VirtuCards

CS 30700 Sprint 2 Retrospective

TEAM 8

June Seo

Ryan Hawks

Aryan Wadhwani

Kade Boltjes

Shayne Marques

Umang Sharma

What went well during Sprint 2?

The time we spent familiarizing ourselves with new technologies during Sprint 1 paid off, as we were able to focus solely on our implementation of different desired functionalities in the application. We were able to spend a significant amount of time perfecting UI components - animations, loading screen, labels, etc.

We were also able to incorporate quality-of-life features (ability to hide certain UI components, a turn timer, shuffling the played cards back into the undealt cards pile, etc.) to help the user tailor their application to their preferred styles. We also added scripted games in this sprint, allowing players to play unique actions like asking other players for cards during Go Fish or flipping cards during War.

We all worked together quite seamlessly due to our modified approach of assigning bigger tasks to individuals to minimize the overhead of passing down tasks to other teammates. We created a new method of documenting bugs to ensure that everything we find is accounted for, and will eventually be fixed. This is something we will expand upon in sprint three to ensure an even more efficient development process of Virtucards.

Another observation was that during this sprint, there were much fewer issues with version control. Merge conflicts were much less common, and usually not difficult to resolve. This can be attributed to better coordination of what tasks are done at a time, to anticipate possible conflicts and finding solutions to them in advance.

We managed time significantly better than the first sprint. During the last week, the majority of the group was done with their tasks and only working on test cases or debugging Virtucards. This made for a much less stressful presentation week and gave us more time to fine-tune the details for our presentation.

With that, our presentation for this sprint was a lot more streamlined than our first one. We were able to better demonstrate our user stories in a faster, more efficient manner.

User Story #1
As a user hosting the game, I would like to have the cards played by the game players on their mobile appear on the shared screen.

#	Description	Estimated Time	Owner
1	Design UI for the host card decks.	4 hours	Umang
2	Design UI for the cards themselves.	3 hours	Umang
3	Create an animation for a user playing a card on the client-side.	3 hours	Shayne
4	Create an animation for cards being drawn on the common-screen	2 hours	Shayne
5	Create an animation for a user playing a card on the host-side.	2 hours	Shayne
6	Design a test case to verify animations work properly.	1 hour	Umang

Completed

When the user is in a game on the client application they are able to view the cards with a great design that identifies the suit and the rank of the card. The host side application reflects a similar design for the cards, when displaying the played cards. When a user draws and plays a card on the client side, there are corresponding animations that show a card entering and leaving their hand. Similarly, on the host side, there is animation for when a card is played and then added to the table. For the above animations, there is a respective test case (#12) in the Test Cases directory of our repository.

User Story #2
As a user hosting the game, I would like to have the option to play games with rules, like Go Fish.

#	Description	Estimated Time	Owner
1	Create options for more games such as War and Go Fish to be selected	2 hour	Kade
2	Design gameplay scripts for War and Go Fish	8 hours (Cumulative)	Kade and Ryan
3	Modify game screen UI depending on the game that has been selected. For example, Go Fish has no central deck, and War has 2 decks.	3 hours (Cumulative)	Kade and Ryan
4	Ensure that the host is unable to change the game, once the selection has been made.	1 hour	Shayne
5	Create test cases for verifying different games work properly.	2 hours	Aryan

Completed
The user is prompted with options to choose Go Fish and War in the Game Creation screen in the Choose Game drop-down menu. When the user selects War or Go Fish as a mode, there are appropriate changes made to the UI of the game. Additionally, there is no means for the game mode to change during the game.

User Story #3

As a user hosting the game, I would like to have the option to play games whose rules aren't in the library of VirtuCards.

#	**Description**	**Estimated Time**	**Owner**
1	Add Freeplay as an option for the host to select	1 hour	Ryan
2	Add an extra number of settings to the settings UI designed for Freeplay	2 hours	Ryan
3	Add custom card rules in this interface (i.e. picking a standard 52 card deck or remove specific ranks). Allow the host to check and uncheck individual settings for their freeplay game such as certain cards allowed, and other game features.	3 hours	Ryan
4	Update the freeplay settings according to the settings chosen	3 hours	Ryan
5	Create test cases to verify that a host can create a free play game.	1 hours	Ryan

Completed

The drop-down on the host landing screen now has the option to select a Freeplay game mode. When the host selects Freeplay and creates a game, there is an additional Freeplay settings panel so that the host can make Freeplay-related changes such as removing suits, and displaying or hiding the last played card. When the host makes these selections, the deck and UI are updated accordingly so that the gameplay corresponds with the changes.

User Story #4
As a user hosting the game, I would like to declare a winner for games whose rules aren't in the library of VirtuCards.

#	**Description**	**Estimated Time**	**Owner**
1	Add a declare winner button to the host side	1 hour	Ryan
2	After the declare winner button is clicked show a list of players in the lobby and allow the host to choose one to win.	3 hours	Ryan
3	Add an end game button to the host side, which ends the game with no winners.	2 hour	Ryan
4	When a winner is chosen, create a message that displays the name of the winner on the common screen.	1 hour	Ryan
5	When a player has won the game update their statistics in Firebase.	3 hours	Aryan
6	Create test cases to verify that a host can end a game with a winner or without a winner, depending on the button they click in Freeplay.	1 hours	Ryan

Completed

When the host has selected Freeplay as their chosen game mode there is now a "Declare Winner" button that appears on the host side. When clicked, this button displays a list of the players in the current lobby and allows the host to choose one to win. When chosen, all of the player's screens including the host screen are updated accordingly to show the winner of the game. In addition to this, there is now an "End-Game" button that allows the host to end the game without a winner. When either of these buttons is utilized, the player statistics are all updated in Firebase.

User Story #5

As a user hosting the game, I would like to be able to choose my preferred game easily.

#	**Description**	**Estimated Time**	**Owner**
1	Create a dropdown to select the available games in the host.	2 hours	Kade
2	Implement UI such that the Host's Waiting Screen reflects the selected game.	2 hours	Shayne
3	Implement UI such that the Client's Waiting Screen reflects the selected game.	2 hours	Ryan
4	Design a test case to verify that the waiting screens reflect the game information properly.	2 hours	Shayne

Completed

The dropdown on the host landing screen shows all the available games: War, GoFish, Freeplay, and a testing game. The host's waiting screen displays that name after they have pressed the create game button. After each client has joined, their waiting screen displays the maximum number of players along with the game selected.

User Story #6

As a user hosting the game, I would like to be able to mute the chat if needed.

#	**Description**	**Estimated Time**	**Owner**
1	Repurpose the disable chat toggle in the host waiting screen into a dropdown to include muting chat.	2 hour	June
2	Repurpose the disable chat feature on the host game screen into a dropdown to include muting chat.	1 hour	June
3	When the host disables the chat, the chat window is hidden on all the clients.	4 hours	June
4	Create a test case for verifying that the chat can be muted and disabled.	1 hour	June

Completed

We were able to replace the toggle on the host side with a dropdown with mute chat. It was successful that when the host disables the chat mid-game, nobody can text anything and when the host mutes the chat mid-game, the chat doesn't show on the host side, but players in the client are still able to chat normally.

User Story #7

As a user hosting the game, I would like to be able to set a timer for each player to play their move.

#	**Description**	**Estimated Time**	**Owner**
1	Implement a timer into the game settings scripts	4 hours	Kade
2	Create an interface for setting the timer	3 hours	Kade
3	Create a timer UI component that can easily be reused	1 hour	Kade
5	Pass the turn to the next player if a player does not make a move in the given time. Penalise if the game rules specify.	2 hours	Kade
6	Implement a test that verifies the timer works properly.	1 hour	Kade

Completed

When a host wants to add the timer functionality to the game, there is a toggle in the pregame settings, which allows them to enable it and set the time. When it reaches 30 seconds, the timer turns red and a popup appears beneath it for 3 seconds saying “Warning only 30 seconds.” If the timer reaches 0, it skips that player’s turn. If the host disables the timer mid-game, it disappears from all clients and the host screen and is disabled.

User Story #8

As a user hosting the game, I would like to be able to shuffle the deck of cards easily.

#	**Description**	**Estimated Time**	**Owner**
1	Create a method in the game logic to shuffle the played deck of cards into the draw pile.	2 hour	Umang
2	Create a shuffle button on the host side.	1 hour	Umang
3	Create the animation for shuffling.	3 hours	Umang
4	Prevent players from performing actions while shuffle animation is playing.	2 hours	Umang
5	Create a test case that verifies the host can shuffle and players can draw from those shuffled cards.	1 hours	Umang

Completed

The user has the ability to shuffle the game deck when there are insufficient cards. There is a corresponding button for it on the UI screen and a corresponding animation that prevents users from making moves. The button is located on the right side of the game screen. The animation appears over all the other UI elements and appears similar to a buffering animation. The shuffling functionality also has a respective unit test.

User Story #9

As a user, I would like to be able to change my username.

#	**Description**	**Estimated Time**	**Owner**
1	Display details of the user on the Profile Screen with an edit button.	3 hours	Shayne
2	Allow the username and name fields to be changed upon clicking the edit button.	2 hours	Shayne
3	If the current user is signed in anonymously, they should see an error message when they click the edit button.	1 hour	Shayne
4	If the user changes either field, when they click the edit button once again, their changes should be validated and saved.	2 hours	Shayne
5	If the username already exists, or either field is blank, an appropriate error message should be shown.	2 hour	Shayne
6	Create a unit test that verifies a username can be changed.	1 hours	Shayne

Completed

Users can now view their profile information on the client side. By pressing the Profile button on the top left corner of the screen after a user is logged in, they are able to view their name, username, email, games played, games lost and games won. If a user wishes to edit their name and username, they can do so by clicking the Edit button. On the Edit screen, they can enter an updated name and username which is then updated in the player database as well. We have also added functionality to prevent a guest/anonymous user from editing their profile data.

User Story #10

As a user, I would like to view the cards that I have been dealt at the beginning of the round from my mobile device.

#	**Description**	**Estimated Time**	**Owner**
1	Add a loading panel for when players first enter the game screen	3 hours	Aryan
2	Depending on the game selected, give every player a certain amount of cards to start with from the global deck.	2 hour	Aryan
3	Update every player's UI to show the cards the same as if they were to draw them	2 hour	Aryan
4	Add a unit test to check that depending on the game selected the client gets the correct amount of starting cards.	1 hours	Aryan
5	Construct a test case that verifies the presence of the loading screen	1 hours	Aryan

Completed

When the host starts the game, a loading screen is placed until cards are distributed. There is also a loading screen present while signing in and attempting to join the lobby. The cards are added to a carousel on the client, with a glow indicating whether a certain card is legal or not.

User Story #11
As a user, I would like for the cards held by each player to be visible on the shared screen, hidden face down.

#	**Description**	**Estimated Time**	**Owner**
1	Keep track of the cards present with each user on the host.	2 hours	Kade
2	Display a face-down card for each user with a number indicating their hand size, on the host.	2 hours	Ryan
3	Set up the Scene to handle any range of players, from 1 up to 9 players.	3 hours	Ryan
4	When cards are played or drawn by the user, the view on the host should reflect those changes.	2 hours	Kade
5	Create a test case that verifies if facedown card piles associated with each player are shown on the host screen.	1 hour	Kade

Completed
When the game is started, an icon for each player is shown across the top of the host screen. It shows their current card count, score, and username. If a player leaves mid-game, their icon is removed. Whenever a player receives or has a card removed from their hand, their card count is updated accordingly.

User Story #12

As a user, I would like to be able to send and receive private messages to others in the same game lobby.

#	**Description**	**Estimated Time**	**Owner**
1	Add a dropdown list with the list of available players in the chat interface and a “Public” option.	4 hours	June
2	Allow selecting a player to send a private message to.	2 hours	June
3	Select the player in Photon, and send messages specifically to the player.	2 hours	June
4	Add an indicator for private messages	2 hours	June
5	Retain the selection of the specific player even if they hide the chat window temporarily	2 hours	June
6	Write test cases to ensure messages are sent only to the intended recipient.	1 hours	June

Completed

We were able to make a dropdown on the client-side with all of the players in the game above the chat that the player can choose who to private message. The dropdown has “Public Chat as the first option, but if the player presses any name under the public chat, it will be a private message. When the private message is sent, “(private)” will show before the username and that specific text will not show up on the host chat, but it will only show up on the screen of the player and the intended receiver of the message. If the host disables the chat or the player hides the chat, the option will not go away and the chat will still be in private message mode.

User Story #13

As a user, I would like to use a default messaging system to say something very fast.

#	**Description**	**Estimated Time**	**Owner**
1	Create a chat bubbles of default chat messages in the chat UI.	2 hours	Aryan
2	Add default messages in the chat bubbles.	1 hour	Aryan
3	Create a method that sends a message when a message has been clicked from the list.	2 hour	Aryan
4	Write a test case to verify that the default messages are being sent when pressed.	1 hour	Aryan

Completed

When the clients join the game, they can send default messages, available as bubbles above the chat box. These messages are sent to the recipient selected by the dropdown, which is either “Public” or a specific player.

User Story #14

As a user, I would like to have an indication of a winner, allowing us to end a game.

#	**Description**	**Estimated Time**	**Owner**
1	Add a win condition that is checked after every card is played in all game modes besides Freeplay.	2 hours	Kade
2	When a player has won, update their screen to show that they have won the game.	2 hours	June
3	When a player has won, show their name on the common screen.	1 hour	June
4	Once a player has won, allow users to leave easily and connect to the Photon LeaveRoom method.	2 hours	Umang
5	Once the game is over, update all player's statistics.	3 hours	Umang
6	Create a test case that verifies a winner can be declared.	1 hour	Umang

Completed

The user can press the *Exit* button once a game is over or a winner has been declared in order to leave the game session and return to the Waiting Screen. Upon exiting, the user's statistics will be updated on Google Firebase and these changes will reflect on their profile screen. There is also an appropriate test case (#29) for it in the Test Case directory.

User Story #15
As a user, I would like to have the games I play have the rules related to the game enforced, ensuring only legal moves are allowed.

#	**Description**	**Estimated Time**	**Owner**
1	Implement game rules correctly depending on which game the user picks.	2 hours	June
2	Ensure that the game progresses if the player plays a legal move	1 hour	June
4	Create an *Invalid Move* error screen.	1 hour	June
5	Implement functionality to check for invalid moves and display the *Invalid Move* screen.	1 hour	June
6	Create test cases that verifies that the game handles legal and illegal moves properly	1 hour	June

Completed
We have successfully implemented this user story by having a boolean in the game chat to see if the move can be played or not. If the move can be played, the game continues smoothly without any delays, but if the move is not a valid move, the move cannot be played and it shows a sign that says "Invalid move" and fades. Since each game has different rules, the code in each game shows that if it's a legal or illegal move or not.

User Story #16

As a user, I would like to be able to hide my cards on my mobile device if I need to set my device in view of other players if time allows.

#	**Description**	**Estimated Time**	**Owner**
1	Add a cover cards button	1 hour	Shayne
2	Create a method that updates UI to show the backs of the players cards when the button is clicked	2 hours	Shayne
3	Create a visual effect that indicates which card is selected in the hand.	2 hours	Shayne
4	Create a unit test that verifies that the cards are properly covered when the button is pressed	2 hours	Shayne

<u>Completed</u>

On the client application, users have an option to hide their cards. Upon clicking the hide cards button, the cards present in the hand of the player are all flipped over with animation. This animation rotates the card and then shows the back of the card - hiding any information about the cards from other players. While their cards are flipped, users are not able to play or draw more cards. When the user would like to view their cards again, they can do so by pressing the same button again which flips the cards back with an animation and they are able to view the cards face up.

User Story #17

As a user, I would like to be able to change game settings easily.

#	**Description**	**Estimated Time**	**Owner**
1	Create a settings page on the client-side game screen.	4 hours	Umang
2	Create toggle functionality for adjusting the visibility of the chat and timer.	3 hours	Umang
3	Implement a test that verifies the settings functionality works.	1 hour	Umang

Completed

There is a panel in the client-side game screen that appears after pressing the Cog-icon on the right-side of the screen. There are buttons in the panel that toggle the visibility of both chat and the timer. There is also an appropriate test case to test the functionality of this panel.

User Story #18

As a user, I would like to be able to exit a game without disrupting the flow of play.

#	**Description**	**Estimated Time**	**Owner**
1	In the settings window in the game scene, add a button to leave the game.	1 hour	Aryan
2	Upon clicking the button, disconnect the player from the room and redirect them to the join game page.	2 hours	Aryan
3	On the host, retrieve the cards from the user when they disconnect and add them to the draw pile, with reshuffling.	2 hours	Aryan
4	Remove the disconnected user from the game screen.	2 hours	Aryan
5	If there are no players left in the game, return to the waiting screen.	2 hours	Aryan
6	Send a message in the chat, if enabled, that a user has disconnected.	1 hour	Aryan
7	Write test cases to ensure the game correctly handles each situation.	1 hour	Aryan

Completed

When a player leaves the game, they are disconnected from the host. A message appears in the chat, coming from the host stating the user had left. Any cards that the user had were taken and added to the draw pile, and reshuffled. Also, if it was currently the user's turn, it was passed to the next player. To leave the game, the user can either leave the game from the settings window or close the app entirely.

User Story #19 (Unfinished tasks from Sprint 1)
As a developer, I would like to test the functionality implemented into both the VirtuCards client and host.

#	**Description**	**Estimated Time**	**Owner**
1	Implement unit tests to debug creating Photon rooms with the unique code.	1 hours	Umang
2	Create unit tests for clients joining a game room.	1 hours	Shayne
3	Discard name upon exiting the game if the user is signed in anonymously.	1 hour	Aryan
4	Add unit tests that check if a player can play as a guest, receive a random username and ensure name is unique	1 hours	Shayne
5	Create unit tests to ensure a person can join as a player from their mobile device if they have already created a lobby from their laptop or desktop computer	1 hours	Aryan
6	Design test cases that evaluate the propagation delay between user actions and the appropriate response from the server	.5 hours	June
7	Create unit tests that check that a player can draw cards successfully	1 hours	Kade
8	Create test cases that verify that a player can play valid cards during a game	1 hours	Ryan
9	Implement unit tests that check a player can skip their turn	1 hours	Kade
10	Create unit tests that verify messages can be sent from a player to all other players	1 hours	June
11	Design unit tests to verify that the public messages are displayed	1 hour	Kade

Completed

We were successful in catching up on our work from Sprint 1. We have four test cases within each test case document and made a doc and a pdf file for easier access.

What did not go so well in Sprint 2?

Overall, we think this was a great Sprint in terms of completing and assigning tasks, bug fixes and presenting during the Sprint Review. We all worked well together and completed our individual tasks in a timely manner.

One aspect that could have been better is the Test Cases. While we all were assigned our own test cases to work on, most of the test cases were not completed until the last week of the Sprint. This minimized our time to work on last minute bug fixes.

Due to the nature of our application being a game, there are a lot of edge cases and random scenarios that are unaccounted for during development. This leads to us noticing a considerable amount of bugs. While we were able to fix a good amount of bugs, there could have been a better system for logging bugs then assigning the bug fix task to a team member.

While drafting our Sprint 2 Planning Document, we should have paid more attention to our wording for certain tasks and acceptance criteria. This would have made our tasks more clear and concise without any room for obscurity.

How should we improve?

On the final couple days of Sprint 2, we found a much better way to track bugs was to keep a google spreadsheet that listed them all, along with who was working on them. That kept us on track to finish up final bug fixes, and let us see which bugs were the most necessary to fix. During sprint 3, we will extensively use that sheet to keep track of them.

In Sprint 3, we hope to finish the spread out time we spend working on test cases to ensure that we can spend some significant time logging bugs and then fixing them towards the end of the Sprint.

Another point of improvement is better discussing and planning out our acceptance criteria. This means researching in advance regarding how a certain criteria should be implemented. This will be necessary during the implementation of music playback during this sprint. Also having the criteria read by more teammates during the planning section, so that we have a better understanding of how we all wish to approach this.

Additionally, we plan on incorporating screenshots and explanations for said screenshots into our test cases when they're written to avoid having to come back and update them in the future.