Class	Attribute	Method
Player	**+Name**: • **Description:** Stores the name of the player. • **Type:** String **+Progress**: • **Description:** Tracks the player's progress in the game, including unlocked levels. • **Type:** Dictionary or data structure to store level status. **+Inventory**: • **Description:** Stores the collected samples by the player. • **Type:** List or dictionary to manage collected items. **+Points**: • **Description:** Tracks the points earned by the player. • **Type:** Integer **+Currency**: • **Description:** Tracks the amount of Basalt currency collected by the player. • **Type:** Integer **+CurrentLevel**: • **Description:** Keeps track of the current level the player is on. • **Type:** Reference to level	**+set_name(name: str) -> None**: • **Description:** Sets the name of the player. **+update_progress(level: int, status: str) -> None**: • **Description:** Updates the progress of the player in the game (unlocking levels). **+add_to_inventory(RegionalObject reference) -> None**: • **Description:** Adds a collected sample to the player's inventory. **+update_points(points: int) -> None**: • **Description:** Updates the player's points based on certain actions/events. **+update_currency(amount: int) -> None**: • **Description:** Updates the player's Basalt currency based on collection or transactions. **+start_level(level: reference) -> None**:

	object. **+CurrentTime**: ● **Description:** Tracks the current time elapsed in the current level. ● **Type:** Integer or float (representing seconds or minutes).	● **Description:** Initiates a new level for the player to play. **+complete_level(time_taken: int) -> None**: ● **Description:** Handles actions when the player successfully completes a level within the time limit. **+encounter_failure() -> None**: ● **Description:** Handles actions when the player fails to complete a level due to dust storms or meteoroids. **+attempt_puzzle(puzzle:reference, answer: str) -> str**: ● **Description:** Handles the player's attempt to solve a puzzle and returns feedback (correct or incorrect). **+buy_item(item: reference) -> None**: ● **Description:** Allows the player to buy items from the store using Basalt currency. **+show_inventory() -> None**:

		• **Description:** Displays the player's inventory with collected samples/items. **+handle_time_limit() -> None**: • **Description:** Manages actions when the player exceeds the time limit for a level. **+handle_menu_option(option: str) -> None**: • **Description:** Handles the player's selection from the main menu. **+choose_shovel(shovel_type: str) -> None**: • Description: Allows the player to select a shovel from the inventory. **+use_shovel(object_name: reference) -> None** • **Description:** Initiates the extraction process for a given object using the selected shovel. **+choose_suit(suit_type: str) -> None:** • **Description:** Allows the player to select a suit from

		the inventory. **+use_suit(suit: reference) -> None:** • **Description:** Utilizes the benefits of the currently equipped suit during gameplay.
RegionalObject	**+Name**: • **Description:** Stores the name of the object. • **Type:** String **+Level**: • **Description:** Indicates the level in which the object is found. • **Type:** Integer or reference to level object. **+ExtractionTime**: • **Description:** Represents the time required to extract information from the object. • **Type:** Integer or float (representing seconds or minutes). **+Type**: • **Description:** Indicates the type/category of the object (e.g., Basalt, Shergottites, etc.). • **Type:** String or enumeration representing object types.	**+get_name() -> str**: • **Description:** Retrieves the name of the object. **+get_level() -> int**: • **Description:** Retrieves the level in which the object is found. **+get_extraction_time() -> int**: • **Description:** Retrieves the extraction time required for the object. **+get_type() -> str**: • **Description:** Retrieves the type/category of the object. **+set_name(name: str) -> None**: • **Description:** Sets the name of the object.

	+AmountCollected ● **Description:** Number of the object collected by the player. ● **Type**: Integer	**+set_level(level: int) -> None**: ● **Description:** Sets the level in which the object is found. **+set_extraction_time(time: int) -> None**: ● **Description:** Sets the extraction time required for the object. **+set_type(object_type: str) -> None**: ● **Description:** Sets the type/category of the object. **+getSample()->reference to itself:** ● **Description:** After extraction, returns reference to itself as a sample to be added to the inventory. **+analyze() -> str**: ● **Description:** Initiates the analysis process for the object and returns the analysis results for the level it is in. **+interact() -> None**: ● **Description:** Represents the interaction of the player with the object, triggering extraction and

		analysis processes. **+increase_count()->None** • **Description:** Increase value of the amount of object collected by the player while interaction.
Shovel	**+Color**: • **Description:** Represents the color of the shovel. • **Type:** String **+Speed**: • **Description:** Indicates the extraction speed multiplier provided by the shovel. • **Type:** Integer or float **+Price**: • **Description:** Represents the price of the shovel in Basalt currency. • **Type:** Integer	**+get_color() -> str**: • **Description:** Retrieves the color of the shovel. **+get_speed() -> int**: • **Description:** Retrieves the extraction speed multiplier provided by the shovel. **+get_price() -> int**: • **Description:** Retrieves the price of the shovel. **+set_color(color: str) -> None**: • **Description:** Sets the color of the shovel. **+set_speed(speed: int) -> None**: • **Description:** Sets the extraction speed multiplier provided by the shovel. **+set_price(price: int) ->**

		None: ● **Description:** Sets the price of the shovel. **+buy_shovel(player_currency: int) -> bool**: ● **Description:** Allows the player to purchase the shovel if they have enough Basalt currency. Returns True if purchase is successful, False otherwise. **+use() -> None**: ● **Description:** Allows the player to equip and use the shovel during gameplay, affecting the extraction speed of objects. **+get_shovel()->reference:** ● **Description :** Returns reference to itself to be added to the store.
Suit	**+Type**: ● **Description:** Indicates the type or level of the suit. ● **Type:** String or enumeration representing suit types.	**+get_type() -> str**: ● **Description:** Retrieves the type of the suit. **+get_color() -> str**: ● **Description:** Retrieves the color

	+Color: ● **Description:** Represents the color of the suit. ● **Type:** String or enumeration representing suit colors. **+SpeedMultiplier**: ● **Description:** Represents the speed multiplier for the player while wearing the suit. ● **Type:** Integer or float representing speed multiplier. **+JumpForce**: ● **Description:** Represents the force or height multiplier for the player's jumps while wearing the suit. ● **Type:** Integer or float representing jump force multiplier. **+Price**: ● **Description:** Represents the price of the suit in Basalt currency. ● **Type:** Integer	of the suit. **+get_speed_multiplier() -> int**: ● **Description:** Retrieves the speed multiplier of the suit. **+get_jump_force() -> int**: ● **Description:** Retrieves the jump force multiplier of the suit. **+get_price() -> int**: ● **Description:** Retrieves the price of the suit. **+use_suit() -> None**: ● **Description:** Equips the suit for the player to use during gameplay, enhancing their speed and jump abilities. **+buy_suit(player_currency: int) -> bool**: ● **Description:** Allows the player to purchase the suit if they have enough Basalt currency. Returns True if purchase is successful, False otherwise. **+get_suit()->reference:**

		• **Description :** Returns reference to itself to be added to the store.
Store	**+AvailableSuits**: • **Description:** A list or dictionary containing references to available suits in the store. • **Type:** List or dictionary of Suit objects. **+AvailableShovels**: • **Description:** A list or dictionary containing references to available shovels in the store. • **Type:** List or dictionary of Shovel objects.	**+add_suit(suit: Suit) -> None**: • **Description:** Adds a new suit to the available suits in the store. **+add_shovel(shovel: Shovel) -> None**: • **Description:** Adds a new shovel to the available shovels in the store. **+display_available_suits () -> None**: • **Description:** Displays the available suits for purchase in the store, showing details such as type, color, speed multiplier, jump force, and price. **+display_available_shovels() -> None**: • **Description:** Displays the available shovels for purchase in the store, showing details such as type, color, speed multiplier, and price. **+buy_item(item_name:**

		str, player_currency: int) -> bool: ● **Description:** Allows the player to purchase an item from the store by providing the item's name and the player's available currency. Returns True if the purchase is successful, False otherwise. **+handle_menu_option(option: str) -> None**: ● **Description:** Handles the player's selection from the store menu, allowing them to view available items and make purchases.
Inventory	**+CollectedObjects**: ● **Description:** A list or dictionary containing references to the regional objects collected by the player. ● **Type:** List or dictionary of RegionalObject objects.	**+add_object(obj: RegionalObject) -> None**: ● **Description:** Adds a new regional object to the inventory. **+remove_object(obj: RegionalObject) -> None**: ● **Description:** Removes a regional object from the inventory. **+display_inventory() ->**

		None: **Description:** Displays the collected regional objects in the inventory.
Puzzle	**+Level**: ● **Description:** The level in which the puzzle is present. ● **Type:** Integer **+Region**: ● **Description:** The region where the puzzle is located. ● **Type:** String **+PuzzleText**: ● **Description:** The text describing the puzzle. ● **Type:** String **+PossibleAnswers**: ● **Description:** The possible answers to the puzzle. ● **Type:** List of strings **+CorrectAnswer**: ● **Description:** The correct answer to the puzzle. ● **Type:** String **+Completed**: ● **Description:** Indicates whether the puzzle has been solved. ● **Type:** Boolean	**+display_puzzle() -> None**: ● **Description:** Displays the puzzle text to the player. **+check_answer(player_answer: str) -> str**: ● **Description:** Checks if the player's answer matches. Returns a message indicating whether the answer is correct or not. **+check_inventory(inventory: Inventory) -> bool**: ● **Description:** Checks if the player has collected all the required samples for solving the puzzle by accessing the inventory. **+is_completed() -> bool**: ● **Description:** Checks if the puzzle has been solved.Returns true if the puzzle is solved, False

		otherwise. **+disable_puzzle() -> None**: ● **Description:** Disables the puzzle icon once it's solved. **+get_level() -> int/reference**: ● **Description:** Retrieves the level of the puzzle. **+get_region() -> str**: ● **Description:** Retrieves the region where the puzzle is located. **+get_puzzle()->reference :** ● **Description:** Returns reference of itself.
Level	**+LevelNumber**: ● **Description:** The number or identifier of the level. ● **Type:** Integer **+Region**: ● **Description:** The region where the level is located. ● **Type:** String **+TimeLimit**: ● **Description:** The time	**+unlock_level() -> None**: ● **Description:** Unlocks the level for the player. **+complete_level() -> None**: ● **Description:** Marks the level as completed. **+add_puzzle(puzzle: Puzzle) -> None**: ● **Description:** Adds a puzzle to the

	limit for completing the level. • **Type:** Integer (seconds) **+AvailablePuzzles**: • **Description:** List of puzzles available in the level. • **Type:** List of Puzzle objects **+Completed**: • **Description:** Indicates whether the level has been completed. • **Type:** Boolean **+Unlocked**: **Description:** Indicates whether the level is unlocked for the player. **Type:** Boolean **+HideoutLocation**: • **Description:** The location where the player starts and can return to for safety. • **Type:** String **+MeteoroidsFalling**: **Description:** Indicates whether meteoroids are falling in the level and if falling then for how many seconds. **Type:** Integer(Seconds) **+DustStorm**: **Description:** Indicates whether a dust storm is occurring in the level and if occurring then for how many seconds.	level. **+remove_puzzle(puzzle: Puzzle) -> None**: • **Description:** Removes a puzzle from the level. **+start_dust_storm() -> None**: • **Description:** Initiates a dust storm event in the level. **+stop_dust_storm() -> None**: • **Description:** Stops the dust storm event in the level. **+start_meteoroids_falling() -> None**: • **Description:** Initiates meteoroids falling event in the level. **+stop_meteoroids_falling() -> None**: • **Description:** Stops the meteoroids falling event in the level. **+get_time_limit() -> int**: • **Description:** Retrieves the time limit for completing

	Type: Integer(Seconds)	the level. **+is_completed() -> bool**: ● **Description:** Checks if the level has been completed. **+is_unlocked() -> bool**: ● **Description:** Checks if the level is unlocked for the player. **+calculate_points(seconds_saved: int) -> int**: ● **Description:** Calculates the points earned by the player for completing the level. **+show_weather_forecast()->None:** ● **Description:** Shows realtime martian weather data on display.
DustStorm	**+Duration**: ● **Description:** The duration of the dust storm event. ● **Type:** Integer (seconds) **+Severity**: ● **Description:** The severity level of the dust storm. ● **Type:** String **+AffectedArea**: ● **Description:** The area affected by the dust	**+start() -> None**: ● **Description:** Starts the dust storm event. **+end() -> None**: ● **Description:** Ends the dust storm event. **+get_duration() -> int**: ● **Description:** Retrieves the duration of the dust

	storm. • **Type:** String **+VisibilityReduction**: • **Description:** The extent to which visibility is reduced during the dust storm. • **Type:** Float (percentage)	storm. +**get_severity() -> str**: • **Description:** Retrieves the severity level of the dust storm. **+get_affected_area() -> str**: • **Description**: Retrieves the area affected by the dust storm. **+get_visibility_reduction () -> float**: • **Description:** Retrieves the extent to which visibility is reduced during the dust storm.
MeteoriteFalling	**+Duration**: • **Description:** The duration of the meteorite falling event. • **Type:** Integer (seconds) **+Impact_intensity**: • **Description:** The intensity of the meteorite impacts. • **Type:** String **+Affected_area**: • **Description:** The area affected by the meteorite	**+start() -> None**: **Description:** Starts the meteorite falling event. **+end() -> None**: **Description:** Ends the meteorite falling event. **+get_duration() -> int**: **Description:** Retrieves the duration of the meteorite falling event. **+get_impact_intensity()**

	falling. • **Type:** String	**-> str**: **Description:** Retrieves the intensity of the meteorite impacts. +**get_affected_area() -> str**: **Description:** Retrieves the area affected by the meteorite falling.
WeatherForecast	**+Date**: • **Description:** The date for which the weather forecast is applicable. • **Type:** Date **+MaxTemperature**: • **Description:** The maximum temperature forecasted. • **Type:** Float **+MinTemperature**: • **Description:** The minimum temperature forecasted. • **Type:** Float **+UvIndex**: • **Description:** The UV index forecasted. • **Type:** Integer **+AtmosphericPressure**: • **Description:** The atmospheric pressure forecasted.	**+fetch_weather_data_from_api** (): • **Returns**: Dictionary containing weather data fetched from the MAAS API • **Description:** Fetches weather data from the MAAS API for a specific date. +**get_weather_forecast(date)**: • **Returns:** Dictionary containing weather forecast information • **Description:** Retrieves weather forecast for a specific date by

	• **Type:** Float **+Model** • **Description:**Represents the machine learning model used for weather prediction. **+MaasAPI** • **Description :** API key of Curiosity rover to collect historical data of Martian Weather.	fetching data from the MAAS API and using the ML model to predict the weather. **+display_predicted_data** (PredictedData): • **Description:** Displays the predicted weather data obtained from the ML model.
Map	**+Levels** • **Description:** A list containing Level objects representing the different levels available for exploration on the map. • **Type:** List of Level objects **+UnlockedLevels** • **Description:** A list containing Level objects representing the levels that have been unlocked by the player. • **Type:** List of Level objects	**+add_level(Level reference):** • **Description:** Adds a new level to the map. **+unlock_level(Level reference):** • **Description:** Unlocks a specific level on the map based on its index. **+display_map():** • **Description:** Displays the map, highlighting the unlocked levels and indicating the locked ones. **+select_level(Level reference):** • **Description:** Allows the player to select a level for exploration.

HomePage	+Options ● **Description:** Stores the available options on the homepage. ● **Type:** List of Strings +Choice ● **Description:** Chosen option ● **Type :** String	+input_player_name() -> None: ● **Description:** Prompts the player to input their name and stores it in the Player object's name attribute. +chose_action(Choice)->: ● **Description :** Calls method according to the option chosen by the player. +display_options() -> None: ● **Description:** Displays the available options to the player on the homepage. +resume_game() -> None: ● **Description:** Allows the player to resume the game, directing them to the map screen where previously unlocked levels are visible. +start_new_game() -> None: ● **Description:** Initiates a new game, unfolding the map to reveal the levels available for exploration. Only the first level is initially

		accessible. **+view_high_scores() -> None:** ● **Description:** Allows the player to view the top scores achieved in the game. **+view_about() -> None:** ● **Description:** Displays information about the gameplay, providing details about the game's concept and mechanics. **+view_credits() -> None:** ● **Description:** Displays credits, acknowledging the creators of the game. +**exit_game() -> None:** ● **Description:** Allows the player to exit the game.

Class cards

Player		
Method	**Responsibility**	**Collaboration Class**
update_progress(level: int, status: str)	Updates the progress of the player in the game (unlocking levels).	Level
add_to_inventory(RegionalObject reference)	Adds collected samples to the player's inventory.	Inventory, RegionalObject
start_level(level: int)	Initiates a new level for the player to play.	Level, Map
complete_level(time_taken: int)	Handles actions when the player successfully completes a level within the time limit.	Level
encounter_failure()	Handles actions when the player fails to complete a level due to dust storms or meteoroids.	Level
attempt_puzzle(puzzle: reference, answer: str)	Handles the player's attempt to solve a puzzle and returns feedback (correct or incorrect).	Puzzle
buy_item(item: reference)	Allows the player to buy items from the store using Basalt currency.	Store, Suit, Shovel
show_inventory()	Displays the player's	Inventory

	inventory with collected samples/items.	
handle_time_limit()	Manages actions when the player exceeds the time limit for a level.	Level
handle_menu_option(option: str)	Handles the player's selection from the main menu.	HomePage
choose_shovel(shovel_type: str)	Allows the player to select a shovel from the inventory.	Shovel
use_shovel(shovel: reference)	Initiates the extraction process for a given object using the selected shovel.	Shovel
choose_suit(suit_type: str)	Allows the player to select a suit from the inventory.	Suit
use_suit(suit: reference)	Utilizes the benefits of the currently equipped suit during gameplay.	Suit

RegionalObject		
Method	**Responsibility**	**Collaboration Class**
getSample()	After extraction, it returns reference to itself as a sample to be added to the inventory.	Player, Inventory
interact()	Represents the interaction of the player with the object, triggering extraction and analysis processes.	Player
increase_count()	Increase the value of the amount of the object collected by the player	Player

Shovel		
Method	**Responsibility**	**Collaboration Class**
buy_shovel(player_currency: int)	Allows the player to purchase the shovel if they have enough Basalt currency. Returns True if purchase is successful, False otherwise.	Player, Store
use()	Allows the player to equip and use the shovel during gameplay, affecting the extraction speed of objects.	Player
get_shovel()	Returns reference to itself to be added to the store.	Store

Suit		
Method	**Responsibility**	**Collaboration Class**
use_suit()	Equips the suit for the player to use during gameplay, enhancing their speed and jump abilities.	Player
buy_suit(player_currency: int)	Allows the player to purchase the suit if they have enough Basalt currency. Returns True if purchase is successful, False otherwise.	Player, Store
get_suit()	Returns reference to itself to be added to the store.	Store

Store		
Method	**Responsibility**	**Collaboration Class**
buy_item(item_name: str, player_currency: int)	Allows the player to purchase an item from the store by providing the item's name and the player's available	Player

	currency. Returns True if the purchase is successful, False otherwise.	
handle_menu_option(option: str)	Handles the player's selection from the store menu, allowing them to view available items and make purchases.	Player

Inventory		
Method	**Responsibility**	**Collaboration Class**
add_object(obj: RegionalObject)	Adds a new regional object to the inventory.	RegionalObject
remove_object(obj: RegionalObject)	Removes a regional object from the inventory.	RegionalObject
display_inventory()	Displays the collected regional objects in the inventory.	Player

Puzzle		
Method	**Responsibility**	**Collaboration Class**
check_answer(player_answer : str)	Checks if the player's answer matches. Returns a message indicating whether the answer is correct or not.	Player
check_inventory(inventory: Inventory)	Checks if the player has collected all the required samples for solving the puzzle by accessing the inventory.	Inventory
get_puzzle()	Returns reference to itself	Player, Level

Level		
Method	**Responsibility**	**Collaboration Class**
unlock_level()	Unlocks the level for the player.	Player, Map
complete_level()	Marks the level as completed.	Player, Map
add_puzzle(puzzle: Puzzle)	Adds a puzzle to the level.	Puzzle
remove_puzzle(puzzle: Puzzle)	Removes a puzzle from the level.	Puzzle
start_dust_storm()	Initiates a dust storm event in the level.	DustStorm
stop_dust_storm()	Stops the dust storm event in the level.	DustStorm
start_meteoroids_falling()	Initiates meteoroids falling event in the level.	MeteoroidsFalling
stop_meteoroids_falling()	Stops the meteoroids falling event in the level.	MeteoroidsFalling
calculate_points(seconds_saved: int)	Calculates the points earned by the player for completing the level.	Player
show_weather_forecast()	Displays real time martian weather data on screen	WeatherForecast

DustStorm

Method	Responsibility	Collaboration Class
start()	Starts the dust storm event.	Level
end()	Ends the dust storm event.	Level

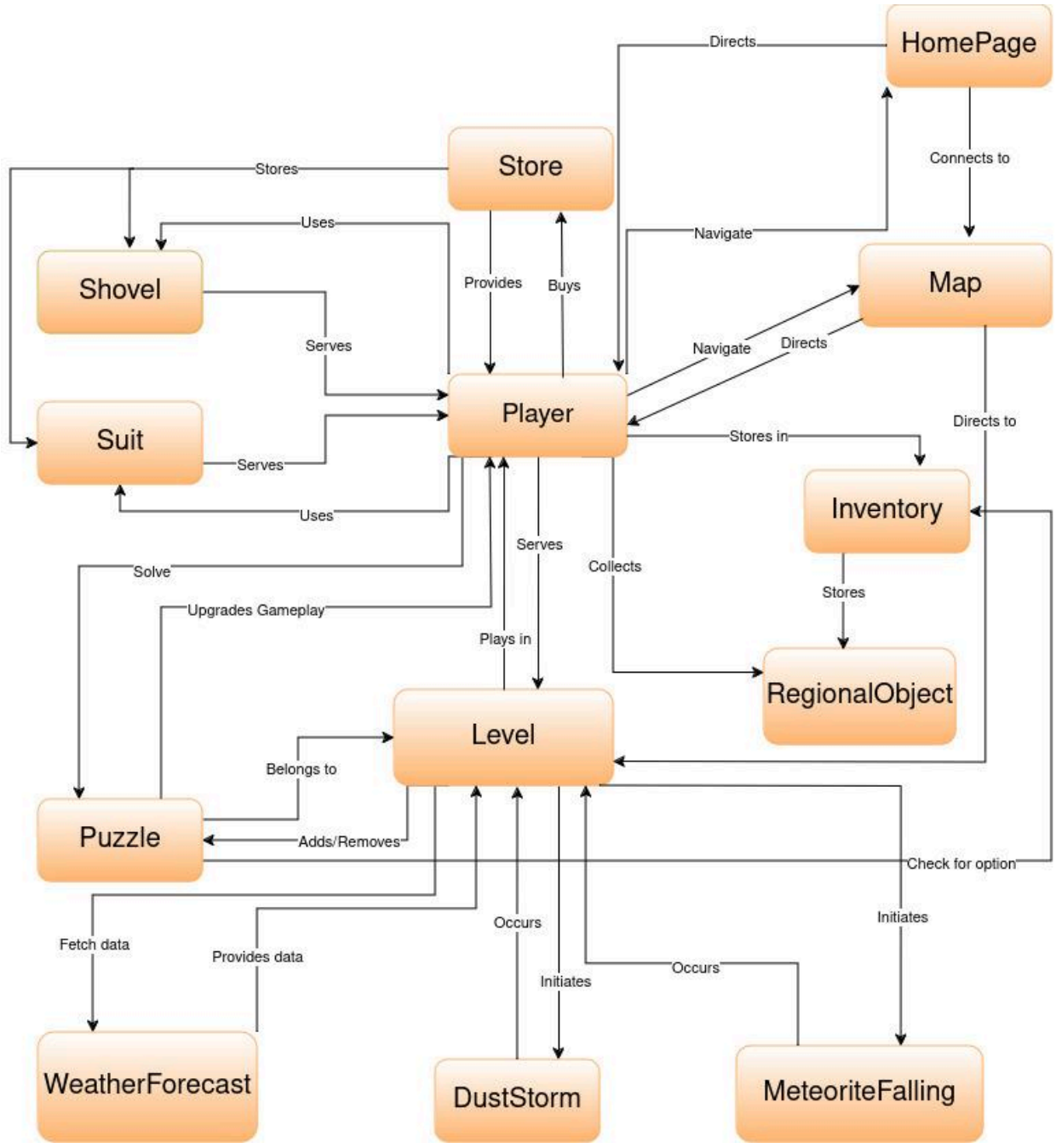
MeteoriteFalling		
Method	**Responsibility**	**Collaboration Class**
start()	Starts the meteorite falling event.	Level
end()	Ends the meteorite falling event.	Level

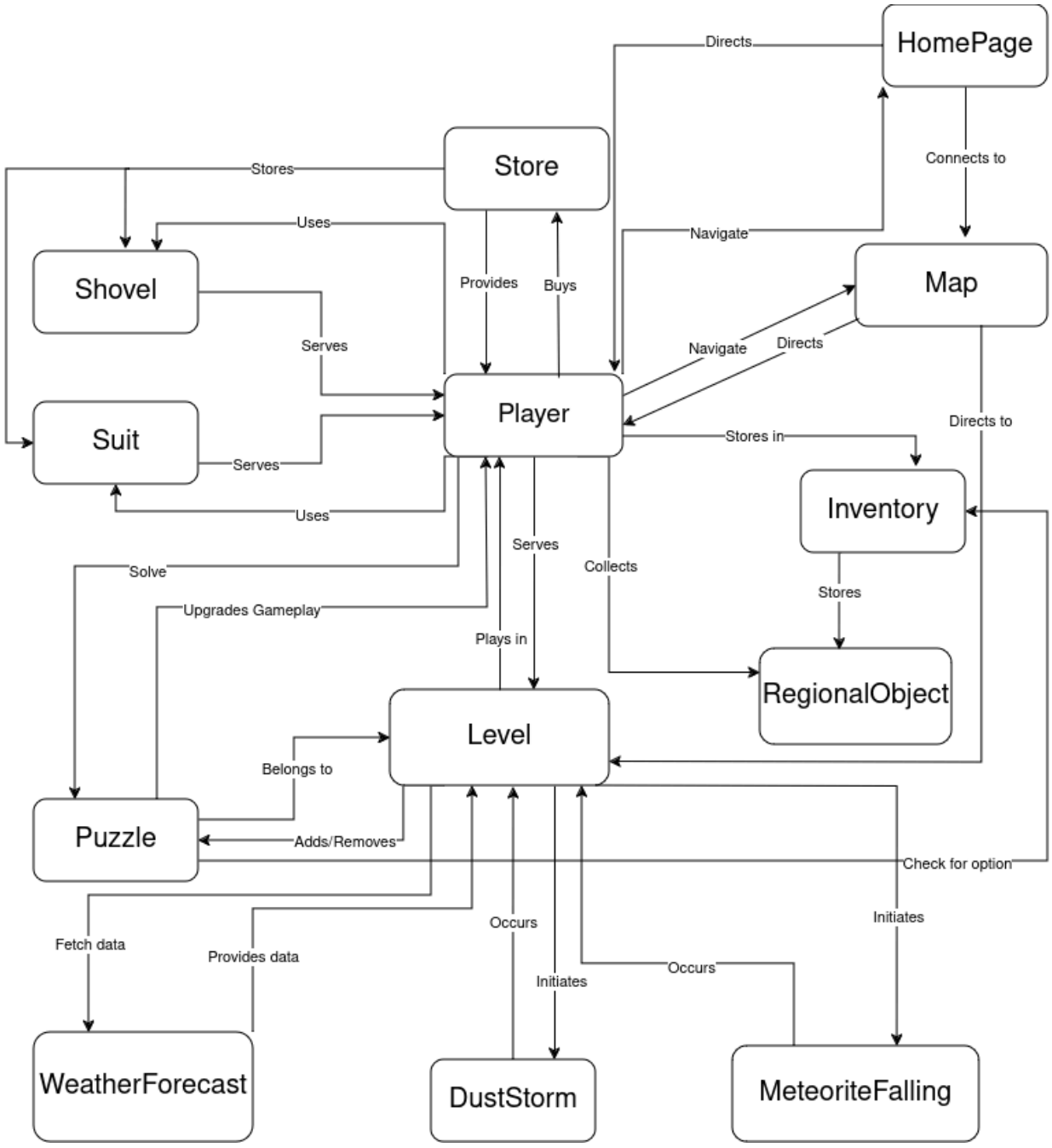
WeatherForecast		
Method	**Responsibility**	**Collaboration Class**
get_weather_forecast(date)	Retrieves weather forecast for a specific date by fetching data from the MAAS API and using the ML model to predict the weather.	Level

Map		
Method	**Responsibility**	**Collaboration Class**
select_level(Level reference)	Allows the player to select a level for exploration.	Player, Level

HomePage		
Method	**Responsibility**	**Collaboration Class**
input_player_name()	Prompts the player to input their name and stores it in the Player object's name attribute.	Player

chose_action(Choice)	Call method according to the option chosen by the player.	Player
resume_game()	Allows the player to resume the game, directing them to the map screen where previously unlocked levels are visible.	Player, Map
start_new_game()	Initiates a new game, unfolding the map to reveal the levels available for exploration. Only the first level is initially accessible.	Player, Level
view_high_scores()	Allows the player to view the top scores achieved in the game.	Player


HomePage
Directs
Store
Stores
Connects to
Uses
Navigate
Shovel
Provides
Buys
Map
Serves
Navigate
Directs
Player
Suit
Stores in
Directs to
Serves
Inventory
Uses
Serves
Collects
Solve
Stores
Upgrades Gameplay
Plays in
RegionalObject
Level
Belongs to
Puzzle
Adds/Removes
Check for option
Initiates
Occurs
Fetch data
Provides data
Occurs
Initiates
WeatherForecast
DustStorm
MeteoriteFalling


HomePage
Directs
Connects to
Store
Stores
Uses
Navigate
Shovel
Provides
Buys
Map
Serves
Navigate
Directs
Player
Suit
Directs to
Stores in
Serves
Inventory
Uses
Serves
Collects
Solve
Stores
Upgrades Gameplay
Plays in
RegionalObject
Level
Belongs to
Puzzle
Adds/Removes
Check for option
Initiates
Occurs
Fetch data
Provides data
Occurs
Initiates
WeatherForecast
DustStorm
MeteoriteFalling