**Use Case 1:** Initialize the main menu

**Scope:** Main Menu

**Level:** Design

**Primary Actor:** System

**Stakeholders and Interests:**

* System: Provides three options to the player.

**Preconditions:**

1. The system gets the command to start the game.

**Success Guarantee:**The main menu is provided to the player with three different options: START, HELP, and EXIT.

**Main Success Scenario:**

1. The player launches the game.
2. A title screen shows up.
3. Main menu shows up.

**Extensions:**

No extensions

**Special Requirements:**

* No special requirements.

**Technology and Data Variations List:**

1a. Player needs a mouse to launch the game.

**Frequency of Occurrence:** Every time you launch the game.

**Open Issues:** No open issues.

**Use Case 2:** Initialize START menu

**Scope:** Main Menu

**Level:** Subfunction

**Primary Actor:** Player

**Stakeholders and Interests:**

* Player: Wants to start playing the game

**Preconditions:**

1. The game application is launched
2. The player clicks the START button on the main menu.

**Success Guarantee:**The player transitions to the "Build Mode" page with empty halls ready to design.

**Main Success Scenario:**

1. The player clicks the START button.
2. The system initializes the "Build Mode" with all halls set to empty.
3. The player begins designing the layout of the halls.

**Extensions:**

No extensions.

**Special Requirements:**

* The START button should be prominently displayed on the main menu.
* A smooth transition should occur when moving to the START page.

**Technology and Data Variations List:**

1a. Player needs a mouse to click on the START button.

**Frequency of Occurrence:** Every time you start the game.

**Open Issues:** No open issues.

**Use Case 3:** Initialize HELP menu

**Scope:** Main Menu

**Level:** Subfunction

**Primary Actor:** Player

**Stakeholders and Interests:**

* Player: Wants to understand the game objects, mechanics, and how to play.

**Preconditions:**

1. The game application is launched.
2. The player clicks the HELP button on the main menu.

**Success Guarantee:** The HELP page is displayed, showing clear and readable instructions about the game.

**Main Success Scenario:**

1. The player clicks the HELP button.
2. The system displays the HELP page.
3. Player gets the information that he wants.
4. Player closes the HELP page.

**Extensions:**

No extensions.

**Special Requirements:**

* HELP text should be legible and organized.
* The HELP page should be visually appealing and intuitive.

**Technology and Data Variations List:**

1a. Player needs a mouse to click on the HELP button.

**Frequency of Occurrence:** When the player needs help about the game.

**Open Issues:** No open issues.

**Use Case 4:** Initialize EXIT case

**Scope:** Main Menu

**Level:** Subfunction

**Primary Actor:** Player

**Stakeholders and Interests:**

* Player: Wants to exit the game.

**Preconditions:**

1. The game is launched.
2. The player clicks the EXIT button on the main menu.

**Success Guarantee:** The game application closes without any errors.

**Main Success Scenario:**

1. The player clicks the EXIT button.
2. The system instantly closes the game.

**Extensions:**

1a. Player clicks the EXIT button accidentally:

1. The system displays a dialog asking for confirmation.
2. The player confirms the action to exit or cancels the action to remain in the main menu.

**Special Requirements:**

* The EXIT button should be easily accessible on the main menu.
* A confirmation dialog should be clear and easy to interact with.

**Technology and Data Variations List:**

1a. Player needs a mouse to click on the EXIT button.

**Frequency of Occurrence:** When the player wants to exit the game.

**Open Issues:** No open issues.

**Use Case 5:** Building the Dungeon

**Scope:** Build Mode

**Level:** User goal

**Primary Actor:** System

**Stakeholders and Interests:**

* Player: Wants to start playing the game by placing objects on an initialized grid map of halls

**Preconditions:**

1. The player should click START button on the Main Menu.

**Success Guarantee:** A gameboard UI along with halls and the game objects is initialized and displayed to the player.

**Main Success Scenario:**

1. An empty game board in a grid form is created.
2. Items slot is placed on the board.
3. Earth, air, water, and fire halls are placed on the board.
4. Player places a minimum of 6 objects in the earth hall.
5. Player places a minimum of 9 objects in the air hall.
6. Player places a minimum of 13 objects in the water hall.
7. Player places a minimum of 17 objects in the fire hall.

**Extensions:**

4-7a. Player tries to place an object to an already occupied space:

1. The system rejects the action and the object remains in its previous place.

4-7b. Not enough number of objects are placed in a hall to meet the minimum requirements:

1. The game remains in the build mode until the minimum criteria for each hall is satisfied.

**Special Requirements:**

* All game entities such as names of the halls, items slot, and items should be clearly visible.

**Technology and Data Variations List:**

● The UI for the game board should be visual and game objects should also be visually identifiable.

● Player should use a mouse

**Frequency of Occurrence:** Each time a new game is started from the main menu.

**Open Issues:** No open issues.

**Use Case 6**: Collecting enhancements

**Scope:** Play Mode

**Level:** User goal

**Primary Actor:** Player

**Stakeholders and Interests:**

- Player: Wants to collect an enhancement and add it to inventory.

-Developer: Wants to check the availability of the inventory, update the count of the items and make sure that enchantments can be used in the next halls.

**Preconditions:**

1. Player must be in playing mode.
2. The enhancement that player wants to collect must be visible in the screen.

**Success Guarantee:** Player collects the enhancement, enhancement added to inventory, inventory is updated.

**Main Success Scenario:**

1. The system puts a random enhancement randomly in the grid every 12 seconds.
2. The player left clicks the enhancement.
3. The system takes the enhancement.
4. The system updates the inventory.

**Extensions:**

1a. Player didn’t get the enhancement in time (6 seconds after it appears in the screen).

1. The Enhancement disappears from the screen.

2a. The inventory is full.

1. The item can not be added to the inventory.

2b. Enhancement is extra-life or extra-time.

1. System automatically uses the enhancement and doesn’t change the inventory, updates the time or life according to it.

**Special Requirements:**

* The image of the enhancement should be clear in the playing grid and doesn’t coincide with another object.

**Technology and Data Variations List:**

       2a. The player must have a working mouse or touchpad.

**Frequency of Occurrence:** Whenever player left-clicks an enhancement on the grid (usually in every 12 seconds).

**Open Issues:** No open issues

**Use Case 7-A:** Enchantment Utilization

**Scope:** Play Mode

**Level:** User goal

**Primary Actor:** Player

**Stakeholders and Interests:**

* Player: Would like to effectively use the reveal enchantment to get a hint about the location of the rune.
* Developer: Aims to temporarily mark a random 4x4 square that contains the rune.

**Preconditions:**

1. The game is in the Play Mode.
2. The player has the reveal enchantment in the inventory.

**Success Guarantee:**The enchantment is successfully used and removed from the inventory.

**Main Success Scenario:**

1. The player presses the button R on the keyboard to use the enchantment.
2. A 4x4 square containing the rune shows up and disappears after 10 seconds.
3. The inventory is updated.

**Extensions:**

1a. The player presses a different button.

1. Reveal enchantment is not used and remains in the inventory.

2a. 4x4 square is out of the bounds of the hall.

1. A new square is attempted until the system finds one that fits into the hall.

**Special Requirements:**

* The marked region should be distinguishable from the rest of the map.
* The image of the inventory UI should be consistent with the gameplay.

**Technology and Data Variations List:**

1a. Player needs a functioning keyboard.

**Frequency of Occurrence:** Every time the player tries to use the reveal enchantment.

**Open Issues:** No open issues.

**Use Case 7-B:** Enchantment Utilization

**Scope:** Play Mode

**Level:** User goal

**Primary Actor:** Player

**Stakeholders and Interests:**

-        Player: Would like to effectively use the cloak of protection to hide from the archer monster.

-        Developer: Wants to make sure that when the player uses the cloak it can be undetected by the monster for exactly 20 seconds.

**Preconditions:**

1. The game is in the Play Mode.
2. The player has the cloak of protection in the inventory.

**Success Guarantee:** The enchantment is successfully used and removed from the inventory.

**Main Success Scenario:**

1. The player presses the button P on the keyboard to use the enchantment.
2. The player is hidden from the archer monster for 20 seconds.
3. The inventory is updated.

**Extensions:**

1a. The player presses a different button.

1.     Cloak of protection is not used and remains in the inventory.

**Special Requirements:**

* The enchantment should have a special effect to indicate that the player is using the cloak of protection.
* The image of the inventory UI should be consistent with the gameplay.

**Technology and Data Variations List:**

1a. Player needs a functioning keyboard.

**Frequency of Occurrence:** Every time the player tries to use the cloak of protection.

**Open Issues:** No open issues.

**Use Case 7-C:** Enchantment Utilization

**Scope:** Play Mode

**Level:** User goal

**Primary Actor:** Player

**Stakeholders and Interests:**

-        Player: Would like to effectively use the luring gem to misdirect the fighter monster.

-        Developer: Wants to make sure that when the player throws the gem at the aimed position, the fighter monster follows it.

**Preconditions:**

1. The game is in the Play Mode.
2. The player has the luring gem in the inventory.

**Success Guarantee:** The enchantment is successfully used and removed from the inventory.

**Main Success Scenario:**

1. The player presses the button B and one of the A, D, W, S buttons to determine the direction.
2. The gem is successfully thrown at the aimed position and the fighter monster follows it.
3. The inventory is updated.

**Extensions:**

1a. The player presses a different button than B.

1. Luring gem is not used and remains in the inventory.

1b. The player presses B and a different button than the direction buttons.

1. The system waits until one of the direction buttons is pressed and then aims the object.

2a. The aimed location intersects with a monster, wall, or an object.

1. The system waits until a valid location is aimed for.

**Special Requirements:**

* There should be a special sound effect once the gem is thrown to lure the fighter monster.
* The image of the inventory UI should be consistent with the gameplay.

**Technology and Data Variations List:**

1a. Player needs a functioning keyboard.

**Frequency of Occurrence:** Every time the player tries to use the luring gem.

**Open Issues:** No open issues.

**Use Case 8:** Player movement

**Scope:** Play Mode

**Level:** User Goal.

**Primary Actor:** Player

**Stakeholders and Interests:**

- Player: Wants to move freely in the current Hall according to the rules of the game.

**Preconditions:**

1.      The game is in play mode.

**Success Guarantee:** Player moves up/down/left/right according to the arrow rune clicked.

**Main Success scenario:**

1. The player is at a random square in the current Hall.

2. One of the arrows is clicked to move the player.

3. The player moves toward the desired direction in one square magnitude.

**Extensions:**

2a. Player clicks a different button than the arrow runes.

1.      The player does not change its location and stays at the exact position.

3a. Player wants to move toward a direction but it is occupied by an object, wall or a monster.

1.      The player stays at the same position.

**Special Requirements:**

● No special requirements.

**Technology and Data Variations List:**

 2a. The player should have a functioning keyboard.

**Frequency of Occurrence:** Every time the player clicks an arrow to move the player.

**Open Issues:** None

**Use Case 9:** Initialize Play Mode

**Scope:** Rokue-Like Game

**Level:** Sub-Functionality

**Primary Actor:** System

**Stakeholders and Interests:**

- Player: Wants the game to transition from Build Mode to Play Mode seamlessly.

- Developer: Ensures all objects, monsters, and game states are initialized correctly for Play Mode.

**Preconditions:**

- The game is in Build Mode, and all required objects have been placed in each hall.

- All halls meet the object placement criteria.

**Postconditions:**

- The game transitions to Play Mode.

- The hero's position is initialized in the first hall (Hall of Earth).

- The timer for the first hall starts.

- All game elements, including monsters and enchantments, are initialized.

**Main Success Scenario:**

1. The player finishes placing objects in Build Mode.
2. The player clicks the "Start Game" button.
3. The system validates that all halls meet the object placement criteria.
4. The system transitions the game to Play Mode.
5. The hero's initial position is set randomly within the Hall of Earth.
6. The system initializes the first hall:
   * Monsters are placed in random locations.
   * Enchantments are set to spawn at timed intervals.
   * The hall's timer begins counting down.
   * The player’s remaining lives and inventory are displayed to the user
7. The game starts in Play Mode, allowing the player to control the hero.

**Extensions:**

2a. **Object Placement Criteria Not Met**:

* The system prevents progression and displays an error message indicating missing or misplaced objects in a hall.

6a. **Hero Position Overlaps with Another Object**:

* The system reassigns the hero's position to a valid location.

7a. **System Fails to Initialize Elements**:

* The system displays an error message and reloads Play Mode.

**Technology and Data Variations List:**

1. The player uses a mouse to place objects in Build Mode.

2. The system assigns initial positions for monsters and enchantments based on a randomization algorithm.

**Frequency of Occurrence:**

-   Occurs once per game session, immediately after Build Mode is completed.

**Open Issues:** None

**Use Case 10:** Collecting runes

**Scope:** Play Mode

**Level:** User goal

**Primary Actor:** Player

**Stakeholders and Interests:**

* Player: Wants to find the rune and go to next hall.

**Preconditions:**

1. Player must be in playing mode.
2. The rune that player wants to collect must be hidden in one of the objects.

**Success Guarantee:** Player collects the rune and goes to the next stage.

**Main Success Scenario:**

1. The system hides the rune in one of the objects.

2. The player goes near an object.

3. The player left-clicks to search for the rune.

4. The player repeats steps 2-3 until success.

5. The current hall is marked as complete

6. The door is opened for passage to the next hall.

**Extensions:**

3a. The rune is not inside the searched object.

1. A special sound effect indicates that the object is empty

**Special Requirements:**

* There should be a special sound effect when the clicked object is empty.
* There should be a visual and sound effect when the player finds the rune.

**Technology and Data Variations List:**

2-4a. The player must have a working mouse or touchpad and a keyboard.

**Frequency of Occurrence:** Whenever player left-click an object on the screen when standing right next to it.

**Open Issues:** No open issues

**Use Case 11:** Moving to the next Hall

**Scope:** Rokue-Like Game

**Level:** User Goal

**Primary Actor:** Player

**Stakeholders and Interests:**

* Player: Wants to progress to the next hall after completing the current hall.
* Developer: Ensures smooth transition between halls without glitches or errors.

**Preconditions:**

* The player has collected the rune in the current hall.
* The current hall is marked as "complete."
* The hero is positioned near the exit door.

**Postconditions:**

* The hero moves to the next hall.
* The timer for the new hall starts.
* Monsters and enchantments for the new hall are initialized.

**Main Success Scenario:**

1. The player collects the rune in the current hall.
2. The system marks the current hall as "complete."
3. The system unlocks the exit door.
4. The player moves the hero to the exit door.
5. The system transitions the hero to the next hall.
6. The system initializes the next hall:

* Monsters spawn at random locations.
* Enchantments are set to spawn at intervals.
* The hall’s timer begins.

1. The hero’s initial position is set near the entry point of the new hall.

**Extensions:**

1a. **Rune Not Collected**:

* The system prevents the player from passing through the exit door.

6a. **System Fails to Initialize the Next Hall**:

* The system reloads the next hall or prompts the player to restart the game.

7a. **Hero's Initial Position Overlaps with a Monster or Object in the New Hall**:

* The system reassigns the hero’s initial position to a valid grid space.

**Technology and Data Variations List:**

4a. The player uses the keyboard to move the hero to the exit door.

**Frequency of Occurrence:**

* Occurs once for each hall, except the final hall (Hall of Fire).

**Open Issues:** None.

**Use Case 12:** Losing the game due to time

**Scope:** Playing Mode

**Level:** Subfunction

**Primary Actor:** Player

**Stakeholders and Interests:**

* Player: Does not want the timer to run out.

**Preconditions:**

1. The player is in playing mode.
2. The game is not paused.
3. The game has not finished by another means.
4. The timer reaches zero.

**Success Guarantee:** The game displays “Game Over” screen, with a text indicating that the player has run out of time.

**Main Success Scenario:**

1. The player enters a hall.
2. The timer initializes to a value of 5 seconds per object.
3. The timer counts down by one every second.
4. The timer reaches 0.
5. The game finishes, the “Game Over” screen is displayed.

**Extensions:**

3a. The game is paused:

1. The timer does not count down.

3b. The player gets an extra time enchantment:

1. The timer increases by 5 seconds.

4a. The player finds the rune:

1. Player moves onto the next hall, and the timer resets.

4b. The health of the player reaches 0:

1. Game is lost, so the timer is irrelevant.

**Special Requirements:**

* The timer should be readable at a glance.
* The timer should warn the players as it gets lower. For example, it might turn red once the timer reaches 10 seconds, or play a different sound cue as the time is running out.
* The information that the player lost due to time should be visible on the “Game Over” screen.

**Technology and Data Variations List:**

2-3a. The computer that is running the game should be able to count seconds.

**Frequency of Occurrence:** Whenever the timer runs out.

**Open Issues:** No open issues

**Use Case 13:** Winning the Game

**Scope**: Gameplay  
**Level**: User Goal

**Primary Actor:** Player

**Stakeholders and Interests:**

1. **Player**: Wants to successfully complete the game by finding all the runes and exiting the dungeon.
2. **Game System**: Ensures all conditions for winning are met and provides a satisfying conclusion to the game.

**Preconditions:**

1. The player has successfully completed all halls (Earth, Air, Water, Fire).
2. The hero has collected all runes in sequence.
3. The hero has at least one life remaining.

**Success Guarantee:** The player completes the dungeon, finds all the runes, and exits successfully, receiving a victory message on the screen.

**Main Success Scenario:**

1. The system validates that the hero has completed all of the halls and displays a victory message or screen

**Extensions:**

No extensions.

**Special Requirements:**

1. The game should visually or audibly celebrate the player's success (e.g., animation, sound effects, or congratulatory message).

**Technology and Data Variations List:**

None.

**Frequency of Occurrence:** Occurs only when the player completes the game successfully.

**Open Issues:** None

**Use Case 14:** Pause/Resume

**Scope:** Play Mode

**Level:** Subfunction

**Primary Actor:** Player

**Stakeholders and Interests:**

* Player: Wants to temporarily stop the game and resume it later without losing progress.

**Preconditions:**

1. The player is in playing mode.

**Success Guarantee:** The game is paused and can be resumed seamlessly from the same point.

**Main Success Scenario:**

1. The player clicks the pause button during gameplay.
2. The system halts all in-game actions (e.g., timer, monster movements, and player controls).
3. The system displays a paused state or menu, indicating that the game is paused.
4. The player clicks the resume button.
5. The system resumes the game from the exact state before the pause.

**Extensions:**

No extensions.

**Special Requirements:**

* The pause button must be clearly labeled and accessible during gameplay.
* Visual feedback (e.g., a pause icon or overlay) should confirm that the game is paused.
* The resume button should revert the game to its active state smoothly.

**Technology and Data Variations List:**

1. The player must have a working mouse, touchpad, or touchscreen to interact with the buttons.
2. The system must support storing the current state of the game when paused.

**Frequency of Occurrence:** Whenever the player wants to temporarily stop gameplay.

**Open Issues:** No open issues

**Use Case 15:** Losing life because of a damage by a monster

**Scope:** Play Mode

**Level:** Subfunction

**Primary Actor:** Player

**Stakeholders and Interests:**

* Player: Wants to run away from the monsters so it doesn’t lose lives and die.
* Monster: Wants to attack and damage the player in their own way.

**Preconditions:**

1. The game must be in play mode.
2. The player must have at least one life.
3. There is at least one fighter or an archer monster in the hall.

**Success Guarantee:** The monster damages and causes the player to lose a life.

**Main Success Scenario:**

1. A fighter or an archer monster gets spawned in an empty location.
2. Player has at least one out of three lives.
3. Character moves to a location that is in the proximity of the monster.
4. Monster attacks and the player loses a life.

**Extensions:**

1a. The location overlaps with a wall, object, monster, or the hero.

1. The system reassigns a new location for the monster.

3a. The hero uses the cloak of protection and the monster is an archer monster.

1. The archer can not detect the player, therefore the player does not lose a life.

3b. The hero throws a luring gem and the monster is a fighter monster.

1. The fighter monster follows the luring gem, the player does not lose a life.

**Special Requirements:**

* A board showing the player’s lives.

**Technology and Data Variations List:**

None.

**Frequency of Occurrence:** Every time a fighter or an archer monster damages the player.

**Open Issues:** No open issues.

**Use Case 16:** Losing the game because of 0 remaining lives.

**Scope:** Play Mode

**Level:** Subfunction

**Primary Actor:** System

**Stakeholders and Interests:**

* System: Wants to make sure that the player loses the game as soon as it does not have any remaining lives.

**Preconditions:**

1. The game is in the play mode.
2. The player does not have any lives left.

**Success Guarantee:** The game displays “Game Over” screen, with a text indicating that the player has run out of lives.

**Main Success Scenario:**

1. The game finishes, the “Game Over” screen is displayed.

**Extensions:**

No extensions.

**Special Requirements:**

* A board showing the player’s lives.
* The information that the player lost due to zero lives should be visible on the “Game Over” screen.

**Technology and Data Variations List:**

None.

**Frequency of Occurrence:** Every time the player loses a game because of zero lives.

**Open Issues:** No open issues.

**Use Case 17:** Exiting the current game session.

**Scope:** Gameplay

**Level:** User goal

**Primary Actor:** Player

**Stakeholders and Interests:**

* Player: Wants to exit the current game session as soon as clicking the exit button.
* Developer: Wants to ensure that the game reverts back to the main menu after the exit button is clicked.

**Preconditions:**

1. The game is in the play or build mode.

**Success Guarantee:** The game goes back to the main menu and the player can start over again.

**Main Success Scenario:**

1. The player clicks the exit button.
2. The system closes the current game session
3. The player is directed to the main menu.

**Extensions:**

1a. Player clicks the exit button accidentally:

1. The system displays a dialog asking for confirmation.
2. The player confirms the action to exit or cancels the action to remain in the game.

2-3a. The system is unable to carry out the operation.

1. The system restarts the game.

**Special Requirements:**

* The exit button should be easily accessible in play and build modes
* A confirmation dialog should be clear and easy to interact with.

**Technology and Data Variations List:**

1a. Player needs a mouse to click on the exit button.

**Frequency of Occurrence:** When the player wants to exit the current game session.

**Open Issues:** No open issues.