Wesley Couturier COMP – 525 11/13/2016
Nick Klardie

Homework 4 – Project Proposal

Dungeons & Dragons App

Motivation – Wesley Couturier

An old familiar game is making a rather huge comeback. Dungeons and Dragons is a Role-Playing game that involves team work, strategy, and a keen sense of what to do with the skills you have. The game has been around for quite sometime, and has been used as a tool to explore different narratives and ideas within the video game community and industry. We see that with this comeback, however, there still isn't an updated way to keep track of who is who, what are their abilities, and how well they can perform when certain situations arise. This is the problem we're attacking, getting rid of the pencil and the piece of paper.

Much like the twenty-sided die, it seems almost impossible to think about Dungeons and Dragons without a character sheet, or without some way of containing notes. It almost is integral to the whole experience. Luckily for the hardcore DND players, Nick and I have decided to tackle trying to update the way we play it, without taking away from the personal experience that is incredibly unique to DND. We want people to have the ability to seamlessly interact with their game, and help make the game run faster, by eliminating paper, having the dungeon master in truly in charge of everyone, and to make the game more connect to the players.

This app we are creating is for every Dungeon and Dragon player out there. It is made by people who love the game, so we understand the perspective of most players. But we can also make this app for the perspective of people who don't know how to play Dungeons and Dragons. We can guide them through a character sheet, and help them understand how the game works, interacts, and progresses throughout the course of a campaign.

Project Objectives – Wesley Couturier

Much of what this app will accomplish has already been described in the previous section. A majority of what we're trying to do is to bring Dungeons and Dragons into the modern age by eliminating paper, and to give full control to the Dungeon Master, a.k.a. the person who controls the story of the game.

In order to eliminate paper for dungeons and dragons, we're going to have to setup a streamlined database to client system where people can store, interact and record what their character is and what they can do. Along with containing the stats of each character, the interactions where the

characters take place will be controlled by the dungeon master, regardless of whether it's giving another character an item, or performing a set skill.

Along with our ability to contain all of the information and actions possible, we will have a help menu that describes accurately what each thing is, does, will do. This will be more or less for new players who don't know how to play the game.

Because of the social requirements of dungeons and dragons, we don't want to eliminate the interactions between people during the course of their campaign. So what we will have is a place where the dungeon master asks someone what they rolled for a specific action. There will be no electronic die for our app.

Approach – Nick Klardie

To meet these objectives, we will use the full capabilities of Django to control and serve content dynamically, as well as the same capabilities of other frameworks. We intend to serve content in an appropriate and efficient form from the database, and modify the database in a similar way. Django provides simple and powerful database abstractions which we intend to take full advantage of. We intend to use such tools as foreign keys and many-to-many fields to avoid repeating data wherever possible.

The front-end will take advantage of Django's powerful template system as well as AngularJS to display and collect data in a simple, general, and dynamic way. It is my personal hope that we can take full advantage of the AngularJS's powerful client-side capabilities to make our templates as simple and straightforward as possible. Wherever data needs to be exchanged between the server and scripting on the front-end, it will be done through regular HTTP requests exchanging JSON data, which is easy to read and easy to work with.

Preliminary Results – Nick Klardie

Our preliminary version of the app displays information about a character stored in the database in a simple HTML table. The only url in the preliminary version that the user is intended to see is `char/view_char` which uses a Django template to display most of the information in the database currently stored for a character. The test character currently in the database can be accessed at `char/view_char?cid=1`. For this page, the relevant data from the database along with some rules related information and the player's level, which is calculated on the spot, is put into the template's context. Notably absent from the page is the character's proficiencies, which are currently stored as a 36-bit bitmask in the database, with the first half representing skills the character is proficient in and

the other half representing skills the character has expertise in. This will most likely be converted to CSV format to simplify use with Django's templates.

In addition to the view_char page, this app currently has a JSON interface at `/char/dump`, which dumps full lines from the database. The test values in the database can be accessed at `/char/dump?cid=1`, `/char/dump?rid=1`, `/char/dump?picd=1`, and `/char/dump?bid=1`. The models behind these pages currently contain only the bare minimum needed to display. For everything but the character model, this is just the name. For the character this is just the character's name, class, background, race, alignment, experience, attribute values, HP, the proficiencies mask mentioned earlier, and miscellaneous notes. The entries for class, background, and race are foreign keys to the other tables. This is more or less the absolute bare minimum needed to represent a character, as the other values can be calculated or referenced from other values. Because it would be helpful for the user, more fields will be put into the database as the project advances. This JSON interface will be used in later versions to power the front-end.

All logic for the template is done in `views.py` before the template is processed, so it is only putting variables in the right spot in the page, except for one variable which it is passing through a humanize filter. The page is a very simple HTML file with a few variables. Later versions will feature fancy CSS and JavaScript to spice it up. As much as possible, logic will be separated from the templates, since they are not really intended to process anything, and as such it difficult to do any real processing in the template. That is the reason the character's level is determined before the template is rendered, because the only way to figure it out and display it dynamically within the template would be to dynamically insert HTML comments around the character's previous levels.

Although our vision for the completed app includes the ability for the user to input and modify information, our preliminary version does not include such functionality. I think this simplified version, even though it is missing features that will be critical to the final product, is good enough at least for a quick preliminary version for demo purposes. Personally, I would have liked to have a rudimentary data entry system for demonstration purposes, but it proved to be too much work to complete this week.

Preliminary Evaluation – Wesley Couturier

Even though the project is in a very rudimentary stage right now, we have not only the ability to store our character information, but extract it in a very nice JSON array. We have our virtual environment setup, Nick understands how we're going to approach our framework, and I understand how we're going to approach the app from an end-user experience.

What we don't have right now is a populated database full of example characters, as well as a layout for a front end. We also don't have functionality for the Dungeon Master. A rather large majority of our code can be carried over into the front end thanks to the abilities of AngularJS. You can

essentially perform server side code into the front end. It might not be the best practice, however for the time being, it seems appropriate.

Nick will continue to grow the back end into what it needs to be. I will be handling a lot of the front end work, as well as the main app logic for the client in AngularJS. Because Nick has done so well, I have to say that he's really propelled this project forward to a point that it needs to be for me to do the best of my ability.

We will keep you updated as to how we're doing, as well as how the project is coming along. Nick and I are pretty self-sufficient, so if we need any help, we know where to go to in case our resources don't give us the answers we need.