Lucid chart:-

https://lucid.app/lucidchart/98319ecc-9917-4945-b386-7ed5e22edae0/edit?viewport_loc=-1081%2C-2588%2C5785%2C2357%2C0_0&invitationId=inv_ee33032b-9f17-4695-8f0f-bf9ea047526f

Changes in the model:

- I have removed the usages of pet interface and pet from the model. You will still find the interface and concrete class. However, it is not being utilized by the any functions in the model.
- I have added a getMappingOfSpaceAndPlayers function in the model needed to get the mapping between the spaces and players.
- I have added the function reload() in the DrLuckyWorld and to load the world specification in from the file.

Diagrams:

At the end.

Milestone 4 Test cases:

Test	Input	Expected values
Loading new game with new specification	startNewGameWithNewWorld	Loaded new game successfully
Test starting new game function with new world	loadNewGame("res/mansion.txt", 3)	View log contains "Adding new world specification !" and Model log contains "Model reset"
Test advancing the target character whenever required by the computer player	advanceTargetCharacter()	Model log contains "called advance target character" and "called next turn!"
Testing reset of the game	resetGame()	View log contains "Resetting the game!" and Model log contains "Model reset"
Testing the whether controller is setting the view to be true and controller is calling the view	playGame()	View log contains "setting the about dialog panel visible" and "setting the view main panel visible"
Testing the process input function when supplied with invalid max capacity	processInput("human", ["v", "-3", "Billiard Room"])	View log contains "ERROR" and "Max capacity cannot be negative!" IllegalArgumentException with appropriate error message
Testing the process input function when supplied with invalid username	processInput("human", ["", "3", "Billiard Room"])	View log contains "Username cannot be empty!" IllegalArgumentException with appropriate error message
Testing the processinput for invalid space name	processInput("human", ["v", "3", "abc room"])	View log contains "Room name is invalid!" IllegalArgumentException with appropriate error message
Testing the isValidMove function for the whether it checks for the valid move	isValidMove(Player("abc", 1), Space("Garden"))	Model log contains "getting neighbors!" and "Checking valid move!" IllegalArgumentException with appropriate error message

Test if the game has ended and controller disallows further actions	processInput()	Game ending message displayed in log
Test the player description	processInput(“playerinfo”, new String[]{v})	Log contains the player details
Test if it's the computer player's turn is simulated	simulateAction()	Computer player makes a move
Testing the space information for retrieving the space information	processInput(“spaceinfo”,new String[]{}	Execute the retriving of space information
Test the look around testLookAround in the command design pattern	processInput("lookaround", [])	Execute the look around method and displays the appropriate message
Test adding the moving human player in the game	processInput ("move", ["Drawing Room"])	Move player success message
Computer Player Info:- Verify computer player information "computer", [] Verify computer player details Comp. Player Pred. Num Test computer player behavior with prediction testComputerPlayerWithPredictableNumbers "computer", [] Verify computer player behavior	simulateAction()	Testing the random behavior of computer player
A Invalid or null Model is the argument	playGame(null)	IllegalArgumentException
Testing an invalid number of max turns	-21	Invalid turns
When turns are greater or equal to MaxTurn and drlucky is killed		Display game over for if turns over for a particular player
Testing DisplaySpaceInfo Throws an exception when not able to display space information	execute(World world, int maxTurns)	Exception
Testing AddHumanPlayerCommand()	execute(World world, int maxTurns)	IllegalArgumentException

For an invalid input to throw error		
Testing PickItem throws exception for wrong name item passed as input	execute(World, int maxTurns)	IllegalArgumentException
Testing MovePlayer throws an exception for an invalid move	execute(World world, int maxTurns)	IllegalArgumentException
Testing player description	execute(World world,int maxTurns)	String containing Player Description
Testing space description	execute(World world,int maxTurns)	String containing Space Description
Testing addHumanPlayer()	execute(World world, int maxTurns)	"Human Player added"
Testing addComputerPlayerCommand ()	execute(World world, int maxTurns)	"Computer Player added"
Testing PickItemCommand successfully	execute(World,int maxTurns)	"Item picked successfully"
Testing movePlayerCommand() for the move Command execution	execute(World world, int maxTurns)	"The player's character should move up on the game board."
Testing player description	execute(World world,int maxTurns)	String containing Player Description
Testing Lookaround	Execute(world, int maxTurns)	String description getting the lookaround

Milestone 3: Test Cases:

https://lucid.app/lucidchart/673deac0-17a7-4fbe-b30f-a41a08f6fd8e/edit?viewport_loc=-6485%2C-4059%2C16652%2C6861%2C0_0&invitationId=inv_274df6a1-3797-45e2-8f18-222b96300338

DrLuckyWorldClass

Test	Input	Expected Values

Test the attack() method by calling attackItem method.	Attack()
	getTargetCharacterDetails()
		Reduction in health and update in the toString of the target Character. Moreover, getHealth should return updated health.
		
		Character Information (Character Name = Lucky, Character Health = 48, " +
		"Character is Target = true)

Test the attack() method by calling attackPoke method.	Attack()
	getTargetCharacterDetails()
		Reduction in health and update in the toString of the target Character. Moreover, getHealth should return updated health.

		Character Information (Character Name = Lucky, Character Health = 49, " + "Character is Target = true)
Test if the computer player performs a poke if no item exists	AttackTargetComputer	Target Character poke by computer player as the output.
Test if the attack is performed with an item by human player.	Attack()	Target character has attacked with an item and that item has been added to evidence list.
Test if during human player's turn, it tries to attack a target character when it is not present in the room	Attack()	IllegalStateException
Testing computer attack by calling the simulate Action	simulateAction()	prevAction should be attack after call of simulate action. Testing using the random interface it should be same as expected.
Testing the computer move by calling the simulate action	simulateAction()	prevAction should be move after call of simulate action. Testing using the random interface it should be same as expected.
Testing the computer pickitem by calling the simulate action	simulateAction()	prevAction should be item after call of simulate action. Testing using the random interface it should be same as expected.
Testing the move pet function	petMove(String spaceName)	Player move the to the space. Pet description should change Pet Name: Dr Fortune, Space: Armory
Testing the display of getTargetCharacter() details	getTargetCharacterDetails()	Health: 50 Name: DrLucky
Test if the target character's pet	getSpaceInfo()	"Space Information (Space Name = Armory, WorldPosition UpperLeft Row = 22,

information is present in the space info.		WorldPosition UpperLeft Column = 23, WorldPosition LowerRight Row = 25, WorldPosition LowerRight Column = 26, Items = Name = Knife, Damage Value= 26) Player in Room: (Name= Vaishnavi, Item in Hand = No item) Pet Info: petName: Dr Fortune Cat, petSPace: Armory”
Test if a player is able to look into the space with the target character pet’s present in this space during lookAround action	lookAround()	Displays a list of spaces that are visible or seen, excluding the one in which the pet is present.
Test the move pet command to invalid space	petMove(String spaceName)	IllegalArgument Exception
Test if move pet command is counted as a turn for the players action	petMove(String spaceName)	numberOfTurns has increased.
Test when the it is the turn of the computer player turns then it simulates the attack command prior to other commands if the target character is in the same room.	simulateAction() AttackTargetComputer()	prevAction should be Attack. Indicating that attack was performed
Test if during the computer player’s and it tries to attack if the target character is not present in the space.	simulateAction() AttackTargetComputer()	illegal State Excpetion
Test the movePet command to an invalid space	movePetSpace(String spaceName)	illegalArgumentException
Test the movePet command to a valid space	movePetSpace(String spaceName)	Pet Moved successfully
Test if the attack to the target character	Attack()	No Reduction in health or upating of the toString of the

is seen by any player, it is disregarded and no damage is done		target Character. Moreover, getHealth should return same health. Character Information (Character Name = Lucky, Character Health = 50, " + "Character is Target = true)
Test if the attack to the target character is seen by any player, it is disregarded and no damage is done and the evidence list is not updated.	Attack()	evidenceList size has not increased.

Test if the attack to the target character
is seen by any player, it is disregarded
and no damage is done and the space info is not updates
	Attack()
	printSpaceInfo()
		The space information has not been updated and still continues to have the item information.

Test if the target character's starting position is same as that of the target	printSpaceInfo (Armory)	Space details should include pet details as well. "Space Information (Space Name = Armory, WorldPosition UpperLeft Row = 22, WorldPosition UpperLeft Column = 23, WorldPosition LowerRight Row = 25, WorldPosition LowerRight Column = 26, Items = Name = Knife, Damage Value= 26) Player in Room: (Name= Vaishnavi, Item in Hand = No item) Pet Info: petName: Dr Fortune Cat, petSPace: Armory"
Testing if the target character pet's description is present in the space description	printSpaceInfo(String spaceName)	Space details should include pet details as well. "Space Information (Space Name = Armory, WorldPosition UpperLeft Row = 22, WorldPosition UpperLeft Column = 23, WorldPosition

		LowerRight Row = 25, WorldPosition LowerRight Column = 26, Items = Name = Knife, Damage Value= 26) Player in Room: (Name= Vaishnavi, Item in Hand = No item) Pet Info: petName: Dr Fortune Cat, petSPace: Armory”
During the computer player's turn, can we confirm that if both the computer player and the target character are present in the same space, the computer player will select the item with the highest damage to attack the target character	getEvidenceList()	getEvidenceList() function returns the getMaxDamageValueItem as added item
Test after attack the item is added to the evidence list.	getEvidenceList()	getEvidenceList() size has increased,
Test after attack the item is removed from the space	printSpaceInfo(String spaceName)	The item is removed from the spaceDescription and print no items.
Test if the target character dies and game ends.	hasGameEnded()	TargetCharacter is dead!!
Test if the target character dies and game ends.	hasGameEnded()	Max turns exhausted.
Test if the attack target character is counted as a turn	Attack()	numberOfTuurns has increment by one.

TargetCharacter

Test	Input	Expected value
Testing if the reduce health function works as expected	reduceHealth(10)	(Assumed initial helath is 50) 40

Testing if the getName of the target character.	getName()	Dr Lucky

PetCharacter

Test	Input	Expected value
Testing if the pet Name is displayed as expected	getPetName ()	Dr Fortune Cat

Game Console Controller:

Test	Input	Expected value
Testing MovePetCommand() is called and moves the player.	execute(World, Appendable, String)	Check if the pet is moved successfully and check if the command is called using the mock model testing.
Testing the AttackTargetCommand	execute(World, Appendable, String)	Target character attack command called successfully.

Milestone 2:-

World Class Table: -

Test	Input	Expected Values
Testing the world object Creation for valid parameters	world = new DrLuckyWorld(12, 8, "Dr Lucky Mansion", new GameCharacter(50, "Lucky"), new ArrayList<>(Arrays.asList(spaceOne, spaceTwo, spaceThree)));	World object created successfully
Testing the getting the name of the world using getName()	world = new DrLuckyWorld(12, 8, "Dr Lucky Mansion", new GameCharacter(50, "Lucky"), new ArrayList<>(Arrays.asList(spaceOne, spaceTwo, spaceThree)));	Dr Lucky Mansion

Testing the number of rows using getRows()	world = new DrLuckyWorld(12, 8, "Dr Lucky Mansion", new GameCharacter(50, "Lucky"), new ArrayList<>(Arrays.asList(spaceOne, spaceTwo, spaceThree)));	12
Testing the number of columns using getColumns()	world = new DrLuckyWorld(12, 8, "Dr Lucky Mansion", new GameCharacter(50, "Lucky"), new ArrayList<>(Arrays.asList(spaceOne, spaceTwo, spaceThree)));	8
Test the total number of spaces using getTotalSpaces()	world = new DrLuckyWorld(12, 8, "Dr Lucky Mansion", new GameCharacter(50, "Lucky"), new ArrayList<>(Arrays.asList(spaceOne, spaceTwo, spaceThree)));	3
Test the getSpaces() that exist in the world.	world = new DrLuckyWorld(12, 8, "Dr Lucky Mansion", new GameCharacter(50, "Lucky"), new ArrayList<>(Arrays.asList(spaceOne, spaceTwo, spaceThree)));	Space Info: Space Information (Space Name = Armory, WorldPosition UpperLeft Row = 22, WorldPosition UpperLeft Column = 19, WorldPosition LowerRight Row = 23, WorldPosition LowerRight Column = 26, Items = [Item Information (Item Name = Billiard Cue, Damage Value = 2)])
Test move target player to next space	world = new DrLuckyWorld(12, 8, "Dr Lucky Mansion", new GameCharacter(50, "Lucky"), new ArrayList<>(Arrays.asList(spaceOne, spaceTwo, spaceThree))); world.moveTargetCharacter()	2

Testing the function getNeigbhors() for space	world = new DrLuckyWorld(12, 8, "Dr Lucky Mansion", new GameCharacter(50, "Lucky"), new ArrayList<>(Arrays.asList(spaceOne, spaceTwo, spaceThree))); world.getNeighbors(Space space)	Neighbors are: Armory, Billiard,
Testing if it returns the correct players list after asking for getPlayers()	world.getPlayers()	Player 1: Vaishnavi, Player 2: Neha
Test addHumanPlayer()	world.addHumanPlayer("vaishnavi", 8, Armory)	Player added successfully in game.
Test move() for a valid space name	world.move(String name)	Player moved successfully to space.
Move a player from the space which has no neighbors	World.move(String)	IllegalArgumentException
Move a player to a space that does not exist	world.move(String)	IllegalArgumentException
Move a player to a space that is not a neighbor	world.move(String name)	IllegalArgumentException
Test pickitem() for a valid item name	world.pickitem (String name)	Player picked item successfully from a space
Test pickitem() from the space which has no items	world.pickitem (String name)	IllegalArgumentException
Test pickitem() from the space does not exist	world.pickitem (String name)	IllegalArgumentException
Move a player to a space that is not a neighbor	world.move(String name)	IllegalArgumentException
Pass null while adding a player	World. addHumanPlayer (null,null,null)	NullPointerException

Item Class table

Test	Input	Expected
Checking if the item object is created Create an item object	DrLuckyItems(Sword, 10)	Item Details: Item Name: Sword, Damage Value: 10
Create an item object with invalid values	DrLuckyItems("",8)	IllegalArgumentException
Create an item object with invalid values	DrLuckyItems("",-8)	IllegalArgumentException
Test getName()	DrLuckyItems(Sword, 10)	Sword
Test getDamageValue	DrLuckyItems(Sword, 10)	10
Test toString()	DrLuckyItems(Sword, 10)	Item Information (Item Name = Sword, Damage Value = 10)
Test equals	DrLuckyItems(Sword, 10), DrLuckyItems(Sword,10)	TRUE
Test equals	DrLuckyItems(Sword, 10), DrLuckyItems(Knife,7)	TRUE

Space Class

Test: Adding Items to the list of items in the space.	addItemToSpace(new Item(“Poison Potion”,100), new Item(“Key”,10))	Item 1: Name: "Health Potion", Type: "Potion" Item 2: Name: "Golden Key", Type: "Key"
Name of the space	DrLuckySpace(“Wine room”,22,23,24,56)	Wine Room
Position of the room	DrLuckySpace(“Wine room”,22,23,24,56)	[22,23,24,56]
Testing Upper left corner values greater than lower right corner values	DrLuckySpace(“Wine room”,29,65,24,56)	IllegalArgumentException
Testing Negative Upper left corner values and lower right corner values	DrLuckySpace(“Wine room”,-22,-23, -24, -56)	IllegalArgumentException
Testing all the items in the space	Space.getItems() A space object which calls the getItems()	List<Item> : a list of items

Get the space information	Space.toString()	"Space Information (Space Name = Armory, WorldPosition UpperLeft Row = 22, WorldPosition UpperLeft Column = 23, WorldPosition LowerRight Row = 25, WorldPosition LowerRight Column = 26, Items = Name = Knife, Damage Value= 26) Player in Room: (Name= Vaishnavi, Item in Hand = No item)
Testing removeItem from space	removeItem(new Item("Knife",23))	Item removed from the room

Player Class

Test	Input	Expected Value
Testing the getName() for the player	AbstractPlayer(vaishnavi)	vaishnavi
Testing if the maximum number if turns are exhausted isMaxTurnsExhausted()	maxTurns = 10 countTurn = 10	TRUE
Testing the current space using getCurrentSpace()	Space space = new Space("Billiard Room", Positionion position) player.move(Space space) player.getCurrentSpace()	Player's Space: (Billiard Room)
Testing if the remaining turns are checked remainingTurns()	player.remainingTurns	8
Test getItems() for a particular space	Player.getItems()	Item Information:(Item Name: Sword, Damage Value: 10)
Test addItem – this is picking up the item.	Item item = new DrLuckyItems("Javelin", 100)	Item Information:(Item Name: Sword, Damage Value: 10)

Tests the description of the player	Player.getDescription()	String consisting the player description

TargetCharacter

Test	Input	Expected value
Test getHealth()	new TargetCharacter*(“Lucky”,50)).getHealth	50
Testing the creation object for negative health	new TargetCharacter*(“Lucky”,-50))	IllegalArgumentException
Testing the creation of object for zero health	new TargetCharacter*(“Lucky”,0))	IllegalArgumentException
Testing the creation of the target for no name	new TargetCharacter*(“”,0))	IllegalArgumentException

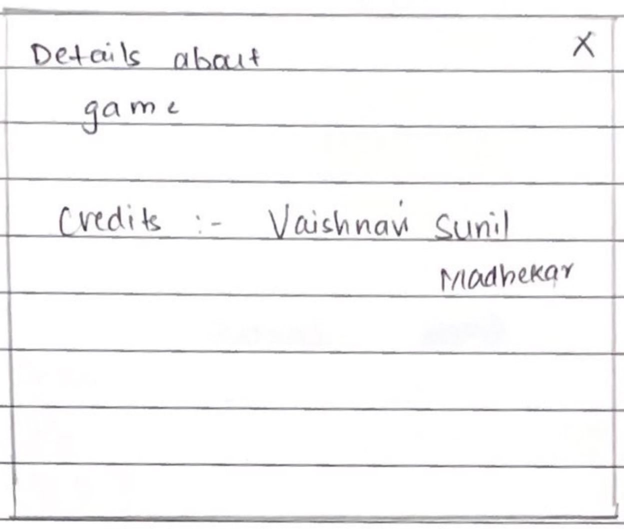
Game Console Controller:

A Invalid or null Model is the argument	playGame(null)	IllegalArgumentException
Testing an invalid number of max turns	-21	Invalid turns
When turns are greater or equal to MaxTurn and drlucky is killed		Display game over for if turns over for a particular player
Testing DisplaySpaceInfo Throws an exception when not able to display space information	execute(World world, int maxTurns)	Exception
Testing CreateGraphicalRep throws exception for when not able to create graphical representation	execute(World world, int maxTurns)	Exception
Testing AddHumanPlayerCommand() For an invalid input to throw error	execute(World world, int maxTurns)	IllegalArgumentException
Testing AddComputerPlayer For an invalid input to throw error	execute(World world, int maxTurns)	IllegalArgumentException
Testing PickItem throws exception for wrong name item passed as input	execute(World, int maxTurns)	IllegalArgumentException

Testing MovePlayer throws an exception for an invalid move	execute(World world, int maxTurns)	IllegalArgumentException
Testing player description	execute(World world,int maxTurns)	String containing Player Description
Testing space description	execute(World world,int maxTurns)	String containing Space Description
Testing DisplayGraphicalRepresentationCommand() displays grid is successful	execute(World world, int maxTurns)	Check if image saved as PNG
Testing addHumanPlayer()	execute(World world, int maxTurns)	“Human Player added”
Testing addComputerPlayerCommand()	execute(World world, int maxTurns)	“Computer Player added”
Testing PickItemCommand successfully	execute(World,int maxTurns)	“Item picked successfully”
Testing movePlayerCommand() for the move Command execution	execute(World world, int maxTurns)	“The player's character should move up on the game board.”
Testing player description	execute(World world,int maxTurns)	String containing Player Description

netsi Northeastern University
Network Science Institute

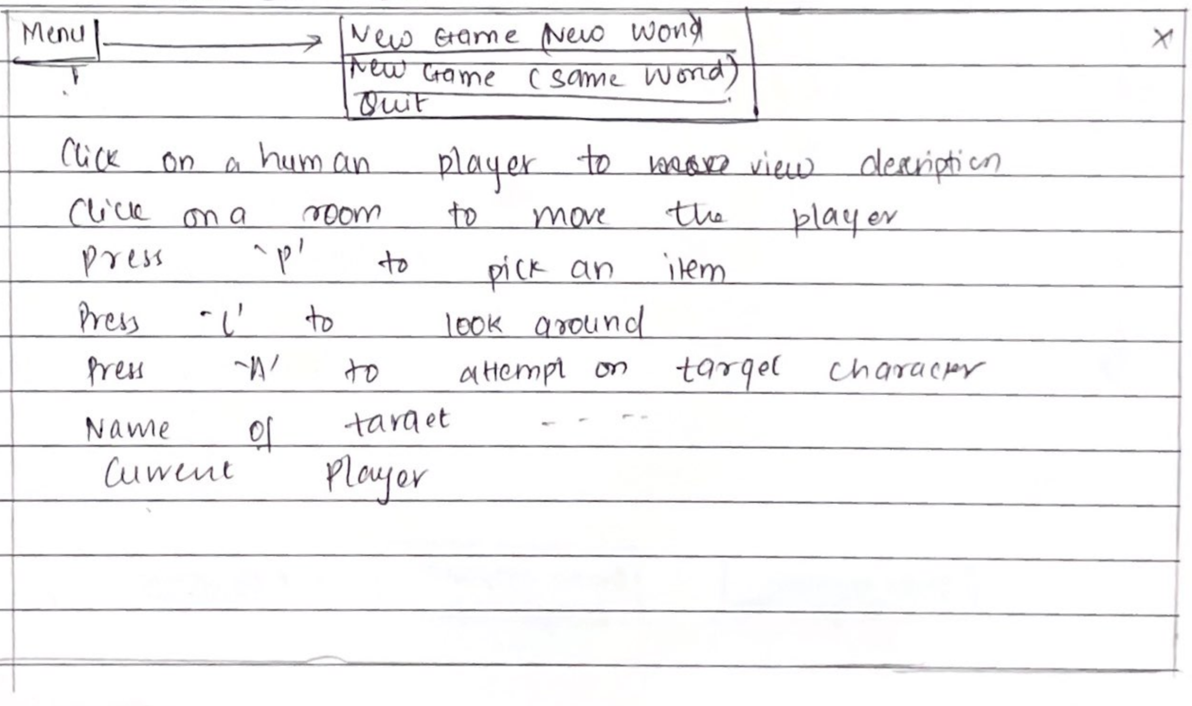
Beginning Dialog Box



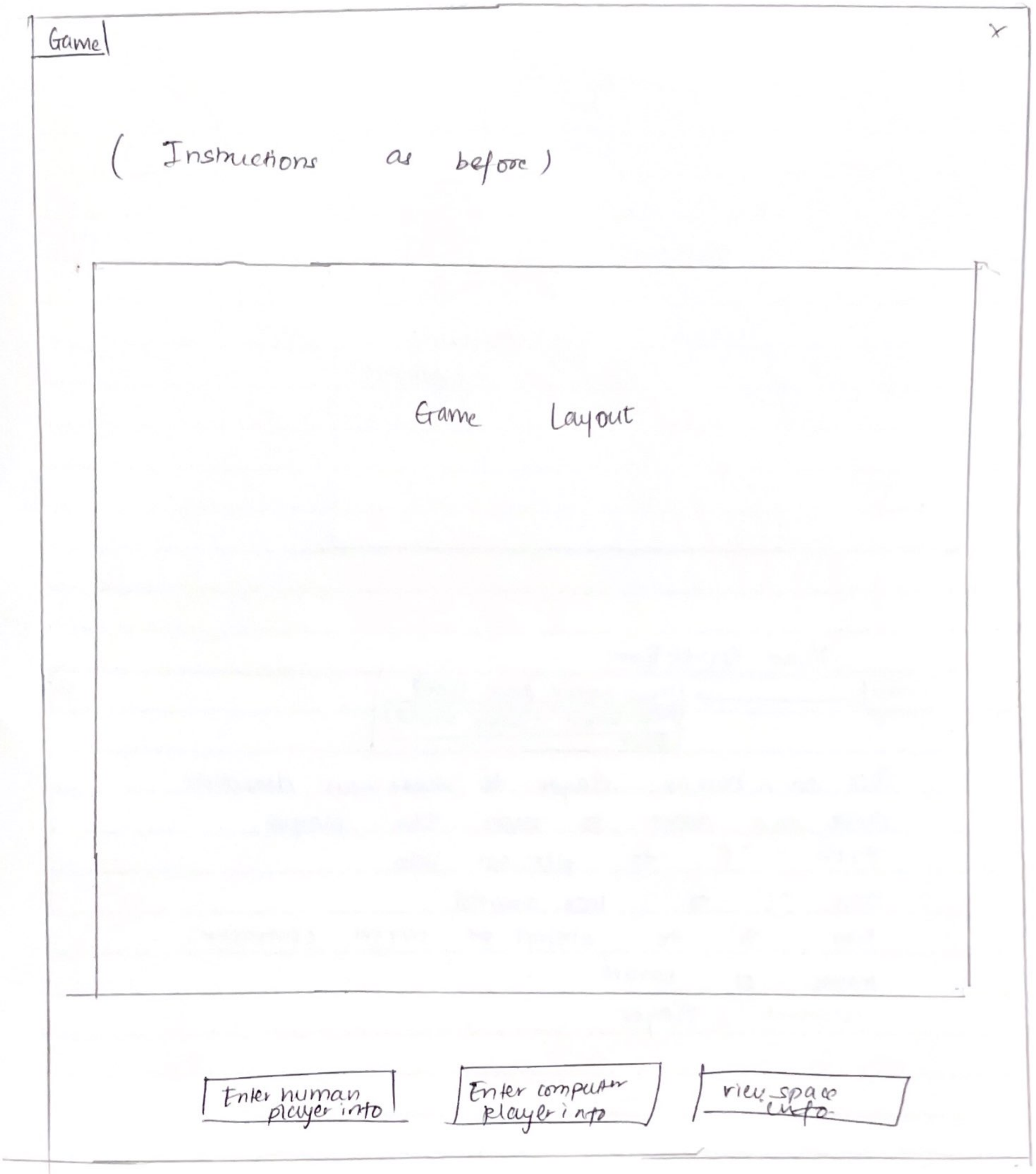
Details about game

Credits :- Vaishnavi Sunil Madhekar

Main ScreenBox



Click on a human player to ~~move~~ view description
Click on a room to move the player
Press 'p' to pick an item
Press 'l' to look around
Press 'A' to attempt on target character
Name of target - - -
Current Player


Game
x
(Instructions as before)
Game Layout
Enter human player info
Enter computer player info
view space info