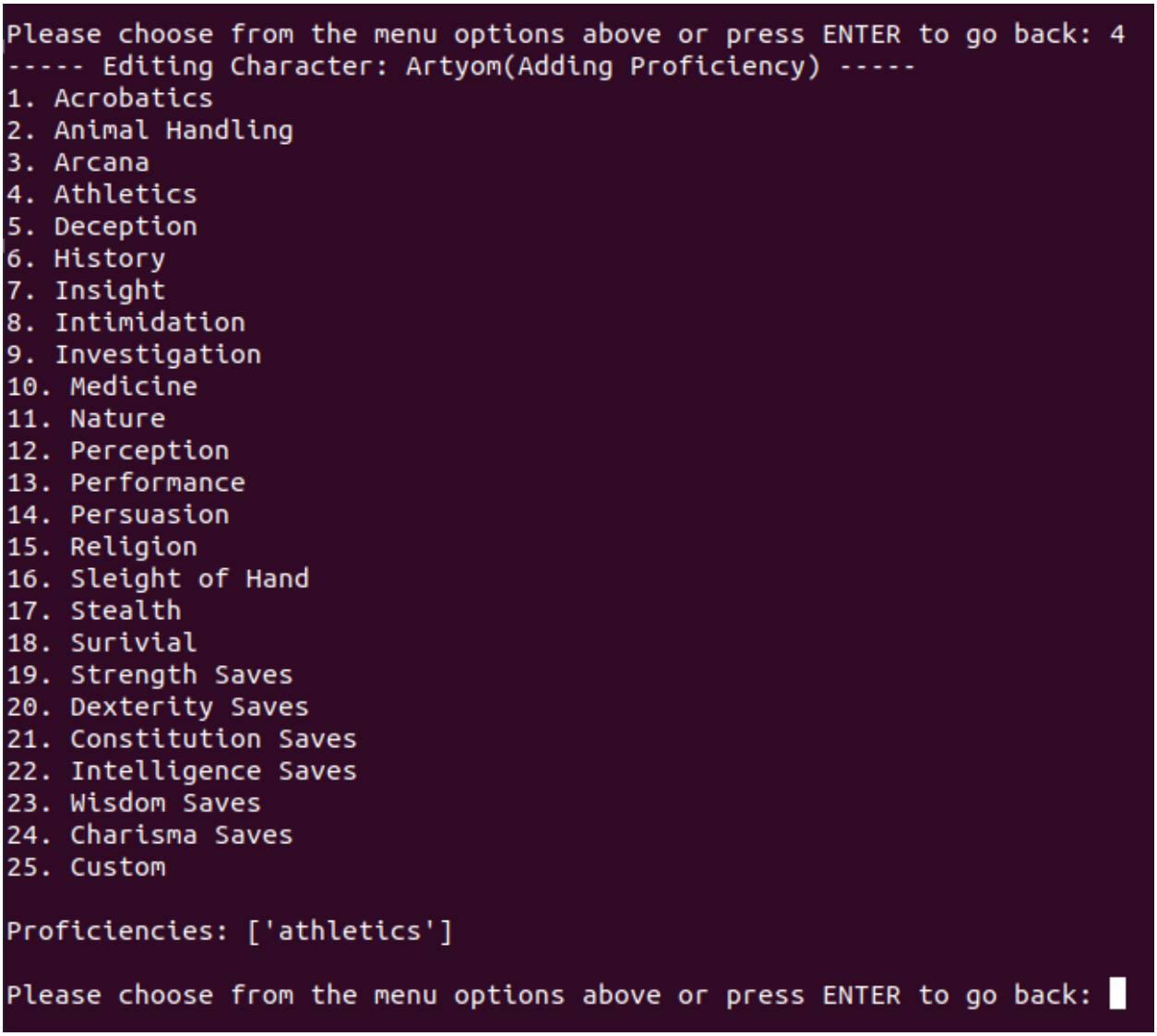

Figure 10 - This displays adding a proficiency to the character

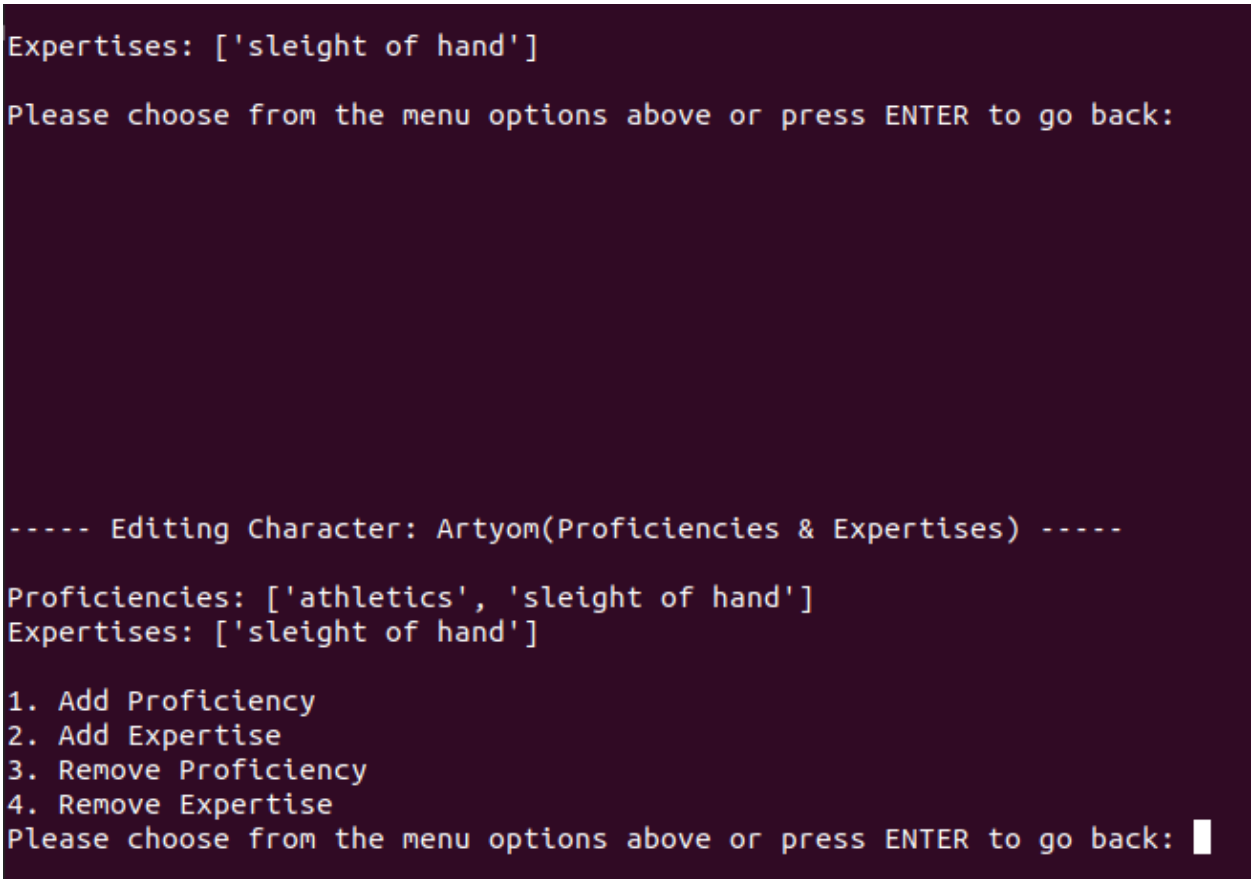

Figure 11 - This displays adding an expertise will also add it to proficiencies (as per the requirement due to game rules)

```
----- Editing Character: Artyom(Remove Proficiency) -----
1. athletics
2. sleight of hand
Please choose from the menu options above or press ENTER to go back: 2
----- Editing Character: Artyom(Remove Proficiency) -----
1. athletics
Please choose from the menu options above or press ENTER to go back:




----- Editing Character: Artyom(Proficiencies & Expertises) -----

Proficiencies: ['athletics']
Expertises: ['None']

1. Add Proficiency
2. Add Expertise
3. Remove Proficiency
4. Remove Expertise
Please choose from the menu options above or press ENTER to go back:
```

Figure 12 - This displays that removing the proficiency also removes the expertise (as per the game rule)

```
----- Editing Character: Artyom(Resistances) -----

Resistances: ['None']

1. Add Resistance
2. Remove Resistance
Please choose from the menu options above or press ENTER to go back: 1
```

Figure 13 - Choosing option 5 displays the resistances screen

```
Please choose from the menu options above or press ENTER to go back: 1
----- Editing Character: Artyom(Add Resistances) -----
1. Acid
2. Bludgeoning
3. Cold
4. Fire
5. Force
6. Lightning
7. Necrotic
8. Piercing
9. Poison
10. Psychic
11. Radiant
12. Slashing
13. Thunder
14. Custom

Resistances: ['acid']

Please choose from the menu options above or press ENTER to go back:
```

Figure 14 - Adding resistances

```
JOAT set to 'True'
----- Editing Character: Artyom -----
1. Base Stats
2. Saves
3. Skills
4. Proficiencies & Expertises
5. Resistances
6. Vulnerabilities
7. Enable/Disable Jack of All Trades
Please choose the section you wish to edit or press ENTER to go back:
```

Figure 15 - Entering "7" toggles the Jack of All Trades boolean

```
----- DND Utilities -----
1: Create Character
2: Edit Character
3: Import Character from PDF
4: View Characters
5. Delete Character
Type 'stop' or press ENTER to quit.
Selection: 3









----- Import Character -----
Please enter the file path of your character PDF or press ENTER to go back: alkjdf
File does not exist. Please check filepath.
Please press ENTER to continue...
```

Figure 16 - Importing a character through pdf with file path validation

```


----- Import Character -----
Please enter the file path of your character PDF or press ENTER to go back: Test_Files/Test.pdf
Sucess! Please add proficiencies, expertises, resistances, and vulnerabilites manually.
Please press ENTER to continue...
```

Figure 17- Displays when file path is found

```
----- View Characters -----

Saved Characters:
Tom
Boblin the Goblin
Gwendolynn
Caoimhe
Artyom
Clementine
Errant

Please type which character you would like to view or press ENTER to go back:
```

Figure 18 - The character is then added to the list of saved characters

```
----- Import Character -----
Please enter the file path of your character PDF or press ENTER to go back: Test_Files/Test.pdf
The character 'Errant' already exists, so no action was taken.
Press ENTER to continue...
```

Figure 19 - Uniqueness checking is done to ensure only one of each character

```
----- View Characters -----

Saved Characters:
Tom
Boblin the Goblin
Gwendolynn
Caoimhe
Artyom
Clementine

Please type which character you would like to view or press ENTER to go back: alsd
Character does not exist. Please check spelling.
Please press ENTER to continue...
```

Figure 20 - View character functionality with error handling

```
Please type which character you would like to view or press ENTER to go back: Tom




----- CHARACTER: Tom -----
Level: 20
Proficiency bonus: +6
Initiative: +3
AC: 18
HP: 27/45
STRENGTH: 22 (+6)
DEXTERITY: 16 (+3)
CONSTITUTION: 10 (+0)
INTELLIGENCE: 8 (-1)
WISDOM: 10 (+0)
CHARISMA: 18 (+4)

----- SAVES -----
Strength Save: +6
Dexterity Save: +3
Constitution Save: +0
Intelligence Save: -1
Wisdom Save: +0
```

Figure 21 - Character displaying functionality

```
----- Delete Character -----

Saved Characters:
Tom
Boblin the Goblin
Gwendolynn
Caoimhe
Artyom
Clementine
Test Char

Please type the character's name you wish to delete or press ENTER to go back: Test Char
Character deleted successfully. Please press ENTER to continue...
```

Figure 22 - Character deletion functionality

Test Plan

Introduction

The goals of this test plan are to ensure the program meets the functional requirements outlined in the program requirements section of this document. Additionally, this test plan will allow for one to ascertain if a functional requirement fails is failed to be met. This test plan will involve system testing and unit testing, allowing for usability testing if time allows.

Goals

This testing document's purpose is to outline the procedures needed to accomplish unit testing and system testing of the program overall. Once testing has been completed, this document will also detail the process in which bugs shall be addressed, fixed, and patched for the user. The testing outlined in this document is not all encompassing and only aims to fix major issues before the product's release. If software bugs are later found in the program, a timely correction of these bugs is to be expected.

Constraints

The main constraints that may cause issue for testing purposes can be broken down into two issues. The first issue will be time constraints. This issue will likely arise from a variant workload and mismanaged objectives for the project overall. The second issue that may arise will be the unfamiliarity of the developer with component testing. Both of these constraints can be overcome with the use of careful planning to overcome time constraints and careful research to familiarize oneself with component testing as a system.

References

This test plan references no other documents other than the full documentation. Please refer to “D&D_Utilities_Documention.pdf” or “D&D_Utilities_Documentation.docx” for other information.

Test Items

This testing procedure will utilize three main instances of software for testing purposes. SQLite browser will be utilized to verify the integrity and correctness of the database file, and VSCode will be utilized to conduct bug testing operations. Finally, Microsoft Excel will be utilized for aggregating test cases and outcomes.

Features to be Tested

The following features will be the main considerations for what aspects of the program will be tested. These specifications are in no particular order but aim to explain the sections of the program that will be evaluated.

- **Character Data Creation:** The program will allow for character data to be created inside the program with no external tools needed. This data will include character name, ac, current and maximum hp, and base statistic scores such as strength, dexterity, constitution, intelligence, wisdom, and charisma. Any data not included in character creation should be able to be edited in the following character edit functionality.
- **Character Data Editing:** The program will allow for character data editing to be done inside the program with no external tools needed. This data that can be edited will include character base statistics such as current and maximum hp, ac, initiative, proficiency bonus, level, and base statistic scores (please refer to above). Additionally, the character

edit screen should allow for the editing of skill modifiers, proficiencies and expertise, resistances, and vulnerabilities.

- **Character Data Importing:** The program will allow for character data to be imported from an official Dungeons and Dragons form-filled character sheet. The data to be imported will include the character's name, level, current and maximum hp, and base statistics (please refer to above). All proficiencies, expertise, resistances, and vulnerabilities will not be included but instead editable from the character edit screen.
- **Character Deletion:** The program will allow created, edited, and imported character data to be deleted from the program's database.
- **Character Data Saving and Loading:** The program will allow created, edited, and imported character data to be saved in a local database file such that each time the program is closed and relaunched, character data will be available from previous sessions.

Features Not to Be Tested

All program features will be tested and evaluated based on testing criteria outlined later in this section.

Approach

Testing will consist of Pass/Fail unit and system testing, including usability testing from users if time allows. Each test criteria will be attempted and assigned a grade of "Pass," if the test is successful as a result of the expected outcome being achieved, or "Fail," if the test is unsuccessful as a result of the expected outcome being different from the expected result. Unit testing will be completed in an agile development strategy, testing each component as it is developed. Finally, system testing will be completed once each component has been

implemented and the product is deemed “finished.” Lastly if time allows, usability testing with test data gathered from users will be completed once unit and system testing has been completed.

Testing Levels

As detailed above, system and unit testing will be the main focus of testing operations utilized for this program. Manual unit tests will be created with an expected input and expected outcome for each test case. Once the program has been completed, a manual system test will be conducted with similar criterion to unit testing. Finally, user usability testing will occur and be graded on a 1-5 Likert scale.

Item Pass/Fail Criteria

For more specific testing pass/fail criteria, please refer to “Test_Cases.pdf” and “Test_Cases.xml”.

- **Application Launches**. When the user inputs python3 dndutilities.py into the command line, the application should launch.
- **Character Data Creation.** In the “Create Character” screen, if the data values “Tom”, “20”, “100” , “100”, “15”, “20”, “10”, “14”, “6”, “10”, “0”, and “y” are entered sequentially, the character displaying the name “Tom” with the statistics of “15 Ac”, “20th level”, “100/100 Hp”, “20 Strength”, “10” Dexterity”, “14 Constitution”, “6 Intelligence”, “10 Wisdom”, and “0 Charisma” should be displayed.
- **Character Data Validation.** In the “Create Character” screen, the letter “a” should prompt the user with an error message if inputted as a character’s level.
- **Character Import.** If “Test_Files/Test.pdf” is used as input for character import, the character “Errant” should be imported into the program.

- **Character Import Validation.** If the file path “Test.py” is used as input for character import, the user shall be prompted that the file does not exist or may be misspelled.
- **Character Data View.** When accessing the “Character View” screen, typing in a character’s name shall provide the database entry for that character accurately.
- **Character Data Deletion.** When accessing the “Character Deletion” screen, typing in a character’s name shall delete the database entry for that character.
- **Character Data Editing.** When accessing the “Character Edit” screen, entering in a character’s name shall bring the user to the edit screen for that character, allowing them to choose from editing the “base statistics”, “skills”, “proficiencies and expertise”, “resistances”, and “vulnerabilities.” Each statistic should be able to be edited and saved in the database in accordance with the data changed.

Suspension Criteria and Resumption Requirements

Suspension Criterion

There are no concerns with testing suspension; however, testing will be suspended in any event that testing becomes dangerous to the system or people involved.

Resumption Criterion

Testing will resume on the condition that all issues that caused testing suspension to occur have been resolved.

Test Deliverables

1. Test Plan
    a. This document is considered the test plan
2. Test Cases
    a. Please refer to “Test_Cases.xml” or “Test_Cases.pdf”

3. Test Scripts
    a. There will be no script-based testing implemented
4. Test Reports
    a. Please refer to “Test_Cases.xml” or “Test_Cases.pdf”

Test Environment

Hardware

Testing will be completed on the following hardware specifications:

- Intel I7-9700k CPU
- Nvidia GTX 3070ti Gigabyte GPU
- G.Skill Trident Z Neo 64GB (4x16GB) DDR4 3600MHz

Software

Testing will be completed with the following software specifications:

- Oracle Virtualbox Version 6.1.30 r148432 (Qt5.6.2)
- Ubuntu Linux, version 20.04.3 LTS
- Microsoft Windows 10 Enterprise, 10.0.19043

Network

Testing will be done with the following network specifications:

- Virtual NAT through Oracle Virtualbox

Cost Estimate

Testing will require no monetary value; however, testing will require time investment in order to be conducted properly, involving a minimum of 56 work hours.

Testing Schedule

Task	Start Date	Completion Date
Begin Testing	April 17th, 2022	April 17th, 2022
Test Character Creation	April 17th, 2022	April 187th, 2022
Test Character Editing	April 18th, 2022	April 18th, 2022
Test Character Import	April 18th, 2022	April 18th, 2022
Test Character Viewing	April 19th, 2022	April 19th, 2022
Test Character Deletion	April 19th, 2022	April 19th, 2022
Finish Testing and Present	April 20th, 2022	April 20th, 2022

Staff and Training Needs

This testing will not require any special staffing or training needs.

Staff Responsibilities

This test plan will require a minimum of two individuals:

- Dr. Sean Hayes: Reviewing documentation and ensuring quality assurance.
- Ethan Powell: Development of software, testing of software, and documentation of software.

Risks and Mitigations

Risks

- The project will fail to be completed.
- Unforeseen hurdles will appear during software development.
- Project and classwork will take priority over project development.

- Equipment and data loss may occur.

Mitigations

- Proper time management will ensure project completion in a reasonable time frame.
- Proper contingency management will allow for major risks to be avoided.
- Proper time delegation will ensure that priorities will be kept in order.
- GitHub data repositories will be utilized to mitigate data loss.

Assumptions and Dependencies

Assumptions

- Time must be delegated between classwork and project completion.
- Software bugs will occur, and additional time will be required to debug.
- A working version of each software piece is available to the developer.

Dependencies

- A working SQLite database is required.
- A working development environment is required.

Approvals

Developer, Ethan Powell, and Lead Assistant, Dr. Sean Hayes, will be required to approve this document.

Test Results

Please refer to the “Test_Cases.pdf” for test result information.

Challenges Overcome

When programming this project, the main challenge that I faced was implementing the program functions in the language that offered the most functionality. When researching, deciding between C++, C#, and Python was a main source of struggle, requiring me to weight the benefits and detriments of each language against one another. Python became my main language of implementation due to the flexibility of many of its libraries and the ease of programming it offered.

The second challenge that I faced was interacting with a PDF. A standard document format to most, the PDF file format became one of the largest thorns in my side, as pulling information from one was not as simple as I had hoped. After digging through documentation and researching different methods of interacting with the file type, PyPDF2 became a significant boon to my journey, allowing the form-filled information from a PDF to be pulled into a Python dictionary. This allowed one of the main functionalities of a PDF import system to be implemented in the final product.

The third challenge that I encountered was program to database interaction. Over the course of all my CSCI courses, I never once was able to interact directly with a database through a program outside of a small introduction when utilizing PHP. This became a daunting task at first, but quickly was lessened when SQL commands became the main interaction method. The small nuances of creating a database connection and closing it were very similar to file interaction, so the continued use of such a method became second nature as I utilized it due to the file interaction background that many of my classes provided.

Lastly, writing extensive documentation was the final hurdle that I had to overcome when creating this project. I had never written any type of program documentation beforehand,

meaning that I had to essentially learn the method from scratch. This provided a much needed insight into the business and customer side of program development, allowing me to truly see what program creation can look like in a business environment.

Future Enhancements

In the future, I wish to fully complete the program as it was originally intended, a Dungeons and Dragons combat encounter simulator. The main issue that I came across was a way to interpret Dungeons and Dragons abilities and spells. As a person, it is easy to read a Dungeons and Dragons spell (much like reading a book) and interpret the meaning behind how an interaction is to take place; however, trying to programmatically interpret the same spell that requires human intuition to properly translate was much more difficult than I anticipated, which led to the main issue behind being able to fully implement the program's functions. This is due to spells and abilities being the main source of interaction in a Dungeons and Dragons combat encounter, leading to the original intention not being able to be achieved.

I wish to also fully implement a graphical user interface. It was my original intent to implement a point-and-click user interface that fully utilized a mouse and visual buttons, but I opted to instead work on a text-based interface to push functionality over appearance. Eventually, upgrading the text-based interface to a point-and-click GUI would greatly improve the user's experience and allow for a more professional feel.

Another implementation that I aspire to add is the ability for the program to be web hosted or allow for mobile device functionality. Because of the program's usability in a tabletop RPG, the ability to allow this data to be accessed from multiple locations (or at least be more portable) would greatly improve the usefulness of the product overall. It is not common that

everyone carries a laptop everywhere they go, but most people have a phone on them in order to interact. This would allow users to connect to the internet with their phone and access the program there or have a local phone application to utilize only on their device.

Finally, cleaning and optimizing the code for speed, readability, and maintainability is always a concern with any program. In its current state, the code is difficult to read, hard to scale, and would later lead to optimization problems if more functionalities would be added. In order to keep the product functioning and professional, the need for code optimization and readability optimization is necessary. This is especially true for this product, but the ideas can be applied to every software product.