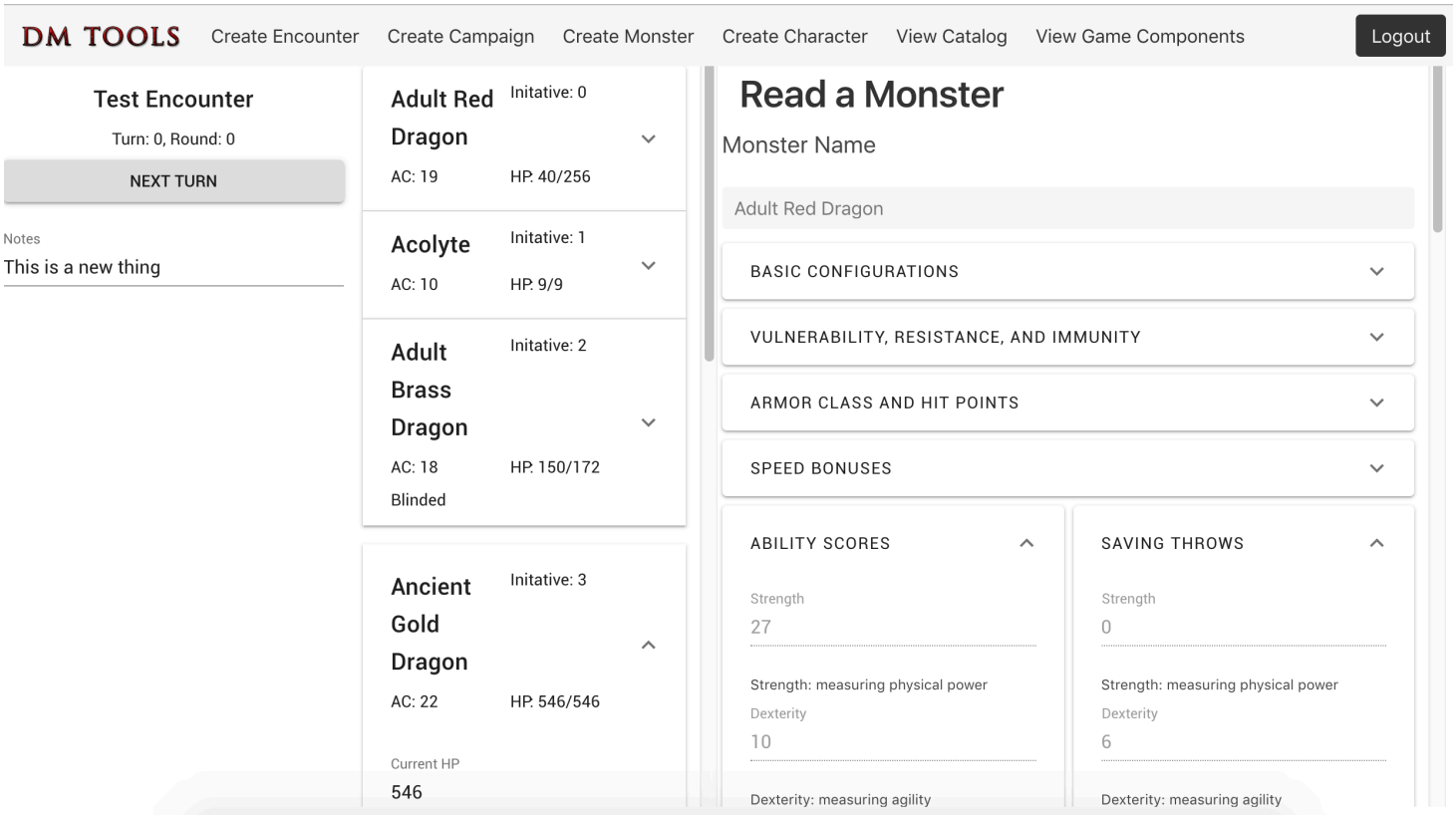
DM TOOLS

By: SPP Team 500 (Heather Kemp, Yusuf Sermet, Muhammed Sit, David McDermott, and Leon Grund)

For: ECE:5830 — Software Engineering Project (SEP)



Abstract

Dungeons and Dragons (D&D) has maintained a vibrant community of enthusiasts for over 45 years, but many aspects of the gameplay are hindered by dependence on traditional pen and paper usage. The goal of our DM Tools product is to enhance the experience of in-person D&D games by providing a centralized means for the Dungeon Master (DM) to generate, organize, and manage the complex interactions of details needed for gameplay.

The DM's role of directing and monitoring gameplay is an astronomical burden that's a critical impediment to successful gameplay. In conventional gameplay, DMs must generate and navigate through dozens to hundreds of pages of custom generated content while also narrating the story. Our product addresses a critical need for easing this burden by providing a centralized means of organizing and generating details for gameplay. Our tool's focus on improving DMs' performances complements other electronic applications focused on enhancing the players' experiences, such as Roll20, and providing DMs a database, like D&D Beyond does. To ensure the accuracy and quality of our product, we introduced cross-platform prototypes developed using Agile methodologies and Electron to stakeholders (local D&D players) every other week for acceptance testing and user story creation.

DMs using DM Tools can easily generate and track details of their campaign encounters, view a creature's information and conditions, and manage turn order in a single location. DM Tools is helping the D&D community leverage modern electronic information management strategies to enhance enjoyment of the game.

Product Vision

Project Title

DM Tools

Project Vision

Unlike with other Dungeons and Dragons (D&D) applications, the focus of DM Tools would be on augmenting and complimenting the personal experience of in-person D&D. To do so, we aren't going to be focusing on entirely digitalizing the game, as platforms like Roll20 do, or essentially providing Dungeon Masters (DMs) a database, like D&D Beyond does, but rather by providing a centralized means of organizing and generating details for gameplay. D&D as a concept has been around for 45 years now, but many aspects of the game play haven't fully moved on from its traditional pen and paper style of execution. DMs are still having to flip through dozens of pages or tabs while still narrating the events of the story in real time. The burden is astronomical – it's why very few players actually want to sign up to be the DM in a game campaign. It's time for D&D to fully enter the 21st century with the help of a digital toolset like DM Tools.

DMs, with our project, will now be able to easily generate and track details of their campaign encounters in one single location and be able to save and share their creations with others with ease. This project will specifically be focusing on alleviating the DM's burden, and thus will not focus on things like player characters or providing interactive maps to replace the in-person experience. Our main focus is rather on the most tedious part of being a DM, which is managing the combat aspect of these encounters. This translates to the features of quick-viewing a creature's information, tracking their condition, and managing turn order, as well as creating the combat encounter itself. These will be the focus of our application, which means that we will also need to provide a simple catalogue of creatures and items for the DM to use to populate such encounters.

Project Narrative

1. **Project Goals**

A successful application for us would involve an application which eases the burden of being a DM for encounter related. We know that this is a fairly lofty and non-descriptive goal, and thus, we hope to define it a bit better here. With our application, users should be able to easily compile and track details of their campaign encounters in one single location and be able to save and share their creations with others with ease through shareable files. 'Compiling' details of their campaign encounters means that DMs will able to add items from their catalogues to the 'encounter' for their campaign, similar to how we add files to a folder or its subfolders in our file systems. By 'tracking' details of their campaign encounters, we mean that DMs can update and control the inner details for each entry (or

file in our metaphor) to their liking. This either means updating the health or status of a creature (like renaming a file) or adding custom notes to the entries (like adding to a read.me in a repository). A quick view tooltip for this information would also be essential. We would also like for DMs to be able to keep track of whose turn it is in combat focused encounters (essentially the same as whose turn it is in a game like chess to move their piece, etc.). To make this application as useful as possible as quickly as possible, we would like to pre-populate our database with current D&D items, creatures, etc.

A tool like this will strengthen the D&D community by finally giving a tool to the DMs to make their job much easier. D&D Beyond often boasts how it can bring new into the D&D fold and support experienced players by making character creation and management a cinch, taking nearly less than a minute to get started. While we can't boast such a claim for DMs, we would like to offer a similar claim of allowing for new DMs to join the DMing experience in a more approachable way while also making more experienced DMs more comfortable with DMing more than a simple oneshot. In doing so, perhaps we will help expand the population of D&D players across the world with more DMs, all while increasing general DM satisfaction with the game.

2. **Priority**

The project is going to developed using Electron as a cross platform native application so that people can run the tool on any operating system. This way, it's compact and portable. By using an online-hosted backend, we're allowing for uses to simply execute the application with no installation required, which removes the technical knowledge requirement to run the application. For offline use, we provide two options. For technically advanced users, they can host their own database, allowing for full offline usage. For normal users, they will be allowed to download the information relevant to their own catalogues/encounters, allowing for a limited use of the application in offline mode. To ensure the project is designed for and will be useful to our community, we're working hand in hand with DMs in our local D&D community to ensure our user stories are valid and that our approach will truly help them better their DM experience. Our features have been given priority based on what the DMs has expressed the most need and want for with consideration to developer focused tasks as pre-requisites for these features.

3. **Sustained Benefits**

One of the benefits of working with a tabletop system like D&D 5E is that once the system is released, no further changed to the core ruleset will be made, meaning features will not need to be updated. It also will stick around and be relevant for a long time to come – many people still play the first and third editions of the game, for example. Even if another edition is released, the catalogue entries will still remain relevant and accurate, as they don't change between editions of the game without qualifying as a new entry altogether (A 5E version of a monster versus a 3.5E version of a monster, for example). The only things that change in these games over time will be new content being added via either official releases or by homebrew (user customized) entries. This feature is taken care of by the fact that we are adding officially designated content creators who will keep the application fresh by adding new content over time and keeping the system relevant to the current D&D scene. The only additional costs would go to developing new and better features for managing the

application experience, perhaps a between UI or new means of integrations with things like D&D Beyond.

4. **Leveraging**

Currently, there is no system comparable to ours. All of the tabletop softwares are entirely focused on the players rather than the DMs, which is why we think we would be able to interface well with these pre-existing applications. D&D Beyond is the leading, official platform for D&D cataloguing and organization, and it only allows people to view single character details at a time while still managing many story details yourself. Roll20, a popular online platform for virtual tabletops, is entirely manual in all aspects of the game, focusing on simply replacing the in-person experience with a virtual one rather than simplifying it. Players will still be able to (and encouraged to) use D&D Beyond while DMs will now have a service dedicated entirely to their organizational struggles, with both being able to use Roll20 for a virtual experience should they choose.

5. **Speed to Functionality**

The project will take approximately 2 sprints to be functional and have positive impacts on the stakeholders involved. Sprint 1 will mainly feature set up, while Sprint 2 will start to introduce the encounter system in terms of viewing, creating, and management. This, at least, will provide the DMs with a simplified collection of materials for their encounters with the test creatures/items/etc that we could provide.

User Manual

Intent

This page is intended to help provide a guide to users and, primarily, developers navigate and interpret the content found in our repository. It is also intended to provide the resources to locate more information should someone wish to look up more.

Wiki page naming protocol

Wiki pages shall start with a ##_ prefix.

The first digit represents the sprint number (i.e. like a chapter in a book): - 0x Global project documents - 1x Sprint 1 documents - 2x Sprint 2 documents - 3x Sprint 3 documents - 4x Sprint 4 documents - 5x Sprint 5 documents - 6x Sprint 6 documents

The second digit represents a subdocument for that sprint (i.e. like a subsection of a chapter).

External Tools

ZenHub

ZenHub Master Page

ZenHub is a free for public use service integrated directly into GitHub. By simply extending from GitHub, this service has up-to-date and accurate information to use in our burndown charts, velocity tracking, and release reports. It also helps us organize our tasks and project by being able to break down more complex tasks into epics and by establishing dependencies and roadblocks. Our scrum meetings are also heightened by being able to use the ZenHub lanes to track the status of our tasks.

CodeCov

CodeCov Reports Master Page

CodeCov is a free for public use hosted service that promotes proper code coverage throughout a project. It receives code coverage reports from each Travis CI build to the project and stores the data and visual representations of the data throughout time. It also will update the build to fail if less than our required coverage is met, ensuring that our testing requirements are met without developers having to run and update the coverage folders to check.

Travis CI

Travis CI Master Page

Travis CI is a free for public use hosted continuous integration service used to build and test projects hosted at GitHub. It automatically detects when pull requests are made in our

repository and then build and tests the project's frontend and backend on that branch. If the build or tests fail, it prohibits the merge from being made, which allows for developers to have instant feedback about the integrity of their pull request while protecting master from breaking code. On successes, it updates the code coverage on CodeCov automatically, allowing for, again, instant and easy tracking of the code coverage throughout the project.

Contacts and Roles

If you have inquiries or need assistance with the following parts of this project, be sure to contact the following people for the appropriate areas of your concerns. They will do their best to either assist with your inquiry or direct you to another person who can. (All members of the team will be responsible for being knowledgable in all of these areas, but the listed members will be the main point of contact, with the supports being the designated backup specialists.)

- Product Owner - Alic Szecsei
- Scrum Master - Heather Kemp
- Product Direction and Ownership, Acceptance Testing, Test User Involvement - Alic Szecsei (Product Owner)
- Team Management and Involvement - Heather Kemp
- Electron Application Deployment, Development, and Execution - Heather Kemp
- Database Design and Management - David McDermott
- Database Seeding - David McDermott and Leon Grund
- Docker Deployment and Management - Leon Grund and David McDermott
- GUI and Application Design - Yusuf Sermet (Support by Heather Kemp)
- Backend Design and Development - Muhammed Sit (Supported by David McDermott and Leon Grund)

Coding Style Guide

This section describes the coding style to be used for our project.

- No end of line white space
- Indentation is only done with tab characters
- Any public values (variables, functions, etc) should be in Pascal Case / upper camel case (i.e. FooBar)
- Any internal values (variables, functions, etc) should be in Dromedary case / lower camel case (i.e. fooBar)
- Any private or protected values (variables, functions, etc) should be in Dromedary case / lower camel case but preceded by an underscore (i.e. _fooBar)
- Names of values (variables, functions, etc) will not be just of single letters (or any other small denomination of letters or symbols). All names will be descriptive and meaningful.
- While naming values (variables, functions, etc), acronyms will be spelled out in all caps for readability.

- No more than 3 blank lines in a row.
- No more than 3 blank lines at the end of the file

Response Time Guide

For our response time guidelines, we will utilize the Usability Engineering guidelines. For feedback on validation on page, we will adhere to the limit of 0.1 second, “about the limit for having the user feel that the system is reacting instantaneously”. For submission of forms and page loading, we will adhere to the limit of 1.0 second, “about the limit for the user’s flow of thought to stay uninterrupted, even though the user will notice the delay”. This is to ensure the best user experience possible.

However, we acknowledge that currently our application is not the most efficient, as it’s currently been in development for one user. When used with multiple users at exportation, it may not meet these standards. That is, caching is not implemented to reduce the number of API/DB calls when obtaining frequently used things like monsters, so the API and DB may struggle to meet with demand. If we meet our MVP before the deadline of the final product, we will be looking to implement this feature.

Issue Creation

Issues will be written in the typical user story format, which is ‘As a [type of user], I want/ would like [some kind of goal], so that [some reason]. This is mandated by the use of a GitHub issue template, with the addition of an ’additional information’ category for either reference material or additional information that could not easily be captured into the user story itself.

Issues Labels

Issues will be used to implement an Agile Project Management system within GitHub:

Zenhub - Agile Project Management user guide

All issues need to have one label from the T[1-3]-* label set, and one label from the S[1-4]-* label set. All non-bug related issues should also be associated with an Epic, or greater feature set.

Label Name	Label Color	Label Description
T1-Defect	#ffd500	This label is for bugs found in the current development or production code
T2-Enhancement	#ffe4e9	This label is for features or add-ons that will improve the performance of the application, but are not a part of the core functionalities outlined by the tasks

T3-Task	#0000FF	This label is for a core functionality to implement
S1-Urgent	#FF0000	This label is for tasks that must be completed ASAP (within 24 hours) and require multiple team members' attention to complete and/or roadblocks the entire project
S2-High	#FAA937	This label is for tasks that are crucial roadblocks to the progression of the project
S3-Medium	#FFFCB5	This label is for tasks that may block features, but the project can still function without it being completed for a while
S4-Low	#00FF00	This label is for a task that either doesn't block tasks or that blocks tasks that are non-crucial and do not needed to be completed soon
C1-GUI	#aaaaaa	This label is to identify the task as primarily GUI related
C2-Testing	#bbbbbb	This label is to identify the task as primarily testing related
C3-Feature	#cccccc	This label is to identify the task are relating to a primary feature
C4-Administrivia	#dddddd	This label is to identify a task as non-coding related, but still required for the overall management of the project as a whole

Issue Completion

When a developer says they have finished their feature, they will attempt to make a pull request to the master branch and move the task to the Review / QA lane. The continuous integration tool will ensure that to make such a pull request, all unit tests pass and that C0 code coverage remains above 80% at the time of the merge request. Assuming that the merge request is allowed to occur, the team will be added as reviewers for the merge request. Developers will go through the review process and either give their feedback for updates and work with the developer to see these either fixed or the concerns addressed in conversation or approve of the merge request should it meet standards. Once at least one developer has approved of the merge request, the pull request will be accepted and merged

into master. The task will go into Done / Accepted, which will tell Alic to pull from the master branch and run through some thorough acceptance tests based on his requirements and understanding of the application/feature. He will give feedback as required (and ensure the requested changes are made, possibly with new issues in extreme cases) or accept the feature as being complete should it meet his standards. Upon receiving Alic's approval, he will close the issue.

Acceptance Testing Feedback Conflict Resolution

During acceptance testing, if, when Alic rolls out the application to other DMs, he finds other DMs have opinions about a feature's execution or deployment, he will enter a discussion with them to attempt to figure out which of their opinions is better for the overall product. If an agreement cannot be reached, Alic's say will be final in which feedback reaches the team, to avoid unnecessary external distractors.

Point Assignment and Team Capacity

Based on university standard, we're expected to complete 3 hours outside of class on the class (including the time in class itself), which leaves us with ((9 * 5) - 15) * 2 as a capacity, or, simplified, 60 story points at full capacity. However, due to many of us having external commitments, each sprint we will take into account whether or not we can reach that actual capacity reasonably and adjust our capacity accordingly.

We used this simplified and not quite accurate means of initial capacity generation to go along with the fact that our point estimates are loosely based off of an estimation of how long it will take for someone to complete the task in hours. This estimate of hours/points is based off of prior time taken to complete similar tasks and/or general difficulty of the task should it be an unfamiliar task area. The points are assigned as a team with individuals giving feedback on their estimates for time while keeping in mind who is attempting the task and their respective skill sets.

Procedure for Incomplete Tasks

At the end of the first week of a sprint, if our completed story points are not below half of our agreed upon capacity, the team will start, at every scrum meeting, addressing contingency plans for how they plan on finishing their tasks. If they believe they can't finish the tasks, they are encouraged to, at this point, inform the scrum master, so procedures can be enacted to contact the product owner and stakeholders and ensure that a reduced capacity request is feasible. If it is not, the team must agree upon a proper contingency plan based on the team's capacity, be it assigning an additional member to the task or having the original team member dedicate more time to the task.

If there are tasks still in the backlog in the last 3 days of the sprint, the emphasis on contingency plans is increased. Team members are encouraged to prioritize their work to remove possible roadblocks, and, if no roadblocks exist, prepare to dedicate the additional time and resources needed to complete the sprint in a timely manner.

If at the end of the sprint there is still unfinished tasks, the team will re-evaluate the tasks at the next sprint's planning meeting. If the task was too large, the team will break up this task into smaller pieces at this time and reassign the points and lanes as appropriate. If the task is still important and necessary for the next sprint, it will remain in the following sprint with an added emphasis on completing the task in this sprint. If the task is deemed as less important, at least relevant to the desired MVP for this sprint, it will be moved back to the icebox for consideration in future sprints.

ZenHub Lanes

In order to organize our stories and manage our progress, we use several different lanes within ZenHub.

Icebox

This lane contains all of the tasks for our project, as well as our epics. Tasks end up in this lane the second that they have a user story associated with them (non-team generated issues will end up in an unassigned lane first). A few tasks will have point values and labels associated with them if they are pulled back from the latter lanes, but minimal information will exist in this ice box. The team acknowledges that there are more features in this icebox than are possible to accomplish in this semester as we wanted to flesh out as many possible features as we could to provide a solid backbone to the project direction.

A task will move forward from the icebox once the team has decided that it is important to accomplish in the next immediate sprint. This means that a task will be assigned labels, an epic, an estimate, and a owner in order to move forward.

Backlog / To Do

This lane contains all tasks for the team's current sprints. These tasks were accepted by the team and approved by the product owner to be most important to accomplish next, but have not yet been started.

A task will move forward from the backlog lane once a team member has start working on that task.

In Progress / Doing

This lane contains all tasks that team members are currently working on. Team members should not have more than one task to their name in this category unless their are either co-listed for a task to do later work in the task or are roadblocked on their current task.

A task will move forward from the in progress lane once a developer believes the feature is completed.

Review / QA

This lane contains all tasks that are currently in review by other developers. If minor updates to the feature (variable names, styling, etc) are under discussion, then the task will remain here until the updates are made.

A task will move back to in progress/doing if major bugs or redesign needs exist in the feature, as discovered during review, so as to properly reflect this status.

A task will move forward to done / accepted if it passed by developer review and is merged in to master.

Done / Accepted

This lane contains all tasks that are currently done with development. They are awaiting to be reviewed by the product owner for accuracy and completeness.

A task will move back to in progress/doing if the product owner does not accept the task's completeness for a major error or missing part of the feature. A task will move back to review / QA if there is a small error in terms of text/appearance/etc that requires minor edits.

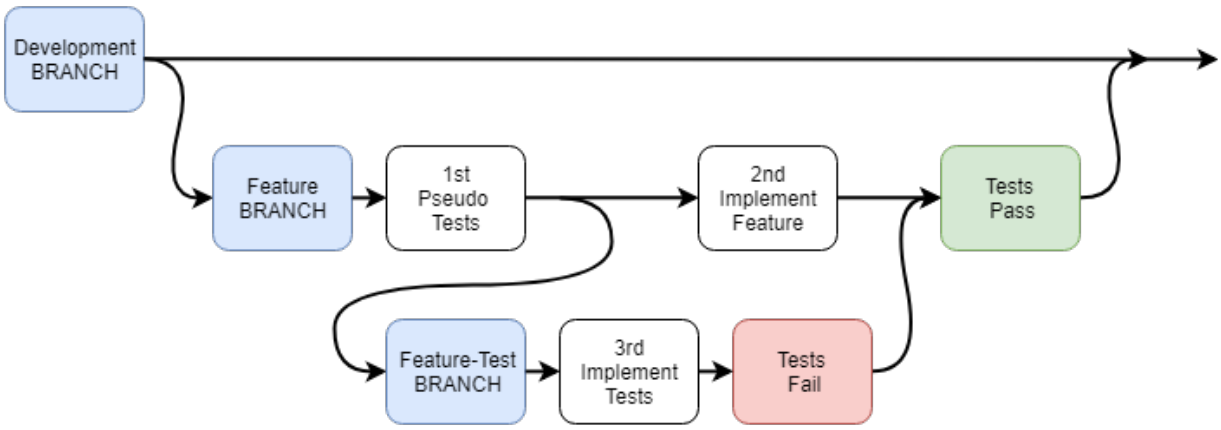
A task will move forward to closed if the product owner entirely accepts the task's completeness.

Closed

This lane contains all tasks that are completed and accepted by the product owner. Tasks cannot leave this lane. If it is revealed that a task is not complete fully due to an error, an issue should be made for that error rather than the corresponding issue being removed from this lane.

Test Plan

We will be using a "test driven development during feature development" protocol using red-green refactor testing principals. Essentially, before the start of the development of the feature, tests will be written out in pseudocode on a testing branch. Then, the feature will be developed onto the feature branch. Back on the testing branch, now that the developer understands the workings of the feature, they can implement the previously pseudocoded tests as well as add any additional tests that they discovered were needed. As these tests are on a separate branch from the feature, they will run red, per red-green refactor. Once they are merged into the feature branch, however, ideally they will turn green. If not, the developer will repeat the process until their code runs green on all test cases, at which point they will merge into the development branch. This process is pictured in the below diagram.



A diagram depicting the red-green refactor strategy described previously.

The following requirements will also be enforced for testing: - 80% code coverage at all times - 100% of written tests must pass at all times - 100% of features must pass acceptance testing by the product owner

A most in-depth overview of the test plan can be found in the official test plan document here.

Task Acceptance Plans / Parallelization

During sprint planning, the product owner identifies the tasks that are important to complete during the sprint and the next sprint. The developers then determine which tasks are pre-requisetes to these tasks. Finally, they work with the product owner to determine the capacity for the sprint based on all of these tasks combined with priority given to getting the desired features for this sprint done and getting as many preparatory tasks for the next sprint done as possible. Then, tasks are assigned importance based on the number of dependencies and due date. These tasks with high importance are assigned with multiple developers, ensuring they will be completed as soon as possible unblock the other tasks. Each team member is also assigned to at least one task they can work on independently of the others (or together with another developer in the case of collaborative tasks) to minimize overlap.

The team aims to front load much of our application set-up, such as the low-fi sketches and database schema, allowing for us to not be roadblocked by such things in future sprints.

Installation

The team will do their best to keep this section of the wiki up-to-date. If you still encounter issues, contact the team per the above roles and contact information.

Before You Get Started

To install a local copy of the repository and ensure you can get the latest updates with ease, be sure to clone the project if you want to run any of the code manually.

```
# copy repository using git clone
git clone https://github.com/UIOWA5830SP19/SPP500.git
cd SPP500
```

Getting Started (End users)

This project is created using Electron to creative native applications (exe's, dmg's, etc) for each of the different operating systems. These native applications will be included in the releases of the project. If you don't see your OS' exe listed, please contact our team, and we'll be happy to assist in adding a new native application for your OS.

Once the team finishes setting up the Azure backend, you will just have to run this native application from your local machine and we'll handle the rest. As of now, however, the team has not finished this task, and so if you wish to run the application in its entirety, please either contact a member of the team for a demonstration or follow the database and backend setup instructions prior to running the application.

Getting Started (Developers)

This project is created using a variety of frameworks and tools as prescribed in the package.json files. There are a few notable ones that we would like to highlight to ensure you are familiar with them before working with the codebase. We provide a native application interface using Electron and host a Hapi backend server using Node, Joi, Boom, and Typescript. Our database is creating using SQL and TypeORM while our testing is accomplished using Jest and Enzyme. We utilize a variety of tools for automation work, including Travis CI for continuous integration, codecov for enforcing code coverage requirements, yarn for package management, and ZenHub for issue tracking.

In future iterations of this project, we will be hosting the backend and database on Azure with assistance from Docker, so developers should be familiar with the Azure pipeline and Docker containers. Developers should not work solely with the Azure servers for developing, however, and should familiarize themselves with the database and backend setup instructions prior to running the application. Developers should also familiarize themselves with the frontend setup instructions, as they should always be running a new local dev build of the front end while working with the backend and database components. Developers should also familiarize themselves with the methods to export the frontend and the backend for new releases.

Front End Installation / Execution

```
# From the applications main directory (SPP500/)
# Navigate to the frontend folder
cd frontend

# Install frontend dependencies
yarn install

# run application in development mode
yarn run dev
```

Front End Exporting

```
# From the application's frontend directory (SPP500/frontend)

# Option 1: (Native OS only unless additional packages installed)
# essentially `yarn compile` & create build with electron-builder for
your native OS
yarn run dist

# Option 2: (Native OS only unless additional packages installed)
# essentially `yarn compile` & create unpacked build with electron-
builder for your native OS
yarn run dist:dir

# Option 3: (Any OS)
# Use Electron Packager https://github.com/electron-userland/electron-
packager

# Install Electron Packager for your CLI
npm install electron-packager -g

# Move back to the main directory (SPP500/)
cd ..

# Export for all OS that you have dependencies/permissions for
electron-packager frontend DMTools --all

# A few examples for single OS exports

# Export for just a Mac OS (dmg)
electron-packager frontend DMTools --platform=darwin

# Export for just a Windows 64 bit (exe)
electron-packager frontend DMTools --platform=win32 --arch=x64
```

To then run this application, either click or run the dmg/exe/appropriate output as you would with any other normal program for your exe. Follow the console text to see where the file(s) were exported.

Back End Installation / Execution

```
# From the applications main directory (SPP500/)

# Navigate to the backend directory
cd backend

# install the backend specific dependencies
yarn install

# run the backend
yarn run start
```

Back End Exporting (Docker)

First, Install Docker Desktop for Windows or Docker Desktop for Mac depending on your OS.

```
# Navigate into the backend folder from the main directory
cd backend

# build and run docker image (Note: this both runs and builds the
docker file sequentially. It will only build it once unless you
specify the build again in the below commands. This only matters to
developers. For users, they will only need to run it once, ideally)
docker-compose up

# (Optional/when changes are made to the local files)
docker-compose up --build
```

Please find additional useful Docker commands at https://devhints.io/docker.

If you run into issues with running the Docker backend, please take the following steps to attempt to fix the issue:

```
# Find all Docker images
docker images

# Delete your images one by one using the ids from the above command
for images
docker rmi <id#>

# Delete all images
docker rmi $(docker images -q)
```

```
# if this causes an issue due to running containers, stop any running
containers using their ids from the above command
docker stop <container_id>

# Remove that container
docker rm <container_id>

# Remove all containers
docker rm $(docker ps -a -q)

# Optional: Just remove all containers/images not in use at once
docker system prune -a

# Rebuild the image and run it
docker-compose up
```

Database Installation

First Install PostgreSQL locally there are many resources for doing this. This team used 10.6 if you wish to keep consistent with our versioning. https://www.postgresql.org/download/

On Linux you can connect to the PostgreSQL server with:

```
# Connect to the postgresql server
sudo -su postgres psql

# The terminal should now have 'postgres=# '

# Only run once, create a new database called devDB
CREATE DATABASE devDB;

# If you want to see all of the databases on your PostgreSQL server
run, q to quit
\l

# Then quit psql
\q

# Navigate into the backend folder from the main directory
cd backend

# Initialize and seed the database using
yarn run seed:run
```

Your database should be completely setup as well as sample data.

Server Deployment Instructions

The server is deployed within a docker image. The docker image is hosted inside a private repository on AWS. This service is called ECR. Before continuing you must install the AWS CLI to deploy new images to the repository. Follow this guide for the CLI: https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-install.html

The server is running on a Fargate cluster. The cluster has a service running which deploys one or more machines to manage requests. At this time the service is setup to create one machine. The ip address is dynamic at this point. We will need to move to a DNS service to get proper constant connections.

Once a push has been made to the docker image repository then the service can be updated. The update process takes about 5-10 minutes.

```
# Retrieve the login command to use to authenticate your Docker client
to the registry.

$ (aws ecr get-login --no-include-email --region us-east-2)

# Build your Docker image using the following command.

docker build -t dm-tools .

# After the build completes, tag your image so you can push the image
to this repository:

docker tag dm-tools:latest 779667361013.dkr.ecr.us-
east-2.amazonaws.com/dm-tools:latest

# Run the following command to push this image:

docker push 779667361013.dkr.ecr.us-east-2.amazonaws.com/dm-
tools:latest
```

Useful background materials

- https://learnui.design/tools/data-color-picker.html#divergent
- https://help.github.com/articles/manually-creating-a-single-issue-template-for-your-repository/
- https://github.com/stevemao/github-issue-templates
- https://help.github.com/categories/managing-your-work-on-github/
- https://help.github.com/articles/closing-issues-using-keywords/

Meeting Minutes

To ensure that every member of our team was aware of the status of every other member of the team, we enforced a policy of 4 standup meetings a week at times that worked best for every team member. These times were Sunday, Monday, and Wednesday at 9:00pm and Thursday at 9:45pm. Each standup was limited to 10 minutes while each planning meeting kickoff was limited to one hours. Notes were taken at each of these meetings to provide a brief snapshot of what each team member did throughout the course of our six sprints, each of two weeks in length. For your reference, the sprints were broken up accordingly:

Sprint 0 (Project Planning):
- Start: 1/16/2019
- End: 1/31/2019

Sprint 1:
- Start: 1/31/2019
- End: 2/17/2019

Sprint 2:
- Start: 2/18/2019
- End: 3/03/2019

Sprint 3:
- Start: 3/04/2019
- End: 3/14/2019

[Spring Break]

Sprint 4:
- Start: 3/25/2019
- End: 4/07/2019

Sprint 5:
- Start: 4/07/2019
- End: 4/18/2019

Sprint 6:
- Start: 4/22/2019
- End: 5/02/2019

[End of Semester]

The scrum meetings following April 30th (4/30/2019) are not included due to the timeframe of Modern Marvels and preparing this binder for presentation purposes.

2019-01-16 In Person

Participation by: Heather, Yusuf, Muhammed, and Leon (Absent: David, he was at a conference)

- The group met up for the first time after class. We all joined a Discord group so we could keep in contact as we started brainstorming our ideas for projects and frameworks.

2019-01-18 In Person

Participation by: Heather, Yusuf, Muhammed, Leon, and David

- We decided upon the D&D project during class to sign up for a pitch time. We're considering Electron and a .NET backend, but we're still keeping our options open.

2019-01-22 In Person

Participation by: Heather, Yusuf, Muhammed, Leon, David

- The team met with Alic to review the project vision. We plan for meeting on Saturday for a D&D oneshot, to solidify everyone's understanding for the project.
- After being unable to support our .NET choice to the professor, we changed to use Node/typescript after seeing they were more of a benefit to our project. It was also supported by our PO.

2019-01-24 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, David

- We met as a team to finalize the product vision, which we had been working on independently until now.

2019-01-25 In Person

Participation by: Heather, Yusuf, Muhammed, Leon, David

- We met to present the product vision to the professor. Afterwards, we reviewed his feedback to make action items on how to proceed with the vision and overall project.

2019-01-28 In Person

Participation by: Heather, Yusuf, Muhammed, Leon, David

- We presented our project vision. It went fairly well, but the professor did bring one of our known fears to question, which is possibly making an application that actually hinders DMs. We re-expressed this concern to Alic and he is going to take extra care to avoid this pitfall as much as possible by getting feedback from multiple DMs.

2019-01-27 In Person

Participation by: Heather, Yusuf, Muhammed, Leon, David

- The team met to do the D&D oneshot. Everyone seems to understand the system, requirements, and purpose of the application now.

2019-01-31 Discord Voice (Due to cold weather)

Participation by: Heather, Yusuf, Muhammed, Leon, David

- Scrum planning meeting with Alic. We planned out all of the issues/stories for the semester independently, and now worked with him to determine what was most important and when to approach it.

2019-02-05 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, David

- Agenda (prepared by Heather)
- Work Completed
  - Heather: TSLint file configuration (pull request made), permissions gotten for CI build, started test plan documentation updates
  - Yusuf: Started analyzing page requirements for the application to create a base for the lo-fi sketches.
  - Muhammed: Mock db connection to feed the API call for the registration
  - Leon: workstation setup
  - David: Basic schema details, talked with Product Owner to get more details on data storage
- To Do by Next Scrum Meeting
  - Heather will: Start configuring the CI build/settings and work on the test plan
  - Yusuf will: Create hand-drawn lo-fi sketches for the main pages for the application. Actually implement the front-end of the register/login pages. Implement the front end for the catalog monsters.
  - Muhammed will: Have a basic registration API call/route finished

    - Leon will: create base node project for backend; restructure git repo
    - David will: Work on the rest of the schema specifically equipment relations
- Team Tasks
    - Test Plan Doc by Friday
        - Let's do it by tomorrow night?
            - Yusuf will do the Unit Testing section
            - Leon will do the Regression Testing section
            - Muhammed will do the Integration Testing section
            - David will do the Control Features section
            - Heather will do the Schedules section
    - Test Plan Presentation by Friday
        - Ideally the doc is done by tomorrow so we can start working on it in the over-morrow.
- When is our next meeting?
    - Wednesday, 9pm
    - Discord Voice

2019-02-06 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, David

- Agenda (prepared by Heather)
- Work Completed
    - Heather: completed sections on the test plan document, started looking into CI options
    - Yusuf: completed assigned sections on the test plan document
    - Muhammed: completed assigned sections on the test plan document
    - Leon: setup node project for back end; dockerized back end
    - David: Worked on equipment separation into weapons, items, and packs
- To Do
    - Heather will: Finish configuring the CI options and attempt to integrate it into the project; look into the powerpoint set up
    - Yusuf will: Continue to work on the sprint goals as detailed in yesterday's scrum meeting.
    - Muhammed will: research about jwt on node.js.
    - Leon will: review merge request; dockerize latest features
    - David will: create relations between the equipment classes.
- Team Tasks
    - Test Plan Doc by Friday
    - Test Plan Presentation by Friday
    - Open Pull Request(s)

- When is our next meeting?
  - Thursday, 9:30pm
  - Discord Voice

2019-02-07 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, David

- Agenda (prepared by Heather)
- Work Completed
  - Heather: Basic CI set up for running the program without errors
  - Yusuf: Completed the hand-drawn lo-fi sketches for landing, login, and register interactive page.
  - Muhammed: Did research on jwt and nodejs
  - Leon: still resolving merge conflict
  - David: Created table for spells, linked up weapon properties, and adventuring gear
- To Do
  - Heather will: Finish setting up the CI
  - Yusuf will: Draw the lo-fi design for the whole application considering the requirements of the minimum viable product
  - Muhammed will: Try to come up with a minimum viable registration api call
  - Leon will: started on next story
  - David will: write sql file to migrate the schema into a postgresql database
- Team Tasks
  - Test Plan Doc by Monday
  - Test Plan Presentation by Monday
  - Open Pull Request(s)
- When is our next meeting?
  - Sunday, 9:00pm
  - Discord Voice

2019-02-10 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, David

- Agenda (prepared by Heather)
- Work Completed
  - Heather: Reviewed pull request for Docker, updated wiki pages
  - Yusuf: Reviewed pull request for CI, hand-draw lo-fi sketch for the application
  - Muhammed: basic registration function
  - Leon: dockerized backend; reviewed merge request

  - David: create SQL script to load equipment schema
- To Do
  - Heather will: update documentation for exporting electron app and opening it, look into finishing the wiki updates
  - Yusuf will: Finalize and update the sketches to GitHub. Will work with Muhammed to implement frontend of the register/login functionality
  - Muhammed will: database related stuff for authentication
  - Leon will: start on next issue #102
  - David will: getting database server setup and more of the schema done.
- Team Tasks
  - Presentation tomorrow in class, update PPT with requirements
  - Open Pull Request(s)
    - All handled!
- When is our next meeting?
  - Monday 9:00pm
  - Discord Voice

2019-02-11 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, David

- Agenda (prepared by Heather)
- Work Completed
  - Heather: Finished updating the Wiki pages as of today
  - Yusuf: Researching electron and react to start the landing page
  - Muhammed: jwt research for logging in tokens etc.
  - Leon: started looking at how to seed database in node
  - David: Updated some of the data types, looked into constraints
- To Do by Next Scrum Meeting
  - Heather will: look into getting tasks approved in a timely manner and start looking at the creating a catalog monster task
  - Yusuf will: Continue with react components for the login/register
  - Muhammed will: implement login, commit register to GitHub
  - Leon will: database setup with David and seed with some sample data
  - David will: Database server setup and monster schema.
- Team Tasks?
  - Open Pull Request(s)
    - None at the moment
- Next Meeting:
  - Wednesday 9pm
  - Discord Voice

2019-02-13 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, David

- Agenda (prepared by Heather)
- Work Completed
  - Heather: initial work done to gather requirements for monster registration, checked in with Alic due to some discrepancies in information
  - Yusuf: Studying installed and init'd React. Actively studying, doing tutorials for React, planning an approach for our app
  - Muhammed: Setup of local PgSQL and TypeORM
  - Leon: started working on sample data set up with David
  - David: SQL schema script coming along well
- To Do by Next Scrum Meeting
  - Heather will: finish working with Alic to gather requirements for the monster registration information and get
  - Yusuf will: work with Muhammed to finish the front end for the landing page/ registration/login
  - Muhammed will: work on finishing registration by tomorrow
  - Leon will: work with David to get script up to convert JSON to SQL for easier data entry
  - David will: Developer SQL server setup, Instructions for running schema script
- Team Tasks?
  - Burndown Rate Check/Overview
  - Open Pull Request(s)
- Next Meeting:
  - Thursday 9:45pm
  - Discord Voice

2019-02-14 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, David

- Agenda (prepared by Heather)
- Work Completed
  - Heather: Worked with Alic to finalize the monster entry information for the first sprint and possible bells and whistles for the next sprint. Basic Electron/React form set up on branch and starting to work with the validation.
  - Yusuf: Initialized the front end, created landing, registration, and login pages and react components. Worked through bugs with the CSS parsing for the Enzyme testing and the lowfi sketches
  - Muhammed: Registration function in JS, tests without mocking the db connection

  - Leon: parse JSON data for spells, skills, conditions, and ability scores
  - David: Finished monster schema, researched TypeORM, decided on Active Record.
- To Do by Next Scrum Meeting
  - Heather will: Finish creating the validations and tests for the basic form of monster creation and try to help Muhammed with the TS-Jest bug
  - Yusuf will: Make pull request with the lowfi sketches and work for the snapshot testing
  - Muhammed will: Registration function in TS, correct testing etc.
  - Leon will: cont. setup backend and create more sample data
  - David will: Setup some entities using TypeORM, and PostgreSQL server setup
- Team Tasks?
  - Burndown Rate Check/Overview
  - Open Pull Request(s)
    - Login Front End
      - Reviewed, awaiting updates
    - Registration Back end
      - Reviewed, awaiting updates
- Next Meeting:
  - Sunday 9:00pm
  - Discord Voice

2019-02-17 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, David

- Agenda (prepared by Heather)
- Work Completed
  - Heather: Updated wiki pages, gone over a few pull requests, created dynamic monster field components, drafted retrospective presentation, fixed CI build issues
  - Yusuf: Initialized the testing environment (jest, enzyme, etc.) for the frontend. Wrote snapshot tests. Created Catalogue viewing page as well as the Monsters Gallery.
  - Muhammed: Fixed mock db connection on registration tests, login implementation with jwt
  - Leon: Reviewed David's pull request, worked with him on typeorm models, and added postgres docker image (can't connect to it yet)
  - David: Setup entities with typeORM, reviewed pull requests.
- To Do by Next Scrum Meeting
  - Heather will: work to finish the retrospective presentation, prepare for sprint 2 kickoff meeting, finalize the basic form fields

  - Yusuf will: study for the presentation. Nothing major.
  - Muhammed will: Start with monster catalog backend.
  - Leon will: Take screenshots of parsed data for presentation
  - David will: Make pictures of the schema for the semi-demo of our application.
- Team Tasks?
  - Burndown Rate Check/Overview
  - Open Pull Request(s)
    - Login Backend
      - Heather, Muhammed, David working on it
- Next Meeting:
  - Today, 10:00pm
  - Discord Voice

2019-02-17 Discord Voice (Pt.2, Retrospective)

Participation by: Heather, Yusuf, Muhammed, Leon, David

- Agenda (prepared by Heather)
- Retrospective
  - What went wrong?
  - What went right?
  - What will we do to fix these things?
  - Comprehensive List in PowerPoint
- Team Tasks
  - Practice presentation
  - Upload presentation *tonight* (must be up by 10am tomorrow)
- Next meeting
  - Monday 7:00pm
  - Main Library Room (See Invite)
  - Sprint Kickoff

2019-03-04 Discord Voice Sprint 3 Kickoff

Participation by: Heather, Muhammed, Leon, David (Absent: Yusuf, gone to his institution's yearly presentations to the senators in Des Moines)

- Agenda (prepared by Heather)
- Capacity
  - Reminder: Normal developer capacity is ~12 for sprints

  - Team Capacity: 52
    - Yusuf: 12 / 12
      - 100% capacity
    - Muhammed: 12 / 12
      - 100% capacity
    - Leon: 10 / 12
      - 85% capacity due to traveling to DC over the weekend on the first week
    - David: 12 / 12
      - 100% capacity
    - Heather: 6 / 12
      - 75% capacity due to scrum master duties & remaining research on dependencies, 75% of that capacity due to being gone to California for a conference starting early Wednesday morning)
      - Accepting 6.5 story points to help with catching up on tasks while gone / hitting MVP in timely manner
  - See review/Q&A tasks for feedback from Alic on the features' acceptance
- New Focus Items?
  - Tasks from the epic for encounter creation
  - Cookies and Session Token Security
  - Skill / Monster DB updates
  - Extract API URL as a variable *somewhere*
  - Optimize Docker image/building
- Next meeting
  - Wednesday 9:00pm
  - Discord Voice

2019-03-06 Discord Voice

Participation by: Heather, Muhammed, Leon, David, Yusuf

- Agenda (prepared by Heather)
- Work Completed
  - Heather: added the milestone and labels for all of the tasks and implemented the docker optimization w/ pull request & reviewed one of David's pull request for dev tools, reran David's travis build issue/fix
  - Yusuf: Reviewed David's pull request on react chrome dev tools and reviewed the issues that has been created for sprint 3 since i was absent from the meeting on Monday due to work.
  - Muhammed: Started with editing encounters.
  - Leon: reviewed pull request David-Monster-Model and user story for scraping ability scores

  - David: Actions are now part of the database, Seeding the database with interfaces is easier, Reviewed a pull request, added Environment to monsters, fixed damage to hitpointdistribution, removed redundant test structure, added react dev-tools to the frontend, made major progress on Bulma Bloomer Refactor. Remade the Navbar, Login header, registration.
- To Do by Next Scrum Meeting
  - Heather will: finish setting up the wiki for the upcoming sprint (milestone, release 2 files, Bulma update / explanation, DevTools update/explanation)
  - Yusuf will: Will work on the dynamic header bar, waiting for the completion of the task "Bulma UI Refactor"
  - Muhammed will: Finish editing encounters backend.
  - Leon will: starting next user story for parsing ability scores
  - David will: Finish the Bulma refactor and push to GitHub.
- Team Tasks?:
  - Open Pull Requests?
    - David minor model fixes
    - Docker Optimization
- When is our next meeting?
  - Thursday at 9:45pm
  - Discord Voice

2019-03-07 Discord Voice

Participation by: Heather, Muhammed, Leon, David, Yusuf

- Agenda (prepared by Heather)
- Work Completed
  - Heather: research extracting api key and starting to try different methods, updated release files and milestones (limited (work done due to being sick, sorry)
  - Yusuf: Waiting for dependent tasks. Haven't done something major since yesterday.
  - Muhammed: nothing major yet, still working on encounter edit backend
  - Leon: creates migration to parse ability scores, almost done
  - David: Finished making changes to the refactor portion of Bulma, Bloomer, ready to update the next components. Made a pull request for the refactor. Updated the testing for many different components. Attempted to get form submission tested but the async nature of those request was making it impossible.
- To Do by Next Scrum Meeting
  - Heather will: finish updating the wiki and finish the lowfi sketches and cookie/session management. Help fix bugs and pull requests as they arise
  - Yusuf will: work on dynamic header bar

  - Muhammed will: work to finish encounter edit backend
  - Leon will: finish parse ability scores issue and start with scraping skills
  - David will: work on refactoring monster component to use Bulma and Bloomer
- Team Tasks?:
  - Open Pull Requests?
    - N/A
- When is our next meeting?
  - Sunday at 9pm

2019-03-10 Discord Voice

Participation by: Heather, Muhammed, Leon, David, Yusuf

- Agenda (prepared by Heather)
- Work Completed
  - Heather: Add modal to monster creation when error, document common error points for psql things, worked with david on resolving hotfixes for db seeding and jwt token update, sprint 2-3 wiki low-fi sketches for encounter and monsters, fix/investigate ideal burndown rate being over inflated during week 1, add all links for meetings
  - Yusuf: reviewed pull request, was waiting on Bulma refactor to start working on header bar which is now completed. Currently working on the header bar.
  - Muhammed: Reviewed two minor PRs. Implemented encounter editing, deleting. Monster actions update to the creation backend.
  - Leon: still working on skills and parsing them right now, just got back from traveling
  - David: worked on monster creation backend with Joi, reviewed pull requests,
- To Do by Next Scrum Meeting
  - Heather will: monitoring pull requests as they come in the next few updates. Updating the Wiki as scrum meetings happen
  - Yusuf will: Work on the header bar
  - Muhammed will: Campaign entity creation
  - Leon will: skills and parsing them
  - David will: redoing Joi implementation especially for the front end
- Team Tasks?:
  - Open Pull Requests?
    - Refactor Monster Backend
      - Needs refactor from latest merges which altered tests, so we can't do anything with this now.
- When is our next meeting?
  - Monday at 9pm

2019-03-11 Discord Voice

Participation by: Heather, Muhammed, Leon, David, Yusuf

- Agenda (prepared by Heather)
- Work Completed
  - Heather: reviewed pull requests and investigated docker errors on local machine (tasks are done, so just on review duty)
  - Yusuf: Was busy with legal stuff. Didn't do a lot course-related
  - Muhammed: started with campaign entity creation
  - Leon: finishing up scraping skills
  - David: refactoring monster creation, finishing refactor of the actions on the backend w/ Joi.
- To Do by Next Scrum Meeting
  - Heather will: Reviewing any new pull requests and mainly preparing for leaving in ~2 days
  - Yusuf will: Continue to dynamic header bar
  - Muhammed will: finish campaign entity creation
  - Leon will: finish skills and seeding DB
  - David will: finish modularizing monster creation
- Team Tasks?:
  - Open Pull Requests?
    - N/A
- When is our next meeting?
  - Wednesday at 9pm

2019-03-13 Discord Voice

Participation by: Leon, Yusuf, Muhammed, David (Heather is gone to GDC [Academic Absence])

- Agenda (prepared by Leon)
- Work Completed
  - Heather: excused absence; traveling
  - Yusuf: Created and completed the frontend and integration for encounter creation
  - Muhammed: Campaign entity creation
  - Leon: seeded db with skills, having trouble with typeorm relations skills and ability scores
  - David: Add actions to the Joi schema, Refactor monster creation to include the new action creation. Change skills back to a dictionary so it is easier to send and

create on the client. Worked on the monster creation, It's in its final state, just need to make changes to the style. The code is passing all of the tests.

- To Do by Next Scrum Meeting
  - Heather will: be excused for next scrum; traveling
  - Yusuf will: Write the tests for encounter creation
  - Muhammed will: Basic campaign creation
  - Leon will: fix relations for skills and ability scores, start working on updating scraping monster issue
  - David will: Finish styling monster creation.
- Team Tasks?:
  - Open Pull Requests?
    - create encounter frontend and integration
- When is our next meeting?
  - Thursday 9:45pm

2019-03-14 Discord Voice

Participation by: Leon, Yusuf, Muhammed, David (Heather is gone to GDC [Academic Absence])

- Agenda (prepared by Leon)
- Work Completed
  - Heather: traveling
  - Yusuf: Completed the testing for Encounter Frontend and Integration. It is now completed and merged to master.
  - Muhammed: started implementing campaign creation.
  - Leon Still fixing skills relations.
  - David: Reviewed encounter creation frontend and integration
- To Do by Next Scrum Meeting
  - Heather will: be back for next scrum.
  - Yusuf will: Create an dependent issue for viewing Monster Objects as part of the View Catalog Monster vision and pull from the backlog. We decided to assign 3 points to that task. Hopefully I'll complete that by next scrum meeting.
  - Muhammed will: finish campaign creation and editing backend calls
  - Leon will: review pull requests and help get MVP ready for deployment.
  - David will: Finish the style of monster creation.
- Team Tasks?:
  - Open Pull Requests?
    - N/A
- When is our next meeting?
  - Sunday 9pm

2019-03-17 Discord Voice

Participation by: Leon, Yusuf, Muhammed, David (Heather is gone to GDC [Academic Absence])

- Work Completed
  - Heather: Was at a conference. Already completed her tasks.
  - Yusuf: Completed the functionality as well as testing for Monster View page as detailed in issue #211. However, most of those changes were affected by another commit for issue #186 that merged to master on Friday. Since that commit changed the architecture and data types considerably, the whole work for Monster View needed to be refactored. That's also done, and pull request is created.
  - Muhammed: Already completed his tasks.
  - Leon: reviewed last pull request for mvp.
  - David: Already completed his tasks.
- To Do by Next Scrum Meeting
  - N/A, Spring Break
- Team Tasks?:
  - N/A, enjoy sprint break
- When is our next meeting?
  - After Spring Break

2019-03-25 Discord Voice (Sprint Kickoff)

Participation by: Heather, Leon, Yusuf, David (Muhammed is preparing for an exam)

- Agenda (prepared by Heather)
- Capacity
  - Reminder: Normal developer capacity is ~12 for sprints
  - Team Capacity: 54
    - Yusuf: 10 / 12
      - 85% capacity due to preparing for PhD exam. Do your best Yusuf! 👍
    - Muhammed: 12 / 12
      - 100% capacity
      - Accepting 13 out of 12 due to necessity for the team being able to catch up to our CRUD MVP goals
    - Leon: 8 / 12
      - 67% capacity due to preparing for switching to frontend
    - David: 18 / 12

        - 150% capacity due to working through Spring break on refactoring efforts.
        - Accepting 19 out of 18 due to prep work being done over break and to help cover the gaps in team capacity.
    - Heather: 6 / 12
        - 75% capacity due to scrum master duties, 75% of that capacity due to organizing an hackathon from March 29th through March 31st
  - See review/Q&A tasks for feedback from Alic on the features' acceptance
- New Focus Items?
  - Finish refactoring monster creation
  - Basic Campaign CRUD
  - Basic Encounter CRUD
  - Basic Monster CRUD
  - Midpoint Review Presentation
- Next meeting
  - Wednesday 9:00pm
  - Discord Voice

2019-03-27 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, David

- Agenda (prepared by Heather)
- Work Completed
  - Heather: all story tickets, latest release, new milestone
  - Yusuf: briefly reviewed some pull requests (i.e. partial review)
  - Muhammed: started paginated monster list backend
  - Leon: Read some react pages and encounters on the frontend.
  - David: Worked on pull request review
- To Do by Next Scrum Meeting
  - Heather will: get outline of presentation done
  - Yusuf will: start working on the tasks
  - Muhammed will: finish paginated monster list backend
  - Leon will: look into pull request, more react state management
  - David will: work on finishing the breakup of components for monster creation.
- Team Tasks?:
  - Open Pull Requests?
    - A few from David's Refactors
- When is our next meeting?
  - Thursday at 9:45pm

2019-03-28 Discord Voice

Participation by: Heather, Yusuf, Leon, Muhammed, David

- Agenda (prepared by Heather)
- Work Completed
  - Heather: completed the presentation & reviewed pull requests
  - Yusuf: view encounters page & tests w/ mocks and reviewing pull requests
  - Muhammed: paginated monster list backend stuff, waiting on review, working on paginated encounter list
  - Leon: hasn't been been able to do much since last scrum meeting
  - David: made sure pull requests went through smoothly, breaking up the components of the monster creation & bug fixes, fixed the health point error we found
- To Do by Next Scrum Meeting
  - Heather will: complete the low-fi sketches
  - Yusuf will: integrate backend to frontend of the encounters db
  - Muhammed will: finish paginated encounter list & then hopefully start other task
  - Leon will: look into react state management and start with the campaign issue
  - David will: finish breaking out components of monster creation and put them into view monster & edit monster and then once that's done, maybe start working on the backend of the monster edit
- Team Tasks?:
  - Open Pull Requests?
    - David monster breakup
    - Malisit monster list
  - Prepare for presentation tomorrow at the earliest
- When is our next meeting?
  - Sunday at 9pm

2019-03-31 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, David

- Agenda (Prepared by Heather)
- Work Completed
  - Heather: remade presentations and completed lowfi sketches
  - Yusuf: Reviewed some PRs. Implemented a new mechanism and frontend to paginate monster catalogue. Integrated the backend to actually retrieve the monsters from the database instead of the dummies.
  - Muhammed: Implemented paginated encounter reading. Started implementing read campaign backend.

  - Leon: Reviewed Monster Breakup pull request and started create campaign frontend.
  - David: Finished breaking the monster into individual components
- To Do by Next Scrum Meeting
  - Heather will: review powerpoint and pull requests if they're not done. Otherwise, start looking at last task
  - Yusuf will: Continue PR reviews.
  - Muhammed will: finish read campaign backend.
  - Leon will: Finish create campaign frontend.
  - David will: Going to work on combining these components to update view monster and edit monster.
- Team Tasks?:
  - Open Pull Requests?
    - Yusuf monster view integration and pagination
    - David monster breakup
- When is our next meeting?
  - Monday 9pm

2019-04-01 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, David

- Agenda (Prepared by Heather)
- Work Completed
  - Heather: Finished the implementation of my task, need to update the tests
  - Yusuf: We all have done the presentation as a team. Didn't do a lot beside that.
  - Muhammed: working on to read campaign
  - Leon: working on create campaign frontend.
  - David: Worked on pull requests and getting monster create, edit, delete working
- To Do by Next Scrum Meeting
  - Heather will: Do my best to update the tests and get them working
  - Yusuf will: work on integrating the encounter frontend and backend
  - Muhammed will: finish read campaign backend
  - Leon will: finish create campaign frontend.
  - David will: Big pull request coming soon.
- Team Tasks?:
  - Open Pull Requests?
    - N/A
- When is our next meeting?
  - Wednesday 9pm

2019-04-03 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, David

- Agenda (Prepared by Heather)
- Work Completed
  - Heather: Tried to get nock stuff working. Basic testing for feature done and pull request made for what I could. Also reviewed Yusuf's feature.
  - Yusuf: Completed the encounter front-backend integration with tests. Created a new component for pagination to enable easy reuse.
  - Muhammed: Still working on campaign reading backend
  - Leon: working on campaign creation frontend
  - David: Getting nock to work when making a request from the constructor (now from "componentDidMount") Making a lot of progress on testing monster creation / edit / delete
- To Do by Next Scrum Meeting
  - Heather will: review pull requests as they come up/if they come up and look into monitoring feature pull request review
  - Yusuf will: Do the delete encounter
  - Muhammed will: finish read campaign backend
  - Leon will: finish create campaign frontend
  - David will: make pull request with his features
- Team Tasks?:
  - Open Pull Requests?
    - Encounter Refactor with Monsters from DB
- When is our next meeting?
  - Thursday 9:45pm

2019-04-04 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, David

- Agenda (Prepared by Heather)
- Work Completed
  - Heather: updated wiki and ZenHub statuses and monitored pull requests
  - Yusuf: Delete encounter is completed and currently in review. Reviewed David's pull request.
  - Muhammed: Implemented read campaign backend
  - Leon: Finished create and edit campaign for frontend.
  - David: Finished the Monster Create, Edit, and Delete.
- To Do by Next Scrum Meeting
  - Heather will: help review any last pull requests and prepare for planning meeting

  - Yusuf will: help review PRs
  - Muhammed will: Implement read campaign frontend, delete campaign backend/frontend
  - Leon will: Fix tests for create and edit campaign frontend.
  - David will: Working on tests for the backend of get one, edit and delete monster.
- Team Tasks?:
  - Open Pull Requests?
    - Read Campaign Back End
    - View Edit Create Monster
    - Delete Encounter Integration and testing
- When is our next meeting?
  - Sunday 9:00pm retrospective

2019-04-07 Discord Voice (Retrospective)

Participation by: Heather, Yusuf, Muhammed, Leon, and David

- Agenda (Prepared by Heather)
- Work Completed
  - Heather: organized general tasks for planning and met with Alic to talk about the product
  - Yusuf: Reviewed some PRs
  - Muhammed: Read and delete campaign frontends in collaboration with Yusuf.
  - Leon: Create campaigns done, editing campaigns logic is but not supported by backend.
  - David: Finished View-Edit-Delete Monsters and got it pulled into master. Reviewed pull requests. Looked into IPC Main in electron, https://electronjs.org/docs/api/ipc-main Looked into storing JSON in electron for performance. https://www.youtube.com/watch?v=AJw4KpNdWPM
- To Do by Next Scrum Meeting
  - Heather will: meet with Alic about product vision
  - Yusuf will: not do a whole lot until tomorrow since the sprint is just starting (not assigned tasks yet)
  - Muhammed will: Wait for new tasks?
  - Leon will: start rework on campaign backend routes and validations
  - David will: Work on running encounters for the frontend.
- Team Tasks?:
  - Open Pull Requests?
    - N/A
- When is our next meeting?
  - Now, sprint planning

2019-04-07 Discord Voice (Kickoff)

Participation by: Heather, Yusuf, Muhammed, Leon, and David

- Agenda (prepared by Heather)
- Capacity
  - Reminder: Normal developer capacity is ~12 for sprints
  - Team Capacity: 53
    - Yusuf: 10 / 12
      - 85% capacity due to preparing for PhD exam Part 2. Do your best Yusuf! 👍
    - Muhammed: 12 / 12
      - 100% capacity
    - Leon: 13 / 12
      - 100% capacity
      - Accepting extra 1 point to help catch up from last week's mishap
    - David: 11 / 12
      - 92% capacity due to taking a break after last sprint, taxes, and machine learning project due this week that will take attention
    - Heather: 7 / 12
      - 75% capacity due to scrum master duties, 75% of that capacity due to running a conference on April 13th and being gone throughout the week doing prep work for that.
  - See review/Q&A tasks for feedback from Alic on the features' acceptance
- New Focus Items?
  - Start player characters
  - Get a strong backbone for starting to run through an encounter
  - Optimize CRUD of encounter and campaign
  - Update testing with nock fixes
- Next meeting
  - Wednesday 9:00pm
  - Discord Voice

2019-04-10 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, and David

- Agenda (prepared by Heather)
- Work Completed
  - Heather: made all the tasks, updated wiki with new milestone, met with Alic to discuss some product vision things for player character
  - Yusuf: Started making a work plan for this sprint's tasks.

  - Muhammed: Started working on Get Campaign by ID (Backend)
  - Leon: refactoring backend for Campaign.
  - David: Finished a few components
- To Do by Next Scrum Meeting
  - Heather will: finish the release and and meet with Alic about encounter playing
  - Yusuf will: Look at an efficient way to perform refactoring to Encounter CRUD.
  - Muhammed will: finish Get Campaign by ID (Backend)
  - Leon will: finish setting up routes and Joi validations.
  - David will: Finish the rest of the components for monster.
- Team Tasks?:
  - Open Pull Requests?
    - N/A
- When is our next meeting?
  - Thursday 9:45pm

2019-04-11 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, and David

- Agenda (prepared by Heather)
- Work Completed
  - Heather: created lowfi sketches for sprint 5 features, met with alic to discuss goals of sprint 5 features. Made the release for sprint 4.
  - Yusuf: Is continuing to CRUD encounter refactoring
  - Muhammed: still working on Get Campaign by ID (Backend)
  - Leon: finished rework on routes and validations
  - David: Finished the more of the monster components
- To Do by Next Scrum Meeting
  - Heather will: Look into the helper text registration task work.
  - Yusuf will: Continue to CRUD encounter refactoring
  - Muhammed will: finish Get Campaign by ID (Backend)
  - Leon will: create migration for backend changes
  - David will: Make a pull request for the monster component changes. Look at the lo fi for running an encounter
- Team Tasks?:
  - Open Pull Requests?
    - Sprint 5 LoFi
- When is our next meeting?
  - Sunday 9:00pm

2019-04-14 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, and David

- Agenda (prepared by Heather)
- Work Completed
  - Heather: been busy with conference, so not a lot happened. Helper text was added to registration and snapshot testing added to registration.
  - Yusuf: Is continuing to CRUD encounter refactoring
  - Muhammed: working on get campaign by id and get encounter by id
  - Leon: migration is ready to be run but getting some error on my machine.
  - David: Made a pull request for monster component changes. Reviewed pull requests
- To Do by Next Scrum Meeting
  - Heather will: attempt to review pull requests as they come in, otherwise just catch up on work for other classes to be able to focus on these tasks more fully
  - Yusuf will: Continue to CRUD encounter refactoring
  - Muhammed will: continue to work on get campaign by id
  - Leon will: get migrations to work
  - David will: Start working on running an encounter
- Team Tasks?:
  - Open Pull Requests?
    - 286: David monster expansion panels
- When is our next meeting?
  - Monday at 9pm

2019-04-15 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, and David

- Agenda (prepared by Heather)
- Work Completed
  - Heather: completed her other classes' assignments to free up some of her time for this class
  - Yusuf: Reviewed David's PR (286). Continuing to CRUD encounter refactoring
  - Muhammed: Still working on Get Campaign by ID (Backend)
  - Leon: still sorting out issues with running migrations
  - David: Created a component for basic encounter running
- To Do by Next Scrum Meeting
  - Heather will: investigate Registration Testing Fixes and make as much progress as I can on this

  - Yusuf will: Continue to CRUD encounter refactoring
  - Muhammed will: finish Get Campaign by ID (Backend)
  - Leon will: get migrations to run
  - David will: Get the encounter running to a solid state and polish after pull request.
- Team Tasks?:
  - Open Pull Requests?
    - N/A
- When is our next meeting?
  - Wednesday at 9pm

2019-04-17 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, and David

- Agenda (prepared by Heather)
- Work Completed
  - Heather: completed registration test updates and made that pull request. Also re-finished the presentation for today.
  - Yusuf: Reviewed Heather and Muhammed's PRs.
  - Muhammed: implemented get campaign by id and get encounter by id
  - Leon: Abandoned get migration to work on my machine, David is going to generate the migration for my backend changes once I'm done with the unification process. Campaign can be created, viewed, and edited - logic resides in CampaignCreation and CampaignDetails. Editing Campaign Integration issue is ready for review but will have to wait till Unify campaign CRUD issue is done.
  - David:Worked on tests for running encounters. Reviewed pull requests
- To Do by Next Scrum Meeting
  - Heather will: set up a meeting with Alic for abstract and helper text
  - Yusuf will: Continue to CRUD encounter refactoring
  - Muhammed will: work on to create the character entity
  - Leon will: Fix deleting campaigns to finish Unify campaign CRUD issue. Write more tests for campaign. If I have time, change Encounter Id input box to Encounter Id's checkboxes.
  - David will: continue to work on testing run encounter.
- Team Tasks?:
  - Open Pull Requests?
    - Yusuf refactor encounter crud
- When is our next meeting?
  - Thursday at 9:45pm

2019-04-18 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, and David

- Agenda (prepared by Heather)
- Work Completed
  - Heather: met with Alic to outline the helper text and scheduled another meeting for the abstract review after initial abstract creation
  - Yusuf: Completed the refactoring and implementation of CRUD for encounters with tests. Created the PR and waiting for review. This PR includes both of my tasks for this sprint.
  - Muhammed: Working on character entity creation
  - Leon: Fixed delete campaign and backend tests for delete.
  - David: Wrap up testing encounter run
- To Do by Next Scrum Meeting
  - Heather will: update the helper text and have abstract done after meeting w/ Alic
  - Yusuf will: Be finished with my tasks for this sprint, and will hopefully continue to help with PRs
  - Muhammed will: finish character entity
  - Leon will: Work on frontend tests for Campaign CRUD.
  - David will: Make an encounter run pull request
- Team Tasks?:
  - Open Pull Requests?
    - Yusuf refactor encounter
- When is our next meeting?
  - Sunday at 9pm

2019-04-21 Discord Voice (Kickoff)

Participation by: Heather, Yusuf, Muhammed, Leon, and David

- Agenda (prepared by Heather)
- Capacity
  - Reminder: Normal developer capacity is ~12 for sprints
  - Team Capacity: 40
    - Yusuf: 5 / 12
      - 42% capacity due to taking PhD exam Part 3. Do your best Yusuf! 👍
    - Muhammed: 9 / 12
      - 75% capacity due to very important exam coming up. Do your best Muhammed! 👍

        - Leon: 7 / 12
            - 59% capacity due to lots of outside work and class project being due. Do your best Leon! 👍
        - David: 13 / 12
            - 108% capacity to help finish the features of this sprint in time.
        - Heather: 8 / 12
            - 75% capacity due to scrum master duties, 90% of that capacity due to being out of town this weekend and having other class projects due.
    - See review/Q&A tasks for feedback from Alic on the features' acceptance
- New Focus Items?
    - Player Characters
    - Final presentation materials
    - Running encounter features
- Next meeting
    - Monday 9:00pm
    - Discord Voice

2019-04-22 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, and David

- Agenda (prepared by Heather)
- Work Completed
    - Heather: created all issues and milestone for GitHub
    - Yusuf: On leave for PhD exam.
    - Muhammed: On leave for an important exam.
    - Leon: No work on SEP, caught up on other school work.
    - David: Getting started on character create read update.
- To Do by Next Scrum Meeting
    - Heather will: start working on the final presentation
    - Yusuf will: Keep being on leave for PhD exam.
    - Muhammed will: Keep being on leave for an important exam.
    - Leon will: Fix campaign tests.
    - David will: Work on finishing the crud and then create a catalog so we can access these functionalities.
- Team Tasks?:
    - Open Pull Requests?
        - Get One PC (Reviewed, waiting changes)
- When is our next meeting?
    - Wednesday at 9pm

2019-04-24 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, and David

- Agenda (prepared by Heather)
- Work Completed
  - Heather: outlined final presentation
  - Yusuf: On leave for PhD exam.
  - Muhammed: On leave for an important exam.
  - Leon: Reviewed CharacterCRUD pull request. Moved campaign tests into separate files.
  - David: Finished CharacterCRUD merged into master
- To Do by Next Scrum Meeting
  - Heather will: finish making the presentation
  - Yusuf will: Keep being on leave for PhD exam.
  - Muhammed will: Keep being on leave for an important exam.
  - Leon will: Still working on campaign tests and doing some refactoring on CampaignCRUD to match format of Characters.
  - David will: work on character catalog
- Team Tasks?:
  - Open Pull Requests?
    - Character Catalog (Roadblocked by Get Many PCs task)
- When is our next meeting?
  - Thursday at 9:45pm

2019-04-25 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, and David

- Agenda (prepared by Heather)
- Work Completed
  - Heather: created final presentation
  - Yusuf: On leave for PhD exam.
  - Muhammed: was on leave for an important exam.
  - Leon: Still working on campaign tests.
  - David: Finished the most of the pieces for catalog
- To Do by Next Scrum Meeting
  - Heather will: make release for last sprint and start compiling docs for binder
  - Yusuf will: Keep being on leave for PhD exam.
  - Muhammed will: do Get Player Characters (Many, Catalog View) [Backend] and handle requested changes for Get Player Character (One, By id)

  - Leon will: Finish campaign tests and start next issue Add Player Characters to a Campaign [Frontend]
  - David will: Work on getting catalog into master and then verify that everything works when the backend server is done.
- Team Tasks?:
  - Open Pull Requests?
    - Character Catalog (Roadblocked by Get Many PCs task)
    - Presentations
- When is our next meeting?
  - Sunday at 9:00pm

2019-04-28 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, and David

- Agenda (prepared by Heather)
- Work Completed
  - Heather: was out of town, compiled some materials for the binder
  - Yusuf: Was on leave for Phd exam
  - Muhammed: requested changes for Get Player Character (One, By id) are done by David. I'm still working on Get Player Characters (Many, Catalog View) [Backend]
  - Leon: Getting bad request in CampainCreation test when it should create campaign with name only. It's not due to me refactoring CampaignCRUD, campaign creation works like it suppose to.
  - David: Worked on encounter running start looking at adding player characters
- To Do by Next Scrum Meeting
  - Heather will: get binder and materials for the physical handout and review poster if it comes in
  - Yusuf will: stop being on leave for Phd exam.. And work on the poster.
  - Muhammed will: have progress on Get Player Characters (Many, Catalog View) [Backend].
  - Leon will: Find out why campaign creation test fails.
  - David will: add player characters to encounter
- Team Tasks?:
  - Open Pull Requests?
    - Character Catalog (Roadblocked by Get Many PCs task)
    - Presentations
- When is our next meeting?
  - Monday at 9:00pm

2019-04-29 Discord Voice

Participation by: Heather, Yusuf, Muhammed, Leon, and David

- Agenda (prepared by Heather)
- Work Completed
  - Heather: got the binder materials and compiled 90% of the materials for the binder
  - Yusuf: Created the poster design and initial content placement
  - Muhammed: Implemented Get Player Characters (Many, Catalog View) [Backend].
  - Leon: Still fixing CampaignCRUD tests.
  - David: Adding player characters is almost done, waiting on my pull request. My pull request is blocked by the backend get many functionality.
- To Do by Next Scrum Meeting
  - Heather will: finish adding the latest scrum meetings to the binder and finish it to get it printed by Wednesday
  - Yusuf will: Finalize the draft for the poster and will send it to Heather for review and finalization.
  - Muhammed will: Push Get Player Characters (Many, Catalog View) [Backend] to GitHub and implement Delete Player Character
  - Leon will: Fixed all tests for CampaignCRUD and start on adding characters to campaign
  - David will: will: review “get many” on the backend and finish working on characters added to encounters.
- Team Tasks?:
  - Open Pull Requests?
    - Character Catalog (Roadblocked by Get Many PCs task)
- When is our next meeting?
  - Wednesday at 9:00pm

Sprint Reviews
> Sprint 1

Sprint 1 Overview

In this sprint, we will be setting up the database and boiler plate for the application, including basic user log in and registration to access the application. The MVP for the end of sprint 1 (for users) is an exported native application that a user can register for and log in to, as then start to be able to see and create basic monsters in their catalogue. For the developers, the MVP will be to have most of the project set up completed, such as with continuous integration and linting.

Product Owner: Alic Szecsei

Scrum Master: Heather Kemp

Team Capacity

We believe our team's capacity for Sprint 1 is 45 story points. Due to most of the team being unfamiliar with the technologies being used in this project, we decided that our capacity was actually limited to about 75% of our normal capacity due to the amount of time needed to properly research these topics. We chose to do this rather than just creating a research task to fill in the remaining story points, as the latter didn't seem in line with our Agile manifesto.

Resources Needed

- Azure account for the database will be configured (Heather will look into setting this up). The Postgres database will need to be designed and configured (David will lead this effort).
- GitHub, Travis CI, and CodeCov will need to be configured for the Continuous Integration set up (Heather will look into this).
- Docker set up will need to be configured (Leon with look into this)
- GitHub project board for Sprint 1 Backlog identified.

DM TOOLS

SEP Team 500

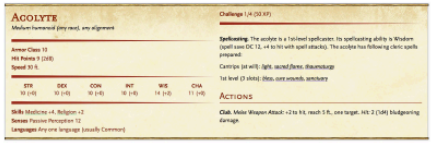
Our Vision

D&D BEYOND
- Viewing one character's details at a time
- Limited interaction with characters' information
- Managing story details by yourself

Roll20
- Generic for all tabletop games
- Manual data entry and managing details of the game
- Focused on replacing the in-person experience with a virtual one

DM TOOLS
- Augmenting and complimenting the personal experience of in-person D&D
- Providing a centralized means of organizing and generating details for gameplay

The Challenges of Being a Narrator (The Scope of DM Tools)



Things the narrator keeps track of:
- Where are the characters?
- How healthy are the characters?
- What *can/would* each of the characters do?
- *How* can each of the characters do this?
- What does the character have with them and why?

And more...

Our MVP (& Framework Stack)

- Viewing multiple characters' information at the same time
- Assisted crafting of a story and tracking aspects around combat sequences
- Partially automated generation and management of character statuses
- Save and load combat sequences

Electron | Node JS | Jest | Enzyme | PostgreSQL

Test Plan for DM Tools

SEP Group 5

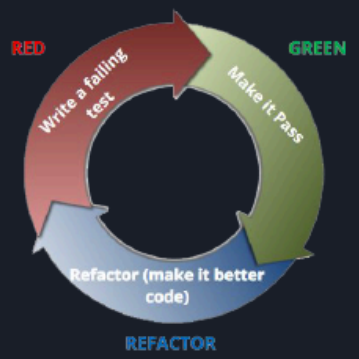
Testing Strategy

Unit test
- At least 80% of all feature code have to be covered

Integration test
- Comprehensive, independent tests for each interaction medium

Regression test
- At least 80% of codebase must be covered at all times
- 100% of the tests for the codebase must pass at all times

Acceptance test
- Product Owner pilots feature to local DMs



Testing Tools

Enzyme JS:
- React Integration
- Front End Testing

Jest JS:
- API Testing Integration
- Backend and Scripting Testing

Travis CI:
- Continuous Integration
- Ensures code coverage on all pull requests

CodeCov:
- Enforces 80% code coverage on repo
- Provides extra code coverage reports over time

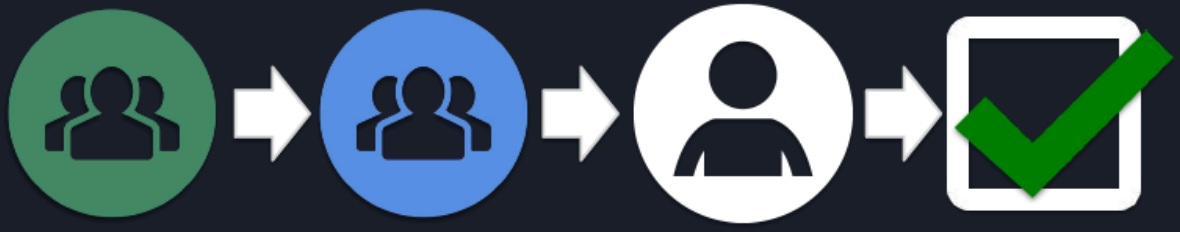
Testing Responsibilities

Feature developer(s) are responsible for implementing all code-based tests for their feature

Code reviewer are responsible for verifying that the code-based tests are accurate and comprehensive for the feature

Product owner is responsible for (official) acceptance testing

Feature Acceptance Criteria and Process



Feature Developer(s):
- 80%+ Code Coverage
- Feature Implementation and Testing

Other Developers:
- Code Style
- Code and Test Correctness

Product Owner:
- Acceptance Testing

Feature Acceptance

Questions?

Sprint 1 Retrospective

SEP Team 5

Things That We Did Well

Stable and Robust Project Foundation

- Proactive fixes of unmaintainable code over temporary fixes to complete sprint
- Thoroughly documented wiki pages and processes

Pro-active Roadblock Removal

- Strong database schema and creation during sprint 1
- Robust component creation for re-use in future sprints

Strong Product Owner Communication

- Feedback on story acceptance in < 24 hours of completion

Things That We Didn't Do Well

MVP Task Delivery Time

- Local database setup was delayed until late Saturday night, 2nd week
- Initial front end tasks were delayed until mid second week

Pull Request Review Response Time

- Pull requests went for up to 3 days without being receiving initial review

Late Project Set-up and Dependency Adding

- Frameworks (i.e. React/Joi/TypeORM) were delayed into the later half of the sprint
  - Unexpected dependencies due to late set up to avoid duplicate work

Team Member Capacities

- Initial estimates of capacity did not include time dedicated to test plan documents, presentations, etc.
- Scrum Master duties cut into development time

Story and Feature Break Down

X4 - 1 point

X1 - 2 Point

X4 - 3 point

X2 - 5 Point

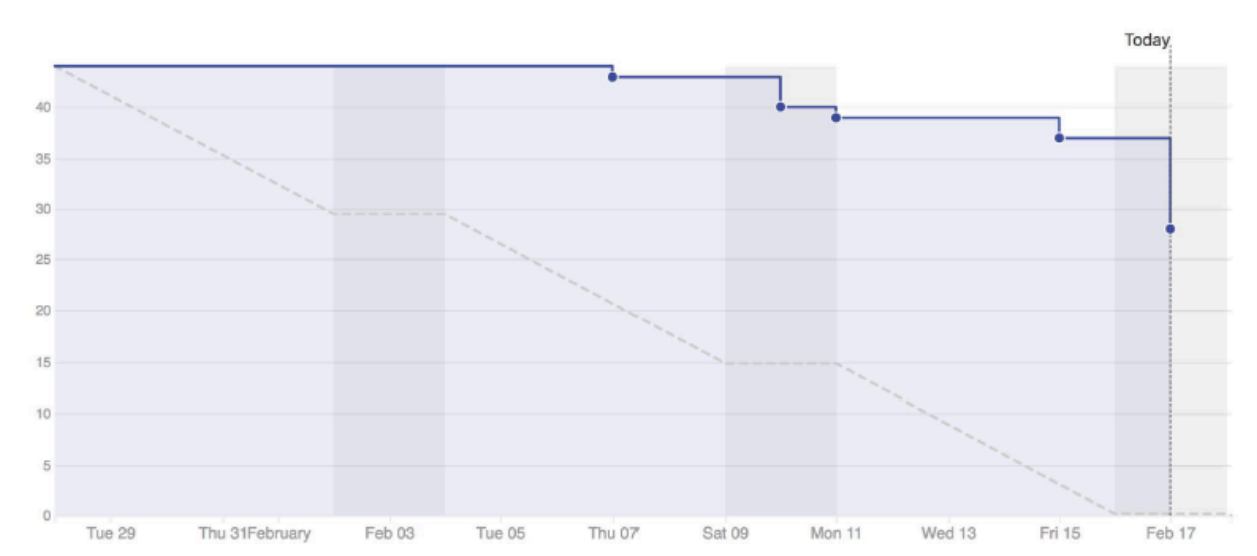
X2 - 8 Points

Total: 45 Points

Story and Feature Break Down

- Continuous Integration Set-Up (Included Code Coverage set up)
- Log In (Included Testing and React set up)
- Log in Backend (Included Testing and Enzyme set up)
- Database set up (Included orm set up)

Burn Down Rate



Interteam Communication

- Minor confusions about tasks and errors that weren't brought up until scrum meetings

Actions We'll Take to Improve Next Sprint

MVP Task Delivery Time

- Break tasks even smaller to encourage faster delivery of smaller parts

Pull Request Review Response Time

- Reminders of open pull requests are now a part of our scrum meetings

Late Project Set-up and Dependency Adding

- Dependency setups are going to be in separate tasks/pull requests
- Most of the dependencies are already added, so future dependency adding will be limited

Team Member Capacities

- Before assigning workload capacity, review the class schedule and our own schedules

Story and Feature Break Down

- Research tasks in advance before accepting or creating the stories
- Put more details into what *exactly* we want for a story before beginning it to prevent feature creep

Interteam Communication

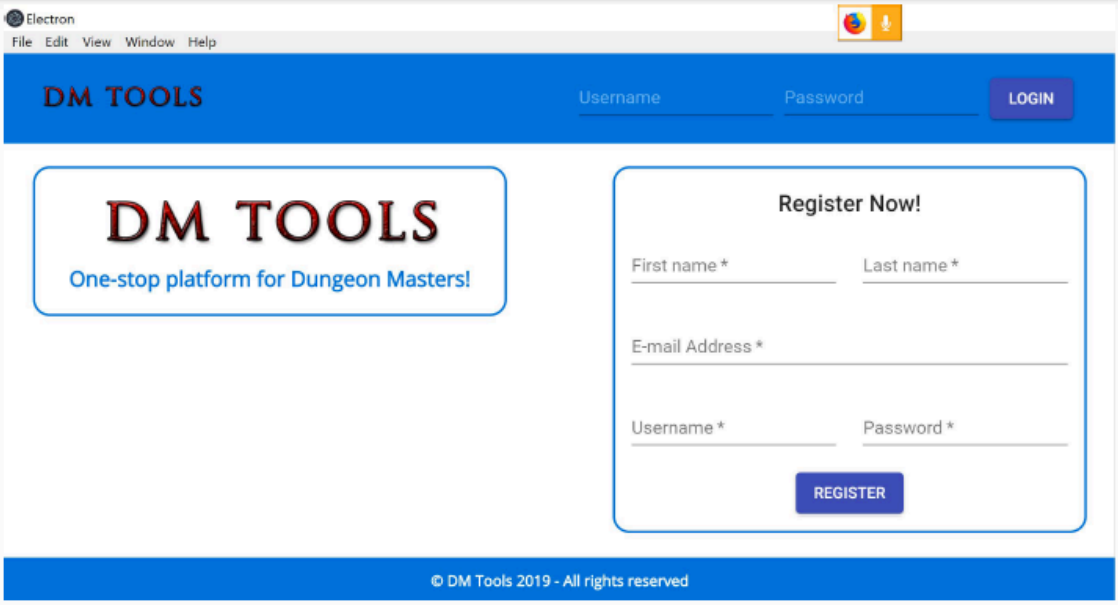
- Encourage team to post errors and seek for help as soon as issues arise

Burn Down Rate Catch Up

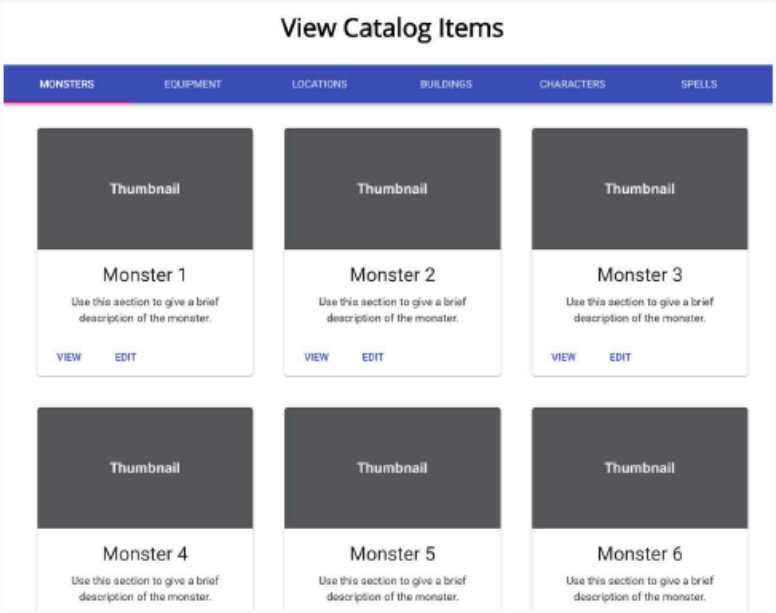
- Break tasks up into more manageable chunks to encourage smaller features being accepted sooner

Outcome

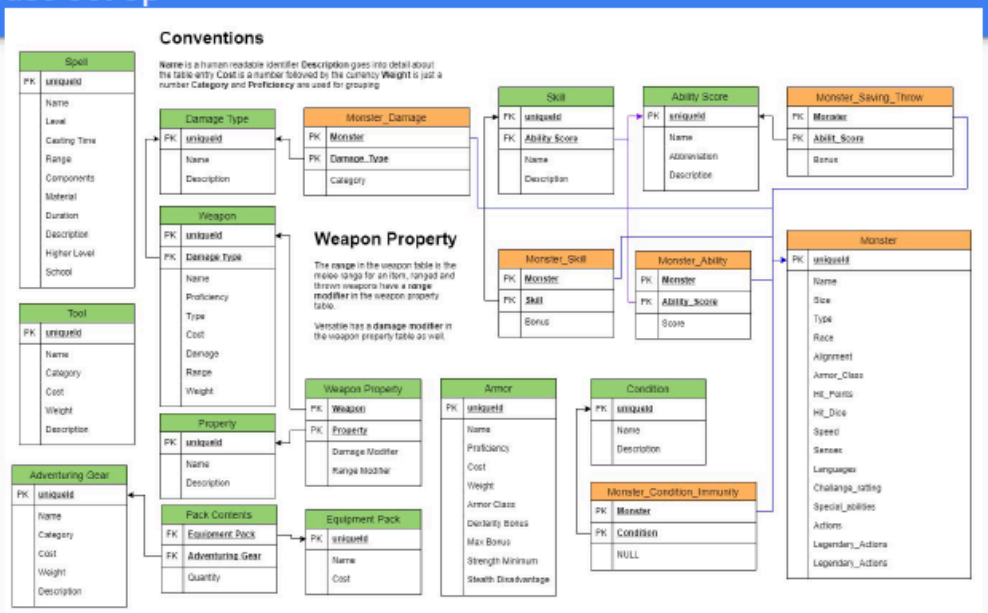
Landing / Registration / Login Page



View Game Catalogue / Monsters Gallery



Database Set Up



> Sprint 2

Sprint 2 Overview

In this sprint, we will be finishing up the application setup, including integrating user log in and registration to access the application. The MVP for the end of sprint 2 (for users) is an exported native application that a user can register for and log in to with a basic navbar, then be able to see and create basic monsters in their catalogue. For the developers, the MVP will be to have project set up completed, such as with react router, storing session tokens as cookie, and having monster related data already being in the database.

Product Owner: Alic Szecsei

Scrum Master: Heather Kemp

Team Capacity

We believe our team's capacity for Sprint 2 is 43.8 story points. While most of the team is familiar with the technologies being used in the project, a few members are still unsure if they can contribute at 100% full capacity yet. Furthermore, 2 team members will be gone for half of the sprint, leading to about half capacity for them. Hopefully this will put us on track for our estimations versus our velocity.

Resources Needed

- GitHub project board for Sprint 2 Backlog identified.
- React Router setup

Sprint 2 Retrospective

SEP Team 5

Things That We Did Well

Caught up on Sprint 1 tasks

- All tasks from sprint 1 are now complete and additional tasks to get us set up for sprint 3 were completed

Pro-active Dependency and Resource Adding

- Many crucial things were accomplished within the first week
  - Dependencies were added
  - Examples of pages and application structure
  - Database entities

Modular and Robust Code Features

- External files containing commonly reused features
  - Enums for monster related information
  - Cookie handling class
- 91% Code Coverage (Sprint 1) → 95% Code Coverage (Sprint 2)

Team Member Capacity

- No one was extremely overworked or had excessive tasks leftover at the end of the sprint

Story and Feature Break Down (Sprint 2)

- React Router Set Up
- Creating a Catalog Monster
- Integrated login and registration (Sprint 1 leftover tasks)
- Creating an Encounter (backend)
- Finish dockerizing the application backend for local development
- Deploy to AWS
- Data parsing and importing tasks for database, migrations
- Database Entity set up for Monsters and Encounters (Sprint 2 and 3 tasks)

Things That We Didn't Do Well

Dev Workstation Setup

- Workstations were only set up for the tasks that the developer was working on (frontend, backend, server)
- Workstation setup was delayed until the end of the first week while completing current tasks
  - Delayed pull requests

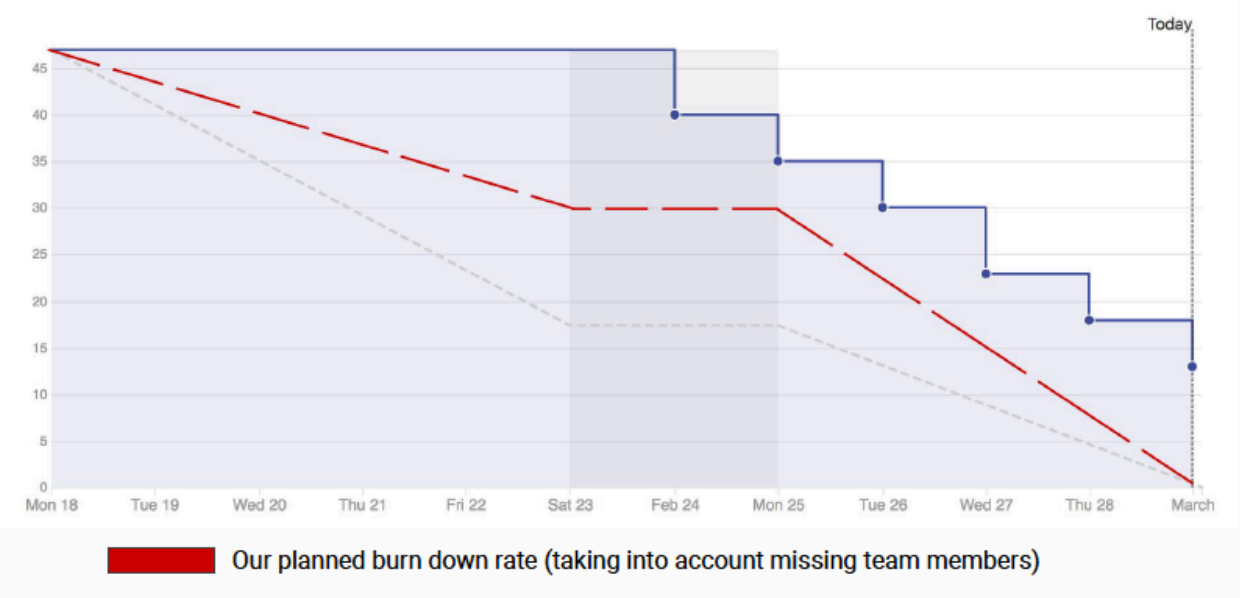
Team Member Capacities

- We took into account absences but not *when* those absences were
  - Two team members were only here the first 2 days and the last 2-3 days of the sprint
  - Limited development time and impact on burndown rate
  - Development time went a little over the end of the sprint (last pull request at 4am on Saturday)

Pull Request Review Response Time

- Due to half of the team being gone the first week, pull requests review capacity was limited
- Request review ability was also limited by workstation setup
- Request review conversations mainly took place in Discord for ease of conversation

Burndown Rate



Actions We'll Take to Improve Next Sprint

Workstation Setup

- We're meeting as a team early this week to go over complete workstation setup
  - Catching up teammates who were gone and solidifying understanding

Pull Request Review Response Time & Burndown Rate Catch Up

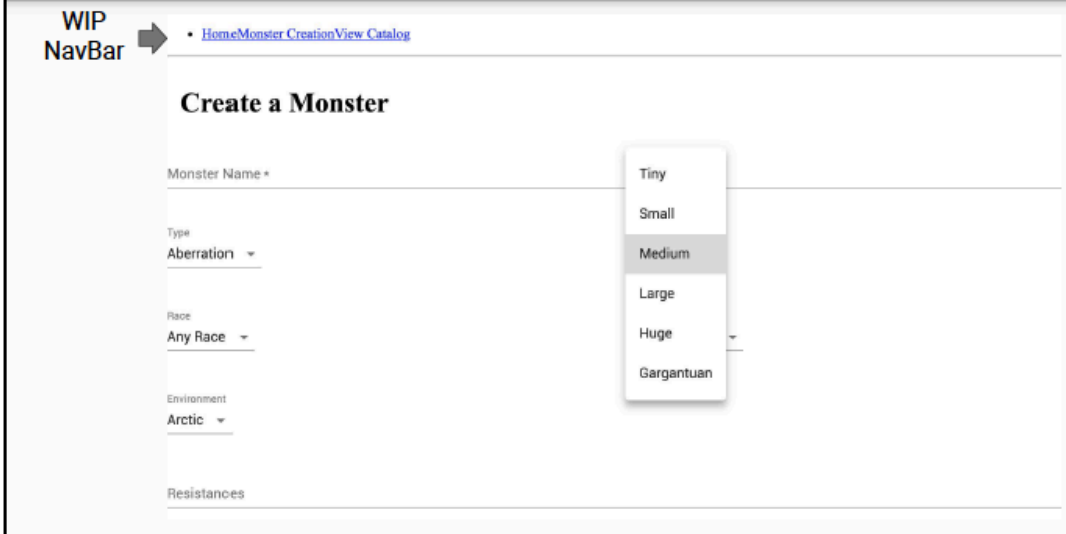
- Ensure workstations are set up for all aspects of the project
- Ensure someone is always available and ready to review incoming pull requests
- Ensure pull request conversations (or at least summaries of them) happen on GitHub

Team Member Capacities

- If someone is going to be gone for most of the sprint, ensure that the groundwork is laid out for their task first to allow for a smooth finish for tasks

Outcome

Monster Creation Page



54 inputs with various constraints

> Sprint 3

Sprint 3 Overview

In this sprint, we will be focusing on modularizing, optimizing, and securing our application, updating our GUI, and creating encounters. The MVP for the end of sprint 3 (for users) is an exported native application that builds upon sprint 2's MVP, allowing a user to create an encounter and only access pages they have access to. For the developers, the MVP will be to have the project converted from Material UI to Bulma and Bloomer, secured through session token checking, cookie storing complete, monster skill data added to the DB, and optimized API URL extraction and docker building.

Product Owner: Alic Szecsei

Scrum Master: Heather Kemp

Team Capacity

We believe our team's capacity for Sprint 3 is 52 story points. The team is now familiar with the stack so the team can contribute at 100% capacity if they are here throughout the full sprint. However, 2 team members will be absent for brief periods during the sprint (one for the last 3 day of the sprint, one for the 1st week's weekend), resulting in our capacity being slightly under the max.

Resources Needed

- Team will need to be on-boarded for using all aspects of the application (locally deploying server using Docker, locally running frontend using yarn and webpack)
- GitHub project board for Sprint 3 Backlog identified.

DM TOOLS

Mid-semester Reviews

SEP Team 5

Product Overview (Review)

Goblin

Small humanoid (goblinoid), neutral evil

Armor Class 15 (Leather Armor, Shield)
Hit Points 7 (2d6)
Speed 30 ft.

STR	DEX	CON	INT	WIS	CHA
8 (-1)	14 (+2)	10 (+0)	10 (+0)	8 (-1)	8 (-1)

Skills Stealth +6
Senses Darkvision 60 ft., Passive Perception 9
Languages Common, Goblin
Challenge 1/4 (50 XP)

Nimble Escape. The goblin can take the Disengage or Hide action as a bonus action on each of its turns.

Actions

Scimitar. Melee Weapon Attack: +4 to hit, reach 5 ft., one target. Hit: 5 (1d6 + 2) slashing damage.

Shortbow. Ranged Weapon Attack: +4 to hit, range 80/320 ft., one target. Hit: 5 (1d6 + 2) piercing damage.

Acolyte

Medium humanoid (any race), any alignment

Armor Class 10
Hit Points 9 (2d8)
Speed 30 ft.

STR	DEX	CON	INT	WIS	CHA
10 (+0)	10 (+0)	10 (+0)	10 (+0)	14 (+2)	11 (+0)

Skills Medicine +4, Religion +2
Senses Passive Perception 12
Languages Any one language (usually Common)
Challenge 1/4 (50 XP)

Spellcasting. The acolyte is a 1st-level spellcaster. Its spellcasting ability is Wisdom (spell save DC 12, +4 to hit with spell attacks). The acolyte has following cleric spells prepared:

Cantrips (at will): *light, sacred flame, thaumaturgy*
1st level (3 slots): *bless, cure wounds, sanctuary*

Actions

Club. Melee Weapon Attack: +2 to hit, reach 5 ft., one target. Hit: 2 (1d4) bludgeoning damage.

Things the narrator keeps track of:

- Where are the characters?
- How healthy are the characters?
- What *can/would* each of the characters do?
- *How* can each of the characters do this?
- What does the character have with them and why?

And more...

Our Goal: Providing a centralized means of organizing and generating details for playing through encounters inside a campaign

Framework Plans

Sprint 1:

- Electron and Electron Webpack
- Material UI
- Enzyme and Jest Testing
- TypeScript
- PostgresSQL
- Azure Hosting
- React
- Node JS
- Docker

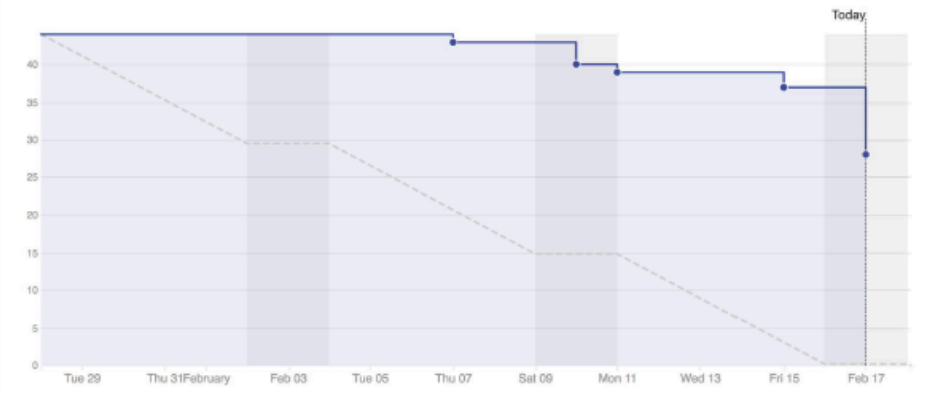
Sprint 4+:

- Electron and Electron Webpack
- ~~Material UI~~ ***Bloomer/Bulma***
- ***TS***-Enzyme and ***TS***-Jest Testing
- TypeScript
- PostgresSQL ***and TypeORM***
- ~~Azure Hosting~~ ***Amazon Web Services***
- React ***and React Dev Tools***
- Node JS ***with Hapi, Joi, and Boom***
- ***Nock (Http Mocking)***
- ***React Router***
- ***Electron Store (Cookie Management)***
- ***JSON Web Token (JWT)***
- Docker

Sprint 1 Milestones

- Basic application setup
- Completed database schema
- Frontend and Backend for login and registration completed
- Frontend for monster creation started
- Initial lofi sketches made
- Adding Electron, Node, Docker, AWS, TypeORM, and Material UI

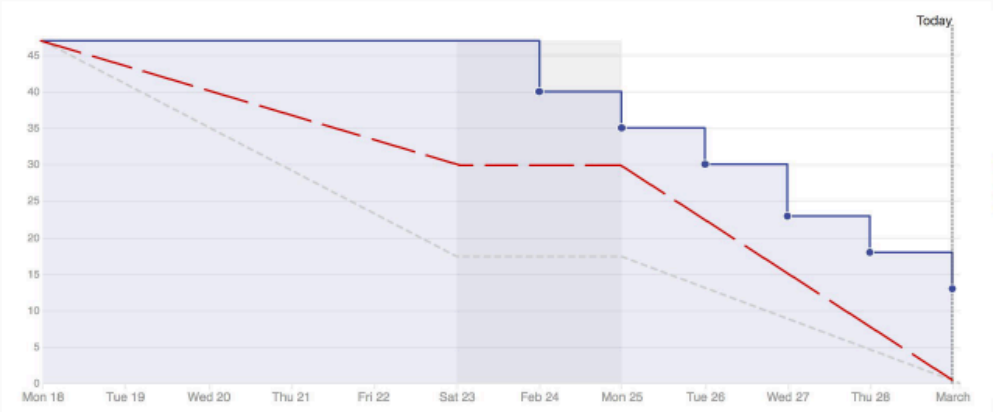
- 91% code coverage
- 45 point capacity



Sprint 2 Milestones

- React Router added (+ navbar)
- Session tokens and cookie management started
- Database populated with pre-gathered data for current features
- Prior features integrated from sprint 1
- Switch to TS testing frameworks, add devtools, add type checking to backend, and http mocking

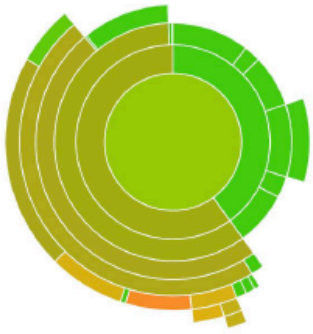
- 92% code coverage
- 43.8 point capacity



Sprint 3 Milestones

- Material UI -> Bulma and Bloomer
- Secured application via session token checking for pages
- Cookie management
- Optimized API URL referencing and Docker Building
- Updated DB w skill data
- Create encounter

- 93% code coverage
- 52 point capacity



Future Milestones

MVP:

- Update and Delete Monster **[Sprint 4]**
- Read, Update, Delete Encounters **[Sprint 4]**
- CRUD Campaigns **[Sprint 4]**
- Paginated Views of Catalog Entries (Monsters, Encounters, Campaigns) **[Sprint 4 - Sprint 5]**
- CRUD basic entries for player characters **[Sprint 5]**
- Running through an Encounter **[Sprint 5 - Sprint 6]**

Enhancements to MVP:

- CRUD for additional catalog entries (Items, Locations, etc) **[Sprint 6]**
- Enhancements to running through an encounter (Auto generated fields, etc) **[Sprint 6]**

Product Demonstration

> Sprint 4

Sprint 4 Overview

In this sprint, we will be focusing on finalizing the basic form of CRUD for the main features of our MVP. That is, our basic CRUD for monster creation, encounter creation, and campaign creation will be created. The MVP for the end of sprint 4 (for users) is an exported native application that builds upon sprint 3's MVP, allowing users to edit and delete monsters and encounters as well as create, edit, and delete campaigns. For the developers, the MVP will be to have monster creation more robust to allow for easier viewing and modeling for future features and a midpoint presentation to the stakeholders.

Product Owner: Alic Szecsei

Scrum Master: Heather Kemp

Team Capacity

We believe our team's capacity for Sprint 4 is 54 story points. As we've caught up on most of our backend features, we're going to focus on diversifying our team member's skills by helping them transition into multiple roles so that everyone can accept tasks from the frontend and backend. Furthermore, two team members will be occupied with exams/ events during the sprint, resulting in our capacity being slightly even further under the max.

Resources Needed

- Team on-boarded on frontend and backend development of the application
- GitHub project board for Sprint 4 Backlog identified.

> Sprint 5

Sprint 5 Overview

In this sprint, we will be focusing on adding a few features crucial to our MVP as well as optimizing our previous codebase. We will get a solid start on the backend for player characters, unify the code structure of CRUD for encounters and campaigns, and start basic features of running an encounter. The MVP for the end of sprint 5 (for users) is an exported native application that builds upon sprint 4's MVP, updating the editing for encounters and campaigns and unifying the CRUD of all features, added helper text for forms, and a introductory view to running an encounter for a campaign. For the developers, the MVP will be to have all forms be more robust to allow for easier viewing and modeling for future features, and have added ORM models, creation and reading for player characters.

Product Owner: Alic Szecsei

Scrum Master: Heather Kemp

Team Capacity

We believe our team's capacity for Sprint 5 is 52 story points. We're continuing to focus on diversifying our team member's skills, as most of our tasks are centered on the frontend of the application. We'll be taking an even greater hit to our capacity this sprint because three team members will be occupied with exams/events during the sprint, especially during the weekend, which is the brunt of our development time.

Resources Needed

- GitHub project board for Sprint 5 Backlog identified.
- In-depth low-fi sketches for new features

DM TOOLS

Progress Review and Testing Overview

SEP Team 5

Progress Review

MVP Progress Review

MVP:

- Catalog view of monsters, player characters, encounters, and campaigns
- CRUD for monsters, player characters, encounters, and campaigns
- Basic running through an encounter of a campaign

Left to Do and Scheduling Plan:

- UD Front end integration for Encounters and Campaigns **[U - Sprint 5, D - Sprint 6]**
- CRUD integration for player characters **[Sprint 6]**
- Catalog view of player characters **[Sprint 6]**
- Basic running through an encounter **[4/7 tasks for Sprint 5, remaining 3 tasks for Sprint 6]**

Testing Overview

Testing Tools

TS-Enzyme:
- React Integration
- Front End Testing

TS-Jest:
- API Testing Integration
- Backend and Scripting Testing

Travis CI and CodeCov:
- Continuous Integration
- Ensures 80%+ code coverage on all pull requests

Nock:
- Mocks HTTP requests and their returns

Testing Responsibilities

Feature developer(s) are responsible for implementing all code-based tests for their feature

Code reviewer are responsible for verifying that the code-based tests are accurate and comprehensive for the feature

Product owner is responsible for (official) acceptance testing

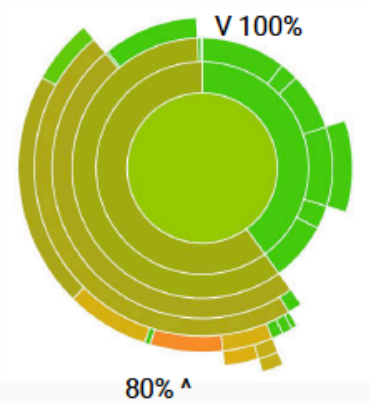
Testing Standards (Unit & Regression Testing)

Unit Testing

- At least 80% of all feature code has to be covered
  - *Usually close to 90%, with a few 80%'s*

Regression Testing

- At least 80% of codebase must be covered at all times
  - *Consistently has been around 93%*
- 100% of the tests for the codebase must pass at all times

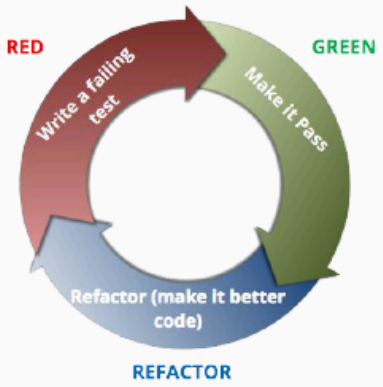


Testing Standards (Acceptance Testing)

- External Product Owner pilots feature to local DMs currently involved in campaigns
- Piloting observed by scrum master and reported back to team
- All features must be accepted by the external product owner and (within reason) local DMs
  - Procedure for conflict between DMs and team/product owner documented on Wiki

Testing Implementation Strategy

- Snapshot Testing
- Happy/Sad Path Testing
- Pseudo-Red-Green Refactor
- Isolate Components and Interactions via Mocking when not implementing integration testing



Product Demo

> Sprint 6

Sprint 6 Overview

In this sprint, we will be focusing on finishing our MVP as well as optimizing our previous codebase. We will finish implementing player characters and the basic features of running an encounter. The MVP for the end of sprint 6 (for users) is an exported native application that builds upon sprint 5's MVP, allowing for them to create, read, update, and delete monsters, encounters, player characters, and campaigns, as well as run the encounters of their campaigns from the catalog. For the developers, the MVP will be to have all tests updated to be automated, have the backend for all of the features completed, and have the documentation done for the end of the class.

Product Owner: Alic Szecsei

Scrum Master: Heather Kemp

Team Capacity

We believe our team's capacity for Sprint 6 is 40 story points. Many of our team members are busy with exams and other end of semester tasks along with travel, as detailed in our meeting minutes.

Resources Needed

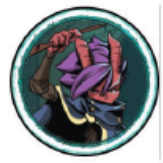
- GitHub project board for Sprint 6 Backlog identified.

DM TOOLS

End of Semester Presentation

SEP Team 5

Product Overview (Review)



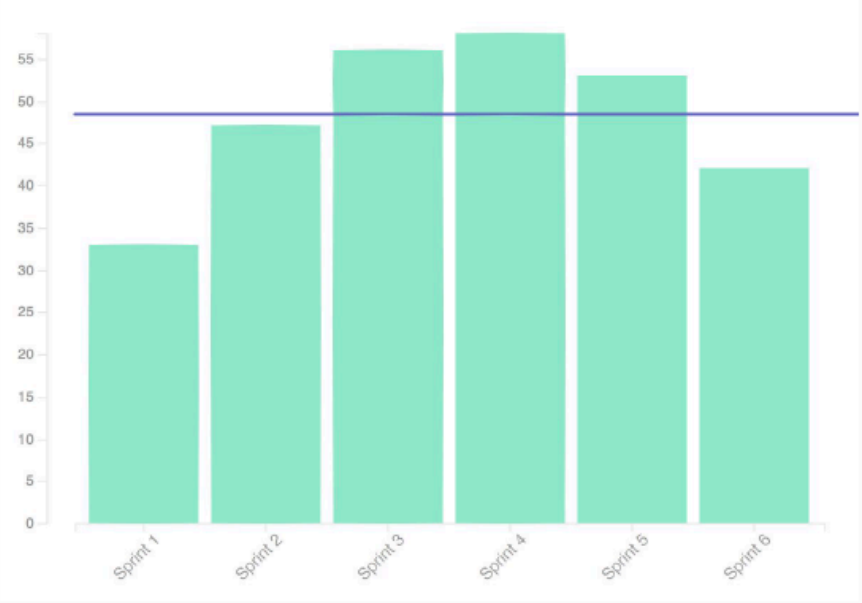
Things the narrator keeps track of:

- Where are the characters?
- How healthy are the characters?
- What *can/would* each of the characters do?
- *How* can each of the characters do this?
- What does the character have with them and why?

And more...

Our Goal: Providing a centralized means of organizing and generating details for playing through encounters inside a campaign

Semester In Review: Sprint Velocity



Completed:
222 Issues
298 Story Points

Technology Platforms Used (Frontend)

Frontend Development

Electron Store

webpack

React Dev Tools

Electron Webpack, Electron Store, React, React Dev Tools, and React Router

CSS / UI

Bulma, Bloomer, and Material UI

Technology Platforms Used (Backend and Testing)

Backend Development

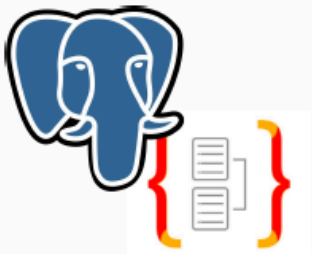
Node JS, TypeScript, Hapi JS, Joi JS, and Boom JS

Fullstack Testing Suite

TS-Enzyme, TS-Jest, and Nock

Technology Platforms Used (Server/Hosting/CI)

Database Development



PostgresQL and TypeORM

Database/Server Hosting

Amazon Web Services, JSON Web Token (JWT), and Docker

Continuous Integration

Travis CI and CodeCov

Process Strengths

- PO was in our communication channels
  - Quick responses to questions/etc.
- PR test running and code coverage requirements enforced by CI
- Roadblocks were identified and removed in advance
- (Following Sprint 2) Accurate and meaningful capacities and estimates
- Modular components allowed for straightforward stack/feature changes

Process Weaknesses

- Class and work loads are time intensive
  - Quick yet in-depth PR reviews are hard to keep up with
  - Full capacity was never available
  - Scheduling (regular) meetings is hard
  - Switching to text meetings instead of voices after a while lead to decreased communication within the group
- For quick starting the project, we broke up into specialty areas, leading to lots of catch up time later

Lessons Learned

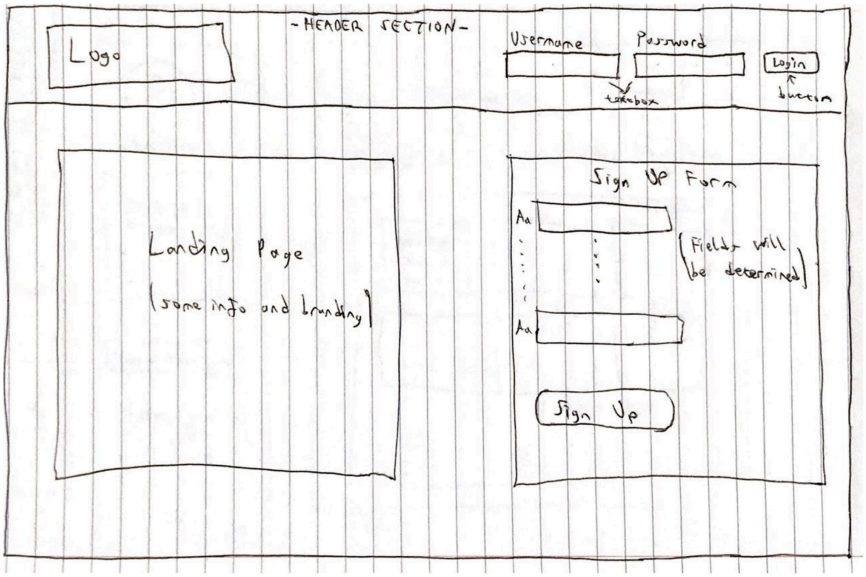
- Agile's strength are highlighted when the team can dedicate fully to it
- Strong communication with PO/stakeholders is essential
  - Especially important before sprint closure
- Well-rounded teams take more time to get setup but perform better overall
- Team standards work best when ideals are shared rather than enforced
- Product stack changes should be researched and communicated to the team in advance

Product Demonstration

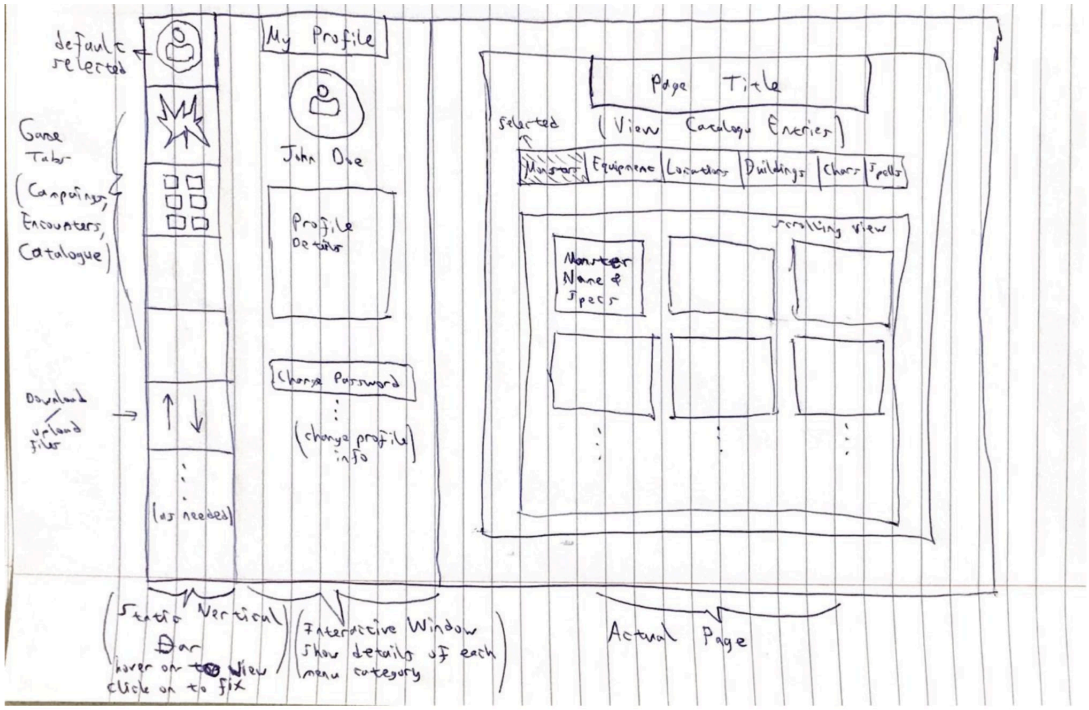
Appendices

> LowFi Sketches

Landing Page, Registration, and Login

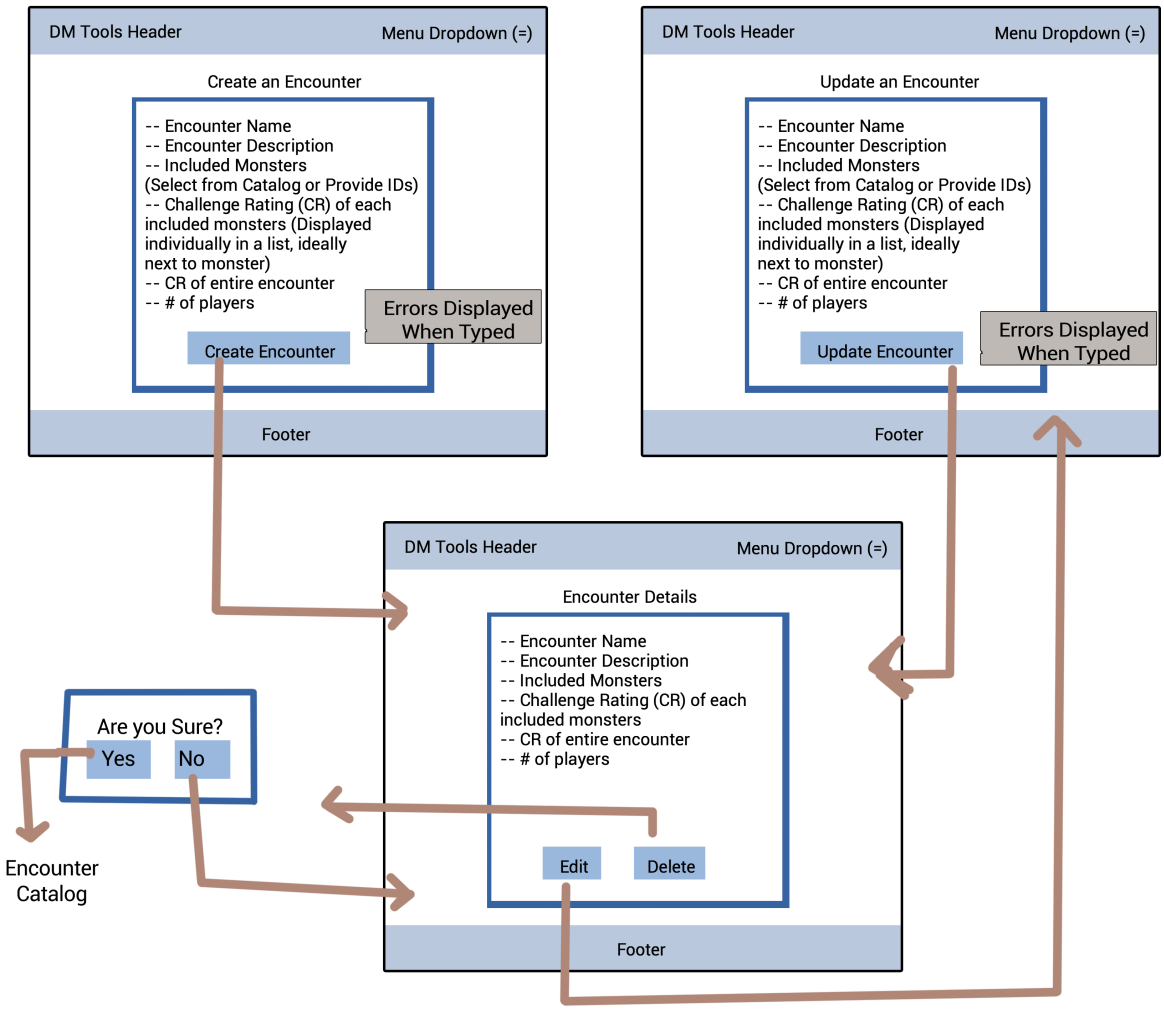


Main Page

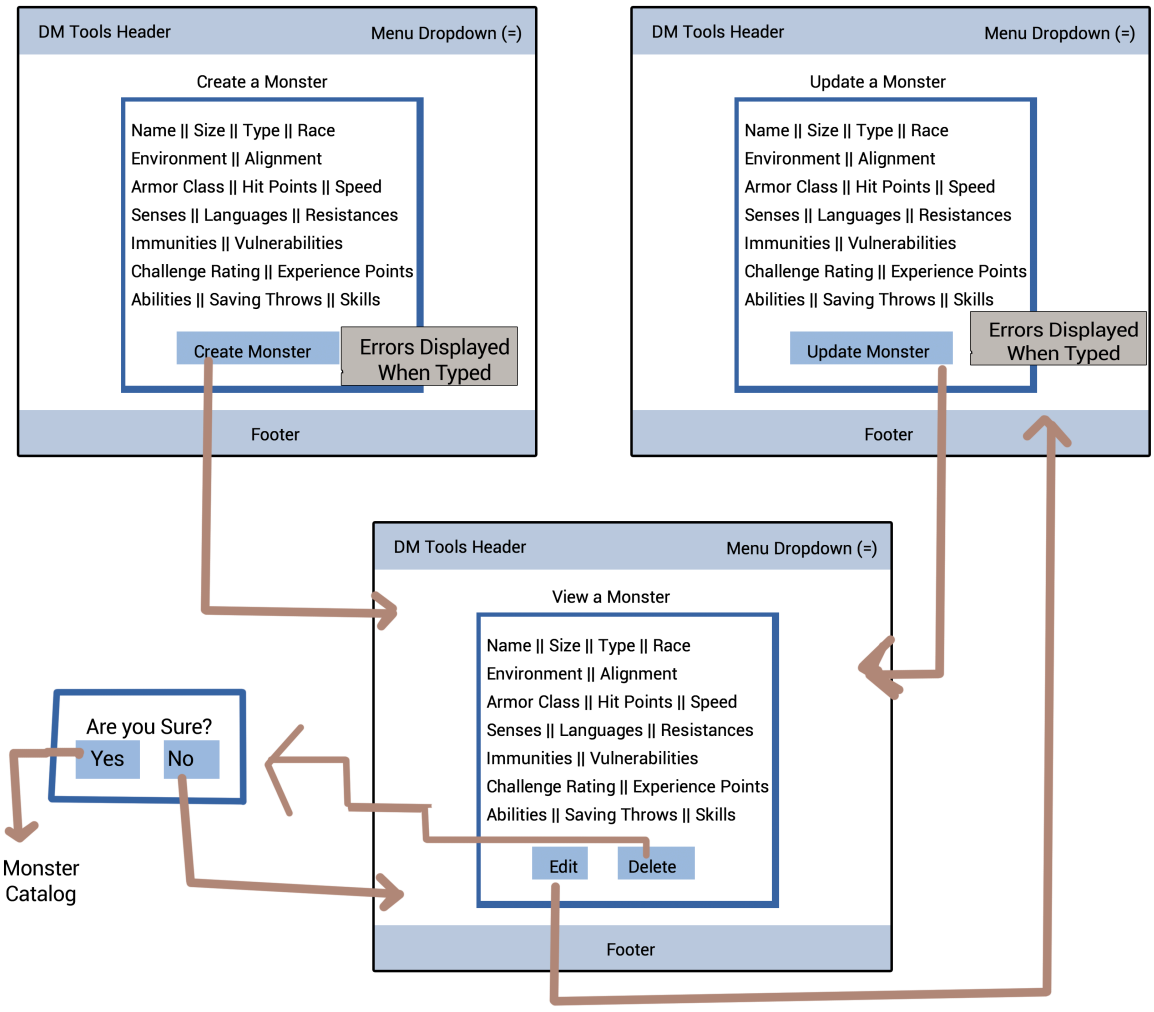


CRUD Interactions

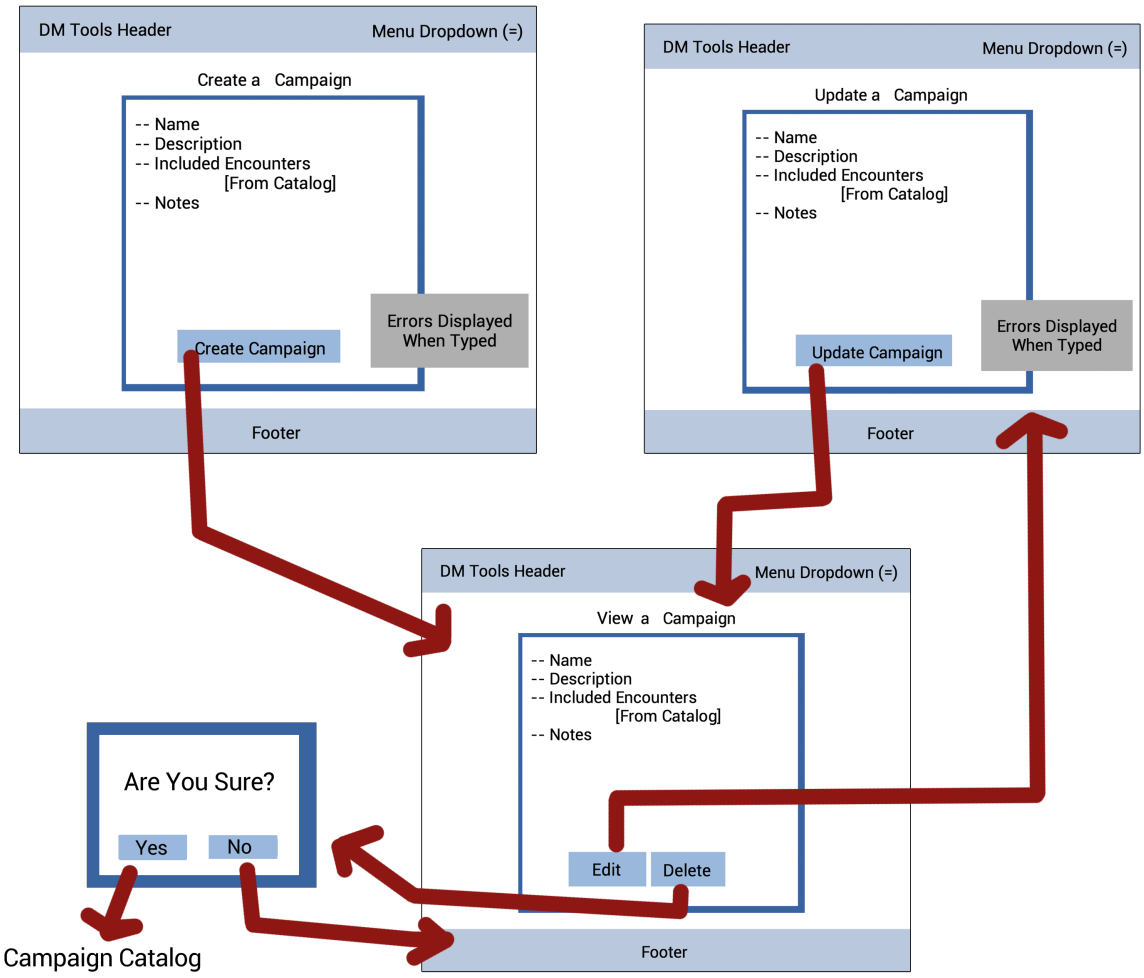
Monster CRUD



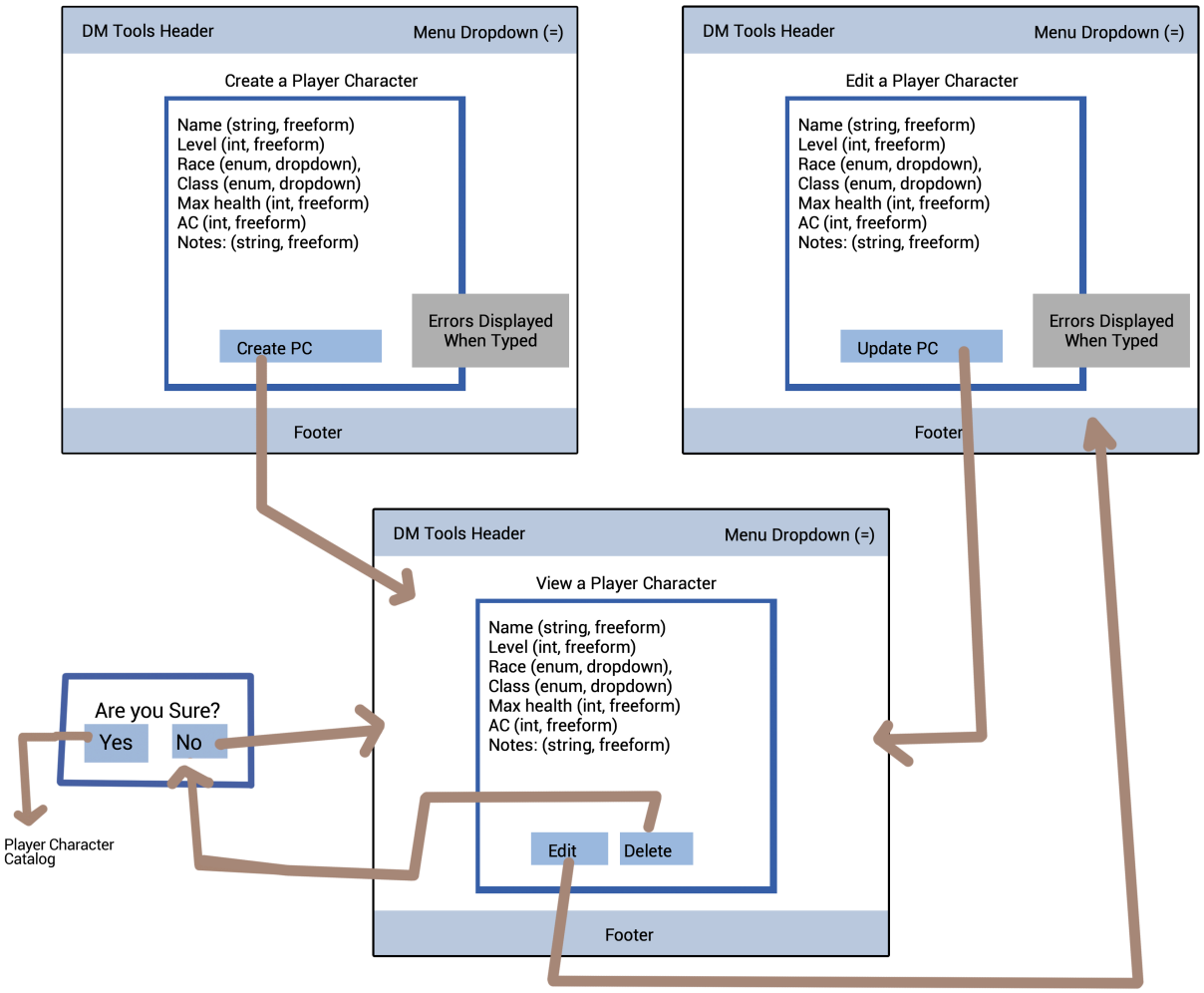
Encounter CRUD



Campaign CRUD

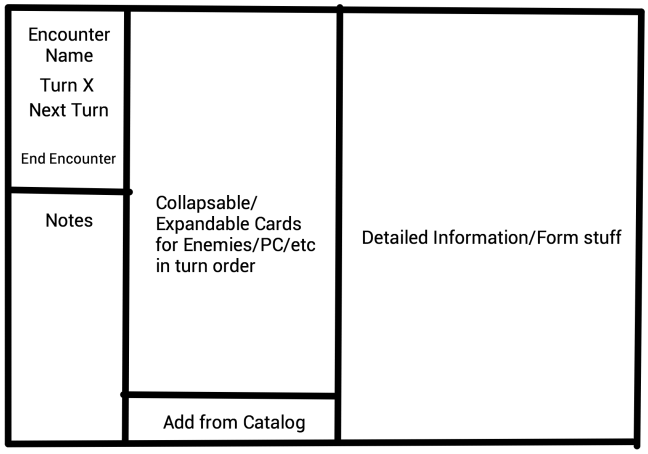


Player Character CRUD



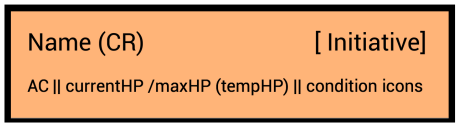
Running an Encounter

General Running Encounter Structure

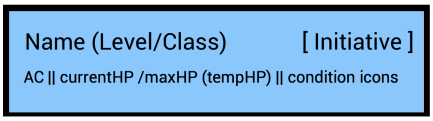


Monster and Player Character Components in Running of Encounter

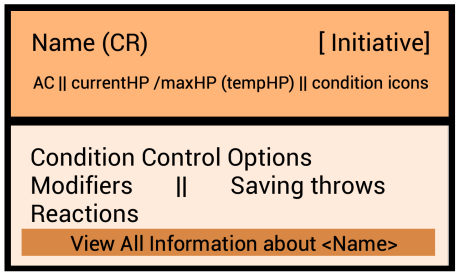
Collapsed Monster View:



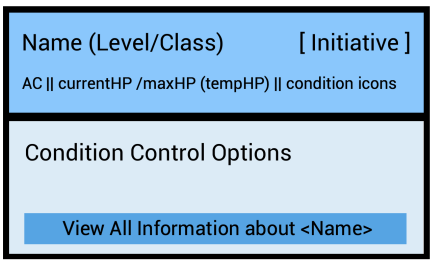
Collapsed Player Character View:



Expanded Monster View:

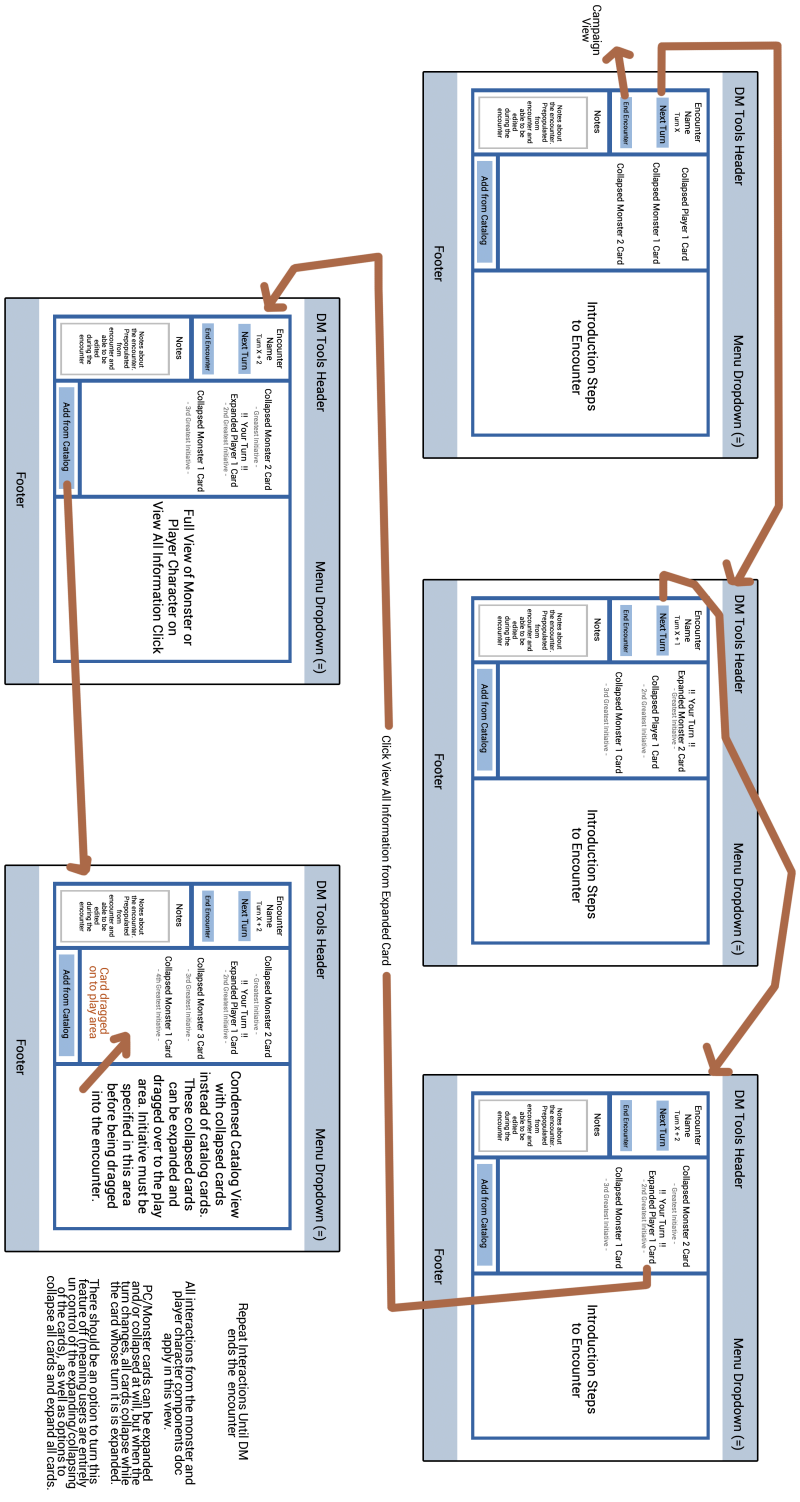


Expanded Player Character View:



Interaction with the Views:
- Initiative can be updated until turn 1 starts
- AC, current HP, max HP, and temp HP are freeform (integer) boxes
- Conditions are controlled by check boxes for the specific conditions in the expanded view

Running of an Encounter (Detailed)



> Tools Used

External Tools

For explanations on what external tools were used for the project and why, see the user manual.

General Tools

Bulma / Bloomer

Bulma is a free, open source CSS framework based on Flexbox and built with Sass. It's 100% responsive, fully modular, and available for free.

Bloomer - a cool set of React Stateless components for Bulma

To allow for a application wide CSS base, we chose to use Bulma and Bloomer, which fit the styling of our application well. Since it was just CSS, it didn't add bells and whistles that made it obtrusive to use (as with other frameworks).

Material UI VS Bulma / Bloomer

We chose to use Bulma and Bloomer over Material UI because Material UI buries its onclick method inside several layers of abstraction. This made testing with Jest nearly impossible because it required knowing the abstraction layers, which aren't easily accessible from their documentation. To improve our ability to test, Bulma and Bloomer, which only applied CSS, seemed like the best choice.

React Router

Declarative routing for React.

To allow for easier navigation of our application, we added the ability to navigate between 'pages' of our application. It doesn't restrict our 'navigation' to using links like this, so we can still switch between being modular and using linking whenever works best for us.

React Dev Tools

React Developer Tools is a Chrome DevTools extension for the open-source React JavaScript library. It allows you to inspect the React component hierarchies in the Chrome Developer Tools.

As we use Electron, which uses Chromium, using React Rev Tools to allow for us to get simple debugging of our React state and components seemed like a no brainer, especially with how easy it was to import it.

Electron Store

Simple data persistence for your Electron app or module - Save and load user preferences, app state, cache, etc

Electron doesn't have a built-in way to persist user preferences and other data. This module handles that for you, so you can focus on building your app.

We found storing session information to be very cumbersome to do ourselves. As this library's read.me promised, storing and retrieving cookies was now as simple as a function call. It does have the caveat of (seeming to be) using a dependency that requires node version 11.7 and higher, which is a downside, so there *may* be a need to find a better solution in the future.

Docker

Docker provides this same capability without the overhead of a virtual machine. It lets you put your environment and configuration into code and deploy it. The same Docker configuration can also be used in a variety of environments. This decouples infrastructure requirements from the application environment.

We chose to use Docker to assist with our local deployments/execution to avoid having to deal with the setup of each developer's codebase workstation with dependencies. By having it's own environment, it makes errors a lot easier to isolate, and also provides an easy 1 line way to run both the database and server..

Electron / Electron Webpack

Electron: Build cross platform desktop apps with JavaScript, HTML, and CSS

Modern web development practices today require a lot of setup with things like webpack to bundle your code, babel for transpiling, eslint for linting, and so much more that the list just goes on. Unfortunately when creating electron applications, all of that setup just became much more difficult. The primary aim of electron-webpack is to eliminate all preliminary setup with one simple install so you can get back to developing your application.

Our ideal application works across multiple platforms and can also be run natively. This is because we'd like to have an offline version later on in the project's lifecycle, and thus we wanted to start with using Electron early on in the development to not have to port it later.

Hapi / Joi / Boom

Hapi is a rich framework for building applications and services. It enables developers to focus on writing reusable application logic instead of spending time building infrastructure.

Joi is an object schema description language and validator for JavaScript objects.

Boom builds off of Hapi, providing HTTP-friendly error objects.

We chose to use this stack of 3 libraries to allow for an easy to set up and maintain backend server. Joi assists with verification of inputs while Boom allows us to give more friendly error messages as well.

TypeScript

TypeScript is a typed superset of JavaScript that compiles to plain JavaScript.

TypeScript seemed like a obvious choice while working with JavaScript, to ensure types for our objects.

TypeScript VS JavaScript

We chose to use TypeScript rather than JavaScript simply because we liked the structured nature that TypeScript provided, forcing us to stick to a standard which was, at the end of the day, easier to test.

TypeORM

TypeORM is an ORM that can run in NodeJS, Browser, Cordova, PhoneGap, Ionic, React Native, NativeScript, Expo, and Electron platforms and can be used with TypeScript and JavaScript (ES5, ES6, ES7, ES8). Its goal is to always support the latest JavaScript features and provide additional features that help you to develop any kind of application that uses databases - from small applications with a few tables to large scale enterprise applications with multiple databases.

TypeORM supports both Active Record and Data Mapper patterns, unlike all other JavaScript ORMs currently in existence, which means you can write high quality, loosely coupled, scalable, maintainable applications the most productive way.

We chose to use TypeORM to support our application because we found active record style database interactions like from SELT last semester were very helpful for maintaining our application.

Pure SQL vs TypeORM

Without TypeORM we would not be able to use active record style database interactions, which is what our team wanted to support. Pure SQL wasn't robust enough and was, in fact, limiting at times with a lot more effort than we desired. TypeORM extracted these burdens from us and seemed to be the better choice in the longrun.

(TS-)Jest

Jest is a delightful JavaScript Testing Framework with a focus on simplicity. It provides easy mocking, coverage reports, and built in tool options like snapshot testing.

We chose to use Jest because it integrates with Enzyme and allows for easy testing of our application. The snapshot tests are great for basic tests to start off with.

Enzyme

Enzyme is a JavaScript Testing utility for React that makes it easier to test your React Components' output. You can also manipulate, traverse, and in some ways simulate runtime given the output.

We chose to use Enzyme to test the actual React output of our application, allowing for more in-depth testing and interaction for the rendered components.

Nock

Nock can be used to test modules that perform HTTP requests in isolation.

We chose to use Nock as it was one of the few request mocking tools we could get to work with our system.

JSON Web Token (JWT)

JWT is an open standard (RFC 7519) that defines a compact and self-contained way for securely transmitting information between parties as a JSON object. This information can be verified and trusted because it is digitally signed.

We chose to use this for our user authentication as it allows for easy verification and also easy token expiration/logging out.

> Term Glossary

Alic Szecsei (@aszecsei)

Alic Szecsei (username @aszecsei) is the product owner for DM Tools. Alic has been running D&D games for several years, and is currently running a campaign during which this product will be tested and used. In addition, he is heavily involved in tabletop gameplay, discussing DM Tools with other Dungeon Masters to ensure that the tools provided are extensible enough to be useful not only to him, but others as well.

On a technical level, he's also experienced in web development and so is familiar with many of the risks and challenges associated with this sort of project.

Dungeons and Dragons

Dungeons & Dragons (abbreviated as D&D) is a fantasy tabletop role-playing game (RPG) originally designed by Gary Gygax and Dave Arneson. It was first published in 1974 by Tactical Studies Rules, Inc. (TSR).

The basic premise of D&D is fairly simple. The Dungeon Master (DM) provides a basic plot and general goal for the players to accomplish. Each player creates and takes on the persona of a single character. The players explain who their characters are, what they do, how they do it, and so on and so forth, while the DM explains how the world around them reacts to their choices. Without getting into the details of the game itself, D&D is essentially just a group of friends sitting together and telling a story together.

There are multiple editions of D&D that have been released so far. This project focuses on the most recent release, which is the 5th edition, also known as 5e.

Dungeon Master (DM)

The Dungeon Master (often abbreviated as DM) is the game organizer and narrator. They are in charge of creating the details and challenges of a campaign, while maintaining a realistic continuity of events. When the game is played, the DM describes an encounter and manages its progression. This includes revealing details about the locations, actions that each player can make, and adjusting for the conclusion based on character actions. They aren't out to 'win' the game, but rather ensure that everyone is having a fun time while they all craft a story together.

There is usually only one DM, but in the case of larger groups or newer DMs, there may be two per campaign.

Campaign

A campaign is a sequence or branching of encounters. Essentially, a campaign can be boiled down to the story that the DM and players are telling together.

Encounters

An encounter is an interaction between creatures and or people around a iconic location. Creating encounters requires combining cities, towns, buildings, people, and creatures in an interesting way.

Loot

Essentially, 'loot' is items that can be gained from a creature through various means. Creatures carry items, such as equipment, with them during encounters. If a creature is defeated, or otherwise distracted, players can attempt to take items from the creature. Thus, DM's need to know what items the creature is carrying with them, and as a result, what loot players can gain

Challenge Rating (CR)

Challenge rating is only a guidepost that indicates at what level that monster becomes an appropriate challenge. When putting together an encounter or adventure, especially at lower levels, the DM must exercise caution when using monsters whose challenge rating is higher than the players' levels. Such a creature might deal enough damage with a single action to overwhelm players of a lower level. Even though an ogre has a challenge rating of 2, for example, it can kill a 1st-level wizard or sorcerer outright with a single blow.

> Application Requirements

Required for Local Execution/Running

Node JS

Please install a version of Node JS at or above 11.7 from their download page.

You will see `Module not found: Error: Can't resolve 'worker_threads' in '\node_modules\write-file-atomic'` when you run the application if you have a Node version less than 11.7.

Yarn

To install yarn, please visit their download page.

Optional

Docker

Download Docker for your operating system from their download page.

> Poster Design