TEST PLAN

Presented to Sean Hayes

Presented by Ethan Powell

## Introduction:

The goals of this test plan are to set guidelines for how the program will be evaluated by providing a specific overview of the requirements and testing procedures that will be implemented. The items that will be covered in this document include: Testing Methods, Testing Levels, Testing Pass/Fail Criteria, and overviews of requirements needed for testing to occur.

## Goals:

This testing document’s purpose is to ensure that proper testing is completed such that he program will perform according to the documented specifications. Each test case will be created or the purpose of ensuring that a component of the software functions correctly, has a predictable outcome, and, in the event of a software bug being discovered, allow for quick correction of the issue. After testing has been completed, the final product will be a production-ready software that a user can utilize to improve a tabletop gaming experience.

## 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 projects 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 document will reference two documents: Project\_Proposal.docx and Project\_Requirements.docx

## Test Items:

### Software

There are several software pieces that will be utilized to create and test this product, including but not limited to: SQLite, Ubuntu Linux (for development purposes), and Elements (a graphical C++ library).

## Features to be Tested:

The following features will be the main considerations for what 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 Tracking: The program will allow for character data to be saved for reference throughout the rest of the program. This will include any pertinent character statistics, skill proficiencies, status effects, resistances, vulnerabilities, and capabilities to deal damage or cause status effects to other entities.
* Character Data Upload: The program will allow for character documents to be uploaded through the use of PDF form-fill in documents or by manual data entry. This function will rely on standard PDF form fill in format and will not support other document formats such as text files, docx, and more.
* Spell and Attack Data Referencing: The program will have the ability to reference Dungeons and Dragons fifth edition spell and attack data from the SQLite database and interpret if any applicable status effects, damage dice, or movement should be applied to the respective target, or targets. The program will then apply the proper reductions, if damage and applicable, or status effects, if applicable, to the respective targets. If the referenced spell or attack requires user interaction, the program will display what actions should be taken and noted.
* Minimal UI Functionality: The program will provide a reoccurring update to the user so that each character can be easily referenced from a visual standpoint.

## Features Not to Be Tested:

All program features will be tested and evaluated based on testing criteria.

## Approach:

Testing will consist of Pass/Fail component testing. 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. Once each component test has been completed, each Failed testing outcome will be evaluated and fixed. Component testing will then resume from the beginning, attempting each test case once more. Once all component test cases have been assigned a grade of Pass, manual testing of visual elements will occur.

## Testing Levels

Component testing will be the main form of testing that will be utilized for this program. Much of the component testing will occur in an automated sense, crafting test cases using code and then having the code assign a “Pass” or “Fail” grade automatically. However, when manual testing begins, each visual aspect of the system will be inspected manually, assuring that correct data is being displayed as well as visual alignment and general feel of the program is as expected.

## Item Pass/Fail Criteria:

### Pass Criteria

* Character Data Tracking: The program should properly track and store inputted character data without corruption or unintended modification. This proper tracking and storage should include the ability to input data through a manual or PDF format and have the same data be stored in the program’s memory. This data should then be able to be recalled at a later time such that each time the program is started, the data will not be required to be re-entered. Finally, when calculations are done on the character data, the calculations should provide correct solutions, not allowing for any errors to occur when altering data values.
* Character Data Upload: The program should properly interpret PDF document data entry without data fragments or unintended data entry. When manual data entry occurs, data entered should be properly recorded and stored such that it exactly reflects data entered.
* Spell and Attack Data Referencing: Proper data storage for Dungeons and Dragons fifth edition spells and attacks should be done such that the program can easily reference data for calculation and data modification purposes. A user should be able to access this data and interpret any additional information when the program cannot do automatic calculations due to data complexity.
* Minimal UI Functionality: The user should be able to visually reference current player data values after each data modification in a clear and readable format.

### Fail Criteria

* Character Data Tracking: The program provides an incorrect interpretation of data such as an incorrect calculation applied to any data value. All data should be correctly tracked at all times, as one incorrect calculation will result in multiple data calculation failures at later points in the program’s runtime. Data corruption may also occur, resulting in the failure of proper data tracking.
* Character Data Upload: The program fails to allow the user to input data manually or by document upload. This criterion can also fail in the event of data upload, whether through manual or document upload, fails to reflect the data entered.
* Spell and Attack Data Referencing: The program fails to interpret SQLite database information, resulting in incorrect data calculation or data corruption. Data that is housed in the SQLite database becoming corrupted also constitutes program failure, as well as the incorrect storage of the data through data insertion into incorrect columns. Finally, unresponsive SQLite database queries is considered failure of data referencing.
* Minimal UI Functionality: The program fails to visually display correct information at any point during the program’s runtime.

## Suspension Criteria and Resumption Requirements:

### Suspension Criteria

Testing will be suspended upon catastrophic program failure. This includes but is not limited to: The complete failure of program capabilities such that the program is incapable of performing testing procedures, the complete loss of program data, or testing requirements are unobtainable by any means.

### Resumption Criteria

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

## Test Deliverables:

1. Test Plan
   1. This document is considered the test plan.
2. Test Cases
   1. Test cases have yet to be created.
3. Test Scripts
   1. Test scripts have yet to be created.
4. Test Reports
   1. Test reports have yet to be created.

## Test Environment:

### Hardware

Testing will be done 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 done 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

## Estimate:

Testing will require no monetary value; however, testing will require significant amounts of time in order to conduct properly, involving a minimum of 56 work hours.

## Schedule:

* SCHEDULE TO BE PLACED HERE –

(I do not currently have access to my proposal document for consistent scheduling requirements, so this will be implemented later.)

## Staffing and Training Needs:

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

## Responsibilities:

This test plan requires a minimum 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.
* Projects 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 each moment will be utilized.
* GitHub data branching 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.