# 1. Guide Utilisateur - Introduction au jeu

Bienvenue dans Backrooms, un jeu d'exploration et de survie où le joueur navigue de salle en salle dans un environnement mystérieux et captivant. Chaque salle possède ses propres caractéristiques, quêtes et personnages non-joueurs (PNJ) avec lesquels le joueur peut interagir. Votre mission est de progresser dans cet univers tout en évitant les dangers et en relevant les défis.



# 1. Guide Utilisateur - Règles et Commandes

Règles du jeu et conditions de victoire/perte :

* Condition de victoire :

* Atteindre la dernière salle en vainquant le boss final.

* Condition de défaite :

* Perdre tous ses points de vie.

Commandes disponibles :

* Se déplacer : Utilisez la commande 'go <direction>' pour passer d'une salle à une autre (N, E, S, O).

* Interagir avec un objet : Utilisez 'prendre <objet>' ou 'poser <objet>'.

* Parler à un PNJ : Tapez 'parler <nom\_du\_pnj>'.

* Inspecter la salle : Tapez 'inspecter'.

* Consulter l'historique des salles : Tapez 'historique'.

* Quitter le jeu : Tapez 'quit'.
* Attaque un PNJ : taper attaquer <PNJ>
* Poser un objet : taper poser <objet>

# 2. Guide Développeur - Modules

Le jeu est développé en Python en utilisant une architecture modulaire. Voici les modules principaux:

Classe Actions

Elle regroupe les fonctions permettant au joueur d’interagir avec le jeu. Fonctions disponibles :

* go(), help(), back(), look(), check(), take(), drop(), enter().

La fonction enter() utilise l’interface graphique Tkinter. Paramètres communs à toutes les fonctions :

* game : Game
* list\_of\_words : list[str]
* number\_of\_parameters : int

Classe Character

Elle définit les caractéristiques des personnages non-joueurs (PNJ). Attributs :

* name : str
* description : str
* current\_room : Room
* msg : list[str] Méthodes :
* \_\_init\_\_(self, name, description, current\_room, msg) → None : Initialise un personnage.
* \_\_str\_\_(self) → str : Retourne une description du PNJ.
* move(self) → None : Fait bouger le PNJ de manière aléatoire.
* get\_msg(self) → str : Retourne un message du PNJ.

Classe Command

Elle gère la logique des commandes entrées par le joueur.

Attributs :

* command\_name : str Méthodes :
* execute(self, game, args) → None : Exécute une commande spécifique.

Classe Config

Elle contient les configurations globales du jeu.

Attributs :

* settings : dict[str, Any] Méthodes :
* load(self, file\_path) → None : Charge une configuration à partir d’un fichier.
* save(self, file\_path) → None : Sauvegarde la configuration actuelle

Classe Direction

Elle gère la normalisation des directions de déplacement.

Attributs :

* valid\_directions : list[str] Méthodes :
* normalize(self, direction) → str : Normalise une direction donnée.

Classe Game

Elle coordonne le jeu et gère les niveaux.

Attributs :

* current\_level : int
* player : Player Méthodes :
* start(self) → None : Lance le jeu.
* stop(self) → None : Arrête le jeu.

Classe Item

Elle définit les objets du jeu. Attributs :

* name : str
* description : str Méthodes :
* use(self, player) → None : Applique l’effet de l’objet sur le joueur.

Classe Player

Elle modélise le joueur, son inventaire, et ses déplacements. Attributs :

* name : str
* inventory : list[Item] - current\_room : Room Méthodes :
* move(self, direction) → None : Déplace le joueur dans une direction donnée.
* add\_item(self, item) → None : Ajoute un objet à l’inventaire.

Classe Room

Elle définit les caractéristiques des salles et leurs interactions. Attributs :

* name : str
* description : str
* connected\_rooms : dict[str, Room] Méthodes :
* connect(self, direction, room) → None : Connecte une salle dans une direction donnée. - get\_room(self, direction) → Room : Retourne la salle dans une direction donnée

# 3. Perspectives de Développement

Perspectives de développement:

1. Ajouter davantage de salles avec des thématiques uniques.

1. Enrichir les interactions avec les PNJ (arcs narratifs, dialogues).

1. Introduire des quêtes secondaires.

Optimisations:

* Améliorer les performances pour gérer un plus grand nombre de salles.

* Ajouter des sauvegardes et chargements de parties.

Extensions possibles:

* Multijoueur : Collaboration pour explorer les Backrooms.

* Graphismes améliorés avec Pygame ou une autre bibliothèque.

* Version mobile ou web.

# 4. Vidéo de Démonstration

Vidéo de démonstration (3 minutes):

1. Présentation de l'univers du jeu:

* Carte des salles.

* Description des personnages principaux (ex: Gandalf).

* Objets interactifs.

2. Démonstration:

* Parcours perdant : Le joueur échoue après une série de mauvaises décisions.

* Parcours gagnant : Le joueur progresse jusqu'à la dernière salle et vainc le boss final.

Espace pour la vidéo: (Veuillez insérer votre vidéo ici.)