

Projet Blackjack

Table des matières

1. Analyse	2
2. Conception	3
3. Réalisation	4
4. Tests	5
5. Mise en service	5
6. Maintenance	5



1. Analyse

Objectif principal :

L'objectif du projet est de développer un jeu de Blackjack multijoueur local en Python, avec une interface graphique (GUI) réalisée à l'aide de Tkinter. Le jeu permet à deux joueurs de s'affronter contre un croupier contrôlé par l'ordinateur, tout en intégrant un système de mise et de calcul automatique des scores. Les contraintes principales concernent l'utilisation exclusive de Python et Tkinter, et la gestion locale sans connexion réseau.

Exigences et contraintes :

- Interface intuitive (boutons Tirer, Rester, Rejouer).
- Gestion automatique du score et du croupier.
- Mode local uniquement.
- Gestion des mises et du solde.
- Python + Tkinter uniquement (pas de base de données externe).

Livrables de cette phase :

- Cahier des charges (fonctionnalités listées).
- Schéma d'interface et d'organisation du code.

2. Conception

Interface graphique :

- Zone d’affichage des cartes.
- Zone de mise.
- Boutons : Tirer / Rester / Rejouer / Quitter.
- Zone d’affichage du score et du solde.

Schéma UML ou organigramme :

[Cliquer pour voir.](#)

- Démarrage et initialisation (vert)
- Fin (rose)

Codes couleurs :

- Vert = Début/Gains
- Rouge/Orange = Pertes
- Bleu = Égalité
- Jaune = Blackjack
- Orange = Décisions

3. Réalisation

Structure du projet :

- main.py → Lancement du jeu.
- gui.py → Interface Tkinter.
- blackjack.py → Logique du jeu.
- player.py → Classe joueur (nom, solde, cartes).
- dealer.py → Classe croupier.

Technologies :

- Python
- Tkinter
- Modules standard (random, time...)

Développement prévu en sprints :

Sprint	Durée	Objectif principal
Sprint 1	1 semaine	Interface Tkinter de base + affichage des cartes
Sprint 2	1 semaine	Logique du jeu (distribution, score, As, croupier)
Sprint 3	1 semaine	Gestion des mises et solde + affichage des résultats
Sprint 4	1 semaine	Améliorations visuelles et tests utilisateurs

4. Tests

Types de tests :

- Tests unitaires : distribution correcte, calcul du score, gestion de l'As.
- Tests fonctionnels : déroulement complet d'une partie.
- Tests d'interface : vérification des boutons, affichage, rejouer.

5. Mise en service

- Exécutable .py prêt à lancer.
- Présentation du projet en classe.
- Documentation simple (manuel utilisateur).

6. Maintenance

- Correction des bugs mineurs.
- Amélioration possible : ajout d'un mode en ligne, animation des cartes, historique détaillé.