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 3](#_Toc101039096)

[Project Description 4](#_Toc101039097)

[Research and Background 4](#_Toc101039098)

[Deciding a Language 4](#_Toc101039099)

[C++ - Using a sledgehammer to kill a fly 4](#_Toc101039100)

[C# - Getting better, but not quite there 5](#_Toc101039101)

[Python – Easy to use, lots of support. 5](#_Toc101039102)

[GUI Implementation Prospects 5](#_Toc101039103)

[C++ - Hardware not software 5](#_Toc101039104)

[C# - Better for GUI, but still involved 6](#_Toc101039105)

[Python – A good option that went underused 6](#_Toc101039106)

[Library Availability 6](#_Toc101039107)

[Documenation 6](#_Toc101039108)

# 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.

# 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.

# 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

## Documenation

Implementation Languages

This project will utilize Python as its main implementation, providing a scripted language that doesn’t rely on operating system or hardware in order to run. This will be supported by several libraries, as well as SQL implementation for back-end database support. The database will be accessed through the use of SQL queries using the SQLite python library, providing a reliable, local database for interaction with the software.

1. Statement of Purpose (with the Problem Statement)
2. Research & Background
3. Project Language(s), Software, and Hardware
4. Project Requirements
5. Project Implementation Description & Explanation  
   *This section details the design and features of the in paragraph form with references to the following:*
   * screenshots with numbered captions (e.g., *Fig 1. The loading screen.*)
   * a link to the source code repository
6. Test Plan
7. Test Results
8. Challenges Overcome
9. Future Enhancements
10. Defense Presentation Slides