Épiém.io - Cahier des charges

Athène Rousseau Rambach et Adrien Verstrepen

Table des matières

Présentation du projet

1. Objectif

L'objectif du projet Épidém.io est de réaliser une simulation de propagation d'une épidémie. L'utilisateur pourra régler différents paramètres pour adapter la simulation à ses critères. L'implantation se fera avec Python et des bibliothèques associées. L'utilisateur verra la maladie se répandre au fur et à mesure. Il y aura des itérations successives de la propagation de l'épidémie que nous afficherons sous forme de schéma.

2. Contexte

Ce projet est réalisé dans le cadre de la matière "Développement de logiciel". L'objectif est de concevoir un logiciel orienté objet. Le choix du thème de la propagation d'une épidémie a été fait pour que nous puissions apprendre à faire de la modélisation, de l'affichage et de l'analyse de données.

3. Contraintes techniques

Le logiciel sera développé en Python. Nous utiliserons des bibliothèques adaptées pour la visualisation, comme Tkinter ou PyQt pour l'interface et Matplotlib pour les graphiques. Le logiciel devra fonctionner sur les systèmes d'exploitation Linux et Windows.

Description de l'application

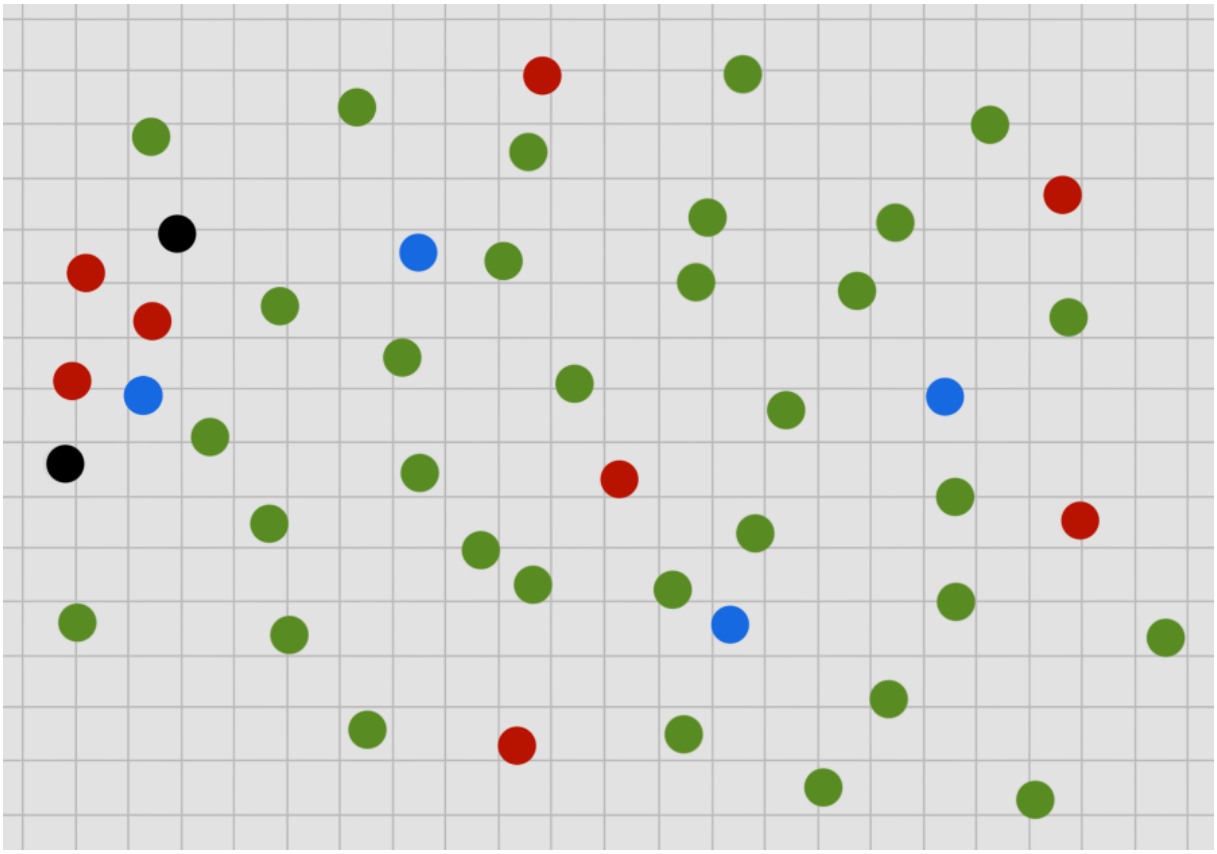
1. Affichage de la simulation

La partie principale de l'écran sera la fenêtre d'affichage de la simulation. Les personnes seront donc placées dans un espace en 2D et seront représentées par un état. L'attribut état est obligatoire et exclusif : il y a quatre états, une personne en a nécessairement exactement un parmi ces choix. Les quatre états sont liés à une couleur pour la représentation visuelle :

- l'état sain, représenté en vert : la personne n'est pas malade ;
- l'état infecté, représenté en rouge : la personne est infectée, mais n'est pas morte ;
- l'état mort, représenté en noir : la personne a été infectée et est morte à cause de la maladie ;
- l'état immunisé, représenté en bleu : la personne ne peut pas être contaminée.

La simulation fonctionnera donc avec des itérations, et nous mettrons à jour toutes les x secondes l'affichage, faisant une pseudo animation (la valeur de x sera déterminée lors du développement, certainement entre 0.2 et 1).

Exemple d'arrêt sur image de la simulation :



2. Éléments de la simulation réglables par l'utilisateur

L'utilisateur pourra régler différents éléments grâce à un menu dans l'interface. Cela lui permet de simuler une épidémie qui correspond le plus possible à celle qu'il cherche à recréer. Par exemple, l'épidémie du COVID-19 ne peut pas être simulée comme celle de la

peste au XIVe siècle. Parmi les indicateurs que l'utilisateur pourra donc régler, on compte trois types :

a. les indicateurs liés à la population :

- le nombre de personnes totales ;
- le pourcentage de personnes immunodéprimées : par exemple, au Japon 30.2% de la population est âgée de 65 ans ou plus, alors que c'est 1.7% de la population en Ouganda, c'est donc un facteur intéressant ;
- le nombre de médecins : ce sont des personnes qui peuvent augmenter les chances de survie des gens autour (ça ne sera pas du 100%, mais on pourrait dire que les personnes proches de ce point à une certaine distance verraient leurs chances de survie multipliées par deux) ;

b. les indicateurs liés à la maladie :

- le taux de létalité (ou pourcentage de mortalité) : la simulation ne sera pas pareil si on représente la tuberculose, qui a un taux de létalité de 43%, et la grippe A, qui a un taux de létalité inférieur à 0.1% ;
- la distance d'infection : la distance qu'il faut entre deux personnes pour que la personne infectée contamine l'autre ;
- le pourcentage de risque de transmission : ce n'est pas la même chose si une maladie a seulement 10% de risques d'être transmise, ou 95% ;
- l'immunité après guérison : il arrive que certaines maladies ne puissent plus s'attraper après qu'une personne l'ai attrapée et soit guérie, on peut donc mettre un indicateur qui voudra "oui" ou "non" pour l'obtention d'une immunité contre cette maladie une fois guéri ;
- le temps de guérison : certaines maladies ne nous affectent que quelques jours, comme la grippe, mais d'autres restent à vie par exemple, comme le VIH ;

c. les indicateurs liés à la simulation :

- le nombre de personnes infectées au début : le nombre de personnes qui seront les "patients 0" ;
- la vitesse de déplacement des points

Organisation du projet

1. Répartition des rôles

Athène (profil data) : s'occupe de la modélisation de la population, des règles de propagation, du calcul des états et du stockage des données pour les statistiques ;

Adrien (profil interface/visualisation) : s'occupe de l'affichage de la simulation, du menu de réglage des paramètres, des boutons (lancer, pause, reset) et de l'intégration des graphiques Matplotlib

2. Tâches à réaliser

- Implémentation de l'interface graphique :

    - création de la fenêtre principale
    - mise en place d'une toile pour afficher la simulation en 2D
    - ajout d'un menu de réglage des paramètres :
        - champs numériques pour les valeurs entières (la taille de la population, le nombre de médecins, rayon de contagion…) ;
        - cases à cocher pour les options booléennes (immunité après guérison, présence ou non des médecins,…) ;
        - curseurs pour les valeurs continues (risque de transmission, taux de létalité, distance d'infection, vitesse de déplacement, pourcentage de personnes immunodéprimées…)
    - ajout des boutons démarrer, mettre en pause et réinitialiser
    - mise à jour de la toile à chaque itération pour animer les déplacements et les changements d'état (couleur selon l'état)
    - ajout d'un bouton pour télécharger les statistiques
    - gestion de la fermeture propre du programme
- Implémentation du système de gestion des statistiques :
    - création de la classe statistiques pour enregistrer les données à chaque itération
    - définition d'une structure interne pour stocker le nombre total de personnes totales et contaminées
    - ajout d'une méthode appelée à chaque mise à jour de la population pour stocker les informations
    - génération d'un graphique matplotlib affichant l'évolution de la contamination au fil du temps
    - ajout de la possibilité d'exporter les données
- Implémentation de la simulation :
    - création des classes principales personne, population et épidémie
    - implémentation d'un système de calcul des distances entre individus pour déterminer les contaminations
    - mise en place d'un pas de temps entre chaque itération où chaque individu :
        - prend une nouvelle position ;
        - se déplace selon sa vitesse ;
        - vérifie les contacts à proximité pour une éventuelle contamination ;
        - met à jour son état (guérison, mort, immunisation)
    - test et comparaison de différentes approches pour le déplacement des individus, type algorithmes de simulation de déplacement de foule (exemple d'approches possibles : déplacement totalement aléatoire, déplacement avec inertie ou direction persistante, gestion des collisions pour éviter la superposition des points, déplacements inspirés de Boids…)
    - choix et implémentation de l'approche la plus stable et visuellement claire pour la simulation
    - synchronisation de la simulation avec l'affichage
    - mise en place de comportements spécifiques (par exemple : les médecins augmentent les chances de guérison autour d'eux, les immunodéprimés ont un risque de mortalité plus élevé…)

    - gestion de la fin de la simulation (tous guéris, morts, aucune infection active, ou au contraire régénération de la population avec des enfants : à déterminer lors du développement)
    - gestion de l'approche concernant le temps entre les itérations (temps que peux déterminer l'utilisateur, temps fixe, temps variable en fonctions d'indices comme le nombre d'individus…)
    - détermination des limites de l'algorithme (par exemple : pas plus de x personnes car risques trop élevés que la simulation ne supporte pas une quantité trop élevée)
- Tests de l'application :
    - vérification des données entrées par l'utilisateur dans les champs à remplir
    - vérification des effets de bord
    - surveillance des bugs algorithmiques
    - tests de performance (par exemple : limitation de la population à simuler, mise en place d'une limite minimum pour le temps entre chaque itération…)
    - vérification de l'intégrité des données récupérées entre chaque itération
- Rédaction de la documentation et du support de présentation
    - rédaction de la documentation technique des classes et méthodes
    - ajout de commentaires dans les parties principales du code
    - rédaction d'une notice utilisateur décrivant le déroulement de la simulation et les options disponibles
    - création d'un schéma UML final mis à jour selon les choix d'implémentation effectués
    - rédaction d'un support de présentation
    - réalisation de tests finaux et correction des éventuels bugs avant la présentation et la livraison du projet

3. Rétroplanning

Sprint 0 : du 12/09/2025 au 06/10/2025 (24 jours)

Objectif : réalisation du cahier des charges

- Athène :
    - présentation du projet
    - diagramme de classes
    - organisation du projet
- Adrien :
    - diagramme de Gantt

Sprint 1 : du 06/10/2025 au 16/10/2025 (10 jours)

Objectif : mise en place des bases du modèle et de l'interface

- Athène :
    - implémentation de la classe personne avec les attributs de base : état, immunodéprimé, position et vitesse (mise de côté temporaire des médecins)

    - implémentation de la classe population pour faire la création et mise à jour simple des personnes
    - première boucle d'itération console (propagation simple sans déplacement)
    - stockage des données par itération (nombre infectés, morts, immunisés, positions…)
- Adrien :
    - implémentation d'une maquette d'interface : structure avec menu, zone principale et boutons vides
    - préparation des widgets pour les futurs paramètres (sliders/champs non reliés)
    - classe simulation squelette pour organiser la logique future

Sprint 2 : du 16/10/2025 au 20/11/2025 (35 jours)

Objectif : affichage visuel, premiers contrôles utilisateur et simulation plus réaliste avec paramètres complets et premier déplacement

- Athène :
    - ajout des états : sain, infecté, immunisé, mort
    - ajout des paramètres simples : nombre initial d'infectés, probabilité de transmission…
    - mise en place de la propagation (première étape sans déplacement)
    - gestion de la mortalité, immunité après guérison (oui/non), immunodéprimés…
    - premier déplacement aléatoire temporaire simple pour chaque personne (mouvement limité par les bords) pour tester pour la contagion
    - organisation des données collectées par itération pour le module statistiques
- Adrien :
    - ajout de la visualisation dans l'interface de la simulation
    - ajout de la partie extraction et visualisation des données dans l'interface
    - liaison entre le modèle et l'interface : toile affichant les points colorés selon l'état
    - rafraîchissement automatique de la toile toutes les x seconde(s) (approche à déterminer) avec mise à jour des paramètres
    - boutons fonctionnels pour lancer, mettre en pause et réinitialiser
    - info bulle pour que les utilisateurs puissent comprendre exactement à quoi correspondent les paramètres

Sprint 3 : du 20/11/2025 au 03/12/2025 (13 jours)

Objectif : algorithme de déplacement des personnes (foules) et amélioration de l'interface

- Athène :
    - prise en compte de la distance de contagion et de la probabilité de transmission selon la proximité
    - tests de plusieurs modèles de déplacement (aléatoire simple, inertiel, avec évitement, Boids ajusté aux foules humaines…)
    - comparer leurs impacts sur la propagation, leur fidélité par rapport au phénomène à représenter et la stabilité visuelle

- Adrien :
    - liaison de tous les paramètres du menu à la simulation (taille population, distance contagion, mortalité, vitesse…)
    - interface plus ergonomique : disposition claire des contrôles, labels, validation des champs (amélioration de l'UI/UX)
    - ajustement de la toile pour suivre les déplacements des points à chaque itération
    - mise en place de tests de non régression
    - ajout de cercles rouges autour des points pour montrer la distance d'infection entre deux points lorsque l'utilisateur règle la distance d'infection pour qu'il puisse bien se représenter la distance

Sprint 4 : du 03/12/2025 au 08/12/2025 (5 jours)

Objectif : ajout des médecins et gestion des statistiques extraites

- Athène :
    - ajout des actions des médecins sur leurs voisins et leurs paramètres associés (pourcentage de survie augmenté, distance nécessaire pour soigner les personnes…)
- Adrien :
    - graphiques matplotlib (schéma avec les itérations en abscisse et le pourcentage d'infectés (morts inclus) en ordonnée…)
    - téléchargement du graphique réalisé
    - amélioration visuelle : taille des points, contraste, lisibilité, disposition générale… ce qui est nécessaire

Sprint 5 : du 08/12/2025 au 15/12/2025 (7 jours)

Objectif : stabilisation du projet et amélioration l'ergonomie

- Athène :
    - tests du modèle complet : cohérence propagation + mouvement
    - vérification cohérence des statistiques enregistrées
    - rédaction de la documentation technique
- Adrien :
    - tests de l'interface : réactivité des boutons, gestion des états (pause, reprise, réinitialisation)
    - correction des bugs potentiels
    - diagramme de classes final à jour
    - rédaction de la documentation utilisateur

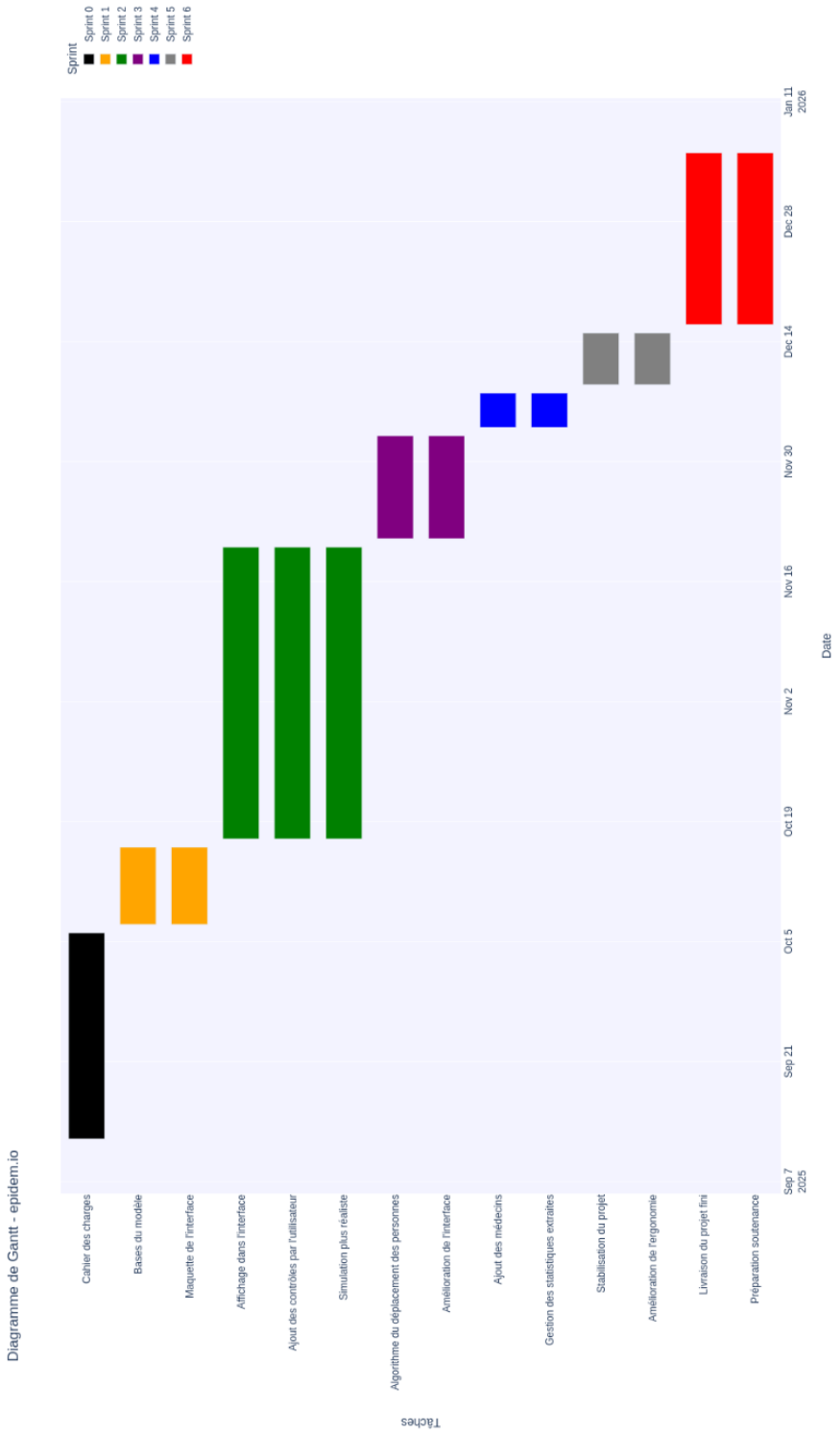
Sprint 6 : du 15/12/2025 au 05/01/2026 (21 jours)

Objectif : livraison du projet fini et préparation de la soutenance

- Athène :
    - nettoyage et commentaires du code côté modèle

    - rédaction de la partie modèle du rapport
    - correction des bugs
- Adrien :
    - nettoyage et commentaires du code côté interface
    - préparation des diapositives et de la démonstration
    - correction des bugs

4. Diagramme de Gantt



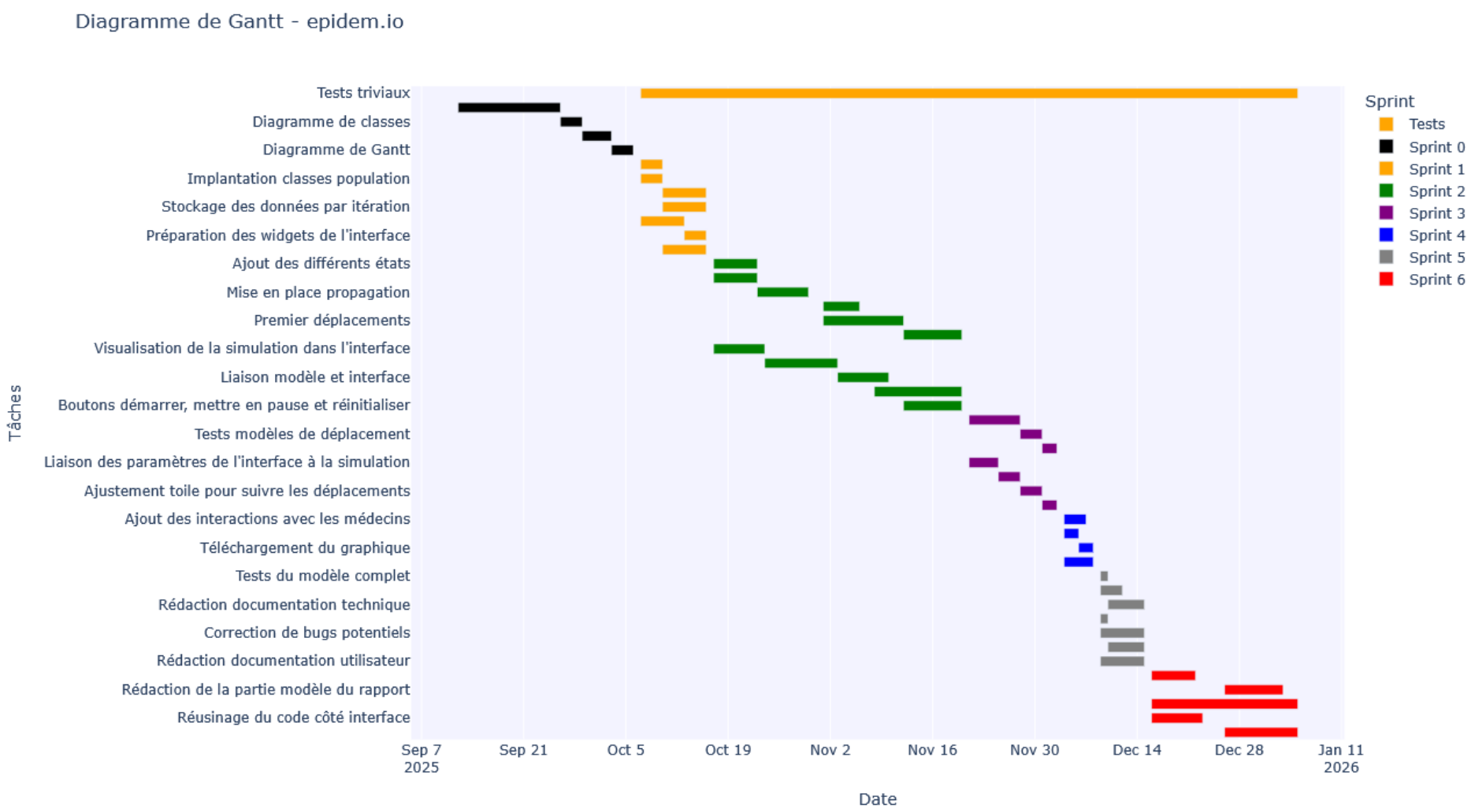


Diagramme de Gantt - epidem.io
Tâches
Tests triviaux
Diagramme de classes
Diagramme de Gantt
Implantation classes population
Stockage des données par itération
Préparation des widgets de l'interface
Ajout des différents états
Mise en place propagation
Premier déplacements
Visualisation de la simulation dans l'interface
Liaison modèle et interface
Boutons démarrer, mettre en pause et réinitialiser
Tests modèles de déplacement
Liaison des paramètres de l'interface à la simulation
Ajustement toile pour suivre les déplacements
Ajout des interactions avec les médecins
Téléchargement du graphique
Tests du modèle complet
Rédaction documentation technique
Correction de bugs potentiels
Rédaction documentation utilisateur
Rédaction de la partie modèle du rapport
Réusinage du code côté interface
Sep 7 2025
Sep 21
Oct 5
Oct 19
Nov 2
Nov 16
Nov 30
Dec 14
Dec 28
Jan 11 2026
Date
Sprint
Tests
Sprint 0
Sprint 1
Sprint 2
Sprint 3
Sprint 4
Sprint 5
Sprint 6

Diagramme de classe

Le logiciel sera organisé autour de classes principales :

- Personne : contient les informations pour chaque personne représentée dans la simulation
- Simulation : contient une liste de personnes et gère les interactions entre elles, gère les différentes interactions avec les autres classes qui influencent ou nécessitent les données des personnes
- Épidémie : règle la propagation selon les paramètres de la maladie
- Interface : gère l'affichage et l'interaction avec l'utilisateur
- Statistiques : gère le stockage des données au fur et à mesure des itérations pour exporter les résultats