SCHOOL OF COMPUTER SCIENCE

A User-Centred Approach to Developing a Tool for TTRPG Game Masters

Briar [redacted]

A dissertation submitted to the University of Bristol in accordance with the requirements of the degree of Master of Engineering in the Faculty of Engineering **worth 40CP** .

Monday 30th June, 2025

Abstract

Table-top Roleplaying games have seen recent surges in popularity with the effects of the COVID-19 pandemic and the success of the 5th edition of Dungeons and Dragons. These games are often run by a person known as the Gamesmaster who is expected to perform considerable admin to manage the game. Many existing software tools exist to aid with these tasks but none appear to garner user interests above others. Through my research, I have found that these tools are either developed by hobbyists who develop their own interests or by corporations who aim to earn a profit. This project, therefore, outlines a process of developing a TTRPG tool prototype using existing standards and research into User Centred Design. Based on an initial requirements analysis a two-step iterative process was followed to produce the final prototype. The prototype, my experience, and user feedback were then used to evaluate the effectiveness of the User Centred Design frameworks for this form of project as well as theorise as to the influence this effectiveness might have had on why they are not already used by existing developers in the TTRPG space.

Declaration

I declare that the work in this dissertation was carried out in accordance with the requirements of the University's Regulations and Code of Practice for Taught Programmes and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, this work is my own work. Work done in collaboration with, or with the assistance of others including AI methods, is indicated as such. I have identified all material in this dissertation which is not my own work through appropriate referencing and acknowledgement. Where I have quoted or otherwise incorporated material which is the work of others, I have included the source in the references. Any views expressed in the dissertation, other than referenced material, are those of the author.

AI Declaration

I declare that any and all AI usage within the project has been recorded and noted within Appendix A or within the main body of the text itself. This includes (but is not limited to) usage of text generation methods incl. LLMs, text summarisation methods, or image generation methods.

I understand that failing to divulge use of AI within my, work counts as contract cheating and can result in a zero mark for the dissertation or even requiring me to withdraw from the University.

Contents

Table of figures

Table of tables

Ethics Statement

This project fits within the scope of ethics application 6683, as reviewed by my supervisor [redacted].

Supporting Technologies

- Rust and its build system were used to develop this project.
- Yew [6] is the library for WASM [7] generation in which I developed my application in.
- Yewdux [8] and `serde` [9] were used for the management and exporting of global application state.
- Rust Fuzzy Search [10], [11] is a library implementing a fuzzy search algorithm to sort a list of strings by a search term. It implements the algorithm outlined by Kevin Aleman [12]. It is used to power the application's search bar.
- Tailwind CSS [13] is a CSS framework for web development which I utilised to style my application.
- `evalexpr` [14] is a library for evaluating mathematical expressions from Strings with optional context. It was used for the statistics and formula system.

My use of these technologies is evaluated in Section 4.3.

Notation and Acronyms

- TTRPG: Table-top Roleplaying Game
- GM: Gamesmaster
- DM: Dungeon-master
- D&D: Dungeons and Dragons
- NPC: Non-player Character
- LLM: Large Language Model
- SA: A short-answer question
- UCD: User-centred design
- UOD: User-oriented design
- W3C: World Wide Web Consortium
- UI: User Interface
- HCI: Human-Computer Interaction
- OOP: Object-Oriented Programming
- ISO 9241-210: Ergonomics of human-system interaction Part 210
- DOM: Document Object Model
- DSL: Domain Specific Language

Chapter 1
Introduction

Figure 1: Google search engine trend data for “ttrpg” over 5 years [1]

Table-top Roleplaying Games (TTRPGs) have had a cultural boom over recent years, and with influence from the COVID-19 pandemic, the success of Critical Role, the Stranger Things television show, and the release of a new (fifth) edition in 2014. Systems like Dungeons and Dragons (D&D) have gone from obscure geekdoms[1] to a pop culture hobby. A study of 1,479 Merchoid (A geek merchandise store) customers in June 2023 showed that 37% of responders were spending more time on tabletop and board games than they were before COVID-19 lockdowns [15]. Additionally, media reports suggest that Dungeons and Dragons sales increased by 33% during 2020 [16].

Almost all TTRPG systems are run by a gamemaster (GM) who needs to manage a significant amount of admin. The official D&D Beyond “How to Be a Dungeon Master” guide states:

> “the main responsibilities of the DM are to narrate the adventure, control the monsters during combat, roleplay any non-player characters (NPCs) that the party might encounter, and help move the story along”
>
> — R. Silverman [17]

[1] geekdom: The world or sphere of geeks. https://en.wiktionary.org/wiki/geekdom#English

While there exist many tools to aid in the preparation and playing of TTRPGs, I have found that users haven't found a tool that satisfies them all (See: Appendix C).

Due to this, I have created a prototype software tool for TTRPG GMs focused on user feedback and requirements. I have followed existing research and frameworks for user-oriented software development to develop a software solution for the aforementioned TTRPG use case. This project acts as a case study into the process and effectiveness of following these methodologies on a smaller-scope project.

1.1 Aims And Objectives

1. Perform Requirements Analysis on TTRPG players and GMs
2. Iteratively design and develop a solution and its requirements
3. Test and evaluate the current solution on TTRPG players and GMs
4. Evaluate the final user response, my own experience, and what that could mean for user-oriented design approaches.

1.2 Challenges

- I am relatively new to Yew as a framework and while I am experienced with Rust there will be some additional learning required.
- My access to users for testing is limited, as is my ability to incentivise them to engage in my study.
- There is limited time to both survey the user base, design a solution, and then also implement a prototype. To ensure that all tasks are achieved, time management and prioritisation is critical.

Chapter 2
Background

2.1 TTRPGS

Table-top Roleplaying Games is an umbrella term for story-focused games involving the portrayal and development of characters.

2.2 History

The first commercial edition of Dungeons and Dragons (OD&D) was published in 1974 [18] and is considered the first TTRPG (as we define it in the modern day). Since then there have been 8 additional versions and revisions which are shown in Table 1.

Year	Edition
1974	Dungeons & Dragons (original edition) [OD&D]
1977	Advanced Dungeons & Dragons (1st edition) [D&D 1e]
1989	Advanced Dungeons & Dragons (2nd edition) [D&D 2e]
1995	Advanced Dungeons & Dragons (revised 2nd edition) [D&D 2e 1995]
2000	Dungeons & Dragons (3rd edition) [D&D 3e]
2003	Dungeons & Dragons v3.5 [D&D 3.5e]
2008	Dungeons & Dragons (4th edition) [D&D 4e]
2014	Dungeons & Dragons (5th edition) [D&D 5e]
2024	Dungeons & Dragons (5th edition, 2024 version) [D&D 2024e/5.5e]

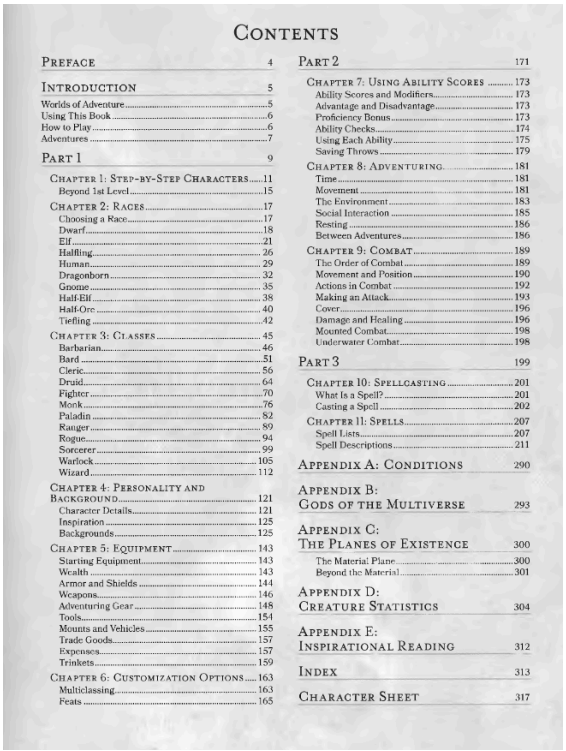
Table 1: D&D editions and their release years

For other popular TTRPGs: Call of Cthulhu first came out in 1981, Vampire: The Masquerade came out in 1991, and Pathfinder in 2009. All later received new updates or new editions similar to Dungeons and Dragons.

2.3 Dungeons and Dragons

Dungeons and Dragons (D&D) is the most popular and widely recognised TTRPG. Like many TTRPGs, it is generally played in extended stories called Campaigns which can last up to multiple years. They are usually played throughout numerous sessions, ordinarily 2-6 hours long.

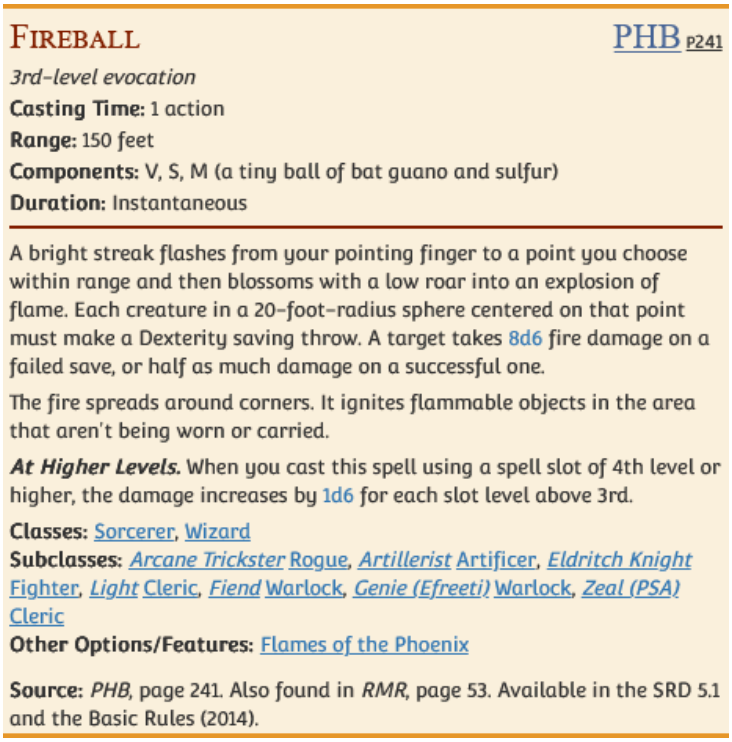
It has 19 source books that are generally considered official (i.e. full sourcebooks published by Wizards of the Coast) as well as a range of partnered content and small extras. The two main books are The Player's Handbook (PHB) [2]; which is designed primarily for players but contains lots of information that the GM will need to understand in order to run the game, and The Dungeon Master's Guide (DMG) [19] which includes additional rules, information and advice which is needed by the GM only. Combined the PHB and DMG include 618 pages of content, any of which could end up relevant during play. The GM can choose which (if any) of the other materials they allow their players to use, but it is common for GMs to allow most or all of the remaining 17 main sourcebooks. It is, of course, not expected for the players or GM to remember all of the information, however, trying to find the specific description of a rule via the physical books alone will take up lots of time that could be spent enjoying the hobby.



CONTENTS

Figure 2: Contents page of The Player's Handbook [2]

As an example of the large quantity of information included in a TTRPG such as D&D: there are 516 spells within the aforementioned 19 source books. These are spells that players and NPCs may both learn and cast during play. In order to handle these spells, what the spell does will always need to be known. Figure 3 shows the description for the spell Fireball, one of the 516 spells[2], it includes rules such as the spell's range, components, radius, saving throw requirement, damage dice, rules for upcasting etc. It also refers to other rules such as Saving throws, Spell schools (akin to type of spell) and fire damage the details of which are not included in the spell's description and will instead be found elsewhere within the books.



FIREBALL PHB p241

3rd-level evocation

Casting Time: 1 action

Range: 150 feet

Components: V, S, M (a tiny ball of bat guano and sulfur)

Duration: Instantaneous

A bright streak flashes from your pointing finger to a point you choose within range and then blossoms with a low roar into an explosion of flame. Each creature in a 20-foot-radius sphere centered on that point must make a Dexterity saving throw. A target takes 8d6 fire damage on a failed save, or half as much damage on a successful one.

The fire spreads around corners. It ignites flammable objects in the area that aren't being worn or carried.

At Higher Levels. When you cast this spell using a spell slot of 4th level or higher, the damage increases by 1d6 for each slot level above 3rd.

Classes: Sorcerer, Wizard

Subclasses: *Arcane Trickster* Rogue, *Artillerist* Artificer, *Eldritch Knight* Fighter, *Light* Cleric, *Fiend* Warlock, *Genie (Efreeti)* Warlock, *Zeal (PSA)* Cleric

Other Options/Features: Flames of the Phoenix

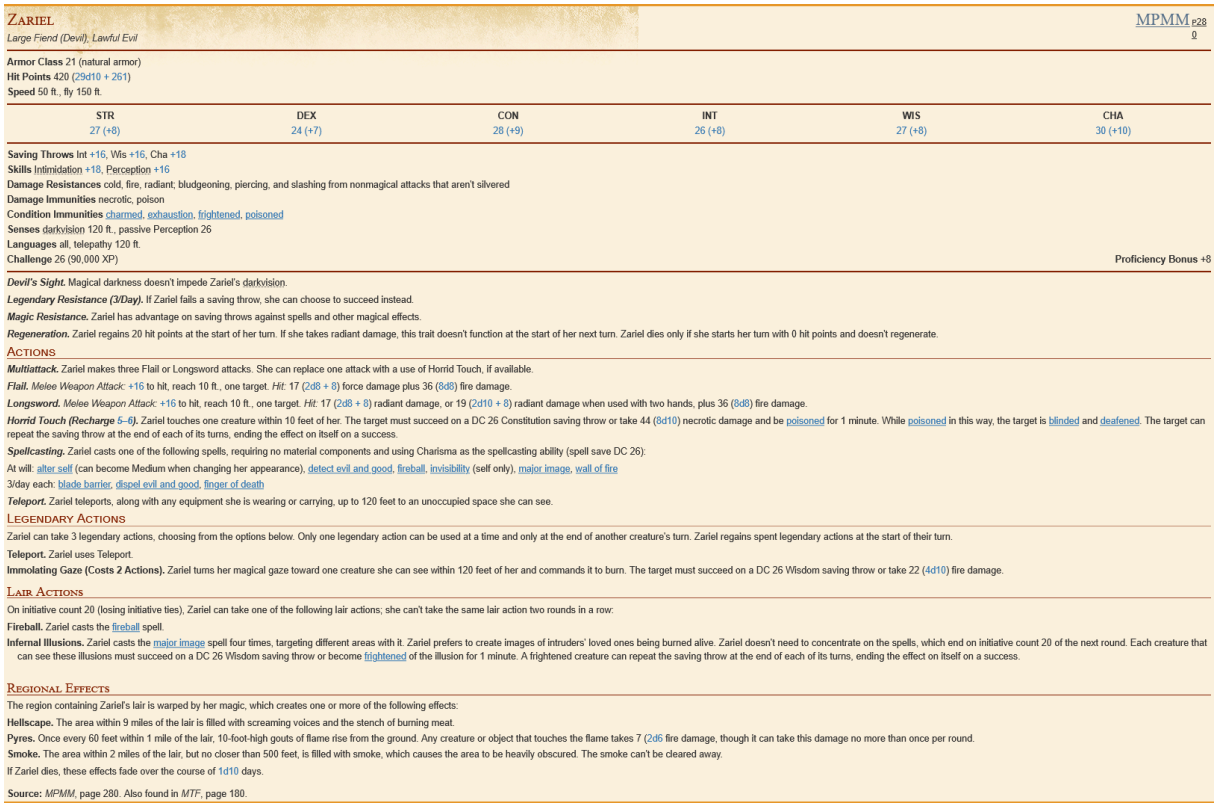
Source: *PHB*, page 241. Also found in *RMR*, page 53. Available in the SRD 5.1 and the Basic Rules (2014).

Figure 3: Description of the fireball spell from the Players Handbook [2] taken from 5etools [3]

Another example would be the NPC. The bestiary contains a colossal number of different characters and creatures (2020 in the 19 source books) and their respective statistics (known as a statblock). These could be animals, monsters, shopkeepers, heroes, allies, villains and henchmen. Figure 4 showcases a particularly large statblock for the Arch-devil Zariel, as a powerful and influential being her statblock includes high numbers and many abilities/ actions. While most stat blocks won't be as large as Zariel's almost all will

[2] And one of the most famous

include similar sections of information: their 6 attributes (STR, DEX, CON, INT, WIS, CHA), their armour class, speed, hit points, resistances and immunities, challenge rating, special senses, languages, actions, etc. Without understanding what these attributes actually represent, it is still clear that there is a lot of information to be aware of when using a statblock.



ZARIEL

Large Fiend (Devil), Lawful Evil

MPMM p28

Armor Class 21 (natural armor)
Hit Points 420 (29d10 + 261)
Speed 50 ft., fly 150 ft.

STR	DEX	CON	INT	WIS	CHA
27 (+8)	24 (+7)	28 (+9)	26 (+8)	27 (+8)	30 (+10)

Saving Throws Int +16, Wis +16, Cha +18
Skills Intimidation +18, Perception +16
Damage Resistances cold, fire, radiant; bludgeoning, piercing, and slashing from nonmagical attacks that aren't silvered
Damage Immunities necrotic, poison
Condition Immunities charmed, exhaustion, frightened, poisoned
Senses darkvision 120 ft., passive Perception 26
Languages all, telepathy 120 ft.
Challenge 26 (90,000 XP) **Proficiency Bonus** +8

Devil's Sight. Magical darkness doesn't impede Zariel's darkvision.
Legendary Resistance (3/Day). If Zariel fails a saving throw, she can choose to succeed instead.
Magic Resistance. Zariel has advantage on saving throws against spells and other magical effects.
Regeneration. Zariel regains 20 hit points at the start of her turn. If she takes radiant damage, this trait doesn't function at the start of her next turn. Zariel dies only if she starts her turn with 0 hit points and doesn't regenerate.

ACTIONS

Multiattack. Zariel makes three Flail or Longsword attacks. She can replace one attack with a use of Horrid Touch, if available.
Flail. *Melee Weapon Attack:* +16 to hit, reach 10 ft., one target. *Hit:* 17 (2d8 + 8) force damage plus 36 (8d8) fire damage.
Longsword. *Melee Weapon Attack:* +16 to hit, reach 10 ft., one target. *Hit:* 17 (2d8 + 8) radiant damage, or 19 (2d10 + 8) radiant damage when used with two hands, plus 36 (8d8) fire damage.
Horrid Touch (Recharge 5–6). Zariel touches one creature within 10 feet of her. The target must succeed on a DC 26 Constitution saving throw or take 44 (8d10) necrotic damage and be poisoned for 1 minute. While poisoned in this way, the target is blinded and deafened. The target can repeat the saving throw at the end of each of its turns, ending the effect on itself on a success.
Spellcasting. Zariel casts one of the following spells, requiring no material components and using Charisma as the spellcasting ability (spell save DC 26):
At will: alter self (can become Medium when changing her appearance), detect evil and good, fireball, invisibility (self only), major image, wall of fire
3/day each: blade barrier, dispel evil and good, finger of death
Teleport. Zariel teleports, along with any equipment she is wearing or carrying, up to 120 feet to an unoccupied space she can see.

LEGENDARY ACTIONS

Zariel can take 3 legendary actions, choosing from the options below. Only one legendary action can be used at a time and only at the end of another creature's turn. Zariel regains spent legendary actions at the start of their turn.
Teleport. Zariel uses Teleport.
Immolating Gaze (Costs 2 Actions). Zariel turns her magical gaze toward one creature she can see within 120 feet of her and commands it to burn. The target must succeed on a DC 26 Wisdom saving throw or take 22 (4d10) fire damage.

LAIR ACTIONS

On initiative count 20 (losing initiative ties), Zariel can take one of the following lair actions; she can't take the same lair action two rounds in a row:
Fireball. Zariel casts the fireball spell.
Infernal Illusions. Zariel casts the major image spell four times, targeting different areas with it. Zariel prefers to create images of intruders' loved ones being burned alive. Zariel doesn't need to concentrate on the spells, which end on initiative count 20 of the next round. Each creature that can see these illusions must succeed on a DC 26 Wisdom saving throw or become frightened of the illusion for 1 minute. A frightened creature can repeat the saving throw at the end of each of its turns, ending the effect on itself on a success.

REGIONAL EFFECTS

The region containing Zariel's lair is warped by her magic, which creates one or more of the following effects:
Hellscape. The area within 9 miles of the lair is filled with screaming voices and the stench of burning meat.
Pyres. Once every 60 feet within 1 mile of the lair, 10-foot-high gouts of flame rise from the ground. Any creature or object that touches the flame takes 7 (2d6 fire damage, though it can take this damage no more than once per round.
Smoke. The area within 2 miles of the lair, but no closer than 500 feet, is filled with smoke, which causes the area to be heavily obscured. The smoke can't be cleared away.
If Zariel dies, these effects fade over the course of 1d10 days.

Source: *MPMM*, page 280. Also found in *MTF*, page 180.

Figure 4: Statblock for Zariel the Archdevil from Mordenkainen Presents: Monsters of the Multiverse (MPMM) and 5etools [3]

2.3.1 The role of the GM

TTRPGs are generally led by a moderator often known as the Gamemaster (also known as Dungeon-master, storyteller, narrator etc.). The roles of the GM include [17]:

- Narrate the story during play
- Roleplay as the Non-Player Characters (NPCs)
- Control the enemies during combat
- Design the story and world the players are a part of
- Plan or improvise the scenarios players may come across
- Often have a superior understanding of the TTRPG system being played.

Stories being run can be both created by the GM themselves and also in settings books provided by the system's developer. Custom stories allow for greater combustibility, both for the GM to make their story but also for the story to adapt to the players' actions, however, running a custom story requires more effort and improvisational skills to be done well. As the final adjudicator when running the game, it is also the GM's job to interpret a system's rule-set wherever the wording may be ambiguous.

The management and fulfilment of all these tasks, therefore, can include a substantial amount of administration and so there is room for the assistance of a software tool. D. Acharya et al [20] found that GMs were interested in the potential for computational tools to help with GMing and that many requested features to help sort, filter and categorise campaign information. Similarly, Musacchio et al [21] found success exploring the capabilities of Large Language Models (LLMs) in creating Dungeons and Dragons spells.

2.3.2 The role of the player

Alongside the Gamemaster are the players. Players (usually around 5 in a group) are responsible for creating and roleplaying their characters, controlling that character's actions during combat and other encounters, and interacting with the world the GM has laid out. While the role is simpler than being a GM, it can still benefit from improved knowledge of a system's rule set, and there will be a minimum requirement of understanding in order to play the game. They also need to understand the character being played. Therefore, software aid can be useful here as well.

2.4 Existing Tools For TTRPGs

There are some existing tools and applications for information searching in TTRPGs. Notably, 5etools [3] and Wikidot [22] are available for the D&D 5e rule set. Depending on other systems they may have their own sites of varying quality or simply nothing available at all. Additionally, some tools and applications serve purposes less similar to the prototype I have created: virtual table-tops like Foundry[23] and Roll20[24] aim to replicate the experience of a table for uses like battle maps to facilitate playing TTRPGs online, virtual character sheet tools (such as Scoundry for Blades in the Dark [25]) aim to digitise the process of creating and playing a character (via the character sheet) sometimes having integration with the aforementioned virtual table tops. My application's approach is unique by being based on an initial requirements

analysis: allowing me to fill in gaps left by the existing tool sets (according to the users).

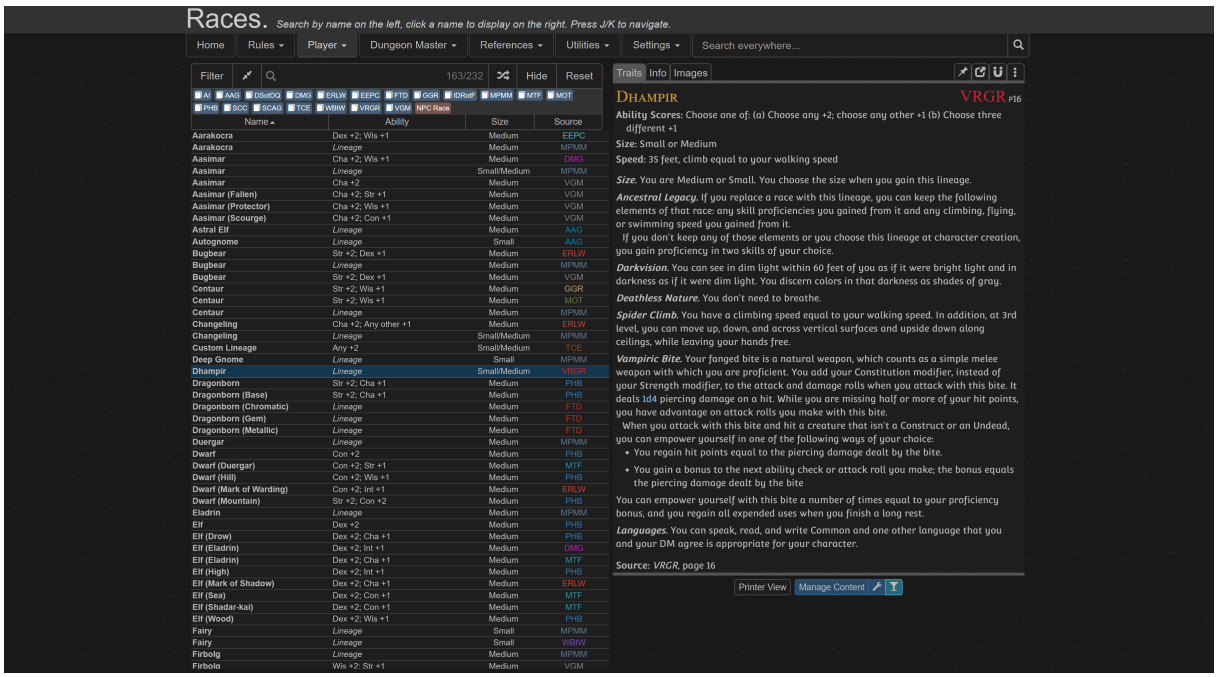


Figure 5: Races list page from 5etools [3]

2.5 User Focused Development

There are existing approaches and frameworks for user-focused software development. Perhaps most famously Donald A. Norman's User-centred design approach [26] (UCD). It draws on other existing frameworks: Cooperative design [27], Participatory design [28], and Contextual design [29]. UCD and similar approaches are designed for varied software projects and thus cover many principles which aren't necessarily all relevant or are simply outside the scope of this dissertation. However, UCD does cover two concepts relevant to my approach:

- First, "satisfaction of a human user as a criterion for evaluation" [26] which means, the resultant software solution and its prototypes are judged not on its technological efficiency or achievement but on how satisfied the potential users are with the application.
- Second, definition and refinement of requirements through investigative methods, meaning that the requirements of the software should be defined by investigating the user and their wishes and then continuously refined by similar investigations throughout the development and design process.

2.5.1 Industry Standard

To select a specific standard to follow (although they are generally similar) I have selected Ergonomics of human-system Interaction Part 210 (ISO 9241-210:2010) [30] due to its relevance to software development and popularity within industry and academia (the 2019 edition was not used as I was unable to acquire a copy). Specifically outlined are the following steps:

1. The design is based upon an explicit understanding of users, tasks and environments;
2. Users are involved throughout design and development;
3. The design is driven and refined by user-centred evaluation;
4. The process is iterative
5. The design addresses the whole user experience;
6. The design team includes multidisciplinary skills and perspectives.

— ISO 9241-210:2010 §4.1

Unfortunately, I cannot implement point six due to this being a single-person project. Across different fields, numerous literature exists which follows the ISO standard during software development [31], [32], [33]: Farinango et al [31] designed a personal health record system and report final efficiency and effectiveness of their system evaluated at 100%, Aziz et al [32] design the interface for an E-learning tool and found an average value of SUS Testing of 94.17 on their sample of 6 users, lastly, König et al [33] design a planning tool for air traffic controllers and find benefits to following the defined process. Farinango et al and Aziz et al also make use of the System usability scale (SUS) [34] to test their applications. While König et al briefly discuss the limitations of the standard for their specific project, room exists for further description. Additionally, these projects all focus purely on designing a potential solution rather than also providing an implementation. This project includes development, implementation, and further reporting on the benefits and limitations of the standard.

2.5.2 Thematic Analysis

As part of the previously covered “investigative methods” I perform user surveys and interviews which are analysed with thematic analysis. This approach was utilised as it better suited the smaller sample size of my surveys: with only ~20 responses it would’ve been unsuitable to use context analysis.

Additionally, thematic analysis allows the project to benefit from my own experience as a software developer and TTRPG player. Thematic analysis is best outlined by Braun et al [35], [36]. They cover six phases to follow:

1. Read the data to become familiar with it. Create a plan for potential codes.
2. Collapse data into patterns and labels to create initial codes.
3. Combine codes into overarching themes.
4. Evaluate found themes and continue coding if necessary.
5. Define each theme and what they contribute to the data.
6. Check step 5 definitions against the original data to ensure they are correct.

2.5.3 Autoethnographic Influence

Due to the limits of a small sample size of users to survey, as well as the potential benefit my experience as both a developer and a TTRPG player could provide: Autoethnography was used throughout the research and development process of my application.

> Autoethnography is a research method that uses personal experience ("auto") to describe and interpret ("graphy") cultural texts, experiences, beliefs, and practices ("ethno").
>
> — TE Adams [37]

The influence and limitations of this are discussed further in Section 3.3.1.1 and Section 4.5.

Chapter 3
Project Execution

Per point 1 of ISO 9241-210:2010 [30]. I would first perform a requirements analysis to acquire an understanding of users and the environment in which a software solution could be used. Section 6.2 of ISO 9241-210:2010 outlines four pieces of the context to understand:

1. The users and other stakeholder groups
2. The characteristics of the users or groups of users
3. The goals and tasks of the users
4. The environment(s) of the system

While I will define part 1 as experienced TTRPG players and DMs. The other context parts can be discovered through an initial requirements analysis.

3.1 Initial Survey Design

Due to the limited time frame of this project: it was essential for the initial survey to be released as soon as possible. Because of this, there was less opportunity to research and design the best survey according to user-oriented design techniques. That said, thought was still taken, where possible, to ensure questions were sensible, length varied, and that all participants gave full informed consent before answering. Questions were asked about their experience with TTRPGs and GMing (context part 2), what features and tools they would like a software tool to provide (context part 3), about any accessibility or usability requirements, and in which form they would like the program to take, in other words, which platform they preferred as well as if the application should be a website or application (context part 4). The survey questions are shown in Appendix B.

In accordance with university ethics policy, an initial consent form was included to give responders informed consent and the opportunity to contact

me before answering. This then led to the true survey which was anonymous. Microsoft Forms was used to provide both these surveys due to its link to the university network, its trustworthiness, and its technological ability to avoid confusion (for example: on paper, a user may not understand how they are expected to fill in a checkbox, while on the digital form, there is only one possible way to do so (by clicking it)).

The survey was sent to the University of Bristol Gamesoc[3], a student union society for board games and tabletop roleplaying games. The society includes a large number of members within the target audience of my application, however, this does introduce some bias as they are likely to be English-speaking, student-aged, people who live in Bristol, UK: which is a stricter sample than my outlined user and stakeholder group. Additionally, thanks to my supervisor, the survey was sent to the computer science staff group chat, the Bristol Interaction Group group chat, the Software Engineering Project (a university unit) group chat, and the group chat for this dissertation unit.

3.2 Analysis and Results

Following the steps of Section 2.5.2, the responses (available in supplement file `Briar_Baker_S1_Results.xlsx`) were first read through and I came up with the initial concepts as to what the codes should be. Then re-reading the data, I generated the codes in Table 5 (Appendix C). As part of the survey was asking about which existing tools participants were already aware of and potentially using, these were counted separately as this question was more statistical (see Table 6 (Appendix C)). Afterwards, I distilled the codes into higher-level themes in Table 2. Then analysing these themes against the original data I decided they were satisfactory and ready to be designed around.

Themes
D&D most popular system by large
Majority of responders rated themselves as 4/5 experienced with GMing
Many who had issues with a website were due to wanting to use it offline[4]
D&D Beyond, Roll20 are very well known
Foundry, 5etools and World Anvil are also popular
Clear most requested features are information searching and Ease of use

[3] https://www.bristolsu.org.uk/groups/gamesoc-4e16

[4] Many respondents who favoured offline use, did not select *website* earlier in the form as they held the belief that a website was inherently online only (see: Section 3.3.3)

Next priority features are: note-taking, entity management, Ability to customise, speed of use
Next player sheet links and battle maps (more online game style features)
Suggestions for dark mode and other colour-scheme controls for accessibility and preference

Table 2: First Round Themes

3.2.1 Quantitative Data

In addition to the long/text answer questions, the survey included some quantitative questions with the following results (this data can be seen directly in the supplement file `Briar_Baker_S1_Results.xlsx`):

3.2.1.1 Which TTRPGs have you played or GMed?

D&D was by far the most popular system with 100% of responders claiming to have played or GMed it. Call of Cthulu, Cyberpunk RED, and Vampire: The Masquerade were the next most popular (and specifically outlined) systems with 31.6%, 26.3%, and 26.3% respectively. The other two I had outlined (Pathfinder, and Blades in the Dark) had only been played by 10.5% of responders. Interestingly 36.8% of responders had played other systems, although there was no pattern in what those systems were.

3.2.1.2 How much experience with GMing do you have (across systems)?

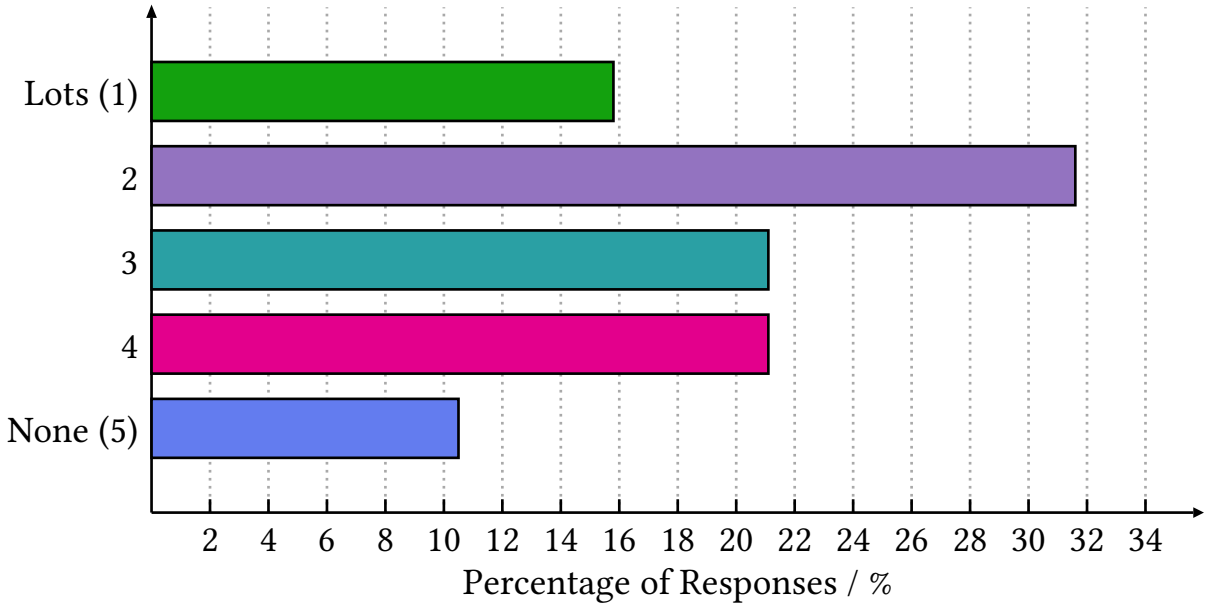


Figure 6: “How much experience with GMing do you have (across systems)?”

3.2.1.3 A tool made specifically for D&D would likely be able to cover more specific details, while a program made system agnostic could cover more TTRPGs. Would you prefer a more feature detailed tool or a tool which can be used for any system?

53% of responders preferred a D&D-specific tool, while 47% wished for system agnostic. While a preference for D&D is shown, D&D was favoured by only a narrow margin.

3.2.1.4 What form would you like a potential GM tool to take?

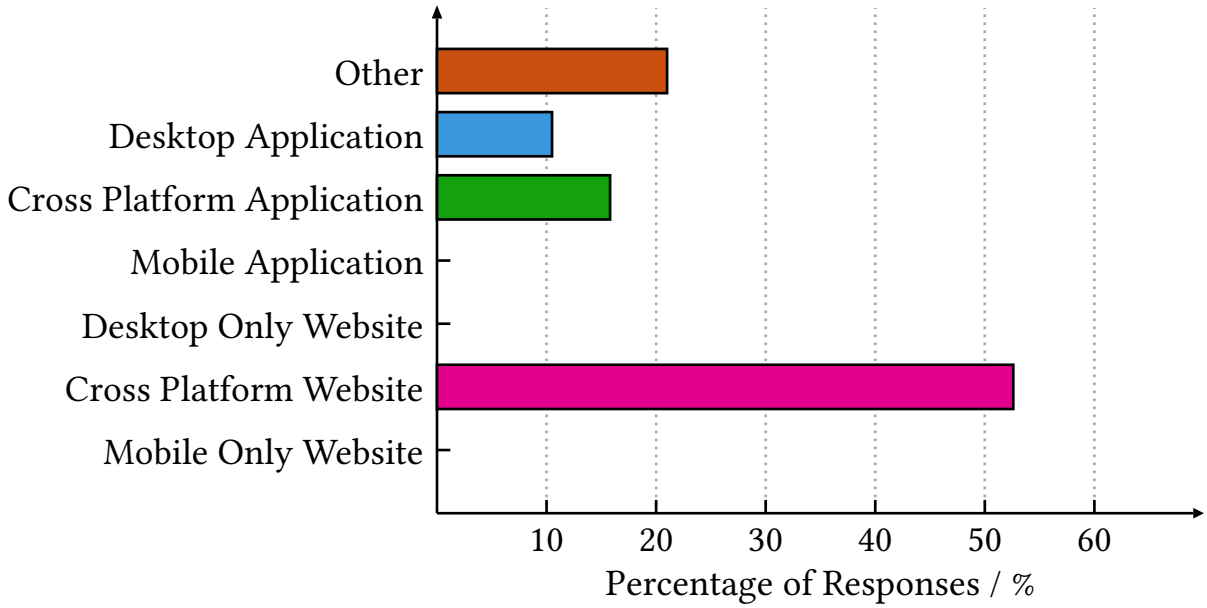


Figure 7: “What form would you like a potential GM tool to take?”

3.2.1.5 To summarise:

- D&D was the most played system by large.
- Responders are generally experienced with GMing.
- A slight preference for a D&D-specific tool.
- A strong preference for a cross-platform website.

3.2.2 Conclusions

Section 6.3.3. of ISO 9241-210:2010 summarises that specification of user requirements shall include:

1. The intended context of use;
2. Requirements derived from user needs and the context of use — for example, there might be a requirement for a product to be used outdoors;
3. Requirements arising from relevant ergonomics and user interface knowledge, standards and guidelines (e.g. accessibility requirements are found in ISO 9241-20 and ISO 9241-171);

4. Usability requirements and objectives, including measurable usability performance and satisfaction criteria in specific contexts of use — for example, an objective might be that 90 % of the intended users can successfully divert an incoming call to voice mail, or for the aesthetic design of a Web page to achieve a given user satisfaction score;
5. Requirements derived from organizational requirements that directly affect the user — for example, a call centre system might require that customer calls be answered within a specific time frame.

— ISO 9241-210:2010 §6.3.3

From my initial survey, I define:

1. The context of use is planning or preparing a TTRPG game
2. This is an indoor activity but user devices will vary: with some users on smaller devices such as mobile phones and others on larger screens such as a laptop: the solution should be cross-platform to accommodate these needs. Additionally, as previously mentioned, the website should work offline once loaded as in many contexts an internet connection may be unreliable or unavailable.
3. The interface should be designed to be accessible with room for alternative colour schemes (See: Section 3.3.1)
4. This requirement is difficult to reach with my small sample size of users
5. While there exist organisations for TTRPG running, this software is aimed at hobbyists and therefore there are no organisational requirements.

3.3 Prototype-1 Plan

With a strong preference for a website with the compatibility to work cross-platform, I planned to make a website. Since the most requested type of tool was something to help with information. I designed a tool inspired by websites I've used and found well made (Notably: *Garland Tools* [38] and *5etools* [3]). Information would be broken down into blocks (I decided to call **cells**), these blocks would take the form of individual rules, characters, stat-blocks[5], areas etc. These cells would be editable within the application and able to be found with a search bar in addition to hyperlinks between cells (similar to Wikipedia linking between pages).

[5] A Stat-block is the statistics of a particular creature or NPC

Since a desire to work offline was expressed Service Workers would be used to allow the browser to cache and use a local copy of the website without an internet connection (See: Section 3.3.3).

Many of the existing TTRPG tools also have a statistics and formulae system. Meaning, when the result of a rule includes a reference to a cell's statistics (For example Damage = 1d4 + DEX modifier + 1) the system can recognise the reference to these statistics and perform the required computation for display (For example: if DEX mod is 2: Damage = 1d4 + **2** + 1 = 1d4 + 3).

This system wouldn't benefit from being D&D-specific due to its focus on information searching and organisation so I made the decision to disregard the surveyed preference for a D&D-specific system.

Additionally, I planned to have a Tree-structure organisation, where each cell has a place within a file-system-like tree-style structure, with folders and cells. However, unlike a file system, cells can appear in multiple places as the locations in this structure are just references. This allows for more customised and intuitive organisation. For example: An evil cave monster stat-block may be included in both the folder for the cave in which it lives (allowing a GM to find it easily when looking at the location's information) and also in a list-of-monsters folder (allowing the GM to pick preplanned monsters to generate an encounter on the fly).

Finally, I planned for the exporting and importing of cells and folders so that communities can represent a system's rule set and let others utilise it without the effort of adding it to the software themselves. In addition, this allows a user to utilise this software over multiple devices as they can share their custom information between the devices with ease.

3.3.1 Interface Design

Inspiration was taken from the accessibility and simplicity of Google's Material Design [39]. Favouring simple block colours, icons, and the use of shadows to hint to the user how the User Interface (UI) can be interacted with. Nerd fonts [40] provides a convenient source of material design icons (as well as icons from other famous fonts/frameworks). Icons have been a staple in GUIs since the start [41], [42] and research [43], [44] has shown the effectiveness of icons for fast and intuitive Human-Computer Interaction (HCI). With more time, I would have first performed a user evaluation period of UI mockups.

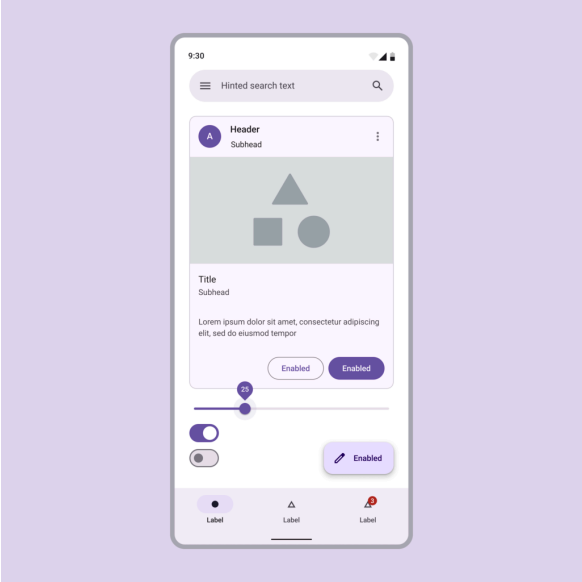


Figure 8: Material You (On Mobile) [4]

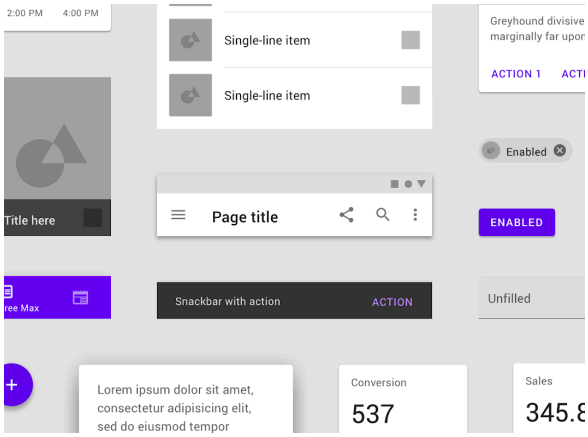


Figure 9: Material Design 2 Components [5]

3.3.1.1 Bias

Since my sample of users as well as my autoethnographic (Section 2.5.3) influence are English-speaking, UK-based, TTRPG players: there will certainly be some bias as to what appears to be a good design. Research [45] suggests that the effectiveness of icons and their potential different designs can be influenced by the user's culture.

For this project, I outline my target user base as those in the English-speaking TTRPG community with potential for future work (Section 5.3) to explore how to make the design process and resultant software more culture and experience-agnostic.

3.3.2 Front-end Only

An information handling app for TTRPGs shouldn't be handling any sensitive data or information, which is the usual reason a full stack system is used (Front end application that communicates back and forth with server back end). Additionally, the app should not need to perform any complex computations which would require better performance than the client may be able to provide. Therefore, my application can be made front end only which comes with the following benefits:

- There is no need for costly server hardware or rental to support the app

- The app can work without an active internet connection to the server, and its speeds will not depend on its ability to communicate with a server
- A service worker (Section 3.3.3) will be able to cache the entire application into the user's browser.

3.3.3 Service Workers

> Service workers essentially act as proxy servers that sit between web applications, the browser, and the network (when available). They are intended, among other things, to enable the creation of effective offline experiences, intercept network requests, take appropriate action based on whether the network is available, and update assets residing on the server.
>
> — Mozilla Docs [46]

Service workers exist as part of the World Wide Web Consortium (W3C) standards for websites and amongst other features [47], will allow my website to store a cache of itself in the user's browser so that it can be loaded and used without a present or fast internet connection. This serves to both speed up the use of the application (as refreshes can be handled from the cache and will not always be performing web requests) and also to allow users to utilise the application in situations where they do not have good internet available (e.g. they are on a train).

3.3.4 WebAssembly and Rust

For front-end web development, there are only two options which will be supported by the vast majority of browsers. These are: JavaScript or WebAssembly. To create my website, I decided to use WebAssembly (via Rust) instead of a more standard JavaScript framework. There were four main reasons behind this decision.

- JavaScript implicitly casts between types [48]. For instance, where a function expects a `String` but is given a `Number`, the runtime will attempt to interpret that `Number` as `String` rather than throw an error. While this may be helpful for quick usage of different types (For example: results from user input will always be a `String`, even from a number field), it can lead to unexpected behaviour when erroneous code is run, rather than creating errors which would point directly to the problem (therefore making solving the problem more difficult).

- WebAssembly can be much faster than JavaScript. A. Turner showed that using the WasmBoy benchmarking tool, WebAssembly performed faster than the other options on all mobile and desktop browsers (with extreme improvements over Firefox's JavaScript) [49].
- Rust as a language has a great community and ecosystem, as well as a strict and complex type system which offer many advantages over other languages (as well as being my personal preference as a developer).
- Despite the benefits of WebAssembly and it's existence as part of the W3C standards since 2019 [7]. The ecosystems for WebAssembly are relatively immature. While this could equally be a negative (and is discussed in Section 4) it also allows me to evaluate the current tooling both for its effectiveness and how the User Oriented Design frameworks hold up to this unique situation (of using experimental frameworks). That said, Rust as a language has a more mature ecosystem than many programming languages and therefore I would not be too lost for features or support.

3.3.5 Yew

Yew [6] is one of the frameworks for WebAssembly in Rust. It chose to use this library because it is further developed than alternatives such as Sycamore and is based on ReactJS (A JavaScript framework for front-end web development). Additionally, it hooks into existing libraries like Trunk and Web-Sys which are themselves commonly used and well-developed (Other libraries in the space also do this, but not all).

3.3.6 Tailwind CSS

To style the website I planned to use Tailwind CSS [13]. An open-source CSS framework that provides classes to represent CSS concepts for quick prototyping and designing. Additionally, it hooks into Trunk and therefore can be automatically regenerated when making changes to my code.

3.4 Prototype-1 Implementation

3.4.1 High-level architecture

As planned above, information would be broken into cells represented by Yew components. Each cell would then contain any number of fields (also Yew components). Initially, I implemented a basic text field and a Row-ordered and

Column-ordered field which would allow for more advanced (if manual) formatting of a cell. These later fields would be unique in that they are fields which can themselves contain other fields.

There is also a top-level component known as the Table, which would contain as children all the cells that are currently open.

3.4.2 UI

Figure 10: Inital Mockup of UI using Tailwind CSS

Initially, I blocked out the UI using Tailwind CSS which allowed me to experiment with layout before committing to any features. I decided on a grid of 4 columns to store the cells in as I found them to have superior readability (See: Figure 10).

After implementing the features described in the below sections, I came back to the design of the UI to greatly improve it. Both a Light (Figure 11) and Dark (Figure 12) colour scheme was implemented and the UI was kept minimal with the use of icons as planned in Section 3.3.1.

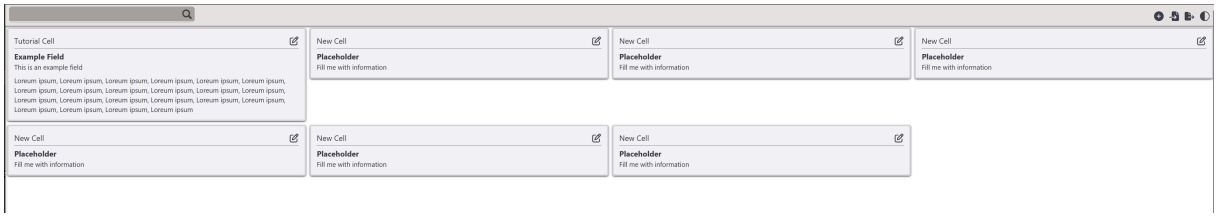


Figure 11: Same UI as Figure 12 but in light mode

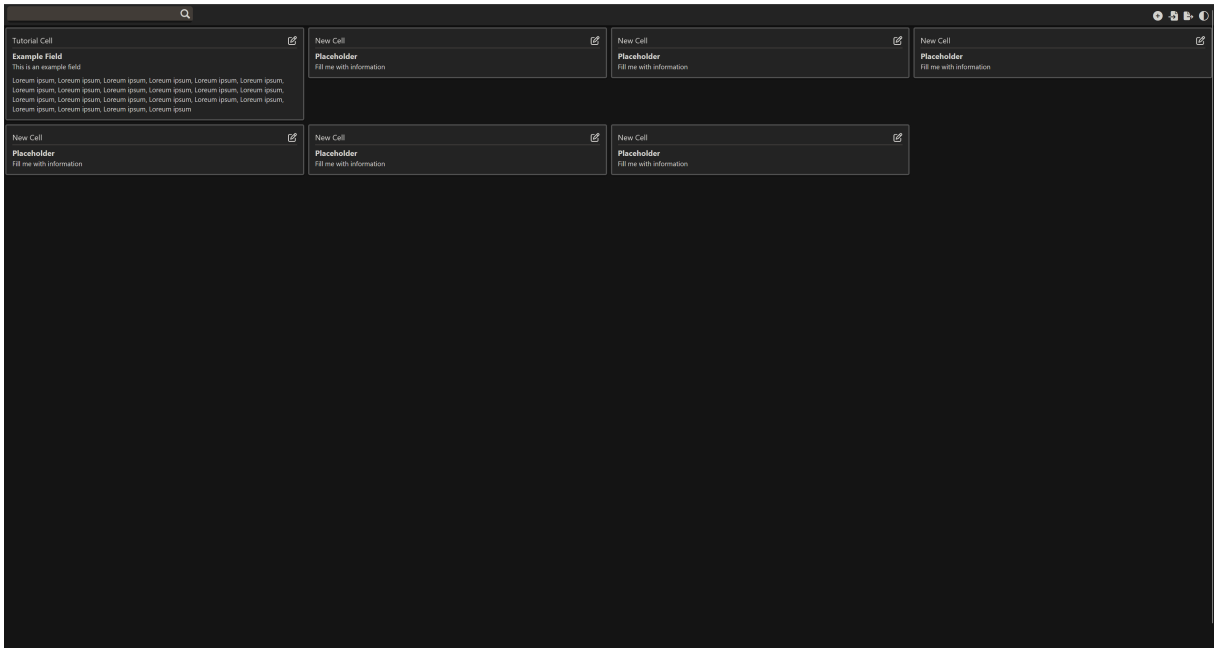


Figure 12: Improved design of UI in Dark Mode

3.4.3 Component System

Components offer an encapsulated approach, splitting pieces of UI into separate and reusable blocks of code. Components handle all the logic of how they should be displayed (i.e. they generate their own HTML) which allows parent components and other logic to abstract away any information about how the component functions. For example: the table component simply lays out a grid and then displays its cells within the slots of that grid, without any knowledge of what those cells are.

> Components are the building blocks of Yew.
>
> They:
> - Take arguments in form of `Props`
> - Can have their own state
> - Compute pieces of HTML visible to the user (DOM)
>
> — Yew Documentation [50]

3.4.3.1 Function Components

Function components, as the name suggests, are built from functions. The properties they inherit are the parameters of the function. And the state

management is done in terms of the scope of the function. They offer a simple and quick method of creating components, however, my prototype needed more advanced units to be able to customise the functionality to a greater extent and to have more complex interactions between components. Therefore, my project utilises Struct components instead.

3.4.3.2 Struct Components

Struct components exist as their own data types. They are more object-oriented in design and are built by creating several `structs`:
1. A `struct` representing the properties the component takes in
2. A `struct` representing the internal state of the component
3. A `struct` representing the component itself

Additionally an `enum` representing the types of messages it can receive (See: Section 3.4.4)

This final `struct` implements the Component `trait` (akin to interfaces in Java and similar OOP languages). This `trait` defines the following methods that I use:

- `create(ctx: &Context<Self>)`: How does the component create itself given the current context (This context will include it's properties (`struct` 1))?
- `update(&mut self, ctx: &Context<Self>, msg: Self::Message)`: How does the component react to receiving a message?
- `fn changed(&mut self, ctx: &Context<Self>, old_props: &Self::Properties)`: How does the component react to a change in it's properties?
- `fn view(&self, ctx: &Context<Self>)`: How does the component display itself/what HTML is generated?

View and Create are required for a component to exist while the others have blanket implementations (generally to do nothing). There are also additional methods to control how the component reacts to being rendered, how it destroys itself, and when using Server-Side Rendering (SSR): how to prepare the state that the client will receive.

My program is front-end only, meaning there is no server involved and the entire application runs on the client . Additionally, none of my components need to react to being rendered and the blanket implementation of how a component should be destroyed is fine, as I am not performing any custom memory management.

3.4.4 Message System

As mentioned above, one of the two ways that a component can react to changes and events is via messages (the other being a change in its properties). They are defined by an `enum` which is a sum type, meaning they are more advanced than the `enums` from languages like C. Each variant (value) of the `enum` can have its own type, generally, these will be tuple types defined within the `enum` itself. For example:

```
pub enum StatUpdateType {
    Name(String),
    Stat(isize),
    Remove,
}
```

Here two tuple types `Name` and `Stat` are defined each having a single subtype (`String` and `isize`). Additionally the variant `Remove` has no typing to it (other than the typing of being an `enum` variant). The variants of an `enum` can also be separately defined types (e.g. `struct` and `enum`). This allows a variety of data to be sent alongside a message. As with the above code snippet: the stat field can send a `String` when it is broadcasting a new name for itself, or a number (`isize`) when it wants to broadcast a new value for itself.

3.4.4.1 Property Changes

The majority of updates were done via the Yew message system as components cannot change their own properties. This restriction means that when updating themselves, components send a value to themselves to then update a local state (`struct` 2).

3.4.5 Callbacks

Another feature from ReactJS implemented in Yew is callbacks.

> Callbacks are used to communicate with services, agents, and parent components within Yew. Internally their type is just Fn wrapped in Rc to allow them to be cloned.
>
> — Yew Documentation [51]

Callbacks are predefined functions, whose reference is remembered and tied to a UI component (e.g. a button) such that when an event occurs (e.g. the button

is clicked) the function is run. Generally, I have utilised these callbacks to create basic functions in which a component can fire a message to itself when events relevant to its parts occur. For example: every field has a delete button, and this button is given a callback when created, this callback fires a Deletion event to the scope of the component (which will be later handled by the `update` method to delete itself).

Callbacks allow my components to link actions and messages to UI events at render time, when the scope is still known, thus making it easier to act on specific information. This means that the scope can be included in the message so, for example, the component knows exactly which button was pressed and also that the button is one that it generated (and not the same type of button from another cell).

3.4.6 Yewdux and Persistence

The program needed to remember its state over refreshes and closures. It would not be ideal if a user closed their tab one day, only to return another day and all the information they had put into the app was gone. Therefore, persistence is needed to store the user's data in a way which survives refreshes and closures. The Web Storage API [52], [53] provides the perfect mechanism to do this: a map (key-value pairs) can be stored on the client's storage in JSON format and this information persists until manually cleared. Unlike cookies, which although similar, have a low maximum capacity (~4KB per cookie, and a varying amount per domain [54]), and are sent with each HTTP request (unnecessary with my front-end only app): `localstorage` has a capacity of 5MiB per domain [55] and is not sent to the server.

To store my data in this map, I needed to convert it into JSON format. This requires two things:
1. A way to access the entire state of my application
2. A method to serialise that state into JSON

Rust has a mature ecosystem for serialisation into JSON and other formats via the package Serde [9]. Therefore, the real challenge of persistence is accessing the data itself. The standard Yew component setup means that components have private access to their local state, and so accessing this data for serialisation is next to impossible. To solve this, a package known as Yewdux [8] provides a top-level global storage where components can store their state rather than locally. This makes it trivial to serialise the entire state (Yewdux

has built-in features to do this via serde) but requires that each component knows where its state is within the global store, and that any deleted component properly clears the area for new data.

3.4.6.1 The arena

The arena datatype (internally a `Vec<Option<...>>`) stores the types in an efficient block of memory, using numbers to access the values the same as any list or array (each component can remember its index). When a component is deleted, the arena marks the area clear (set the `Option` to `None`) and remembers that index for when it next goes to add a component. These remembered indexes are given priority for inserting new data, meaning that old memory regions can be reused and that deleted data cannot insecurely remain accessible.

3.5 Testing phases

> Evaluating designs with users and improving them based on their feedback provides an effective means of minimizing the risk of a system not meeting user or organizational needs
>
> — ISO 9241-210:2010 §4.4

User-focused design requires evaluating the stages of the product with real users and iterating the solution based on their feedback.

3.5.1 Good survey practices

While more concrete, pre-made, surveys exist which have the benefit of being pre-researched (Such as the Computer System Usability Questionnaire [56]) these depend on a more complete project to be truly useful and thus a customised survey was designed.

Survey advice was taken from Research Methods in Human-Computer Interaction [57] Chapter 5 due to its popularity as a textbook for HCI research. Effort was taken to avoid:

- Double-barrelled questions: which ask more than one question in one go
- Negatives: These can confuse the responder (instead ask the positive version and responders can reply with their disagreement if needed)
- Biased wording: questions that lead the responder to a certain answer

- Hot-button words: words that can trigger more biased answers (generally politically charged and so not difficult to avoid in my survey)

As before the survey was provided with informed consent forms first and via Microsoft forms to ensure responders were fully aware of what I expected from them and to avoid potential confusion which coincides with the advice the book gives about presenting surveys. Microsoft Forms also provides additional accessibility options (variable font sizing, read-aloud options, etc.) to meet the book's advice of improved accessibility in the surveys. Since my project involves developing a software tool, the limitation of digital surveys in which they restrict those without electronic access from responding is largely not an issue, as those responders would not be in the target user base. However, bias in favour of wanting a software tool (especially concerning my initial requirements analysis) is likely due to this limitation.

3.5.2 Survey Design

- The survey begins with instructions
- The survey was split into two sections so that users would not feel overwhelmed with the size of the survey.
- Long answer questions were designed not to expect lengthy answers so that users would not need to invest significant time to provide useful results.
- Related questions were grouped together
- Questions are asked about the user interacting with the software in specific ways
- Long and short answer questions are intermixed
- Questions that depend on previous answers are hidden until the relevant answer is given to avoid confusion

The full survey can be seen in Appendix D. As before, the survey was sent to Bristol University Gamesoc[6].

3.5.3 Interviews

I collected 6 individuals to perform one-on-one interviews with via online voice chat on Discord [58]. Discord was selected for its ease of use, the ability for the interviewee to share their screen and the fact that all individuals already made use of the software. Informed consent was collected before asking any questions and where possible, meetings were recorded via OBS so

[6]https://www.bristolsu.org.uk/groups/gamesoc-4e16

that I could re-review them later. These recordings were edited to be as anonymous as is reasonable (and are included as supplementary materials). The interviews were conducted in two batches of 3 interviewees so that two stages of review could be performed, and at both stages: the interviewees had never interacted with the software and this project before (to avoid the bias that could cause). Additional information about each participant is shown in Table 3.

Participant	Gender	Experience (1-5)	Platform	Browser
1	Male	5	Linux (GNOME)	Google Chrome
2	Male	4	Windows	Google Chrome
3	Non-Binary	4	Windows	Mozilla Firefox
4	Male	3	Windows	Mozilla Firefox
5	Non-Binary	3	Windows	Google Chrome
6	Male/Non-Binary	5	Windows	Mozilla Firefox

Table 3: Interviewee descriptions

Research Methods in Human-Computer Interaction [57] was reused to produce higher-quality interviews (Chapter 7).

These interviews followed a similar structure to the surveys (attempting to get the user to perform a specific task, inquiring about their experience, and asking questions about the software overall) but allowed:

- The user to discuss their thought processes when attempting the task
- Me to ask additional questions on the fly
- To clarify questions and answers between myself and the interviewee
- To discuss the software and thoughts in greater detail

To encourage more conversation, the interviews were performed casually without overly formal speech or interaction.

3.5.4 Feedback and Results 1

Unfortunately, far fewer responses were received in the first prototype review (compared to the requirements analysis). Generally, responders and interviewees found the software easy and intuitive to use. A bug with HTML

pasting was discovered, where if you copy from a website or other location with rich text formatting, pasting it into the application would cause additional HTML to be included in the paste (that which provides the rich formatting). Interview Participant 1 discovered that LLMs such as ChatGPT were successful at generating importable JSON in the correct format by being given the example JSON I had prepared and some information to include. Suggestions were made for drag-and-drop reorganising of the cells in the table, templates for cell creation as well as the ability to export specific cells. Support was shown for my plans for additional field types, the formula system, and linking cells to each other.

The results from this stage of surveying can be found in auxiliary file `Briar_Baker_S2_Results.xlsx`.

3.6 Next iteration

3.6.1 Planned Changes

Based on the feedback received and the time available I decided to prepare the following for the next review phase:

- Fix the HTML pasting bug
- Add more field types
- Add linking between cells
- Add a formula system for stat calculations
- Setup service worker for proper offline use

3.6.2 Implementation

The planned steps and fixes proceeded as follows:

3.6.2.1 HTML pasting bug

This was caused due to how the `contenteditable` attribute [59] works in HTML, setting this to `plaintext-only` instead of `true` fixed the issue.

3.6.2.2 Additional field types

An image field was added, due to time constraints, these images would use existing web-stored images only as implementing the ability to upload and store fields in the browser is an unnecessary complication for this prototype.

These images would still be stored in the browser's cache and thus the application will continue to work offline.

A statistics field was added, which includes a number value and a text name. This field is designed to be used with the formula system.

3.6.2.3 Linking between cells

Regular expressions were used on the existing text field to look for instances of `[number]` within the text. If these numbers corresponded with the index of one of the cells, it would be replaced by a hyperlink to open said cell.

3.6.2.4 Formula system

Regular expressions were again used to look for `$formula$` in the text field. This `formula` string would be interpreted by a library `eval-expr` [14] which would interpret the string and perform a range of mathematical operations to arrive at a result. A context based on the cell's statistics fields was generated so that they could be included as variables within the formula (e.g. `2 + DEX_MOD + 1`).

3.6.2.5 Service Worker

As documented by the Mozilla Web Docs [46], a simple script was created to cache the entire application in the browser for offline use of the application.

3.6.2.6 Summary & UI

With these additional field types, more complicated cells could be created. Figure 13 showcases the statblock of a Bat from D&D, including its image, statistics, abilities with formulas that reference these statistics, and a link to another cell (Deafened).

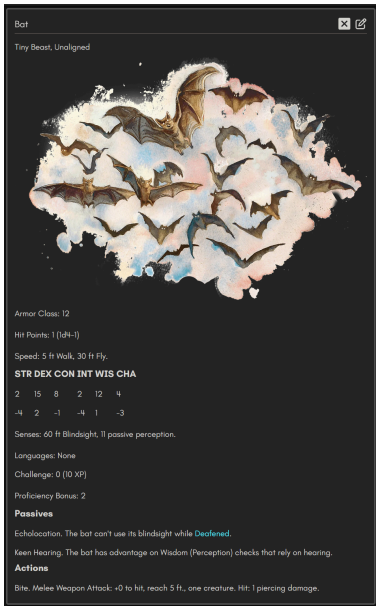


Figure 13: A more complicated cell for a Bat in D&D

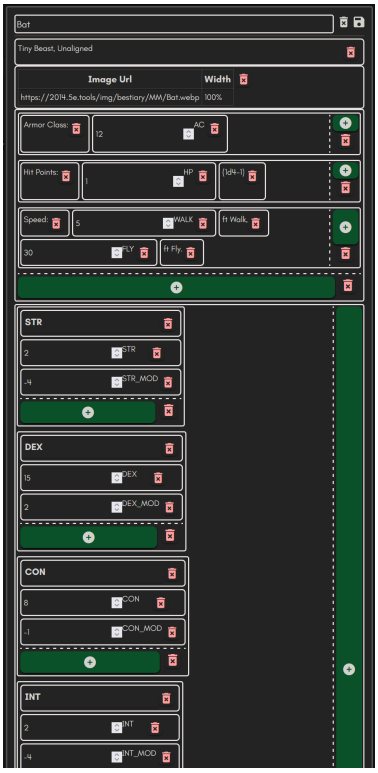


Figure 14: Editing of the bat cell

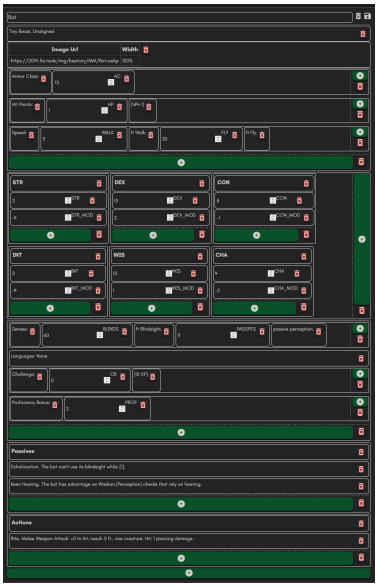


Figure 15: Editing of the bat cell, zoomed out to fit on one screen

Figure 14 shows the editing mode of this same bat cell. Due to the additional information that every part of the cell needs when in editing mode, the formatting flows off the page (Figure 15 shows how this editing mode looks when zoomed out). I considered implementing a system to automatically zoom the editing menu out to better preview all the fields, but since this would make the text and buttons difficult to interact with, I elected to keep it as it was.

Chapter 4
Critical Evaluation

4.1 Final testing

As previously mentioned, the final round of testing followed the same review survey as before and utilised the other 3 prepared interviewees.

Perhaps due to the timing of this second round of prototype feedback (coursework periods for many students), no responses were given to the survey outside of my interviews.

Interviewees favoured the interface layout and had no issues interacting with the software. Due to Participant One's experience with using LLMs before, the opportunity to integrate LLMs with the software was broached but did not receive much favour. Furthermore, The linking system was positively received as a way to interact with related cells.

4.1.1 Autoethnographic use in situ

As well as the user testing, as a potential user myself: I made use of the prototype in a real game of Dungeons of Dragons. I found the prototype useful and suitable for my requirements to searching information however inputting the initial information was quite laboursome. This suggests that priority for future work (Section 5.3) should be placed on implementing the aforementioned template system or creating a form of Domain Specific Language (DSL) for experienced users to outline cells and their information.

4.2 Framework effectiveness & Choice Review

To conclude this project: I discuss the benefits and limitations of both the HCI frameworks used and the software developed with. Reporting on my experience and likelihood to use them again were I to redo this project.

4.2.1 ISO 9241-210

I found the steps and considerations outlined in the standard to be effective when designing and developing my project. A number of the points were obvious to me but I believe they are still worth explaining - as what is obvious likely differs from person to person. The standard does not (and recognises so) go into detail on the methods and approaches that should be used to execute the steps it outlines. Therefore, the work of finding well-tested methods is left as a separate task for the user of the standard. I speculate that including references to recommended materials for such methods would increase the usability of the standards for projects like this one.

As was the trend in HCI standards I discovered in my research for this project, the standard is designed for larger projects - where there are developer teams consisting of multiple persons, longer time scales, and large user samples to test with. While I believe that this is to be expected, as the greatest use case of HCI research will be for large-scale projects, the result is a general lack of well-researched advice for small-scale projects. It is my theory that this may have some influence on the lack of User Oriented Design for smaller projects like a TTRPG assistance tool.

4.2.2 Research Methods in Human-Computer Interaction

In comparison to ISO 9241-210, the book identified by the above heading goes into great and useful detail on the methods that can be undertaken when designing and developing with a user-oriented approach. It provides positives and negatives to different methods, when to best use specific approaches, points to additional materials that may be useful, and outlines the direct steps that should be taken for direct results. While still affected by the same limitations of the standard (in that it was designed for larger-scale projects), the detail and advice provided make it far easier to customise the approach to fit the project.

4.3 Software effectiveness & Choice Review

4.3.1 Rust

Rust's strict and vast type system made developing the software solution especially pleasant. The ability of the compiler and static analysis tools to find

errors at compile time rather than runtime means that useful information could be provided which made fixing bugs far easier. For example: it can easily provide the line number in which errors occur and the type system causing errors provides inherent documentation as to what behaviour is expected and what behaviour is occurring.

The maturity of the community and its libraries meant implementing difficult features far less time-consuming as efficient community-provided solutions such as `evalexpr` for evaluating formulas from Strings can be pulled in.

The Cargo build system (which is an official part of the language unlike other programming languages such as C++) makes managing libraries painless and quick. Additionally, it also ensured that it was simple to get the project building on my different devices since as long as Cargo and Rust were installed, the process of building all dependencies and then the project was handled automatically without issue.

4.3.2 Yew

Yew's React-inspired system was effective to use and supported almost all of my needs without additional libraries. The WASM system resulted in my program running efficiently and had an effective method to report runtime errors, maintaining and reporting more information that was relevant to the original rust code than I had initially expected.

However, I did find the documentation for the library very sparse. While there is automatically generated documentation for everything within the library, without human explanation, many of these features are not intuitive to make use of (even with the benefits that types provide to automatic documentation). Furthermore, many of the features I ended up using had their documentation hidden from the main website, as they are part of a "future version" rather than the "current release". Personally, since the library is still in development, and the "current release" the main documentation refers to is a year older than the latest version (and thus missing many features) I see no reason the latest documentation should be hidden in this way.

4.3.2.1 Yewdux

While Yew did provide most functionality out of the gate, as mentioned in Section 3.4.6. Yewdux was required to provide a global store of information. Despite being a developmental library for a developmental library I found this

crate effective in solving my issue and even got in contact with the developer over Discord to ask for advice on how to utilise the system best.

4.3.3 Tailwind CSS

Tailwind CSS [13] provides a quick way to access CSS actions via only the HTML and is popular in the industry. However, it provides no themed implementation of styling itself, leaving that work to me.

4.3.4 `evalexpr`

This library was used to evaluate the mathematical formulas from strings. Using it was quick and intuitive and offered built-in support for a wide range of operations and easy implementation for using user-defined variables.

4.3.5 Rust Fuzzy Search

I utilised this library to implement the search bar. While it was quick and easy to integrate, the original crate [10] lacks support for sorting references to Strings (rather than Strings themselves) which limits its capability to sort data types like my Cells. Luckily for a past project, I had already customised a fork [11] of this library with my own changes to fix this limitation.

4.4 Choice Review

I have reviewed the methods and technologies I used and whether I would wish to use them again were I to perform this project afresh.

- HCI standards: I would plan to use a customised top-level approach (to replace ISO 9241-210) that better meets the smaller scale of this project and make frequent and immediate use of the methods outlined in the aforementioned book.
- Rust & Web Assembly: I would still aim to use WASM via Rust.
- Yew: I would explore alternative rust implementations for WASM that are potentially more mature (or simply take separate time to learn Yew outside of the project first).
- Yewdux: I would certainly make use of Yewdux to manage my global state as I did in this project if I used Yew again.
- `evalexpr`: I would gladly use this again for purposes I did in this project.
- Tailwind CSS: I would utilise a CSS framework with its own theming rather than using Tailwind to make my own.

4.5 Autoethnographic Benefits and Limitations

As mentioned in Section 2.5.3, the small size of this project (and most importantly the user sample size and single-person development team) meant that the influence of my own bias and experience would be inevitable within this project. It was however often a benefit and not a cost, based on the feedback the prototype has received as well as the experience I had performing this project: I theorise the benefits and limitations of my autoethnographic influence to be the following:

Benefits	Limitations
My experience as a developer allowed me to translate user wants into plans for software development	My bias as someone experienced with software led me to favour designs that were similar to those I am experienced with - independent of the quality of those designs
As a TTRPG player, I was able to build complex plans for the program from the abstract request for information searching without needing additional slow stages of user surveying	While refining the concept myself will have saved time, the resultant concept was strongly influenced by my personal wants rather than confidently reflecting the interests of my user sample
With experience with both software and TTRPGs my perspective was able to improve the quality of the prototype throughout development	As mentioned in Section 2.5, the design team of this project did not reflect multidisciplinary perspectives which affects the cultural and general accessibility of the prototype

Table 4: Benefits and Limitations of my Autoethnographic influence

4.5.1 Result Assessment

As outlined in Section 1.1 the aims for this project were:

1. Perform Requirements Analysis on TTRPG players and GMs
2. Iteratively design and develop a solution and its requirements
3. Test and evaluate the current solution on TTRPG players and GMs
4. Evaluate the final user response, my own experience, and what that could mean for user-oriented design approaches.

4.5.1.1 Requirements analysis (1)

A requirements analysis was successfully performed on a sample user base of TTRPG players and GMs. From this analysis, I was able to design a software solution that would aid in the features requested by the survey responders.

4.5.1.2 Iterations and Evaluations (2 & 3)

After the initial requirements analysis, two iterations of development and review were performed, allowing advice and feedback to change the plan for the application and its requirements. These stages involved testing the stages of the software with TTRPG players and GMs.

4.5.1.3 Final evaluation (4)

4.5.1.3.1 The prototype

The software prototype resulting from this project well consistently well received by users suggesting the effectiveness of User-Oriented Development strategies in making software users favour. However, the scope of the final prototype is quite limited due to the time constraints of the project and the additional time consumed by the User-Oriented Development methods. This could also affect the lack of UOD-developed applications for spaces like a TTRPG app - as those developing the application may be unable or unwilling to devote the extra time required to follow a UOD approach. This time consumption is likely also the reason that existing literature (See: Section 2.5.1) focuses on simply *designing* their solution/interface rather than implementing one.

4.5.1.3.2 The experience

As a report on the effectiveness of my selected resources and User Oriented Design for software development: I have identified several benefits and limitations of the available resources and the ones used. As with the prototype, the evaluation of these strategies would benefit from additional time spent repeating these methods and acquiring further experience to report on.

Chapter 5
Conclusion

5.1 Summary

In this project, I have created a prototype TTRPG software tool. Starting with an initial requirements analysis of 19 potential users from the University of Bristol Students' Union Games Society as well as University staff and student groups, I designed a cross-platform website to aid in the organisation and searching of system rule sets and other information. I then developed this two through two iterations of programming, user testing via surveys, interviews with three potential users, evaluation, and design. Creating an application using Rust, Yew [6], WebAssembly [7], and Tailwind CSS [13]. I made use of existing industry standards in User Centred Design and Human-Computer Interaction (ISO 9241-210 [30] and Research Methods in Human-Computer Interaction [57]) to aid and advise the process and I have evaluated the benefits and limitations of said resources and theorised to their influence on the current lack of UOD processes used for smaller projects such as this one: finding that available resources are designed for larger scope projects than this one.

This project achieved all aims set out in Section 1.1: utilising a requirements analysis to then iteratively develop a software solution using existing User Oriented Development resources and finally reporting on the experience.

5.2 Current Status

The resultant prototype is a Web Assembly website with the following features:

- Information blocks called cells to store information
- Ability to query these cells via the search bar
- Ability to link cells together (e.g. so you can open Cell B from a reference in Cell A)

- Ability to create, modify, and delete these cells.
- Ability to export and import a collection of cells
- Field types including plain text, heading text, web images, statistics, column formatting, and row formatting.

The code for this project can be found at https://codeberg.org/Bloodthorne/Dissertation/src/branch/main/info-app and the hosted version can be found at https://diss.batforge.dev/.

5.3 Future Work

As mentioned in Section 4.5.1.3 both parts of the project would benefit from further work.

5.3.1 Application Development

There were a number of features, both initially planned and later identified via user testing, that were not implemented in the final prototype.

- The tree structure organisation as outlined in Section 3.3 was a lower-priority feature and thus never got implemented. However, it did receive some favour when discussed with interviewees and thus future work would likely prioritise implementing this feature - the existing code base is already designed in such a way to support the implementation of this system.
- Users requested additional field types such as bordered tables and spacing fields which are not yet included. Again, the code base is designed to support the addition of new fields in future development without needless effort.
- Multiple features could be implemented to improve and quicken the information input experience:
  - There is room for integration with Large Language Models to automatically generate valid importable JSON.
  - Additionally, the development of a domain-specific language (DSL) to allow users to outline cells with pure text would allow advanced users to quickly develop cells without going through the existing menus for faster cell creation.
  - Finally, the potential for a template feature, where new cells could be created from copies of existing ones would also provide a method to speed up adding data to the system.
- To help users distribute and acquire pre-prepared JSON files to be imported: the program could be extended to include access to a database - where users could upload their storage rather than downloading the files locally. Users

wanting pieces of specific information such as their system's rule set could then search this database and import the relevant information without a local file
- While the application currently includes both a dark and light mode: support for user-defined colour schemes could improve both user experience and accessibility.
- Currently, the existing search bar queries only the titles of cells: to improve the search functionality, adding the capability for fields to be included in the search data with a given priority could allow users to search for cells based on information in their descriptions.

5.3.2 User-Oriented Development Evaluation

To further explore the effectiveness of the selected user-oriented development strategies without beginning a separate project it would be best to continue the iterative process with further stages of user testing and development. The longer this process is undertaken the more developed the application will be. Both increasing the amount of feedback received and also the detail of questions you are able to ask.

Additional work taken to discover larger user base samples for surveying and interviewing during these stages would be greatly beneficial - as this project was often limited by my limited access to such users.

Once a more finished and polished prototype is ready, exploring more standard HCI surveys such as the Computer System Usability Questionnaire [56], or the System Usability Scale favoured by existing literature (See: Section 2.5.1) would allow for a review of their effectiveness on the software development process for small projects.

Additionally, more in situ testing would be greatly beneficial. Having users attempt to use the tool for real planning and playing of TTRPG sessions would likely uncover additional bugs or feature requests.

As the utilised resources for User Oriented Development are designed for larger projects: analysis of the effectiveness of these resources could be improved by undergoing a separate larger project. While the results of attempting a smaller project are interesting due to their uniqueness, evaluation from a larger project may better reflect the effectiveness of these frameworks on the projects they are intended for.

Bibliography

[1] 'Google Trends ttrpg'. Accessed: Jan. 29, 2025. [Online]. Available: https://trends.google.com/trends/explore?date=today%205-y&q=ttrpg&hl=en-GB

[2] J. Crawford, J. Wyatt, R. J. Schwalb, and B. R. Cordell, *D&D Player's Handbook*, First Edition. Renton, WA: Dungeons & Dragons, 2014.

[3] '5etools'. [Online]. Available: https://2014.5e.tools/

[4] M. Design Google, *English: Material You in light mode.* 2023. [Online]. Available: https://commons.wikimedia.org/wiki/File:Material_you_light.png

[5] 'Material Design'. [Online]. Available: https://m2.material.io/design/introduction#components

[6] 'Yew'. [Online]. Available: https://yew.rs/

[7] 'Web Assembly Core Specification'. [Online]. Available: https://www.w3.org/TR/wasm-core-1/

[8] 'Introduction - Yewdux'. [Online]. Available: https://intendednull.github.io/yewdux/

[9] 'Serde'. [Online]. Available: https://serde.rs/

[10] E. Chiaramello, 'Rust Fuzzy Search · GitLab'. [Online]. Available: https://gitlab.com/EnricoCh/rust-fuzzy-search

[11] B. Baker, 'Bloodthorne/rust-fuzzy-search: Fork - Codeberg.org'. [Online]. Available: https://codeberg.org/Bloodthorne/rust-fuzzy-search.git

[12] K. Aleman, 'Fuzzy Search with PostgreSQL'. [Online]. Available: https://kaleman.netlify.com/fuzzy-search-with-postgresql

[13] ‘Tailwind CSS - Rapidly build modern websites without ever leaving your HTML.’. [Online]. Available: https://tailwindcss.com/

[14] ISibboI, ‘ISibboI/evalexpr’. [Online]. Available: https://github.com/ISibboI/evalexpr

[15] M. Kelly, ‘Rolling Dice in Crisis: The Unexpected Boom of Tabletop Gaming’. [Online]. Available: https://www.merchoid.com/uk/blog/post/rolling-dice-in-crisis-the-unexpected-boom-of-tabletop-gaming/

[16] S. Whitten, ‘Dungeons & Dragons had its biggest year ever as Covid forced the game off tables and onto the web’. [Online]. Available: https://www.cnbc.com/2021/03/13/dungeons-dragons-had-its-biggest-year-despite-the-coronavirus.html

[17] R. Silverman, ‘How to Be a Dungeon Master’. [Online]. Available: http://www.dndbeyond.com/posts/1452-how-to-be-a-dungeon-master

[18] G. Gygax and D. Arneson, *Dungeons & dragons*, vol. 19. Tactical Studies Rules Lake Geneva, WI, 1974.

[19] W. o. t. Coast, *D&D Dungeon Master’s Guide*, 1st edition. Dungeons & Dragons, 2014.

[20] D. Acharya, M. Mateas, and N. Wardrip-Fruin, ‘Interviews Towards Designing Support Tools for TTRPG Game Masters’, in *Interactive Storytelling*, A. Mitchell and M. Vosmeer, Eds., Cham: Springer International Publishing, 2021, pp. 283–287.

[21] E. Musacchio, L. Siciliani, P. Basile, and G. Semeraro, ‘Leveraging Large Language Models for Spell-Generation in Dungeons & Dragons’, in *Proceedings of the 10th Workshop on Games and Natural Language Processing@ LREC-COLING 2024*, 2024, pp. 61–69.

[22] ‘D&D 5e Community Wiki’. [Online]. Available: http://dnd5e.wikidot.com/

[23] ‘Foundry’. [Online]. Available: https://foundryvtt.com/

[24] ‘Roll20’. [Online]. Available: https://roll20.net/

[25] ‘Scoundry - A character keeper for Blades in the Dark’. [Online]. Available: https://scoundry.com/

[26] D. A. Norman and S. W. Draper, *User Centered System Design; New Perspectives on Human-Computer Interaction*. USA: L. Erlbaum Associates Inc., 1986.

[27] J. Greenbaum and M. Kyng, *Design at work: Cooperative design of computer systems*. CRC Press, 2020.

[28] D. Schuler and A. Namioka, *Participatory design: Principles and practices*. CRC press, 1993.

[29] K. Holtzblatt and H. Beyer, *Contextual design: Design for life*. Morgan Kaufmann, 2016.

[30] ISO-9241-210:2010, 'Ergonomics of human–system interaction — Part 210: Human-centred design for interactive systems', Geneva, CH, Mar. 2010.

[31] C. D. Farinango, J. S. Benavides, J. D. Cerón, D. M. López, and R. E. Álvarez, 'Human-centered design of a personal health record system for metabolic syndrome management based on the ISO 9241-210: 2010 standard', *Journal of multidisciplinary healthcare*, pp. 21–37, 2018.

[32] M. F. Aziz, D. P. Satya, and others, 'Designing Human-Computer Interaction for E-Learning using ISO 9241-210: 2010 and Google Design Sprint', in *2020 7th International Conference on Advance Informatics: Concepts, Theory and Applications (ICAICTA)*, 2020, pp. 1–6.

[33] C. König, T. Hofmann, and R. Bruder, 'Application of the user-centred design process according ISO 9241-210 in air traffic control', *Work*, vol. 41, no. S1, pp. 167–174, 2012.

[34] J. Brooke, 'SUS – a quick and dirty usability scale', 1996, pp. 189–194.

[35] V. Braun, V. Clarke, N. Hayfield, and G. Terry, 'Thematic Analysis', in *Handbook of Research Methods in Health Social Sciences*, P. Liamputtong, Ed., Singapore: Springer Singapore, 2019, pp. 843–860. doi: 10.1007/978-981-10-5251-4_103.

[36] V. Braun and V. Clarke, 'Successful qualitative research: A practical guide for beginners', 2013.

[37] T. E. Adams, C. Ellis, and S. H. Jones, 'Autoethnography', *The international encyclopedia of communication research methods*, pp. 1–11, 2017.

[38] ‘Garland Tools Database’. [Online]. Available: https://www.garlandtools.org/db/

[39] ‘Material Design’. [Online]. Available: https://m3.material.io/

[40] ‘Nerd Fonts - Iconic font aggregator, glyphs/icons collection, & fonts patcher’. [Online]. Available: https://www.nerdfonts.com/

[41] Y. Rogers, ‘Icons at the interface: their usefulness’, *Interacting with computers*, vol. 1, no. 1, pp. 105–117, 1989.

[42] ‘The first GUIs’. [Online]. Available: http://www.catb.org/~esr/writings/taouu/html/ch02s05.html

[43] C. Gatsou, A. Politis, and D. Zevgolis, ‘The Importance of Mobile Interface Icons on User Interaction.’, *Int. J. Comput. Sci. Appl.*, vol. 9, no. 3, pp. 92–107, 2012.

[44] J. Ferreira, J. Noble, and R. Biddle, ‘A case for iconic icons’, in *Proceedings of the 7th Australasian User interface conference-Volume 50*, 2006, pp. 97–100.

[45] J. H. Kim and K. P. Lee, ‘Cultural difference and mobile phone interface design: Icon recognition according to level of abstraction’, in *Proceedings of the 7th international conference on Human computer interaction with mobile devices & services*, 2005, pp. 307–310.

[46] ‘Service Worker API - Web APIs MDN_2025’. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Service_Worker_API

[47] ‘Service Workers Nightly’. [Online]. Available: https://w3c.github.io/ServiceWorker/

[48] ‘7.1 Type Conversion | ECMAScript® 2026 Language Specification’. [Online]. Available: https://tc39.es/ecma262/#sec-type-conversion

[49] A. Turner, ‘WebAssembly Is Fast: A Real-World Benchmark of WebAssembly vs. ES6’. [Online]. Available: https://medium.com/@torch2424/webassembly-is-fast-a-real-world-benchmark-of-webassembly-vs-es6-d85a23f8e193

[50] ‘Function Components | Yew’. [Online]. Available: https://yew.rs/docs/next/concepts/function-components

[51] ‘Callbacks | Yew’. [Online]. Available: https://yew.rs/docs/next/advanced-topics/struct-components/callbacks

[52] ‘Web Storage API - Web APIs’. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Web_Storage_API

[53] ‘Local Storage | Mozilla Docs’. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage

[54] ‘Using HTTP cookies - HTTP | MDN_2025’. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/HTTP/Guides/Cookies

[55] ‘Storage quotas and eviction criteria - Web APIs | MDN_2025’. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Storage_API/Storage_quotas_and_eviction_criteria

[56] J. R. L. and, ‘IBM computer usability satisfaction questionnaires: Psychometric evaluation and instructions for use’, *International Journal of Human–Computer Interaction*, vol. 7, no. 1, pp. 57–78, 1995, doi: 10.1080/10447319509526110.

[57] J. Lazar, J. Heidi Feng, and H. Hochheiser, *Research Methods in Human-Computer Interaction*, 2nd ed. Elsevier, 2017.

[58] ‘Discord - Group Chat That’s All Fun & Games’. [Online]. Available: https://discord.com/

[59] ‘contenteditable - HTML_2025’. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/HTML/Reference/Global_attributes/contenteditable

Appendix A: AI Prompts

AI was not used within this project.

Appendix B: Initial Survey

Briar Dissertation Survey One

*Required

1.Which TTRPGs have you played or GMed
Multiple choice.
- D&D
- Pathfinder
- Vampire: The Masquerade
- Blades in the Dark
- Call of Cthulu
- Cyberpunk RED
- Other

2.How much experience with GMing do you have (across systems)?
Likert.

Lots (1) | 2 | 3 | 4 | None (5)

3.Are you aware of existing software tools to help with GMing? And if yes, which ones?
Multi Line Text.

4.Do you use any of these tools, and why/why not?
Multi Line Text.

5.A tool made specifically for D&D would likely be able to cover more specific details, while a program made system agnostic could cover more TTRPGs. Would you prefer a more feature detailed tool or a tool which can be used for any system?
Single choice.
- D&D Specific
- Any System
- Other

6.What features would you wish for/prioritise out of a GM tool?
Multi Line Text.

7.What form would you like a potential GM tool to take?
Single choice.

- Mobile Only Website
- Cross-Platform Website
- Desktop Only Website
- Mobile Application
- Cross-Platform Application
- Desktop Application
- Other

8.Do you have any opinions over what a GM tool's UI should be like?
Multi Line Text.

9.Do you have any accessibility requirements that a GM tool might be able to account for?
Multi Line Text.

10.Any other information you would like to add to influence the tool being made?
Multi Line Text.

Appendix C: Initial Analysis

CODE	COUNT
Searching Information	9
Ease of use	8
Note taking	6
Entity Management	6
Customisable	5
Quick/frequency based	5
Visualise to Players	4
PC sheet access	4
Dark mode	4
Inspiration	1
NPC Creation	3
Random Generators	3
Dice Roller	3
Offline use	3
Colour schemes	3
Map Making Tools	2
Stat tracking	2
Character Sheets	2
Asset Creation	2
Auto Music	2
LLM/AI	2
Sharing Assets	2
UI scaling	2
Dyslexic font	2
Keybinds	2
Obvious information	2
Interaction with other DM tools	2
Encounter planning	1
VTT	1
Local backups	1
Organisation	1
Whiteboard	1
Player Sided (easy to pickup)	1

API	1
Tabs	1
Support all characters (Unicode)	1
Group sheets (e.g. crew sheet)	1
Speech control	1
Screen reader support	1
Symbolism	1
Strong mobile version	1

Table 5: Table of requirements codes

TOOL	COUNT
Roll20	10
D&D Beyond	8
5etools	5
World Anvil	4
Foundry	4
Kobold Fight Club	3
Dungeon Master's Vault	2
ChatGPT	1
D&D Wiki	1
Pathbuilder	1
Dungeon Scrawl	1
Dojon	1
Fantasy Grounds	1
Inkarnate	1
Avrae	1
Arkenforge	1
RPG Engine	1
Wildshape	1
GameMaster Engine	1
DMHelper	1
Dungeon Alchemist	1
Homebrewery	1
Syrinscape	1
Dokuwiki	1
PCGen	1
Campaign Cartographer	1
Pathguy	1
Tabletop Sim	1
Owlbear rodeo	1
Auto roll tables	1
Homebrew creation	1
Natural crit	1

Table 6: “What tools are you aware of?”

Appendix D: Iterations Survey

In accordance to the results of the first survey, a website to help organise and search TTRPG information has been developed. The following link shows an initial developmental copy for your review https:// diss.batforge.dev/. Before you start this survey, visit the link and have a mess around with the system. Alongside the link to this survey should also have been a .json file sent which contains preloaded information for the conditions in D&D 5e. I would like you to attempt the following tasks:

1. Load in the provided conditions information
2. Search for information you want
3. Try adding your own information

*Required

1.Where you able to load in the provided Conditions information?
Single choice.

- Yes, it was intuitive.
- Yes, although it wasn't intuitive
- No, I couldn't figure out how.
- No, I was not provided a file with this survey.
- Other

2.Where you able to add your own information to the system?
Single choice.

- Yes, it was intuitive.
- Yes, although it wasn't intuitive.
- No, I couldn't figure out how.

3.If yes: Please describe your experience?
Multi Line Text.

4.How would you rate the accessibility of the interface? (1 is not accessible, 10 completely accessible)

Rating.
1 2 3 4 5 6 7 8 9 10

5.How easy to use was the website? (1 near impossible to use, 10 completely easy to use)
Rating.
1 2 3 4 5 6 7 8 9 10

Currently this version is just a developmental copy
Planned additional features include:

- Hyperlinks in information blocks so that you can open referenced rule from another
- More field types such as Images and Numbers
- A statistics and formula system so that things such as damage modifiers can be auto calculated based on the statistics on the information block
- File-system like folder organisation, to provide an alternative way to find and store information.
- A better mobile version UI
- Caching so that the website can work offline once it has been loaded once

6.Did you encounter any bugs or similar issues using the website?
Multi Line Text.

7.Do you have any other features or similar you wish to be a part of the final website?
Multi Line Text.

8.Any other information you wish to add (Including any feedback for future surveys)?
Multi Line Text.