Wild Card Bot

Sprint 3 Test Cases

Team 25: Konstantin Liehr, Arun Thirumavalavan, Nikolas Damalas, Kyle Arrowood, Michael Gu, Matthew Wang

Test Case 0001
System: Pokemon Phase: 3
Get pokemon by name
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!poke pikachu”

Expected result

1. Bot responds with an embedded card that gives information about the pokemon pikachu. This includes the name, types, abilities, and statistics, as well as an image.

Test Case 0002
System: Pokemon Phase: 3
Get pokemon by number
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!poke 100”

Expected result

1. Bot responds with an embedded card that gives information about the pokemon voltorb. This includes the name, types, abilities, and statistics, as well as an image.

Test Case 0003
System: Pokemon Phase: 3
Get pokemon by number
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!poke -1”

3. Type in “!poke abc”

Expected result

1. Bot responds with an embedded card for each command that states there was an error loading the pokemon, and that perhaps the number or name is incorrect.

Test Case 0004
System: Pokemon   Phase: 3
Get item
Severity :1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!item lucky egg”

Expected Result:

1. The bot responds with a card that displays the item name, image, and effect.

Test Case 0005
System: Pokemon   Phase: 3
Incorrect Item name
Severity :1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!item abc123”

Expected Result:

1. The bot responds with a card that says the item is not found.

Test Case 0006
System: Pokemon   Phase: 3
Held items
Severity :1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!item held lucky egg”

Expected Result:

1. The bot responds with a card that displays the item name, image, and a list of all the pokemon who can be found holding it in the wild.

Test Case 0007
System: Pokemon   Phase: 3
Shiny pokemon
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!shiny magikarp”

Expected result:

1. The bot responds with a card displaying a shiny magikarp

Test Case 0008
System: Pokemon   Phase: 3
Shiny incorrect name
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!shiny abc1233”

Expected result:

1. The bot responds with a card saying there was an error finding the pokemon.

Test Case 0009
System: Pokemon   Phase: 3
Shiny incorrect name
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!shiny compare magikarp”

Expected result:

1. The bot will respond with two cards, one showing the normal sprite of magikarp and one displaying the shiny sprite of magikarp.

Test Case 0010
System: Pokemon   Phase: 3
Get Move
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!move iron head” into discord

Expected result:

1. The bot will respond with a card displaying all the necessary details of the move.

Test Case 0011
System: Pokemon   Phase: 3
Get Move of one Pokemon
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!move pikachu” into discord

Expected result:

1. The bot will respond with a card displaying all the moves that pikachu can learn.

Test Case 0012

System: Pokemon Phase: 3
Get Move with No Effects
Severity: 3
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!move tackle” into discord

Expected result:

1. The bot will respond with a card displaying the move “tackle”. Since it has no added effect it will state so, and forego showing a 0 for the effect chance.

Test Case 0013
System: Pokemon Phase: 3
Bad Move Name
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!move abc” into discord

Expected result:

1. The bot will respond with a card saying that there was an error getting the requested move.

Test Case 0014
System: Pokemon Phase: 3
Get ability
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!ability overgrow” into discord

Expected result:

1. The bot will respond with a card describing the effect of the ability overgrow.

Test Case 0015
System: Pokemon Phase: 3
Get all pokemon with a specific ability
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!ability all overgrow” into discord

Expected result:

1. The bot will respond with a card listing all pokemon that can have the ability overgrow.

Test Case 0016
System: Pokemon Phase: 3
Incorrect ability name

Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!ability abc” into discord

Expected result:

1. The bot will respond with a card saying that there was an error getting the requested ability.

Test Case 0017
System: Sports Module Phase: 3
Team with an upcoming game
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!score <team>” into discord where <team> is a team that has another game in the near future.

Expected result:

1. The bot will respond with an embed with all of the game information including the accurate time, date, spread, money line, over/under, and records of teams.
2. If the upcoming game is too far out, then the betting odds will likely not yet be posted.

Test Case 0018
System: Sports Module Phase: 3
Team with a live game
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!score <team>” into discord where <team> is a team that is currently playing a game.

Expected result:

1. The bot will respond with an embed with all of the game information including the accurate time remaining, scores, and records.

Test Case 0019
System: Sports Module Phase: 3
Invalid team
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!score abc” into discord.

Expected result:

1. The bot will respond telling the user that the team is invalid.

Test Case 0020

System: Sports Module Phase: 3
Test player stats
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!player Lebron” into discord.

Expected result:

1. The bot will respond with an embed displaying all of LeBron’s current season stats as well as all of his career stats.

Test Case 0021
System: Sports Module Phase: 3
Test player stats
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!player Carson Wentz” into discord.

Expected result:

1. The bot will respond with an embed displaying all of Carson Wentz current season stats as well as all of his career stats.

Test Case 0022
System: Sports Module Phase: 3
Test Invalid Player
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!player abc” into discord.

Expected result:

1. The bot will respond telling the user that the player could not be found and is invalid.

Test Case 0023
System: Pokemon Phase: 3
Get a pokemon evolutionary tree
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!evo pikachu” into discord

Expected result:

1. The bot will respond with a card listing all the pokemon that evolve into and from pikachu, as well as the requirements for evolution.

Test Case 0024

System: Pokemon Phase: 3

Get a pokemon evolutionary tree when they do not evolve

Severity: 1

Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!evo dialga” into discord

Expected result:

1. The bot will respond with a card stating that dialga cannot evolve into and from anything since it has no evolutionary family.

Test Case 0025

System: Pokemon Phase: 3

Evolutions for bad pokemon name

Severity: 1

Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!evo abc” into discord

Expected result:

1. The bot will respond with a card stating that the specified pokemon is not found, and that perhaps it is misspelled.

Test Case 0026

System: Pokemon Phase: 3

Pokemon Type Matchups

Severity: 1

Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!type ground” into discord

Expected result:

1. The bot will respond with a card stating all the type matchups for the ground type.

Test Case 0027

System: Pokemon Phase: 3

Pokemon Specific Type Matchups

Severity: 1

Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!type ground electric” into discord

Expected result:

1. The bot will respond with a card stating the type matchup between the two types.

Test Case 0028

System: Pokemon Phase: 3

Pokemon Incorrect Type

Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!type abc” into discord

Expected result:

1. The bot will respond with a card stating that the type cannot be found.

Test Case 0029
System: Pokemon   Phase: 3
Pokemon Incorrect Number of Types
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!type fire ground water” into discord

Expected result:

1. The bot will respond with a card stating that there is an incorrect number of inputs for the function.

Test Case 0030
System: Pokemon   Phase: 3
Pokemon Egg Groups
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!egg pikachu” into discord

Expected result:

1. The bot will respond with a card stating that pikachu belongs to the fairy and ground egg groups.

Test Case 0031
System: Pokemon   Phase: 3
Pokemon Egg Groups
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!egg fairy” into discord

Expected result:

1. The bot will respond with a card listing all the pokemon in the fairy group.

Test Case 0032
System: Pokemon   Phase: 3
Incorrect pokemon or group name
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!egg abc” into discord

Expected result:

1. The bot will respond with an error card saying that the name or egg group is not valid, and is perhaps misspelled.

Test Case 0033
System: Pokemon   Phase: 3
Pokemon with no egg groups
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!egg dialga” into discord

Expected result:

1. The bot will respond with a card stating that dialga has no egg groups.

Test Case 0034
System: Math   Phase: 3
Simple math problem
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!calc 2+2” into discord

Expected result:

1. The bot will respond with an embed telling the user the correct answer of 4.

Test Case 0035
System: Math   Phase: 3
Complex math problem
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!calc (2 + 5(90^2)/10) % 5” into discord

Expected result:

1. The bot will respond with an embed telling the user the correct answer of 2.

Test Case 0036
System: Math   Phase: 3
Invalid math problem
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!calc 2+2-” into discord

Expected result:

1. The bot will respond telling the user that an invalid math equation was given.

Test Case 0037
System: Math Phase: 3
Algebraic math problem
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!calc 2 + x = 55” into discord.

Expected result:

1. The bot will respond with an embed telling the user that x is equal to 53.

Test Case 0038
System: Math Phase: 3
Complex Algebraic math problem
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!calc (2^(3x)) = 50” into discord.

Expected result:

1. The bot will respond with an embed telling the user that x is equal to 53.

Test Case 0039
System: Math Phase: 3
Invalid Algebraic math problem
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!calc (2^(3x) = 50” into discord.

Expected result:

1. The bot will respond to the user telling them that this is an invalid equation.

Test Case 0040
System: Math Phase: 3
Invalid Algebraic math problem #2
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!calc (2^(3x)) - = 50” into discord.

Expected result:

1. The bot will respond to the user telling them that this is an invalid equation.

Test Case 0041
System: Pokemon Phase: 3

Pokemon nature lookup
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!nature adamant” into discord

Expected result:

1. The bot will respond with a card stating the stats and flavors affected by the adamant nature.

Test Case 0042
System: Pokemon Phase: 3
Pokemon invalid nature
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!nature abc” into discord.

Expected result:

1. The bot will respond with an error card stating the given nature is invalid.

Test Case 0043
System: Pokemon Phase: 3
Pokemon nature search single stat
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!nature +attack” into discord

Expected result:

1. The bot will respond with a card listing all natures that boost attack

Test Case 0044
System: Pokemon Phase: 3
Pokemon nature search two stats
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!nature +speed -defense” into discord

Expected result:

1. The bot will respond with a card showing the nature hasty, which increases speed and decreases defense.

Test Case 0045
System: Pokemon Phase: 3
Pokemon nature invalid stat
Severity: 1

Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!nature +abc” into discord

Expected result:

1. The bot will respond with a card stating that one of the provided stats is invalid.

Test Case 0046

System: Math Phase: 3

Trigonometry Test

Severity: 2

Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!calc sin(pi/3)” into discord.

Expected result:

1. The bot will respond with an embed telling the user that the answer is 0.5 which is the correct value.

Test Case 0047

System: Math Phase: 3

Derivative Test

Severity: 2

Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!derive 2x^3” into discord.

Expected result:

1. The bot will respond with an embed telling the user that the derivative is 6x^2.

Test Case 0048

System: Math Phase: 3

Complex Derivative Test

Severity: 2

Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!derive 2x^3 + sin(5x)” into discord.

Expected result:

1. The bot will respond with an embed telling the user that the derivative is 6x^2 + 5cos(5x).

Test Case 0049

System: Math Phase: 3

Invalid Derivative Test

Severity: 2

Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!derive 2x^3 -” into discord.

Expected result:

1. The bot will respond telling the user that the equation is invalid.

Test Case 0050

System: Math Phase: 3

Indefinite Integral Test

Severity: 2

Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!integrate 2x^3” into discord.

Expected result:

1. The bot will respond with an embed telling the user that the integral is “x^4 / 2 + C”.

Test Case 0051

System: Math Phase: 3

Complex Indefinite Integral Test

Severity: 2

Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!integrate 2x^3 + sin(5x)” into discord.

Expected result:

1. The bot will respond with an embed telling the user that the integral is “x^4 / 2 - cos(5x)/5 + C”.

Test Case 0052

System: Math Phase: 3

Definite Integral Test

Severity: 2

Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!integrate definite 0 5 2x^3” into discord.

Expected result:

1. The bot will respond with an embed telling the user that the definite integral is 312.5.

Test Case 0053

System: Math Phase: 3

Complex Definite Integral Test

Severity: 2

Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!integrate definite 0 5 2x^3 + sin(5x)” into discord.

Expected result:

1. The bot will respond with an embed telling the user that the definite integral is 312.5.

Test Case 0054

System: Math Phase: 3
Invalid Indefinite Integral Test
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!integrate 2x^3 - ” into discord.

Expected result:

1. The bot will respond telling the user that the equation is invalid.

Test Case 0055
System: Math Phase: 3
Invalid Definite Integral Test
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!integrate definite 0 5 2x^3 - ” into discord.

Expected result:

1. The bot will respond telling the user that the equation is invalid.

Test Case 0056
System: Math Phase: 3
Invalid Definite Integral Test
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!integrate definite 0” into discord.

Expected result:

1. The bot will respond telling the user that there are not enough arguments for a definite integral.

Test Case 0057
System: Blackjack Phase: 3
Blackjack Game Initiation Test
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack” into discord.

Expected result:

1. The bot should respond with a message displaying the current statistics of the blackjack game. Round 1 should be indicated and the list of players must include the user that called “!blackjack”. The dealer’s hand should also be displayed.
2. After 15 seconds, the user should receive a direct message indicating an opportunity to change his or her bet.

Test Case 0058
System: Blackjack Phase: 3
Blackjack Game Redisplay Test
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack” into discord.
3. Enter “!blackjack” into discord again.

Expected result:

1. Upon completing step 2, the bot should respond with a message displaying the current statistics of the blackjack game. Round 1 should be indicated and the list of players must include the user that called “!blackjack”. The dealer’s hand should also be displayed.
2. Upon completing step 3, the bot should respond with a message displaying the same content as the message sent in step 2. The message sent in step 2 should also be deleted.
3. After 15 seconds, the user should receive a direct message indicating an opportunity to change his or her bet.

Test Case 0059
System: Blackjack Phase: 3
Blackjack Game Status Test
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack” into discord.
3. After 15 seconds, you should receive a direct message. React with the keycap 10 reaction.

Expected result:

1. The bot should respond with a message displaying the current statistics of the blackjack game. Round 1 should be indicated and the list of players must include the user that called “!blackjack”. The dealer’s hand should also be displayed.
2. After 15 seconds, the user should receive a direct message indicating an opportunity to change his or her bet.
3. Upon reacting to the direct message in step 3, the message in step 2 should be edited by the bot to indicate the current round is in progress.

Test Case 0060
System: Blackjack Phase: 3
Blackjack Game Player Joining Test
Severity: 1
Instructions:

1. Login and open Discord on two separate accounts.
2. To start the bot, at the console, enter: python3 main.py
3. Enter “!blackjack” into discord from one Discord account.

4. From the other Discord account, react with the black joker reaction to the message sent by the command.

Expected result:

1. The bot should respond with a message displaying the current statistics of the blackjack game. Round 1 should be indicated and the list of players must include the user that called “!blackjack”. The dealer’s hand should also be displayed. The bot should also react with the black joker reaction to this message.
2. After 15 seconds, the message in step 2 should be edited and the second user should be included in the list of players.

Test Case 0061
System: Blackjack   Phase: 3
Blackjack Game Player Already Joined Test
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack” into discord.
3. React with the black joker reaction to the message sent by the command.

Expected result:

1. The bot should respond with a message displaying the current statistics of the blackjack game. Round 1 should be indicated and the list of players must include the user that called “!blackjack”. The dealer’s hand should also be displayed. The bot should also react with the black joker reaction to this message.
2. After 15 seconds, the message in step 2 should be unchanged and the user should not be added twice to the list of players.

Test Case 0062
System: Blackjack   Phase: 3
Blackjack Game Round Results For Fold Test
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack” into discord.
3. After 15 seconds, you should receive a direct message. React with the keycap 10 reaction.
4. You should receive another two direct messages. React with the cancel reaction to the second message to fold.

Expected result:

1. The bot should respond with a message displaying the current statistics of the blackjack game. Round 1 should be indicated and the list of players must include the user that called “!blackjack”. The dealer’s hand should also be displayed.
2. After 15 seconds, the user should receive a direct message indicating an opportunity to change his or her bet.

3. Upon reacting to the direct message in step 3, the user should receive another direct message indicating his or her bet for the round.
4. The user should then receive another direct message that displays the user's cards for the round. It should also include three reactions, one to hit, one to stand, and one to fold. The bot should react with all three reactions and the message should indicate which one corresponds to each.
5. Upon completing step 4, the message sent in step 2 should be edited to reflect the round results. It should indicate that the user has folded, and display the amount of money lost, the user's total balance, and the dealer's cards and final value.

Test Case 0063
System: Blackjack Phase: 3
Blackjack Game Round Results For Win Or Lose Test
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter "!blackjack" into discord.
3. After 15 seconds, you should receive a direct message. React with the keycap 10 reaction.
4. You should receive another two direct messages. React with the stand reaction to the second message to stand.

Expected result:

1. The bot should respond with a message displaying the current statistics of the blackjack game. Round 1 should be indicated and the list of players must include the user that called "!blackjack". The dealer's hand should also be displayed.
2. After 15 seconds, the user should receive a direct message indicating an opportunity to change his or her bet.
3. Upon reacting to the direct message in step 3, the user should receive another direct message indicating his or her bet for the round.
4. The user should then receive another direct message that displays the user's cards for the round. It should also include three reactions, one to hit, one to stand, and one to fold. The bot should react with all three reactions and the message should indicate which one corresponds to each.
5. Upon completing step 4, the message sent in step 2 should be edited to reflect the round results. It should indicate that the user has won, lost, or tied, and display the change in balance, the user's total balance, the user's final value, and the dealer's cards and final value.

Test Case 0064
System: Blackjack Phase: 3
Blackjack Game Round Continuation Test
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py

2. Enter “!blackjack” into discord.
3. After 15 seconds, you should receive a direct message. React with the keycap 10 reaction.
4. You should receive another two direct messages. React with the cancel reaction to the second message to fold.

Expected results:

1. The bot should respond with a message displaying the current statistics of the blackjack game. Round 1 should be indicated and the list of players must include the user that called “!blackjack”. The dealer’s hand should also be displayed.
2. After 15 seconds, the user should receive a direct message indicating an opportunity to change his or her bet.
3. Upon reacting to the direct message in step 3, the user should receive another direct message indicating his or her bet for the round.
4. The user should then receive another direct message that displays the user’s cards for the round. It should also include three reactions, one to hit, one to stand, and one to fold. The bot should react with all three reactions and the message should indicate which one corresponds to each.
5. Upon completing step 4, the message sent in step 2 should be edited to reflect the round results. It should indicate that the user has folded, and display the amount of money lost, the user’s total balance, and the dealer’s cards and final value.
6. After 15 seconds, the message sent in step 2 should be edited again to reflect the start of a new round. Round 2 should be indicated, the dealer’s hand should be displayed, and the list of players should include all players from the first round.

Test Case 0065
System: Blackjack Phase: 3
Blackjack Game Hit Test
Severity: 1
Instructions:

1. This test relies on random number generation so it may have to be performed multiple times. The other result is reflected in the bust test.
2. To start the bot, at the console, enter: python3 main.py
3. Enter “!blackjack” into discord.
4. After 15 seconds, you should receive a direct message. React with the keycap 10 reaction.
5. You should receive another two direct messages. React with the green check mark reaction to the second message to hit.

Expected results:

1. The bot should respond with a message displaying the current statistics of the blackjack game. Round 1 should be indicated and the list of players must include the user that called “!blackjack”. The dealer’s hand should also be displayed.
2. After 15 seconds, the user should receive a direct message indicating an opportunity to change his or her bet.

3. Upon reacting to the direct message in step 4, the user should receive another direct message indicating his or her bet for the round.
4. The user should then receive another direct message that displays the user's cards for the round. It should also include three reactions, one to hit, one to stand, and one to fold. The bot should react with all three reactions and the message should indicate which one corresponds to each.
5. Upon completing step 5, another direct message should be sent to the user. It should display the user's cards for the round, which should include an additional card. It should also have two reactions for hit and stand, which the bot should have reacted to already.

Test Case 0066
System: Blackjack Phase: 3
Blackjack Game Stand Test
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter "!blackjack" into discord.
3. After 15 seconds, you should receive a direct message. React with the keycap 10 reaction.
4. You should receive another two direct messages. React with the green cross mark reaction to the second message to stand.

Expected results:

1. The bot should respond with a message displaying the current statistics of the blackjack game. Round 1 should be indicated and the list of players must include the user that called "!blackjack". The dealer's hand should also be displayed.
2. After 15 seconds, the user should receive a direct message indicating an opportunity to change his or her bet.
3. Upon reacting to the direct message in step 3, the user should receive another direct message indicating his or her bet for the round.
4. The user should then receive another direct message that displays the user's cards for the round. It should also include three reactions, one to hit, one to stand, and one to fold. The bot should react with all three reactions and the message should indicate which one corresponds to each.
5. Upon completing step 4, another direct message should be sent to the user. It should display the user's cards for the round and the final value. The hand should be identical to the previous direct message.

Test Case 0067
System: Blackjack Phase: 3
Blackjack Game Fold Test
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter "!blackjack" into discord.

3. After 15 seconds, you should receive a direct message. React with the keycap 10 reaction.
4. You should receive another two direct messages. React with the cancel reaction to the second message to fold.

Expected result:

1. The bot should respond with a message displaying the current statistics of the blackjack game. Round 1 should be indicated and the list of players must include the user that called “!blackjack”. The dealer’s hand should also be displayed.
2. After 15 seconds, the user should receive a direct message indicating an opportunity to change his or her bet.
3. Upon reacting to the direct message in step 3, the user should receive another direct message indicating his or her bet for the round.
4. The user should then receive another direct message that displays the user’s cards for the round. It should also include three reactions, one to hit, one to stand, and one to fold. The bot should react with all three reactions and the message should indicate which one corresponds to each.
5. Upon completing step 4, another direct message should be sent to the user. It should indicate that the user folded and display the money lost. The user should be removed from the round and should not receive a message indicating win or lose.

Test Case 0068
System: Blackjack   Phase: 3
Blackjack Game Bust Test
Severity: 1
Instructions:

1. This test relies on random number generation so it may have to be performed multiple times. The other result is reflected in the hit test.
2. To start the bot, at the console, enter: python3 main.py
3. Enter “!blackjack” into discord.
4. After 15 seconds, you should receive a direct message. React with the keycap 10 reaction.
5. You should receive another two direct messages. React with the green check mark reaction to the second message to hit.
6. Repeat step 5 until you obtain the bust message.

Expected result:

1. The bot should respond with a message displaying the current statistics of the blackjack game. Round 1 should be indicated and the list of players must include the user that called “!blackjack”. The dealer’s hand should also be displayed.
2. After 15 seconds, the user should receive a direct message indicating an opportunity to change his or her bet.
3. Upon reacting to the direct message in step 4, the user should receive another direct message indicating his or her bet for the round.
4. The user should then receive another direct message that displays the user’s cards for the round. It should also include three reactions, one to hit, one to stand, and one to fold.

The bot should react with all three reactions and the message should indicate which one corresponds to each.

5. Upon completing step 5, another direct message should be sent to the user. It should display the user’s cards for the round, which should include an additional card. It should also have two reactions for hit and stand, which the bot should have reacted to already.
6. Upon completing step 6, the bot will send the user a direct message indicating the user has busted, or exceed a final value of 21. It should display the user’s cards and final value, money lost for the round, and also indicate that the user has lost the round.

Test Case 0069
System: Blackjack Phase: 3
Blackjack Game Bet Change Test 1
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack” into discord.
3. After 15 seconds, you should receive a direct message. React with the keycap 1 reaction.

Expected results:

1. The bot should respond with a message displaying the current statistics of the blackjack game. Round 1 should be indicated and the list of players must include the user that called “!blackjack”. The dealer’s hand should also be displayed.
2. After 15 seconds, the user should receive a direct message indicating an opportunity to change his or her bet.
3. Upon reacting to the direct message in step 3, the user should receive another direct message indicating his or her bet of $100 for the round.

Test Case 0070
System: Blackjack Phase: 3
Blackjack Game Bet Change Test 2
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack” into discord.
3. After 15 seconds, you should receive a direct message. React with the keycap 2 reaction.

Expected results:

1. The bot should respond with a message displaying the current statistics of the blackjack game. Round 1 should be indicated and the list of players must include the user that called “!blackjack”. The dealer’s hand should also be displayed.
2. After 15 seconds, the user should receive a direct message indicating an opportunity to change his or her bet.
3. Upon reacting to the direct message in step 3, the user should receive another direct message indicating his or her bet of $200 for the round.

Test Case 0071
System: Blackjack   Phase: 3
Blackjack Game Bet Change Test 3
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack” into discord.
3. After 15 seconds, you should receive a direct message. React with the keycap 5 reaction.

Expected result:

1. The bot should respond with a message displaying the current statistics of the blackjack game. Round 1 should be indicated and the list of players must include the user that called “!blackjack”. The dealer’s hand should also be displayed.
2. After 15 seconds, the user should receive a direct message indicating an opportunity to change his or her bet.
3. Upon reacting to the direct message in step 3, the user should receive another direct message indicating his or her bet of $500 for the round.

Test Case 0072
System: Blackjack   Phase: 3
Blackjack Game Bet Change Test 4
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack” into discord.
3. After 15 seconds, you should receive a direct message. React with the keycap 10 reaction.

Expected result:

1. The bot should respond with a message displaying the current statistics of the blackjack game. Round 1 should be indicated and the list of players must include the user that called “!blackjack”. The dealer’s hand should also be displayed.
2. After 15 seconds, the user should receive a direct message indicating an opportunity to change his or her bet.
3. Upon reacting to the direct message in step 3, the user should receive another direct message indicating his or her bet of $1000 for the round.

Test Case 0073
System: Blackjack   Phase: 3
Blackjack Game Bet Change No Response Test
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack” into discord.
3. After 15 seconds, you should receive a direct message. Wait 15 more seconds.

Expect result:

1. The bot should respond with a message displaying the current statistics of the blackjack game. Round 1 should be indicated and the list of players must include the user that called “!blackjack”. The dealer’s hand should also be displayed.
2. After 15 seconds, the user should receive a direct message indicating an opportunity to change his or her bet.
3. Upon completing step 3, the user should receive another direct message indicating his or her bet of $500 for the round, which is the standard bet amount. This indicates not responding to the direct message in step 3 will result in no change to the user’s bet amount.

Test Case 0074
System: Blackjack   Phase: 3
Blackjack Balance Command Test
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack balance” into discord.

Expected result:

1. The user should receive a direct message from the bot indicating that his or her current balance is $100000, which is the default amount.

Test Case 0075
System: Blackjack   Phase: 3
Blackjack Balance Test
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack” into discord.
3. After 15 seconds, you should receive a direct message. React with the keycap 10 reaction.
4. You should receive another two direct messages. React with the cancel reaction to the second message to fold.
5. After 15 seconds, you should receive a direct message identical to step 3. React with the red cross mark reaction.
6. Enter “!blackjack” into discord.

Expected result:

1. The bot should respond with a message displaying the current statistics of the blackjack game. Round 1 should be indicated and the list of players must include the user that called “!blackjack”. The dealer’s hand should also be displayed.
2. After 15 seconds, the user should receive a direct message indicating an opportunity to change his or her bet.
3. Upon reacting to the direct message in step 3, the user should receive another direct message indicating his or her bet for the round.

4. The user should then receive another direct message that displays the user’s cards for the round. It should also include three reactions, one to hit, one to stand, and one to fold. The bot should react with all three reactions and the message should indicate which one corresponds to each.
5. Upon completing step 4, another direct message should be sent to the user. It should indicate that the user folded and display the money lost. The user should be removed from the round and should not receive a message indicating win or lose.
6. After 15 seconds, the user should receive a direct message indicating an opportunity to change his or her bet again.
7. Upon completing step 5, the user should receive a direct message indicating he or she has left the game and the message in step 2 should be edited to indicate the game has been removed due to no players.
8. Upon completing 6, another game should be started and the bot should send a message similar to the one in step 2. The balance of the user indicated in the list of players should be $99500, unchanged from the final balance of the previous game.

Test Case 0076

System: Blackjack   Phase: 3

Blackjack Balance Result Test

Severity: 1

Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack” into discord.
3. After 15 seconds, you should receive a direct message. React with the keycap 10 reaction.
4. You should receive another two direct messages. React with the stand reaction to the second message to stand.

Expected result:

1. The bot should respond with a message displaying the current statistics of the blackjack game. Round 1 should be indicated and the list of players must include the user that called “!blackjack”. The dealer’s hand should also be displayed.
2. After 15 seconds, the user should receive a direct message indicating an opportunity to change his or her bet.
3. Upon reacting to the direct message in step 3, the user should receive another direct message indicating his or her bet for the round.
4. The user should then receive another direct message that displays the user’s cards for the round. It should also include three reactions, one to hit, one to stand, and one to fold. The bot should react with all three reactions and the message should indicate which one corresponds to each.
5. Upon completing step 4, the message sent in step 2 should be edited to reflect the round results. The change in balance should correspond with the round result. If the user tied, the balance should not be changed. If the user won, the balance should be $101500. If the user lost, the balance should be $99000.

Test Case 0077
System: Blackjack Phase: 3
Blackjack Game Leave Test
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack” into discord.
3. After 15 seconds, you should receive a direct message. React with the red cross reaction.

Expected result:

1. The bot should respond with a message displaying the current statistics of the blackjack game. Round 1 should be indicated and the list of players must include the user that called “!blackjack”. The dealer’s hand should also be displayed.
2. After 15 seconds, the user should receive a direct message indicating an opportunity to change his or her bet.
3. Upon reacting to the direct message in step 3, the user should receive a direct message indicating he or she has left the game and the message in step 2 should be edited to indicate the game has been removed due to no players.

Test Case 0078
System: Blackjack Phase: 3
Blackjack Game Leave Balance Test
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack” into discord.
3. After 15 seconds, you should receive a direct message. React with the red cross reaction.
4. Enter “!blackjack balance” into discord.

Expected result:

1. The bot should respond with a message displaying the current statistics of the blackjack game. Round 1 should be indicated and the list of players must include the user that called “!blackjack”. The dealer’s hand should also be displayed.
2. After 15 seconds, the user should receive a direct message indicating an opportunity to change his or her bet.
3. Upon reacting to the direct message in step 3, the user should receive a direct message indicating he or she has left the game and the message in step 2 should be edited to indicate the game has been removed due to no players.
4. Upon completing step 4, the user should receive a direct message from the bot indicating that his or her current balance is $100000.

Test Case 0079
System: Blackjack Phase: 3
Blackjack Rules All Sections Test

Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack rules” into discord.

Expected result:

1. The user should receive five direct messages, one for each section of the blackjack game rules. This includes the game, cards, turn, outcomes, and play sections.

Test Case 0080
System: Blackjack Phase: 3
Blackjack Rules Game Section Test
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack rules game” into discord.

Expected result:

1. The user should receive a direct message displaying the game section of the blackjack game rules.

Test Case 0081
System: Blackjack Phase: 3
Blackjack Rules Cards Section Test
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack rules cards” into discord.

Expected result:

1. The user should receive a direct message displaying the cards section of the blackjack game rules.

Test Case 0082
System: Blackjack Phase: 3
Blackjack Rules Turn Section Test
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack rules turn” into discord.

Expected result:

1. The user should receive a direct message displaying the turn section of the blackjack game rules.

Test Case 0083
System: Blackjack Phase: 3
Blackjack Rules Outcomes Section Test

Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack rules outcomes” into discord.

Expected result:

1. The user should receive a direct message displaying the outcomes section of the blackjack game rules.

Test Case 0084
System: Blackjack Phase: 3
Blackjack Rules Play Section Test
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack rules play” into discord.

Expected result:

1. The user should receive a direct message displaying the play section of the blackjack game rules.

Test Case 0085
System: Blackjack Phase: 3
Blackjack Rules Unknown Section Test
Severity: 2
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack rules abc” into discord.

Expected result:

1. The user should receive a direct message that indicates the given section was not found. The direct message should also provide a list of all section names.

Test Case 0086
System: Blackjack Phase: 3
Blackjack Leaderboard Test
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!blackjack leaderboard” into discord.
3. React to the message sent in response to “!blackjack leaderboard” with the keycap B reaction.
4. React to the message sent in response to “!blackjack leaderboard” with the keycap M reaction.

Expected result:

1. Upon completing step 2, the bot should respond with a message displaying the players who have won the most blackjack rounds. The bot should react with the keycap B and keycap M reactions to this message.
2. Upon completing step 3, the bot should edit the message in step 2 to display the players who have gotten the most blackjacks. The bot should react with the keycap W and keycap M reactions to this message.
3. Upon completing step 4, the bot should edit the message in step 2 to display the players who have gained the most money from blackjack. The bot should react with the keycap W and keycap B reactions to this message.

Test Case 0087
System: Blackjack   Phase: 3
Slots Spin Test 1
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!slots” into discord.
3. You should receive a direct message. React with the keycap 1 reaction.

Expected result:

1. Upon completing step 2, the bot should direct message the user a message that represents a slot machine. It should include the three slots, the user’s current balance, the reactions that correspond to the bet amounts, the winning combinations, and the winning amounts as a multiplier of the bet. The bot should react with all the reactions for the bet amounts as well as the red cross reaction to remove the slot machine.
2. Upon completing step 3, the user will bet $100, resulting in a balance of $99900. The winning amounts should be updated to reflect the multipliers. The bot will then spin the slots for 5 seconds.
3. Upon finishing the spin sequence, if the user won, the winning amount should be added to the user’s balance and the direct message should be edited to reflect the new balance. The message should also be edited to reflect the result of the spin.

Test Case 0088
System: Blackjack   Phase: 3
Slots Spin Test 2
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!slots” into discord.
3. You should receive a direct message. React with the keycap 2 reaction.

Expected result:

1. Upon completing step 2, the bot should direct message the user a message that represents a slot machine. It should include the three slots, the user’s current balance, the reactions that correspond to the bet amounts, the winning combinations, and the

winning amounts as a multiplier of the bet. The bot should react with all the reactions for the bet amounts as well as the red cross reaction to remove the slot machine.
2. Upon completing step 3, the user will bet $200, resulting in a balance of $99800. The winning amounts should be updated to reflect the multipliers. The bot will then spin the slots for 5 seconds.
3. Upon finishing the spin sequence, if the user won, the winning amount should be added to the user’s balance and the direct message should be edited to reflect the new balance. The message should also be edited to reflect the result of the spin.

Test Case 0089
System: Blackjack	Phase: 3
Slots Spin Test 3
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!slots” into discord.
3. You should receive a direct message. React with the keycap 5 reaction.

Expected result:

1. Upon completing step 2, the bot should direct message the user a message that represents a slot machine. It should include the three slots, the user’s current balance, the reactions that correspond to the bet amounts, the winning combinations, and the winning amounts as a multiplier of the bet. The bot should react with all the reactions for the bet amounts as well as the red cross reaction to remove the slot machine.
2. Upon completing step 3, the user will bet $500, resulting in a balance of $99500. The winning amounts should be updated to reflect the multipliers. The bot will then spin the slots for 5 seconds.
3. Upon finishing the spin sequence, if the user won, the winning amount should be added to the user’s balance and the direct message should be edited to reflect the new balance. The message should also be edited to reflect the result of the spin.

Test Case 0090
System: Blackjack	Phase: 3
Slots Spin Test 4
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter “!slots” into discord.
3. You should receive a direct message. React with the keycap 10 reaction.

Expected result:

1. Upon completing step 2, the bot should direct message the user a message that represents a slot machine. It should include the three slots, the user’s current balance, the reactions that correspond to the bet amounts, the winning combinations, and the winning amounts as a multiplier of the bet. The bot should react with all the reactions for the bet amounts as well as the red cross reaction to remove the slot machine.

2. Upon completing step 3, the user will bet $1000, resulting in a balance of $99000. The winning amounts should be updated to reflect the multipliers. The bot will then spin the slots for 5 seconds.
3. Upon finishing the spin sequence, if the user won, the winning amount should be added to the user's balance and the direct message should be edited to reflect the new balance. The message should also be edited to reflect the result of the spin.

Test Case 0091
System: Blackjack   Phase: 3
Slots No Response Test
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter "!slots" into discord.
3. You should receive a direct message. Wait 60 seconds.

Expected result:

1. Upon completing step 2, the bot should direct message the user a message that represents a slot machine. It should include the three slots, the user's current balance, the reactions that correspond to the bet amounts, the winning combinations, and the winning amounts as a multiplier of the bet. The bot should react with all the reactions for the bet amounts as well as the red cross reaction to remove the slot machine.
2. Upon completing step 3, the user should receive a direct message indicating that the slot machine was removed due to inactivity and the message corresponding to the slot machine should be deleted.

Test Case 0092
System: Blackjack   Phase: 3
Slots Exit Test
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Enter "!slots" into discord.
3. You should receive a direct message. React with the red cross reaction.

Expected result:

1. Upon completing step 2, the bot should direct message the user a message that represents a slot machine. It should include the three slots, the user's current balance, the reactions that correspond to the bet amounts, the winning combinations, and the winning amounts as a multiplier of the bet. The bot should react with all the reactions for the bet amounts as well as the red cross reaction to remove the slot machine.
2. Upon completing step 3, the user should receive a direct message indicating that the slot machine was removed and the message corresponding to the slot machine should be deleted.

Test Case 0093

System: Steam   Phase: 3
Get steam user status
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!steam profile status <USERNAME>”

Expected result:

1. The bot responds with the online or offline status of the requested user.

Test Case 0094
System: Steam   Phase: 3
Get steam user group
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!steam profile group <USERNAME>”

Expected result:

1. The bot responds with a list of the groups of the requested user.

Test Case 0095
System: Steam   Phase: 3
Get current game of steam user
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!steam profile game <USERNAME>”

Expected result:

1. The bot responds with a game the requested user is currently playing

Test Case 0096
System: Steam   Phase: 3
Get profile card of steam user
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!steam profile card <USERNAME>”

Expected result

1. The responds with a profile card of the requested user.

Test Case 0097
System: Steam   Phase: 3
Get current game of steam user
Severity: 1
Instructions:

1. Type in !steam game <game name>

Expected result

2. Will give you a small info box with details about the game including whether it is free, the name, the steam app id

Test Case 0098

System: Steam Phase: 3

Get info on steam game

Severity: 1

Instructions:

1. Type in !steam game <game name>

Expected result

2. If there are not enough arguments there will be an error message printed out to the user

Test Case 0099

System: Steam Phase: 3

Get info on steam game

Severity: 1

Instructions:

1. Type in !steam game <game name>

Expected result

2. If there are too many arguments there will be an error message printed out to the user

Test Case 0100

System: Steam Phase: 3

Get info on steam game

Severity: 1

Instructions:

1. Type in !steam game <game name>

Expected result

1. If the game does not exist the function will print an error message

Test Case 0101

System: Steam Phase: 3

Steam Argument Error Checking

Severity: 1

Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!steam <something> <something>”

Expected result

1. Bot responds with an embed saying “Sorry that tag does not exist”

Test Case 0102

System: Steam Phase: 3

Steam Trending Games
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!steam trending”

Expected result

1. Bot responds with an embed that lists the top 10 games that are trending.

Test Case 0103
System: Steam Phase: 3
Steam Recently Released Games
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!steam recent”

Expected result

1. Bot responds with an embed that lists the top 10 recently released games on Steam that also have some popularity in playrate with their release.

Test Case 0104
System: Steam Phase: 3
Steam Top Played Games
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!steam top”

Expected result

1. Bot responds with an embed of the top 10 most played games on Steam over the last 2 weeks on Steam.

Test Case 0105
System: rpg Phase: 3
Start rpg
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg start”

Expected result:

1. The bot will notify the user the rpg game is starting.

Test Case 0106
System: rpg Phase: 3
Start rpg
Severity: 1

Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg start”
3. React to one of the emojis on the menu

Expected result:

1. The bot will output a menu of all the classes a user can choose
2. When the user has selected a class, the bot will output the class the user has chosen

Test Case 0106

System: rpg Phase: 3

Start rpg

Severity: 1

Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg start”
3. Enter a name for the character

Expected result:

1. The bot will output a menu of all the classes a user can choose
2. When the user has inputted a string, the bot will respond with the name the user has chosen

Test Case 0107

System: rpg Phase: 3

Exit rpg

Severity: 1

Instructions:

1. When the rpg is in progress, type in “!rpg exit”

Expected result:

1. The bot will notify the user that the rpg is ending and their progress is being erased.

Test Case 0108

System: rpg Phase: 3

rpg profile

Severity: 1

Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg start”
3. The bot will say prompt the user that the rpg is starting. The user will then select a class by clicking on one of the emotes.
4. Type in “!rpg myinfo”

Expected result:

1. The bot will respond with a profile card containing ID, class, and name of a player

Test Case 0109

System: rpg		Phase: 3
rpg character stats
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg start”
3. The bot will say prompt the user that the rpg is starting. The user will then select a class by clicking on one of the emotes.
4. Type in “!rpg stats me”

Expected result:

1. The bot will respond with various stats of the player’s character

Test Case 0110
System: rpg		Phase: 3
rpg character equipment
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg start”
3. The bot will say prompt the user that the rpg is starting. The user will then select a class by clicking on one of the emotes.
4. Type in “!rpg equip”

Expected result:

1. The bot will respond with the current equipment the player is using

Test Case 0111
System: rpg		Phase: 3
rpg help
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg help”

Expected result:

1. The bot will respond a help page containing info on all the rpg module commands

Test Case 0112
System: rpg		Phase: 3
rpg inventory
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg inventory”

Expected result:

1. The bot will respond a list of the character’s inventory

Test Case 0113
System: rpg Phase: 3
rpg sell item
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Assuming the player owns the item, type in “!rpg sell <ITEM-NAME>”

Expected result:

1. The bot will remove the item from the player’s inventory if it exists and add the sell price to their character’s balance.

Test Case 0114
System: rpg Phase: 3
rpg sell item
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Assuming the player doesn’t own the item, type in “!rpg sell <ITEM-NAME>”

Expected result:

1. The bot will output an error message notifying the user that they don’t own the item and the sale wasn’t successful

Test Case 0115
System: rpg Phase: 3
rpg sell item
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg sell”

Expected result:

1. The bot will output an error message notifying the user that they must specify an item name and output the correct usage of the command.

Test Case 0116
System: rpg Phase: 3
rpg item profile
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg item <ITEM-NAME>”

Expected result:

1. The bot will output the stats and description of the requested item

Test Case 0117

System: rpg Phase: 3
rpg shop menu
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg shop”

Expected result:

1. The bot will output a list of all the items a player can purchase

Test Case 0118
System: rpg Phase: 3
rpg purchasing item
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg buy <ITEM-NAME>”

Expected result:

1. The bot will subtract the item cost from the user and add the item to their inventory

Test Case 0119
System: rpg Phase: 3
rpg purchasing item
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg buy <ITEM-NAME>”

Expected result:

1. If the user’s balance is less than the item cost, the bot will output and error message notifying the user that they do not have enough money to buy the item

Test Case 0120
System: rpg Phase: 3
List of All Monsters
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg start”
3. The bot will prompt the user that the rpg is starting.
4. Choose one of the classes by selecting an emote.
5. The bot will then prompt the user with many options such as looking at all of the possible enemies.
6. Type in “!rpg monsterlist”.

Expected result:

1. The bot will respond with an embedded list of all of the possible monsters the user can encounter in the dungeon along with the stats of each monster.

Test Case 0121
System: rpg Phase: 3
List of All Monsters Sorted by Rarity
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg start”
3. The bot will prompt the user that the rpg is starting.
4. Choose one of the classes by selecting an emote.
5. The bot will then prompt the user with many options such as looking at all of the possible enemies.
6. Type in “!rpg monsterlist rarity”

Expected result:

1. The bot will respond with an embedded list of all of the possible monsters the user can encounter in the dungeon along with the stats of each monster sorted by rarity, which represents strength and chance of encountering, from least rare to most rare.

Test Case 0122
System: rpg Phase: 3
Specific Monster
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg start”
3. The bot will say prompt the user that the rpg is starting.
4. Choose one of the classes by selecting an emote.
5. The bot will then prompt the user with many options such as looking at all of the possible enemies.
6. Type in “!rpg monsterlist <MONSTER-NAME>”.

Expected result:

1. The bot will respond with an embedded information of the specific monster chosen by name with the stats of that monster alone.

Test Case 0123
System: rpg Phase: 3
Incorrect/Invalid MonsterList Tag
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg start”
3. The bot will say prompt the user that the rpg is starting.

4. Choose one of the classes by selecting an emote.
5. The bot will then prompt the user with many options such as looking at all of the possible enemies.
6. Type in “!rpg monsterlist <INVALID-TAG>”.

Expected result:

1. The bot will respond with an error message which says that the tag is unrecognized/invalid and will return to the options screen.

Test Case 0124

System: rpg Phase: 3

Combat Options Prompted by the Bot

Severity: 1

Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg start”
3. The bot will prompt the user that the rpg is starting.
4. Choose one of the classes by selecting an emote.
5. Type “!rpg dungeon <DUNGEON-NAME>”.
6. The user will be in combat with one or more monster on each floor of the dungeon.

Expected result:

1. When the user is in combat, after the bot has stated what enemies the user is facing currently, the bot will prompt the user with options of attack/defend/heal or use items/flee.

Test Case 0125

System: rpg Phase: 3

Combat Options Chosen by the User

Severity: 1

Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg start”
3. The bot will prompt the user that the rpg is starting.
4. Choose one of the classes by selecting an emote.
5. Type “!rpg dungeon <DUNGEON-NAME>”.
6. The user will be in combat with one or more monster on each stage of the dungeon.
7. The bot will prompt the user with options of attack/defend/heal or use items/flee.
8. Type “!rpg combat <OPTION>”.

Expected result:

1. If the option is attack, the user will attack the monster/s and deal damage and the bot will notify the effects of the damage on the monster/s and their statuses after the damage has been dealt to the user.
2. If the option is defend, the user will defend against the monster/s gaining temporary defensive stats and the bot will notify the user the defensive stats they have gained temporarily.

3. If the option is heal or use and item, the user will either heal if their class has that as a skill or the user will be allowed to use an item from their inventory that will grant them certain effects. The bot will notify the user of the changes in their stats.

Test Case 0126
System: rpg Phase: 3
Invalid Combat Option Chosen by the User
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg start”
3. The bot will say prompt the user that the rpg is starting.
4. Choose one of the classes by selecting an emote.
5. Type “!rpg dungeon <DUNGEON-NAME>”.
6. The user will be in combat with one or more monsters on each stage or floor of the dungeon.
7. When the user is in combat, after the bot has stated what enemies the user is facing currently, the bot will prompt the user with options of attack/defend/heal or use items/flee.
8. Type “!rpg combat <INVALID-OPTION>”.

Expected result:

1. If the user replies with an option that is not listed, the bot will respond with an error message and prompt the options they can choose from again.

Test Case 0127
System: rpg Phase: 3
Flee Option is Chosen by the User
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg start”
3. The bot will say prompt the user that the rpg is starting.
4. Choose one of the classes by selecting an emote.
5. Type “!rpg dungeon <DUNGEON-NAME>”.
6. The user will be in combat with one or more monsters on each stage or floor of the dungeon.
7. When the user is in combat, after the bot has stated what enemies the user is facing currently, the bot will prompt the user with options of attack/defend/heal or use items/flee.
8. Type “!rpg combat flee”

Expected result:

1. If the option is flee, the user will flee the dungeon returning to the dungeon choosing screen with the current stats they have but not any rewards they would have earned for

clearing the monsters they were fighting. Their progress will be saved if any was made throughout the dungeon.

Test Case 0128
System: rpg Phase: 3
User Selects a Dungeon
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg start”
3. The bot will say prompt the user that the rpg is starting.
4. Choose one of the classes by selecting an emote.
5. The bot will then prompt the user with many options such as a list of dungeons the user can enter
6. Type in “!rpg dungeon <DUNGEON-NAME>”

Expected result:

1. The bot will respond with a message that verifies that you have entered a dungeon with the specific dungeon name and starts the combat sequence for that first floor.

Test Case 0129
System: rpg Phase: 3
User Selects an Invalid Dungeon
Severity: 1
Instructions:

7. To start the bot, at the console, enter: python3 main.py
8. Type in “!rpg start”
9. The bot will say prompt the user that the rpg is starting.
10. Choose one of the classes by selecting an emote.
11. The bot will then prompt the user with many options such as a list of dungeons the user can enter
12. Type in “!rpg dungeon <INVALID-DUNGEON-NAME>”

Expected result:

2. The bot will respond with an error message that states that the dungeon trying to be accessed does not exist.

Test Case 0130
System: rpg Phase: 3
User Clears a Floor of the Dungeon
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg start”.
3. The bot will say prompt the user that the rpg is starting.
4. Choose one of the classes by selecting an emote.

5. The bot will then prompt the user with many options such as a list of dungeons the user can enter.
6. Type in “!rpg dungeon <DUNGEON-NAME>”.
7. Clear a floor of the dungeon.

Expected result:

1. The bot will verify that you cleared the floor and state your rewards for that floor and move on to the next one automatically having the user fight more enemies of possible different types.

Test Case 0131

System: rpg Phase: 3

User Gets Loot after Clearing a Floor

Severity: 1

Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg start”
3. The bot will say prompt the user that the rpg is starting.
4. Choose one of the classes by selecting an emote.
5. The bot will then prompt the user with many options such as a list of dungeons the user can enter
6. Type in “!rpg dungeon <DUNGEON-NAME>”
7. Clear a floor of the dungeon

Expected result:

1. The bot will verify that you cleared the floor and state your rewards for that floor. These rewards include a flat amount of currency as well as possible dropped loot from the monsters that is additional currency, if any.

Test Case 0132

System: rpg Phase: 3

RPG Leaderboard

Severity: 1

Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg start”
3. The bot will say prompt the user that the rpg is starting.
4. Choose one of the classes by selecting an emote.
5. The bot will then prompt the user with many options such as a list to see the leaderboard of rpg clears.
6. Type “!rpg leaderboard”.

Expected result:

1. The bot will show the leaderboard of the top 3 users + fastest clear times of the rpg game including all of the dungeons.

Test Case 0133

System: rpg Phase: 3
RPG Dungeon Leaderboard
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg start”
3. The bot will say prompt the user that the rpg is starting.
4. Choose one of the classes by selecting an emote.
5. The bot will then prompt the user with many options such as a list to see the leaderboard of rpg clears.
6. Type “!rpg leaderboard <DUNGEON-NAME>”.

Expected result:

1. The bot will show the leaderboard of the top 3 users + fastest clear times of the rpg dungeon with the specified dungeon name.

Test Case 0134
System: rpg Phase: 3
RPG Leaderboard Replacement
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg start”
3. The bot will say prompt the user that the rpg is starting.
4. Choose one of the classes by selecting an emote.
5. The bot will then prompt the user with many options such as a list to see the leaderboard of rpg clears.
6. Type “!rpg dungeon <DUNGEON-NAME>” until completion of all of the dungeons.
7. Clear all of the dungeons faster than any time on the leaderboard.

Expected result:

1. The bot will prompt the user for a username that will be entered to the leaderboard.
2. The leaderboard for the rpg will be updated adding the newest time in its correct place and moving places for the other times.

Test Case 0135
System: rpg Phase: 3
RPG Dungeon Leaderboard Replacement
Severity: 1
Instructions:

1. To start the bot, at the console, enter: python3 main.py
2. Type in “!rpg start”
3. The bot will say prompt the user that the rpg is starting.
4. Choose one of the classes by selecting an emote.

5. The bot will then prompt the user with many options such as a list to see the leaderboard of rpg clears.
6. Type <!rpg dungeon <DUNGEON-NAME>
7. Clear a dungeon faster than any time on the leaderboard.

Expected result:

1. The bot will prompt the user for a username that will be entered to the leaderboard.
2. The leaderboard for the dungeon will be updated adding the newest time in its correct place and moving places for the other times.