Idée de dev

- RankActor
- Letterboxd Stats
- Six Degrees of Kevin Bacon
- Mon propre jeu vidéo
- Cozy game
- Hugo Pierret V3
- IP Finder
- Minecraft Mod Challenge
- Porfolio Gallery

RankActor

Une plateforme qui permettrait de classer ces acteurs préférer.

1. Base : Tier list personnalisable
   Acteurs et actrices récupérés via l'API TMDB (photos, noms, films, séries, nationalité…).
   Deux colonnes : acteurs à gauche, actrices à droite (ou un filtre pour basculer).
   Drag & drop pour déplacer les portraits entre rangs (S, A, B, C, D par ex.).
   Sauvegarde automatique du classement dans ton compte.
2. Fonctionnalités sociales
   Possibilité de partager son classement via un lien public.
   Mode « anonyme » pour montrer juste le classement sans ton pseudo.
   Comparer ton classement avec celui des autres (stats : % de gens qui ont mis tel acteur en S-Tier).
3. Infos enrichies
   Cliquer sur un acteur pour ouvrir une fiche : filmographie, notes, popularité, récompenses.
   Affichage des notes que toi tu as mises à ses films → cohérence entre tes préférences et le tier.
   Bouton « voir un film au hasard avec cet acteur ».
4. Personnalisation
   Choix du style graphique de la tier list (dark mode, couleurs personnalisées).
   Option pour séparer ou mélanger acteurs/actrices.
   Filtrage par genre de film ou période (années 90, films récents…).
5. Côté technique
   Frontend : React (avec un lib de drag-and-drop comme react-beautiful-dnd).
   Backend : Node.js + Express pour gérer la connexion à TMDB et stocker les classements.
   DB : MongoDB ou PostgreSQL pour stocker utilisateurs et classements.
   Auth via OAuth (Google, GitHub) pour éviter la gestion des mots de passe.
   Cache des requêtes TMDB pour éviter de spammer l'API.

Letterboxd Stats

□□ Statistiques générales

- Nombre total de films vus
- Nombre de films vus cette année / ce mois
- Acteur le plus vu
- Réalisateur le plus fréquent
- Durée totale passée devant des films
- Répartition par genre / pays / décennie

□□ Wrapped (style Spotify / Letterboxd Wrapped)

- Top 5 films les mieux notés
- Premier et dernier film de l’année
- Mois le plus actif
- Graphiques (barres, cercles, etc.)
- Design dynamique (fond, palette couleur, typo ciné)

□□ Export Story Instagram

- Bouton “Générer ma story” → génère une image 1080x1920 avec :
  - le résumé de ton année (ex: “205 films vus en 2025 □□”)
  - ton top films avec affiches
  - quelques stats visuelles

Idée indépendante du movie wrapped de 2025 :

écran séparer en 5 parties chaque partie apparait une après l'autre, celle au centre d'abord (2025 Movie Stats) puis ensuite les 4 autres de haute en bas Pour chaque partie, une stats (nombre de film, réalisateur le plus vu, meilleur films, ...) chaque partie correpond égalment à un plan de film je ne sais pas trop comment introduire les affiches dans ce cas

□□ Structure détaillée de ta story (15s environ) □□ 0–1s : Ouverture (fond noir, son ou texture)

Texte centré :

“My 2025 in films.”

Musique : quelque chose de cinématographique, lent au début puis un léger crescendo (ex : “The Night Window” – Thomas Newman ou “Cornfield Chase” – Hans Zimmer).

Un fade-in doux de lumière ou grain cinéma (façon pellicule).

□□ 1–3s : Bloc central (le premier à apparaître)

Texte : “2025 Movie Stats □□”

Plan de fond : un plan large d’un film calme et évocateur (ex : Lost in Translation, Dune, Her, etc.).

Animation : le bloc se “pose” au centre de l’écran, légèrement flouté derrière.

Astuce affiches : tu peux les faire apparaître en miniature translucide autour du texte (comme des fantômes de souvenirs) — par ex. 3–4 affiches qui s’effacent lentement.

□□ 3–6s : Bloc supérieur

Stat : “□□ 87 films watched”

Plan de fond : un travelling avant vers un écran de cinéma vide ou une foule dans une salle.

Animation : la partie se déploie du haut, comme un rideau.

Affiches : très discret — un reflet ou une bande horizontale de mini affiches qui glissent en bas de cette section.

□□ 6–9s : Bloc inférieur

Stat : “□□ Favorite director: Denis Villeneuve”

Plan de fond : un plan emblématique d’un de ses films (ex. Arrival ou Blade Runner 2049).

Effet : le nom du réalisateur apparaît avec une typographie élégante, légèrement surimposée à l’image.

Affiches : tu peux les intégrer dans un reflet sur l’écran, ou sous forme de mini vignettes qui se fondent dans le plan.

□□ 9–12s : Bloc gauche

Stat : “□□ Best film: The Zone of Interest”

Plan de fond : un plan fixe du film, légèrement désaturé.

Effet : l’affiche se glisse doucement dans le coin inférieur gauche, comme une signature visuelle.

Tu peux ajouter une phrase d’une ligne :

“A haunting masterpiece.”

❤ 12–15s : Bloc droit (clôture)

Stat : “□□ Total runtime: 175 hours of cinema”

Plan de fond : une salle qui s’éteint / un projecteur / rideau qui se ferme.

Texte final au centre :

“See you in 2026.”

Affiches : mini montage en fondu rapide, genre 5–6 qui défilent en arrière-plan avant le noir final.

□□ Conseils techniques :

Musique : choisis un morceau avec 2 montées progressives pour bien rythmer l’apparition des blocs.

Typo : sobre, fine et cinématographique (ex : Neue Haas Grotesk, Futura, Cormorant Garamond).

Palette : 2–3 tons max (ex : noir, beige, doré) pour un rendu “affiche de festival”.

Transitions : fade in/out légers, zoom lent, split reveal pour les blocs.

□□ Idée pour les affiches

Tu n’as pas besoin de toutes les montrer. Voici 3 options élégantes :

Fantômes visuels : affiches en opacité 10–20% dans les arrière-plans, qui bougent lentement.

Bande horizontale de 5–6 mini affiches dans un coin à chaque stat (comme une “pellicule”).

Explosion finale : les affiches apparaissent en mosaïque floutée derrière le texte “See you in 2026”.

Six Degrees of Kevin Bacon

Jeu web inspiré du concept des **six degrés de séparation** (variante "Six Degrees of Kevin Bacon"). Le but : relier deux acteurs entre eux en passant par des films qu'ils ont tournés en commun, en un minimum d'étapes.

- **Dépôt** : https://git.hugo-pierret.be/Hugyouu/six-degrees
- **App** : https://six-degrees.hugo-pierret.be

Fonctionnalités

- **Défi quotidien** — une paire d'acteurs par jour, commune à tous les joueurs
- **Mode infini** — parties en solo avec des paires aléatoires
- **Multijoueur temps réel** — lobbys via WebSocket (SignalR)
- **Calendrier** — historique des défis quotidiens passés
- **Visualisation graphe** — représentation interactive acteur → film → acteur (D3.js)

Stack technique

Couche	Technologie
Backend	ASP.NET Core 8.0 (C#)
Base de données	PostgreSQL + Entity Framework Core 8.0
Temps réel	SignalR
Frontend	React 18 + Vite
Styles	Tailwind CSS + Material Tailwind
Graphe	react-force-graph-2d (D3.js)
API externe	TMDB
Déploiement	Docker + Docker Compose

Architecture

```
six-degrees/
├── backend/
│   ├── SixDegrees.Api/            # Contrôleurs, hubs SignalR, config
│   ├── SixDegrees.Domain/         # Logique métier, services, entités
│   └── SixDegrees.Infrastructure/ # Accès données, migrations EF Core
├── frontend/
│   └── src/
│       ├── pages/                 # Vues (Home, DailyChallenge, Lobby…)
│       ├── components/            # Composants réutilisables
│       └── hooks/                 # Hooks custom (SignalR, username…)
├── docker-compose.yml
└── .env.example
```

Algorithme

Bidirectional BFS pour trouver le chemin le plus court entre deux acteurs (profondeur max = 6 degrés).

Modèle de données

Stocké en PostgreSQL via Entity Framework Core :

- **Person** `{ id, name, popularity }` — acteurs/célébrités
- **Title** `{ id, name, year, type }` — films et séries
- **Relation** `Person → Title` (ACTED_IN)

Données importées depuis l'API TMDB.

Lancer le projet

Prérequis

- Docker et Docker Compose
- Clés API TMDB (compte gratuit sur themoviedb.org)

Démarrage rapide

```
cp .env.example .env
# Remplir TMDB_API_KEY, TMDB_ACCESS_TOKEN, etc.
docker compose up --build
```

Application disponible sur `http://localhost` (ou `APP_PORT`).

Variables d'environnement

Variable	Description
`TMDB_API_KEY`	Clé API TMDB (requise)
`TMDB_ACCESS_TOKEN`	Token d'accès TMDB (requis)
`DB_CONNECTION_STRING`	Chaîne de connexion PostgreSQL
`APP_PORT`	Port exposé pour le frontend (défaut : `80`)

Développement local

Backend

```
cd backend
dotnet restore
dotnet run --project SixDegrees.Api/
# API sur http://localhost:8080
# Swagger : http://localhost:8080/swagger
```

Frontend

```
cd frontend
npm install
npm run dev
# Dev server sur http://localhost:5173
```

CI/CD

Workflow Gitea (`.gitea/workflows/deploy.yaml`) déclenché automatiquement sur chaque tag `v*.*.*` — rebuild et redéploiement via Docker Compose.

```
git tag v1.0.0
git push --tags
```

Notes de conception initiale

Lors de la phase de prototype, plusieurs choix ont été évalués :

Critère	Option retenue	Alternatives envisagées
BDD graphe	PostgreSQL + EF Core	Apache AGE (extension PG), Neo4j
Backend	ASP.NET Core (C#)	Node.js / Express
Algo chemin	Bidirectional BFS	BFS simple, Dijkstra
Source données	TMDB	Wikidata, IMDb

Scoring (mode challenge)

- **Base** : `(6 - degrés) × 100` points
- **Bonus temps** : +50 pts si trouvé en moins d'1 minute
- **Bonus rareté** : +50 pts si le chemin inclut un titre peu populaire

Mon propre jeu vidéo

Genre

- Rogelike
- Metroidvania

DA

- Pixel art

Inspiration

- Hollow Knight
- Celeste
- Wallkeeper, DomeKeeper

Concept

- Boucle temporelle
- zone à explorer durant une durée déterminée
- boss et mob à battre
- quêtes pour survivre au boucle
- boss final genre un gardien temporel

Progression

- Une partie peut durer un certain temps, pas comme un roguelite classique. Donc la carte est assez grande et fixe. Possible de sauvegarder sa progression et le nombre de boucle.
- La forge du temps, le hub central ou tu peux t'améliorer. Utile après chaque fin de boucle puisque tu renviens ici à chaque fois.
- La carte reste la même à chaque fois mais les éléments (pnj, ressources, ...) bougent à chaque fois

Mécanique

- Croiser son précédent "moi" dans les boucles suivantes

- L'environnment se dégrade au fur et à mesure des boucles !! pas sûr de l'idée prcq ça va à l'encontre des boucles temporelles
- Les pnj sont conscient de la boucle donc se trouve à des endroit différents à chaque fois, ils t'aident donc à avancer dans la quête principale et peuvent te donner des quêtes à travers les boucles.

PNJ / Boss

Les bosses qui sont conscient des boucles se souviennent de toi et s'adapte quand tu les recombats

- L'Oubliée : bosse un peu primitif qui n'est pas trop conscient de ce qui se passe
- Miroir brisé
- Gardien du temps
- L'Horlogé
- L'ancre : reste toujours au même endroit
- Les perdus : ils ne retiennent rien et ne sont conscient de rien (mob classique)

But

- Vaincre le boss final en moins de boucles possible
- S'améliorer de plus en plus

Cozy game

Hugo Pierret V3

IP Finder

une simple interface pour afficher les ip utilisée sur mon serveur pour ensuite choisir laquelle je vais bien pouvoir utiliser pour un futur service

Minecraft Mod Challenge

? Concept général

Un mod qui transforme Minecraft en **hub de mini-jeux compétitifs**, jouable en solo ou en multijoueur (LAN/serveur). Le joueur choisit un défi via un menu, une partie démarre avec ses propres règles, son timer et son scoring, et un classement s'affiche à la fin.

L'idée : réutiliser les mécaniques vanilla de Minecraft (craft, exploration, combat) comme base de mini-jeux courts et rejouables, façon "party game".

? Mini-jeux

1. Craft Speedrun

- **Objectif** : être le premier à crafter un item cible (ex: diamond pickaxe, elytra...) depuis un monde fraîchement généré.
- **Règles** : item imposé aléatoirement (ou piochable dans une liste), timer visible, inventaire vidé au départ.
- **Scoring** : temps chronométré, classement des joueurs/parties.
- **Variantes** : - *Item mystère* : l'item cible n'est révélé qu'au lancement. - *Craft imposé en chaîne* : une liste d'items à crafter dans l'ordre. - *Ressources limitées* : biome ou zone restreinte.

2. Guess the Item / Block

- **Objectif** : deviner un item ou bloc à partir d'un indice donné par un autre joueur (ou généré par le mod).
- **Mécanique** : un joueur "maître du jeu" (ou le mod) affiche un indice (silhouette floutée, son du bloc, texture zoomée, recette partielle), les autres proposent une réponse dans le chat ou via un GUI.
- **Scoring** : points selon rapidité de la réponse, bonus pour bonne réponse au premier indice.
- **Variantes** : - *Mode équipe* : un buzzer pour répondre en premier. - *Indices progressifs* : l'indice se précise toutes les 5 secondes (moins de points si on attend).

3. Bingo

- **Objectif** : compléter une grille (3x3 ou 5x5) d'objectifs avant les autres.
- **Contenu de la grille** : items à obtenir, advancements à débloquer, mobs à tuer, structures à visiter.
- **Règles de victoire** : ligne, colonne, diagonale, ou grille pleine (configurable).
- **Scoring** : premier à valider une condition de victoire, ou score cumulé en temps limité.
- **Variantes** : - *Bingo synchronisé* : tous les joueurs ont la même grille (lockout bingo). - *Bingo dynamique* : la grille se régénère à chaque case validée.

3. Blind/Deaf/Mute

- **Objectif** : Finir le jeux avec un joueur muet, un sourd et un aveugle

?? Partie technique

Choix de plateforme

- **Forge** ou **Fabric** pour un mod client+serveur complet (accès plus profond aux mécaniques du jeu).
- **Alternative plugin Paper/Spigot** si on veut rester serveur-only et compatible sans installation côté client.

Architecture proposée

- `ChallengeManager` : gère le cycle de vie d'une partie (start/stop/reset), le timer, le classement.
- Interface `Minigame` (ou classe abstraite) implémentée par chaque mini-jeu (`CraftSpeedrun`, `GuessTheItem`, `Bingo`) avec des hooks `onStart()`, `onEnd()`, `onEvent()`.
- `GameSession` : état d'une partie en cours (joueurs, scores, temps restant).

Événements Minecraft à écouter

- `PlayerJoinEvent` / `PlayerQuitEvent` — gestion des participants.
- `BlockBreakEvent` / `BlockPlaceEvent` — pour le Craft Speedrun et le Bingo.
- `PlayerAdvancementDoneEvent` — objectifs de type advancement.
- `EntityDeathEvent` — objectifs de type kill.
- `InventoryClickEvent` — crafts et items obtenus.

UX / Interface

- GUI (inventaire custom) pour choisir le mini-jeu et voir les règles.

- Affichage du timer et du score via **scoreboard** ou **action bar**.
- Config en YAML/JSON pour définir les listes d'items, grilles de bingo, durée des manches.

Commandes envisagées

- `/challenge start <jeu>` — lance une partie.
- `/challenge join` / `/challenge leave` — rejoindre/quitter.
- `/challenge stats` — voir le classement.

? Idées bonus

- Thèmes saisonniers (grilles de bingo spéciales Halloween, Noël...).
- Power-ups temporaires pendant les speedruns (vitesse, night vision).
- Mode équipe avec score partagé.
- Replay/leaderboard persistant entre sessions.

Porfolio Gallery

Un site vitrine pour afficher toutes ses petites photos prises
Photos accrochées à des files avec des pince à linge
Une sorte de jardin et les cordes sont entre des arbres
Les photos seraient des négatifs à la base\