Team 22 : Memoji

Sprint 2 Planning Doc

Team Members

Scrum Master - Jonathan Poholarz (jpoholar@purdue.edu)
Maxwell Jones (jone1268@purdue.edu)
Manoj Polisetti (mpoliset@purdue.edu)
Andrew Ring (amring@purdue.edu)
Delun Shi (shi272@purdue.edu)

Overview

Since our first sprint focused heavily on creating our networking architecture, we now have much of the code in place for user connection and disconnection to hosted game lobbies. For this next sprint, we will be now focusing on implementing much of the gameplay itself. This will include creating a significant amount of user interface screens as well as implementing the necessary logic for them to work. We will be able to use our networking put in place in sprint 1 to communicate the game information among players, such as the prompts they will be responding to and the answers they will be voting on. Additionally, we will be implementing the interface for placing emojis into the screen which is a core part of our game's hook.

A notable challenge this sprint will be securing a cloud hosted server for testing our gameplay across multiple devices. Moving our server from being locally hosted onto the cloud could result in unforeseen problems that our original networking code did not account for, but it will be hard to plan exactly for these unknowns.

We do not anticipate any major risks this sprint besides this. Since our team is fairly familiar with the workflow in Godot at this point, much of our work will be fairly similar to what was encountered in the first sprint.

Meeting Plan: Wednesday (6:00 pm-7:30 pm), Friday (3:00 pm-6:00 pm)

Sprint Details

Sprint Story #1 (F2) - As a host, I would like to begin the game once enough players have joined the lobby			
#	**Task**	**Time**	**Owner**
1	Enable functionality of “start game” button(s)	1hr	Maxwell
2	Implement a check for if there are enough players joined to start a game.	1hr	Maxwell
3	Implement check for if there are any players connected but have not sent avatar and username.	2hr	Maxwell
4	Implement sending message to all players and audience connected that the game is starting.	3hr	Maxwell
5	Test that a game may only be started with enough players	1hr	Maxwell
6	Test that all players have selected an avatar and username	1hr	Maxwell
7	Test that starting a game moves all participating users to the proper game state and screen	1hr	Maxwell
#	**Acceptance Criteria**		
1	Given that the Host is running, when enough Players join the game, then the Host will be able to start a game.		
2	Given that that there are enough Players joined to a Host, when the Host clicks the start button, then the Host will notify all Players that the game is starting.		
3	Given that the Host is ready to play a game, when the Host clicks the start button, then the Host transitions screens.		
4	Given that the Host started a game, when the Host notifies all Players that the game is starting, then the Players will progress to the waiting screen until prompts are received.		

Sprint Story #2 (F4) - As a host, I would like to display a waiting screen while players are inputting answers to prompts			
#	**Task**	**Time**	**Owner**
1	Design the Host prompt Waiting screen.	4hr	Maxwell

2	Implement transitioning to and away from screen based on game state (in ScreenManager).	4hr	Maxwell
3	Create scene for a prompt object to store the prompt text, responses, and votes.	1hr	Jonathan
4	Create PromptGenerator to randomly select prompts from an external file and track which prompts have already been used to prevent duplicates.	4hr	Jonathan
5	Implement sending prompts to all Players	3hr	Maxwell
6	Implement timer for waiting for responses	2hr	Maxwell
7	Test that screen is implemented correctly.	1hr	Maxwell
8	Test that the screen transitions correctly.	1hr	Maxwell
9	Test that the Host sends all prompts correctly.	2hr	Maxwell
10	Test that prompts generated are unique.	2hr	Jonathan
11	Test that each prompt is only sent to two players.	1hr	Maxwell
12	Test that prompts are handled and distributed for both odd and even numbers of players correctly.	1hr	Maxwell

#	**Acceptance Criteria**
1	Given that there is a game currently running, when the Host sends prompts to all the players, then the Host will display the Waiting screen.
2	Given that the Host is on the Waiting screen, when the Host receives answers but not all answers from the Players and the response period has not ended, then the Host will stay on the Waiting screen.
3	Given that the Host is on the Waiting screen, when the Host receives answers but not all answers from the Players and the response period has ended, then the Host will transition screens and force all Players to the next screen.
4	Given that the Host is on the Waiting screen, when the Host receives all answers from the Players, then the Host will transition screens and force all Players to the next screen.
5	Given that the Host is on the Waiting screen, when the Host is going to send prompts, then the Host will read a prompts file and obtain random prompts based how many players are joined.

Sprint Story #3 (F20) - As a player, I would like to view a prompt response screen on my device for each assigned prompt in each round			
#	**Task**	**Time**	**Owner**
1	Design the Player Prompt screen	2hr	Jonathan
2	Implement functionality to receive the Prompt from the Server using the function in GameStateManager	1hr	Jonathan
3	Implement functionality to Display the Prompt to the Player via GameStateManager and ScreenManager	1hr	Jonathan
4	Implement advancing of prompts to the next available prompt, if it exists, via some sort of Next/Submit button	1hr	Jonathan
5	Implement sending completed prompts back to the Server via a call to Networking from the GameStateManager node	1hr	Jonathan
6	Insert EmojiCanvas scene into prompt screen	0.5hr	Jonathan
7	Implement transition to waiting screen after sending prompts	0.5hr	Jonathan
8	Test that prompts are displayed properly on the Player screens	1hr	Jonathan
9	Test that the screen advances to the next prompt if there is one still unanswered	1hr	Jonathan
#	**Acceptance Criteria**		
1	Given that the game is in the prompt response phase of the round, when the Host sends the Player the appropriate message, then the Player transitions into the prompt response screen.		
2	Given the Host has sent the Player a set of prompts, when the Player transitions to the prompt response screen, then the Prompt will be displayed in a textbox.		
3	Given that a Player has created an emoji answer to a prompt, when the Player clicks the next button, the screen will display the next prompt and a fresh canvas to answer with.		
4	Given that a Player is answering a prompt, when the Player presses the submission button, then the Player will progress to the Waiting screen.		
5	Given that a Player has answered a prompt, when the Player presses the submission button, then the Player will send their submissions to the Host via a server.		

Sprint Story #4 (F23) - As a player, I would like to input emojis in my responses to prompts easily via a straightforward GUI			
#	**Task**	**Time**	**Owner**
1	Create reusable EmojiCanvasContainer to read a set of emojis and display them	2hr	Jonathan
2	Create dictionary of emojis and their corresponding ids	3hr	Jonathan
3	Create EmojiCanvas users can interact with	2hr	Jonathan
4	Create UI for the emoji dock containing emojis sorted by category	4hr	Jonathan
5	Implement serialization of emojis into an array that can be sent to/from the Server and displayed in an EmojiCanvasContainer	2hr	Jonathan
6	Implement de-serialization of the emojis back into a form that can be used by an EmojiCanvas to display them	2hr	Jonathan
7	Test that an EmojiCanvasContainer can properly serialize and un-serialize an encoded set of emojis	2hr	Jonathan
8	Test that the emoji dock contains all of the available emojis	1hr	Jonathan
#	**Acceptance Criteria**		
1	Given that a Player is answering a prompt, when the Player views the emoji dock, then the dock displays all of the emojis which may be placed onto the canvas.		
2	Given that a Player is answering a prompt, when the Player finishes answering, then the placed emojis can be serialized into a form for transmission to the Server or to another EmojiCanvas in the application		
3	Given that a screen contains an EmojiCanvasContainer, when the container is given an encoded set of emojis, then it can decode the emojis and display them in the right positions of its canvas.		
4	Given a screen contains an EmojiCanvasEditor, when the EmojiCanvasEditor is clicked on, then the emoji dock becomes visible to insert emojis into the canvas.		
5	Given a screen needs to display an emoji, when the screen looks up the emoji in the dictionary, then the screen can successfully retrieve the image it should display based on the emoji id.		

Sprint Story #5 (F24) - As a player, I would like to edit my response before submission - adding, moving, or deleting emojis with the GUI			
#	**Task**	**Time**	**Owner**
1	Implement ability to select emojis and put them in the canvas grid	3hr	Jonathan
2	Implement ability to re-position emojis on the canvas.	2hr	Jonathan
3	Implement ability to remove emojis from the canvas.	2hr	Jonathan
4	Test that an emoji may be placed on the canvas once selected from the emoji dock	1hr	Jonathan
5	Test that multiple copies of the same emoji may be placed in the canvas without re-selecting the emoji from the dock	1hr	Jonathan
6	Test that an emoji may be re-positioned once placed	1hr	Jonathan
7	Test that emoji can be removed once placed on the canvas	1hr	Jonathan
#	**Acceptance Criteria**		
1	Given that a Player is answering a prompt, when the Player selects an emoji to add to the canvas, then the selected emoji can be added to the canvas.		
2	Given that a Player is answering a prompt, when the Player selects an emoji to add to the canvas, then multiple clicks on the canvas will insert multiple copies of the emoji.		
3	Given that a Player is answering a prompt, when the Player selects an emoji on the canvas, then the Player may move the emoji to another location on the canvas.		
4	Given that a Player is answering a prompt, when the Player selects an emoji on the canvas, then the Player can delete the emoji.		

Sprint Story #6 (F22) - As a player, I would like to view a waiting screen on my device in-between tasks in which the game requires no further inputs from me			
#	**Task**	**Time**	**Owner**
1	Design Player Waiting screen UI	1hr	Delun
2	Implement transitioning to screen in ScreenManager after the Prompt screen or the voting screen sends a completion signal	1hr	Delun
3	Implement transitioning from screen back to gameplay in GameStateManager when the appropriate signals are processed	1hr	Delun
4	Implement tracking of the current round and state of the game using variables in GameStateManager	1hr	Delun
5	Test that the waiting screen is reached after a prompt round or voting round is completed by the user	1hr	Delun
6	Test that the current round and state variables remain consistent with the game's actual progress	1hr	Delun
#	**Acceptance Criteria**		
1	Given that a player is responding to a set of prompts, when the Player finishes their last prompt, the screen will redirect them to a waiting screen		
2	Given that a player is on the waiting screen, when the game proceeds to a prompt answering or voting stage, then the screen advances out of the waiting screen		
3	Given that a player is on the waiting screen, when the server sends a message to change the game state, then the message is handled by the client and processed appropriately (such as changing the screen and updating the game state variables)		

Sprint Story #7 (F29) - As a player, I would like to view visual confirmation on the host screen when my inputs and votes are properly received			
#	**Task**	**Time**	**Owner**
1	Implement assignment of prompt answers to a data structure on the Host	4hr	Delun
2	Implement visual change in the Host Prompt Waiting screen which changes based on each user's response status	5hr	Delun
3	Implement message handler code in GameStateManager to process receiving of prompt answers	4hr	Delun
4	Test that the host screen correctly displays who has voted correctly	2hr	Delun
#	**Acceptance Criteria**		
1	Given that a Player is answering a set of prompts, when the Player completes his last prompt, then it sends a message to the Server which is delivered to the Host and received correctly.		
2	Given that a Player has answered a set of prompts, when the Host receives a message containing the Player's responses, then the responses are stored with the prompt on the Host machine.		
3	Given that a Host has received a message containing a Player's responses, when that message is processed, then a visual change will occur on the Host waiting screen to confirm that the Player's responses have been received successfully		

Sprint Story #8 (F21) - As a player, I would like to view a voting screen on my device for each round of prompt voting			
#	**Task**	**Time**	**Owner**
1	Design Player Voting screen with boxes for the prompt text and possible voting choices	4hr	Manoj
2	Design alternate Player Voting screen for players who answered a given prompt	4hr	Manoj
3	Design Host Voting screen which displays the prompt and the two choices.	2hr	Delun
4	Implement Host making a call in GameStateManager to Networking for each prompt which sends the prompt text	5hr	Delun

	and the voting options		
5	Implement message handler in Player GameStateManager which inserts the prompt text and voting options to the voting screen	3hr	Manoj
6	Implement Player screen transition when voting options are received	3hr	Manoj
7	Implement Host processing vote received message in Host GameStateManager	6hr	Delun
8	Implement Host transitioning to the voting results stage after all votes are received	5hr	Delun
9	Test that the correct GUI is displayed on the Player and Host during prompt voting	2hr	Manoj/Delun
10	Test that the Player receives and displays prompt responses correctly	2hr	Manoj/Delun

#	**Acceptance Criteria**
1	Given that the player has received the options for the prompt text, design the page to be able to choose between the two options to vote for the option
2	Given that a player has picked an option, implement functionality to direct them to the waiting screen and design the waiting page
3	Given that the round has started,when the prompt and voting options are received from the server, then make sure the correct prompt and the options are received for that round
4	Given that the round has started, when the prompt and the options are received, then redirect the player to the voting page through the GameStateManager
5	Given that the voting for an option is underway, make sure that the host and the player are on the same round with the correct page
6	Given that the game is running, when the voting phase begins, then the Host displays the voting screen, showing a prompt and 2 answers to be voted on.
7	Given that the game is running, when the voting phase begins, then the Host sends the appropriate messages and prompt information to the Players.
8	Given that the game is in the voting phase, when the Host receives all the votes from Players for a given prompt, then it will store the results and transition to the voting results stage.

Sprint Story #9 (F25) - As a player, I would like to vote for answers to the prompts using the GUI			
#	**Task**	**Time**	**Owner**
1	Implement voting functionality to Voting Player screen	4hr	Manoj
2	Implement message sending in GameStateManager to transfer vote information back to the Host	3hr	Manoj
3	Implement screen transition in ScreenManager back to a waiting screen once voting has been completed	4hr	Manoj
4	Test that voting options are received from the host	3hr	Manoj
5	Test that the voting options are displayed correctly	3hr	Manoj
6	Test that options may be selected	3hr	Manoj
7	Test that options selected and sent back to the Host match the Player's selection	3hr	Manoj
#	**Acceptance Criteria**		
1	Given that the Players are voting, when the Host sends the voting options, then the options are displayed for the Player.		
2	Given that the Player is voting, when the Player finishes voting, then the Player's screen transitions to the Waiting screen.		
3	Given that a Player is voting, when a Player casts a vote, then the vote is sent to the server and forwarded to the Host.		
4	Given that the current round is Voting, when the Host sends voting options to the Players, then the correct voting options will be displayed on the Players' screen.		
5	Given that the Player is trying to submit voting for an option, when they try to click on an option, then make sure that only one option can be submitted by the Player		
6	Given that the Player has selected an option to vote, when the submit button is pressed, then make sure the vote is the one selected by the Player and is sent to the Host		

Sprint Story #10 (F6) - As a host, I would like to display a results screen for each prompt of each round			
Sprint Story #11 (F27) - As a player, I would like to view the results of the votes on the host screen			
Note: These stories were merged because they overlap almost entirely, and we did not realize this when initially planning!			
#	**Task**	**Time**	**Owner**
1	Design Host Results screen.	6hr	Andrew
2	Implement Player transition to waiting screen after voting completed	1hr	Andrew
3	Implement scoring methodology to allocate points based on the percentage of votes received	4hr	Andrew
4	Implement awarding players scores based on voting results, updating the score tally in GameStateManager	4 hr	Andrew
5	Implement graphical representation of which players vote for which prompt	5hr	Andrew
6	Implement graphical representation of how many audience members voted for each prompt	1hr	Andrew
7	Test that the results screen is implemented correctly	1hr	Andrew
8	Test that votes are properly received from players	1hr	Andrew
9	Test that votes are scored correctly	1hr	Andrew
10	Test that players that voted are correctly displayed	1hr	Andrew
#	**Acceptance Criteria**		
1	Given that the game is in a voting round on a prompt, when a voting round ends, the host will display the results screen.		
2	Given that a Player is voting on a prompt, when the Player submits their vote, then the Player's screen transitions back to a waiting screen.		
3	Given that a Player is voting on a prompt, when the Player submits their vote, then the vote information is sent to the Host.		
4	Given that a Player has submitted their vote, when the vote is received by the Host, then the vote is tallied for the prompt's results.		

5	Given that Players have voted, when the results screen is displayed, then Players that voted on each response are displayed matching the response they voted for.
6	Given that Audience members have voted, when the results screen is displayed, then the Audience fraction that voted for each response is displayed matching the responses they voted for.

Sprint Story #12 (F7) - As a host, I would like to display a total results screen at the end of each round			
Sprint Story #13 (F28) - As a player, I would like to view round summaries on the host screen			
Note: These stories were merged because they overlap almost entirely, and we did not realize this when initially planning!			
#	**Task**	**Time**	**Owner**
1	Implement screen transition to waiting screen on the Player device while the host displays a summary of scores	1hr	Andrew
2	Design Host Total Results screen.	6hr	Andrew
3	Implement screen transition in ScreenManager once all vote results to this round's prompts have been shown	2hr	Andrew
4	Implement displaying the ranking of players with correct scores	5hr	Andrew
5	Test displaying of correct players with correct scores	1hr	Andrew
6	Test that screen is properly implemented	1hr	Andrew
#	**Acceptance Criteria**		
1	Given the round has ended, and the Host advances the game state to the result screen, then the Player will display a waiting screen on their device		
2	Given that a game is running, when a round of prompts is completed, the total results screen is displayed.		
3	Given that a round of prompts is completed, when the total results screen is displayed, the scores of players are correctly displayed.		
4	Given that a round of prompts is completed, when the total results screen is displayed, players are ranked from highest score to lowest		

Sprint Story #14 (NF9) - As a developer, I would like to host a server in the cloud.			
#	**Task**	**Time**	**Owner**
1	Migrate Node.JS server to Firebase.	5hr	Maxwell
2	Implement Node.JS clusters to make the server robust and stable when error handling.	2hr	Maxwell
3	Implement server Database to store lobby codes, hosts, players, and audience objects.	3hr	Maxwell
4	Update host and port values to connect to Firebase	1hr	Maxwell
5	Test that the Server restarts properly on an error that would crash it.	1hr	Maxwell
6	Test that the Database stores data queried from the Server correctly.	1hr	Maxwell
7	Test that the Server can query data from the Database correctly.	1hr	Maxwell
#	**Acceptance Criteria**		
1	Given that the Server is migrated to Firebase, when a client tries to connect (not on localhost), then the client successfully connects.		
2	Given that the Server is running, when an error occurs on the Server, then the Server will be able to restart itself after the error and continue running as normal.		
3	Given that the Server is running, when the Server receives or creates new data that needs to be stored, then that data is saved on the Database.		
4	Given that the Server is running, when the Server needs to use data, then the Server can pull the data from the Database and convert it into a form that it can use.		
5	Given that the Server crashes from an error, when the Server restarts, then the Database has not lost any information that was in it before the Server crash		
6	Given that the Server crashes, when the Server restarts, then the server will pull all the data it needs from the Database.		

Claimed Hour Totals For Sprint 2	
Teammate	**Hours**
Jonathan Poholarz	45
Maxwell Jones	43
Manoj Polisetti	41
Andrew Ring	41
Delun Shi	42

Backlog

(Stories from last sprint)
(Current Stores for this sprint)

Functional Requirements:

1. **As a host**, I would like to create a game lobby and display the lobby's server generated letter code on screen
2. **As a host**, I would like to begin the game once enough players have joined the lobby
3. **As a host**, I would like to display a lobby screen while waiting for players to join
4. **As a host**, I would like to display a waiting screen while players are inputting answers to prompts
5. **As a host**, I would like to ignore/skip players who have not submitted answers to prompts within the allotted time and continue the game
6. **As a host**, I would like to display a results screen for each prompt of each round
7. **As a host**, I would like to display a total results screen at the end of each round
8. **As a host**, I would like to display a final results screen and winner/credits screen after the game completes
9. **As a host**, I would like to play additional games after the first game without forcing all players to disconnect and reconnect to a new lobby
10. **As a host**, I would like to exit the game properly, disconnecting all players and blocking further connections to the defunct game lobby
11. **As a host**, I would like to allow audience members to join at any time after the main players have connected
12. **As a host**, I would like to set up additional game modes (if time permits)
13. **As a host**, I would like to hear sound effects throughout the game (if time permits)
14. **As a host**, I would like to experience animations that add a personality to the game (if time permits)
15. **As a host**, I would like to host the game via a web-player (if time permits)

16. **As a player**, I would like to connect to a game lobby using my phone through the game application
17. **As a player**, I would like to choose my in-game username and avatar emoji
18. **As a player**, I would like to quit the game in-between rounds if desired, releasing my spot in the player list for a new player to join
19. **As a player**, I would like to receive instructions and prompts from the host computer screen during play
20. **As a player**, I would like to view a prompt response screen on my device for each assigned prompt in each round
21. **As a player**, I would like to view a voting screen on my device for each round of prompt voting
22. **As a player**, I would like to view a waiting screen on my device in-between tasks in which the game requires no further inputs from me
23. **As a player**, I would like to input emojis in my responses to prompts easily via a straightforward GUI
24. **As a player**, I would like to edit my response before submission - adding, moving, or deleting emojis with the GUI
25. **As a player**, I would like to vote for answers to the prompts using the GUI
26. **As a player**, I would like to select multiple answers during the final round when needed
27. **As a player**, I would like to view the results of the votes on the host screen
28. **As a player**, I would like to view round summaries on the host screen
29. **As a player**, I would like to view visual confirmation on the host screen when my inputs and votes are properly received
30. **As a player**, I would like to reconnect to a running game after disconnecting and be able to continue playing as the same player with the same username and same avatar
31. **As a player**, I would like to experience a diverse set of prompts such that at least two consecutive games have little overlap in the prompts chosen
32. **As a player**, I would like to view the remaining time for a question on the host screen
33. **As a player**, I would like to view a synced-with-the-host remaining time counter during answer and voting rounds on my device (if time permits)
34. **As a player**, I would like to connect to the game via a web-player (if time permits)

35. **As a developer**, I would like to maintain a server for which host computers can obtain a letter code for their game lobbies
36. **As a developer**, I would like to maintain a server to facilitate communication between the host computer and player devices in each lobby
37. **As a developer**, I would like to maintain a server which will connect player phones to the correct host computer lobby for each respective letter code
38. **As a developer**, I would like to remove inactive/unresponsive lobbies from the table of hosts on the server
39. **As a developer**, I would like to handle mid-game disconnects such that players can resume their games
40. **As a developer**, I would like to view crash messages in case of errors when possible

Nonfunctional Requirements:

1. Code should be well organized and commented where necessary.
2. Scalability for each individual game session should not be an issue by setting an upper limit for how many players can participate.
3. Game will support Windows/OSX/(Linux if time permits) and Android/iOS
4. Application-server communication should operate responsively with low enough latency to not impact game participation (i.e. it doesn't take too long for answers to be sent)
5. Usability should be simple so that as wide an audience as possible is reached.
6. For security, inputs should be sanitized to prevent communication or usage of ill-formed data which may result in unintended effects
7. There should be a unifying theme across the applications and ports (if time permits)

   = = = = = = = Extra Stories we didn't have before = = = = = = =
8. **As a developer**, I would like to become familiar with Godot
9. **As a developer**, I would like to host a server in the cloud.