SLIDES

Andrew:

Slide 1:

Hello and welcome to our presentation on our Final Project, called Dungeons and Discourse. My name is Andrew Manz, and I will be introducing our project. You will be hearing from my constituents Jason Jellie, Melina Gutierrez, and Scott Kurtz during this presentation.

Slide 2:

To introduce our project, I would like to start with a quote: “It simply isn’t an adventure worth telling if there aren’t any dragons”, attributed to J.R.R Tolkien. While we never had any run-ins with actual dragons during our development of this project, we do feel that the nature of this project ties intrinsically to the concept of fantasy, and it would be remiss not to address that fact going in.

Slide 3:

Our presentation today is divided into four chapters: The idea, The development, Lessons Learned, and finally future plans.

Slide 4:

We begin with the idea. Your Unreliable Narrator.

Slide 5:

So what is DnD Message time? Well, Dnd Message time is a chat application built specifically to be a companion to the tabletop role-playing game Dungeons and Dragons, but can be used with other systems as well, such systems such as Pathfinder, Cyberpunk Red, or Shadowrun as an example. This chat messenger is meant to be used in conjunction with other systems that can assist in creating an online RPG environment, such as Discord and Roll20. This application adds the ability for characters within the game to speak to each other in canon, as if they naturally had the ability to access similar abilities within the game’s canon.

Slide 6:

So why did we make this? Well, the idea for this project was initially created as a supplement to a Dungeon and Dragons campaign being created by the project lead. The specifics of this campaign are still in the works, but one of the major differences between a traditional DnD campaign is the fact it is set in the present, as opposed to a fantasy medieval setting. Characters in said campaign would have access to and be able to communicate across the internet using a messenger application. The application we've created is largely purpose-built to serve this purpose.

Slide 7:

The features of the messenger include the main chat messenger, allowing interperson communication. It includes admin privileges to give the game master extra functionality compared to the players, it includes a built-in Dice roller, allowing for a large number of common tabletop dice combinations such as rolling a d20 for attack rolls. The application also supports the ability to run multiple games across the same server. As a part of the admin functionality, you can scramble text in order to obfuscate the meaning of a message, which can be very interesting in a role-playing environment. Finally, the application has a number of built-in emotes accessible through the messenger through chat commands.

Jason:

Slide 8:

[Transition from previous speaker]

Slide 9:

This is a timeline of some important moments of our project. In the first semester we created a simple forum style chat site to test communication between users over the internet. We also commissioned our emotes and built working prototypes of the dice roller and message scrambler. After winter break we transitioned our server over to the CSUB computer labs server "Bender" which in hindsight wasn't a good idea. Restrictions on file editing on the server massively slowed down progress. To make matters worse, an update to the server caused us to lose practically all of the progress we had made since winter break. Because of this, we decided to separate from Bender and start again, which was for the best, as in only a few short weeks we were able to not only make up for our lost progress but make more progress than we had while on bender.

Slide 10:

Melina:

Slide 11:

Now let's talk about some of the lessons we learned along the way and the huge shift that came with implementing what we learned, as well as exactly what Operation 'Resetti-Spaghetti' was.

Slide 12:

Our original plan was to host the entire project on a departmental server and we were fully configured in our project's original state, however the delays that occured to get there wasted precious time. We needed to make edits to configure the nginx file for our server to perform correctly and we needed basically admin permissions to do so. Once we had gotten set up, there was an issue outside of our control that caused our progress to be wiped and deleted. While we could recover the files, the admin level changes that would need to be done again would have too large a turnaround time to continue in this direction. Due to the setback, we felt it'd be in our best interest to restart with our growth in knowledge. And thus the phase given the codename Operation Resetti Spaghetti was born. This phase would be the most critical for the development of the project.

Slide 13:

We couldn't have the server hosted outside of our own hands anymore. We swapped to hosting the server using an Amazon Web Services node instance which cut down turnaround time almost entirely. Permission issues were also solved as it was now owned within the group. With this change, we needed to change the way the database would be hosted. We decided to swap to using Firebase's Realtime Database hosted by one of the group members as well. Since we used to be working on a previous framework that 3 of the 4 members worked on before this course, we decided to scrap it and start fresh with one we'd no longer have to edit and refactor to fit our needs.

Our largest change was implementing our new "Block System" as a work structure. We distributed pieces of work into a series of blocks for the members and the goal was to complete each block before moving to a new level. The levels for the features in these blocks were assigned mandatory, critical features, and polish to organize tasks by importance.

Slide 14:

Thanks to the block system we were able to successfully rebuild from the ground up. Also a large thanks to some guidance from tutorials and too much caffeine, we had an even more clear and successful framework to develop from over the course of an insane amount of hours spent in calls and developing over the course of six days. Below we have a couple of the blocks we used for this phase of production. Now that we've seen some of the successes, let's see and hear some of this project in action.

DEMOS

Andrew:

To begin, after you login you are dropped upon this opening page this opening page has a sidebar on the left hand side and a number of tabs on the top that include "Conversations," "Contacts," "Dice," and, if you have admin functionality, "Scrambler" as well. We will talk specifically in my part about the Contacts and the Conversations. When a new user joins the room their contact is automatically added to your contacts list, this is basically your log book of people that you're able to message within this application. In order to do so, you come over to the conversations window, you go down to the bottom and say create new conversation and it creates a prompt for you to add people to this conversation you can add one or multiple. After you hit create, the newly created conversation will open in the window you can then type your message to the other person and they will receive it on their end. Every single time a new user joins the game, their username will automatically be added to everyone's contact list. In most cases this functionality works without a hitch. However, if there is a case where one of your users is missing from your contacts list, you can either ask them to refresh their page and it will automatically send across the network, or if that does not work, you can manually add them using the new contact button down at the bottom left hand side of the contacts list. At the bottom left hand side of the page you will see your ID, which is the username you are broadcasting across the network and the game ID which is the game you are sending your messages in. You can also see, next to your id, a logout button which will clear your local storage and send you back to the login menu.

Jason:

[Transition from previous speaker]

The dice roller allows you to easily roll combinations of different sided dice. You can also apply a modifier to your roll and describe it to help keep track of what each roll was for. To roll the dice all you have to do is press the roll dice button, choose how many of each dice you want to roll, optionally describe the roll or apply a modifier, and then press roll. Your roll will appear on the sidebar and will give details on the roll and its outcome. At the top will be the description of the roll if you gave it one, followed by what dice were rolled. The first number is the number of that dice rolled and the second number is how many sides that dice had. Under that, each dice is shown along with the outcome of the individual rolls of each dice, so in this case a D6 was rolled 3 times and the outcome of those rolls were [READ OUTCOME OF D6 ROLLS]. Next is the modifier, and finally the total sum of the roll including the modifier.

Message scrambling can be done by pressing the button next to the send message button. Pressing this button allows you to choose the difficulty of the scramble along with how scrambled you want the message. The difficulty is broken up into three difficulties. Easy scrambles the sentence but keeps letters within their own word, medium allows letters to be swapped with any other letter in the sentence, and instead of rearranging letters hard replaces letters with a random one, making it almost impossible to decipher at higher percentages. As an example I'll scramble this part of my script using the medium difficulty at twenty-five percent. I chose a relatively low percentage so the message is still possible to decode without being too difficult. Now you don't have to worry about your emotes being lost in the scrambling. The system detects emotes and removes them before the message is scrambled. This way, the person you're talking to may not understand what you said, but they can still tell you were excited to say it.

Melina:

Let's look into Dungeons and Discourse's emotes a bit more. As Jason mentioned, while using a chat messenger, it can be difficult to get the full point across, and even more so if a message is scrambled, so we've implemented some special text-based emotes specifically for Dungeons and Discourse. Our emotes can help as context clues for important messages or as a simple way to communicate through images quickly. Since we've decided to base our project as a complement to Dungeons and Dragons, we wanted to pay respect to the original aesthetics. We reached out to an independent artist and commissioned them for a beginning set of 7 custom emotes total, all using the red color scheme you'll see throughout the messenger. Each of the emotes are based on a different race in Dungeons and Dragons, and, if this were a larger scale project with a legitimate budget, we would love to incorporate at least one emote featuring each of the races.

The emotes can be used in a markdown style by surrounding the keyword in colons. The emotes will appear as a standalone image, as seen here with the Pog emote, and bring more personality to the chat. Emotes can be shown using multiple names, in case the player forgets one name for it. With this in mind, there could be an easter egg emote or two that the player could discover just by guessing and checking.

Slide 15:

Scott: Now that Andrew, Melina, and Jason have brought you through our production process and demonstrated to you what our website looks like, and how our application works, I am going to demonstrate to you how our Unreliable Narrator application is compatible with speech reading software and is, therefore, accessible for the blind population. So I have pre-loaded our website.

JAWS Unreliable Narrator v3

Scott: you can see (should have been hear) that we have the unreliable narrator website up and there are 4 options.

JAWS: Conversations tab selected, contacts tab, dice tab, scrambler tab

Scott: now I am going to show you what a conversation looks like

JAWS: unrel conversations tab selected

Scott: so in the conversations tab I am just going to down arrow here.

JAWS: Angie, Steph button Steph buttonJohn, Steph button

Scott: so these are different buttons I can click on for conversations

JAWS: John button unlabeled 4 button your id: blank game ID: 1 new conversation button

Scott: so I am going to select a new conversation

JAWS: enter unreliable narrator game ID: 1 new conversation button enter create conversation close Angie, Scott, Steph, John create conversation close button

Scott: so we are going to create a conversation and I am going to put check marks on the people I want in the conversation

JAWS: Angie check box not checked

Scott: I am going to check her.

JAWS: Angie check box checked Scott Steph check box no checked

Scott: I'll check Steph

JAWS: space Steph check box checked to clear check mark press space bar period John check box to create

Scott so I will create it

JAWS: enter new conversation button to activate

Scott: so I am going to press the 'e' button and that will take me to the edit box

JAWS: edit required invalid entry

Scott: So now I am going to type Let's attack

JAWS: L e t s a t t a c k

Scott: okay, then I am just going to press tab

JAWS: send button to activate press enter period

Scott: I am going to the send button and I am going to send it

JAWS: enter

Scott: and so now

JAWS: send button required invalid entry edit

Scott: it has been sent

JAWS: youLets attack

Scott: and lets attack, I went uparrowed and that is where I heard the message that I sent and so now I can send a new message if I just tab

JAWS: type message

Scott: and that is basically how the conversations tab works.

Slide 16:

Scott:

I am sorry if that was a little hard on your ears. My wife requires that I wear my headphones whenever I use the computer around her. We thought about going through all of the options with you, but in the interest of time we will move on to our next slide and discuss our future plans. Although we feel that we have developed a strong core for our application there is room for improvement. The areas we could work on include developing multiple formats, adding to admins abilities, broadening our database system and strengthening our security.

In order to reach a broader audience we need to develop code so our format can fit on mobile phones and tablets. In our applications current form, it would overwhelm our users with information. In regards to our administrator, we would like to give the DM the ability to observe all of the conversations occurring among the players. Our database is currently sufficient to meet the needs of our program but it may be fun to add more information to the system, especially in the area of non player characters with stats and items. Additionally, we need to implement more security into our app. Unreliable Narrator is an application for fun and there really should not be any extremely sensitive information being shared across our platform, so security was placed a little lower on our list of priorities. However, we don't need annoying hackers working their way into our games, so we need to implement some hashing code and code for validation. Thank you for choosing to check out our presentation for Dungeons and Discourse - Unreliable Narrator application. Please feel free to reach out to us if you have any additional questions. Thank you.