



STREET OF AGES

DMII -1 2019

Iris Mangin

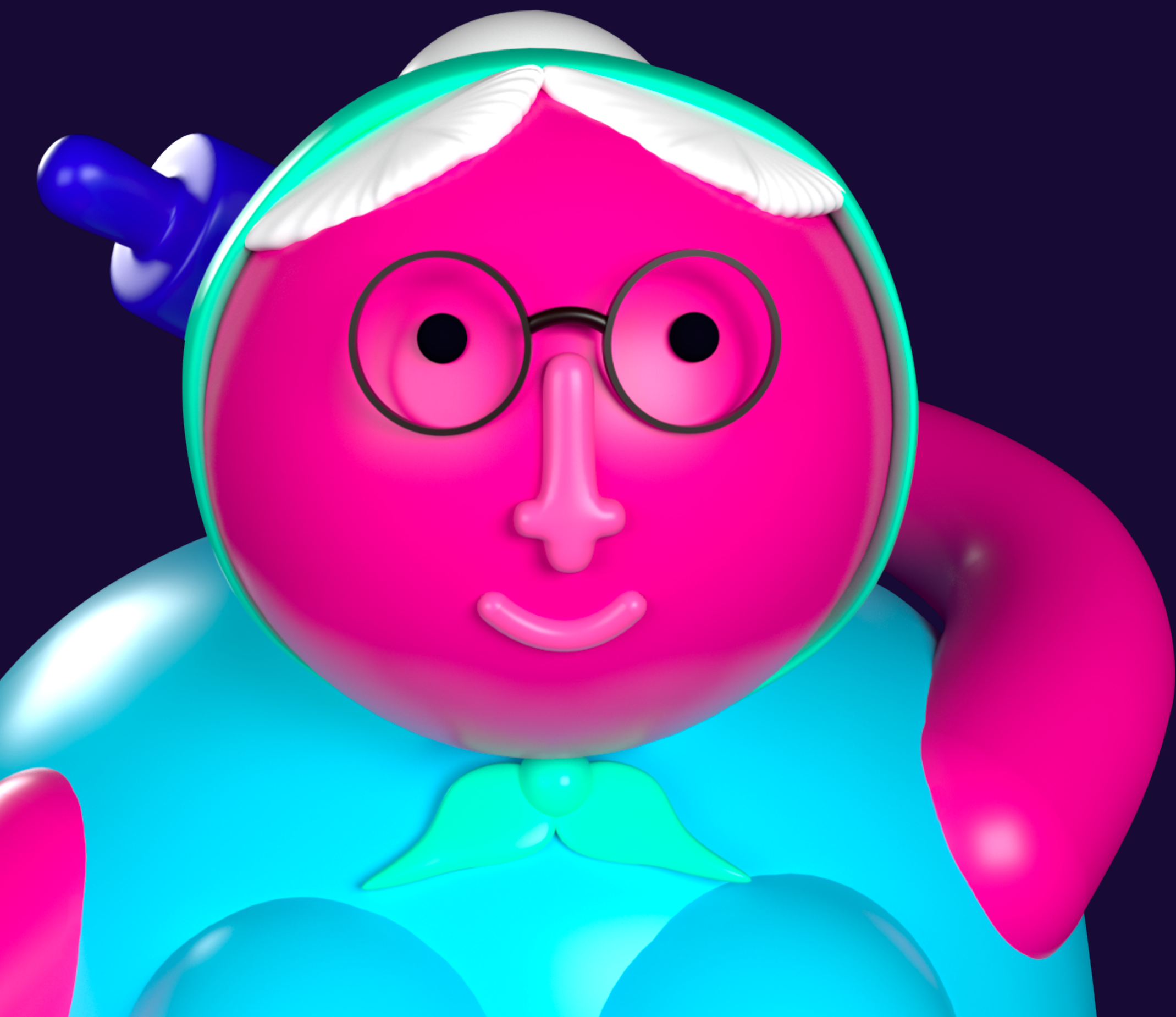
Bastien Lepêcheur

Omar Jbara

Julien Dargelos

Fany Thourain

SOMMAIRE



03 RECHERCHE UTILISATEUR

Le sujet

Le brief

Les interviews

06 RECHERCHE CREATIVE

Les personnages

Le process

Les 3D

La physique

Le moodboard

La map

14 PARCOURS UTILISATEUR

Wireframes

16 IDENTITE GRAPHIQUE

Charte graphique

Map

18 INTERFACE

19 SPECS. TECHNIQUES

24 PLANNING



RECHERCHE UTILISATEUR

LE SUJET

LE CHOC GENERATIONNEL

Vieux vs Jeunes

LES INTERVIEWS

« Les vieux sont lents »

« Les jeunes sont ingrats »

« Les vieux croient tout savoir »

« Les jeunes sont insolents »



RECHERCHE UTILISATEUR

BIREF

CIBLE

16 - 35 ans

Adeptes des jeux mobile
et/ou multijoueurs

EVOCATION

- Rapport de force entre les générations.
- Clichés / stéréotypes
- Combat / Opposition

CONTRAINTE

- Jeu multijoueur en réseau
- Jeu tactile et rapide (5 minutes)
- Jeu au format mobile (petit)
- Traduire et identifier les stéréotypes à petite échelle

TONALITE

Drôle / décalé / satirique

PROBLEMATIQUE

Comment rassembler
chaques générations et
créer un moment de
partage amusant autour
des stéréotypes qu'ont
chaques générations ?

RECHERCHE UTILISATEUR

LES INTERVIEWS

LA NOSTALGIE

« C'était mieux avant »

- Les anciennes générations ne disent pas que "c'était mieux avant" : Quand on interroge des individus nés entre 1938 et 1991 sur les différences entre leur génération et la génération actuelle, seulement 12,5% affirment que « c'était mieux avant ».

- Pourtant elles semblent plutôt le penser : 50% déclarent que la jeunesse actuelle entretient de moins bonnes relations sociales que la leur. 25% déclarent que la jeunesse actuelle est plus inconsciente que la leur.

- Quand on demande à des individus nés entre 1997 et 2003 s'ils entendent souvent l'expression "c'était mieux avant", 75% d'entre eux répondent affirmativement, et 83% de ces mêmes individus déclarent entendre cette expression de leurs parents.

- Les jeunes sont curieux de l'époque de leurs parents: 50% aimeraient vivre dans les années 70, 80 ou 90. Les jeunes fantasment sur les époques antérieures.

- Ils sont nostalgiques d'un passé qu'ils n'ont pas connu : 62,5% d'entre eux se déclarent intéressés par des objets de ces mêmes décennies (Polaroid, Vinyle, Consoles de jeu)

VIEUX VS JEUNES

Vieux vs Jeunes

- Les idées que se font chaque génération entre elles sont souvent empreintes de stéréotypes.

- Vieux comme jeunes ont des idées reçues qui nourrissent les clichés.

- Les jeunes qui sont fénéants, insolents, ingrats et de plus en plus bêtes

- Les vieux qui sont lents, manque d'ouverture, largués, bloqué dans le passé



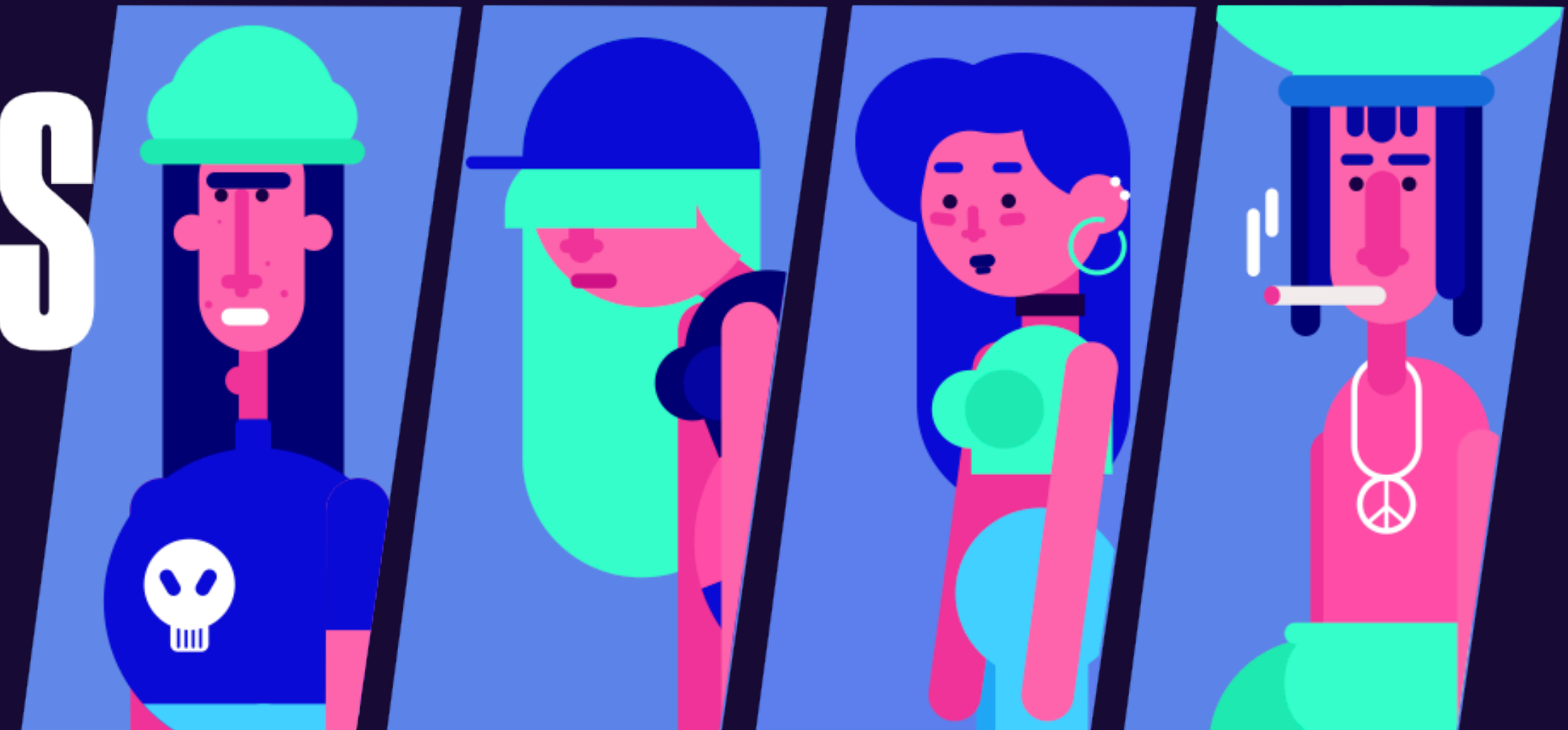
RECHERCHE CREATIVE

LES PERSONNAGES

LES ANCIENS



VS



LES JEUNES

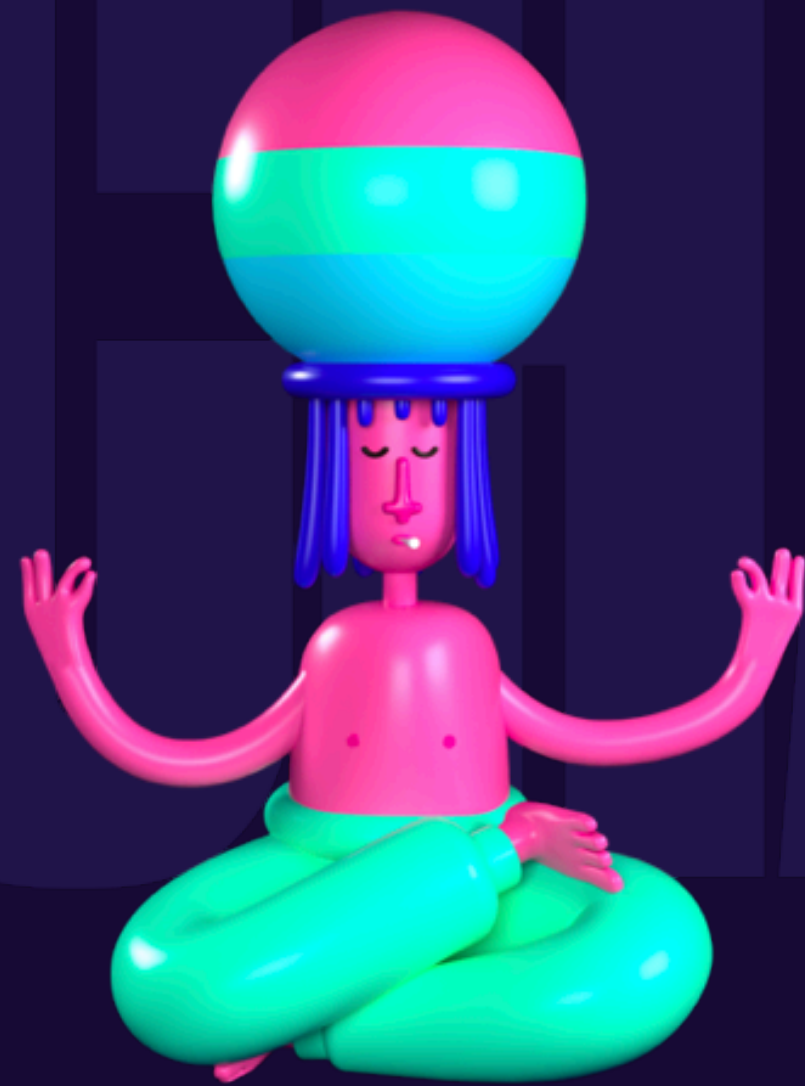
RECHERCHE CREATIVE

LE PROCESS




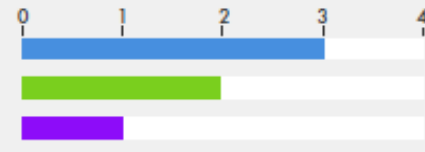



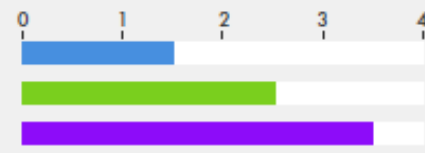

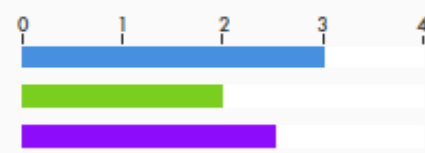

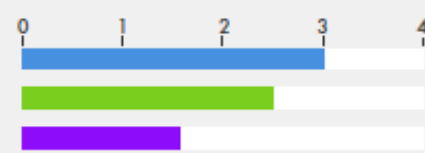

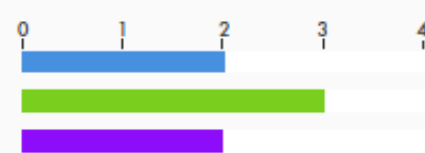

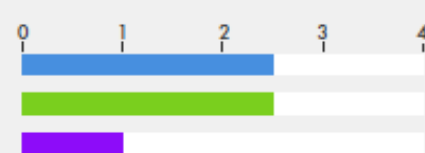

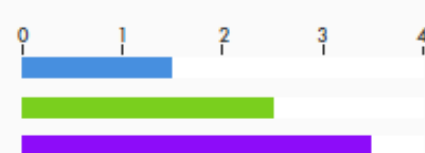
RECHERCHE CREATIVE

LES 3D



RECHERCHE CREATIVE

LA PHYSIQUE

Personnage	Attaque Rapprochée	Dégâts	Attaque Distance	Dégâts	Mouvements	Résumé
 Mamie Gateau	Rouleau a pâtisserie <i>Air d'effet Collé</i>	2 dégâts / Forte projection	Gâteau explosif <i>Masse légère / Rebond très léger / 4 seconde de délais avant explosion / Forte air d'effet</i>	1 dégâts / Très forte projection	Lent 1/4	
 Vieux Raciste	Rot <i>Air d'effet Courte</i>	2 dégâts / Légère projection	Golf canette de bière <i>Masse nul / Rebond élevé / Disparaît après 5 rebond</i>	1 Dégâts par contact avec un joueurs / Projection Légère	Moyen 2/4	
 Vieille Bimbo	Chien qui crache du feu <i>Air d'effet Courte</i>	1 dégâts / Moyenne projection	Lancer de caniche <i>Masse assez lourd / Forte friction / Rebond léger / Disparaît au contact</i>	2 Dégâts / Projection Moyenne	Rapide 3,5/4	
 Vieux musclé	Coup de poing <i>Air d'effet Collé</i>	2 dégâts / Forte projection	Lancer de Bouée <i>Masse légère / Rebond élevé / Disparaît après 5 rebond</i>	1 Dégâts par contact avec un joueurs / Projection Légère	Moyen 2,5/4	
 Jeune Boulette	Coup de Pied <i>Air d'effet Collé</i>	2 dégâts / Forte projection	Strike (ballon) <i>Masse légère / Rebond élevé / Disparaît après 5 rebond</i>	1 Dégâts par contact avec un joueurs / Projection Légère	Lent 1,5/4	
 Jeune métal geek	Riff de guitare <i>Air d'effet Moyenne</i>	1 dégâts / Forte projection	Lancer de Gameboy <i>Tir courbé / rebond moyen / Disparaît au choc</i>	2 Dégâts / Projection Moyenne-Forte	Moyen 2/4	
 Jeune Babos	Envoie du joint <i>Air d'effet Moyenne</i>	2 dégâts / Très légère projection	Lancer de Diabolo <i>Masse légère / rebondissement moyen / Disparaît après 3 rebond</i>	1 Dégâts par contact avec un joueurs / Projection Moyenne	Lent 1/4	
 Jeune Influenceuse	Flash de l'appareil <i>Air d'effet Large</i>	1 Dégâts pour chaque joueurs présent à distance moyenne du joueurs / Très légère projection	Lancer de Bad buzz <i>Masse légère / rebondissement Fort / 4 seconde de délais avant explosion / Forte air d'effet</i>	2 dégâts / Forte projection	Rapide 3,5/4	

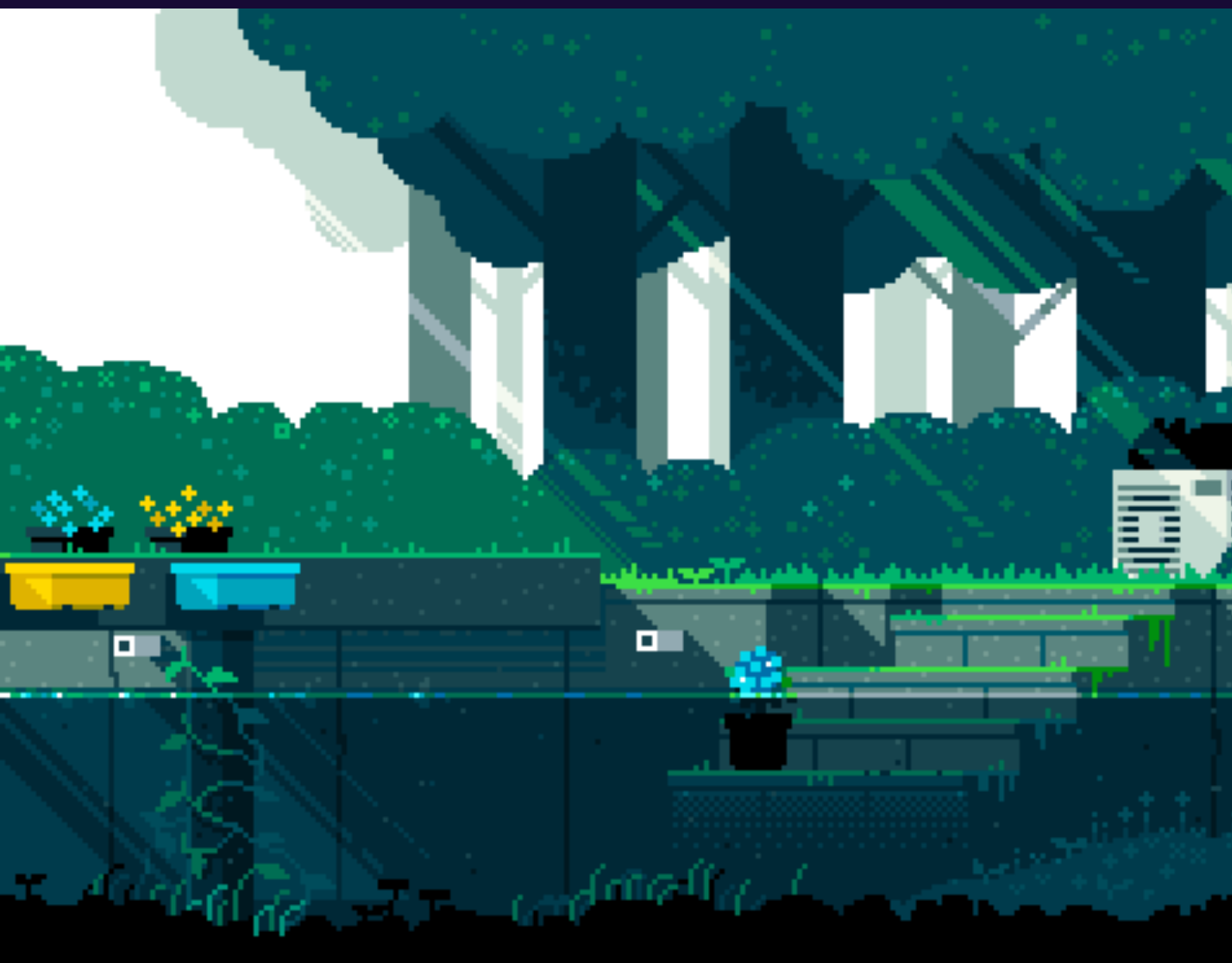
RECHERCHE CREATIVE

LE MOODBOARD/LUMIÈRES



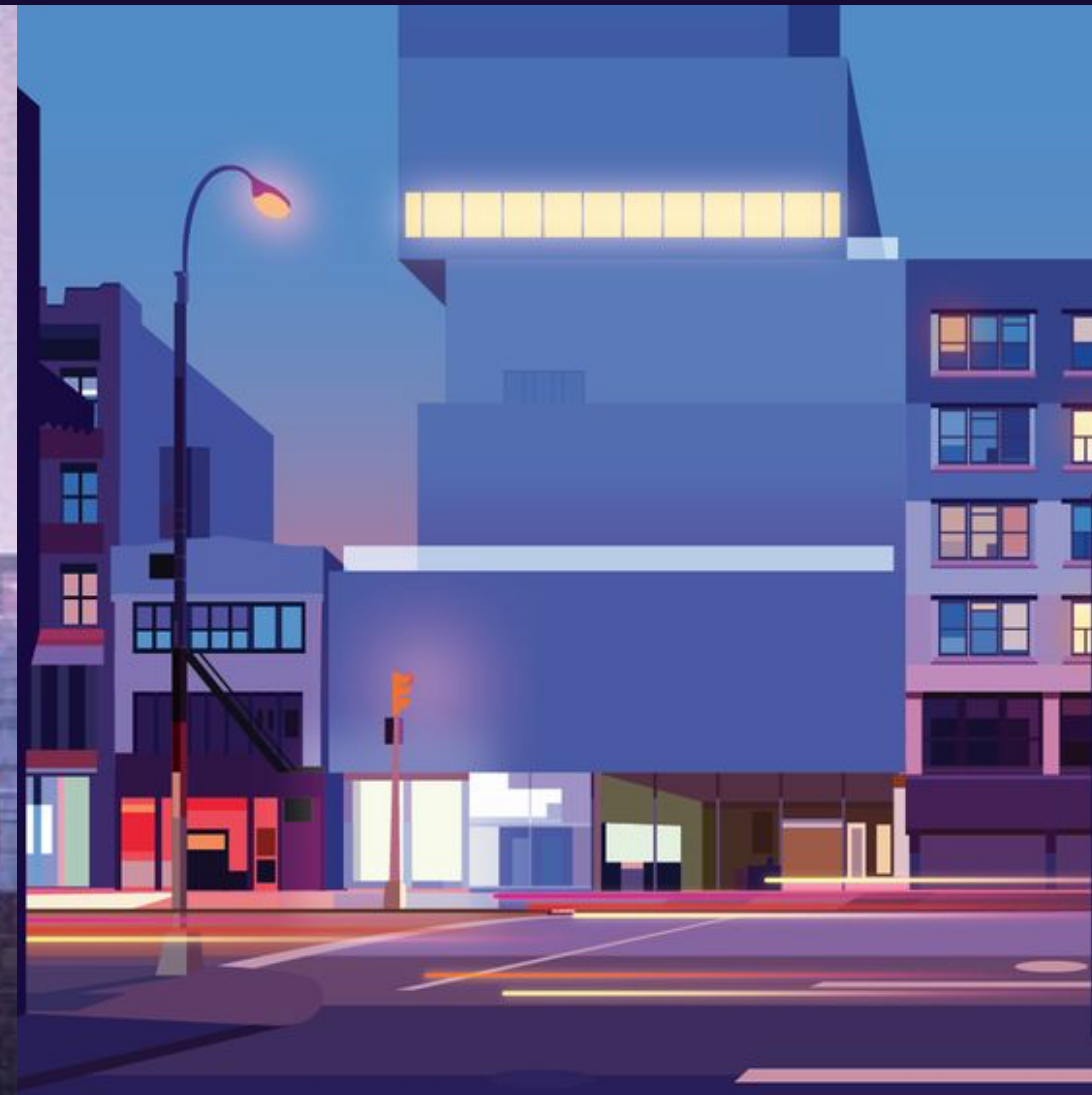
RECHERCHE CREATIVE

LE MOODBOARD/DÉCORS DU JEU



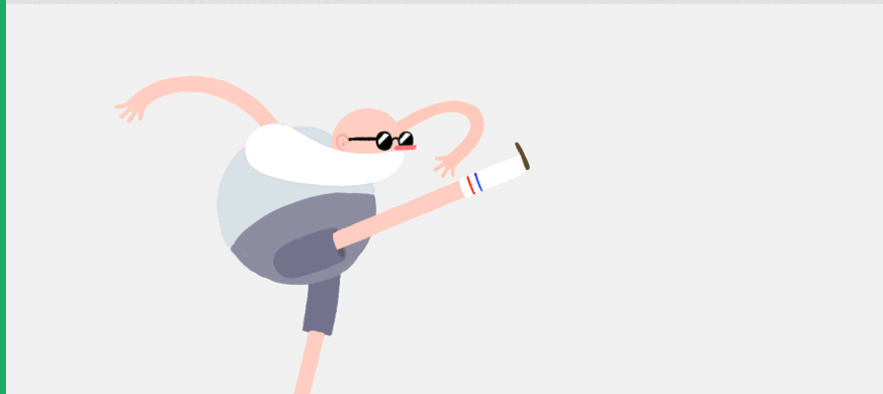
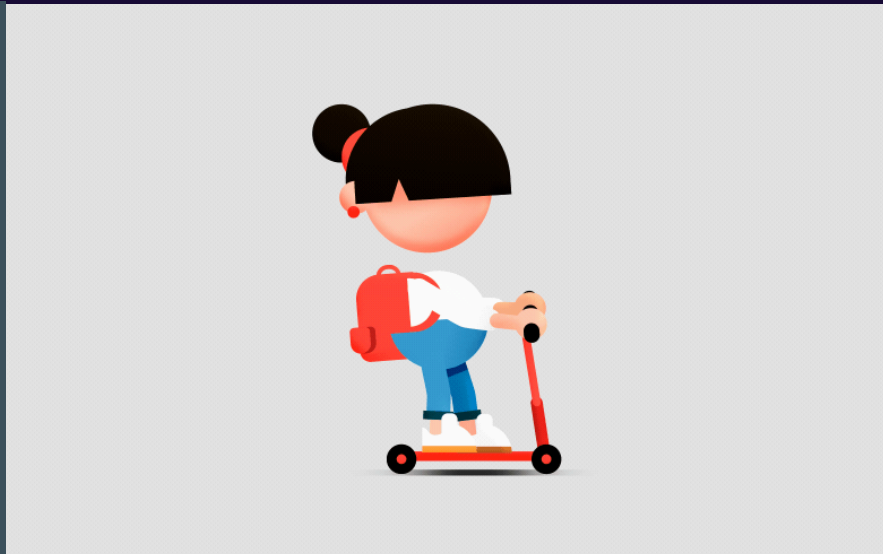
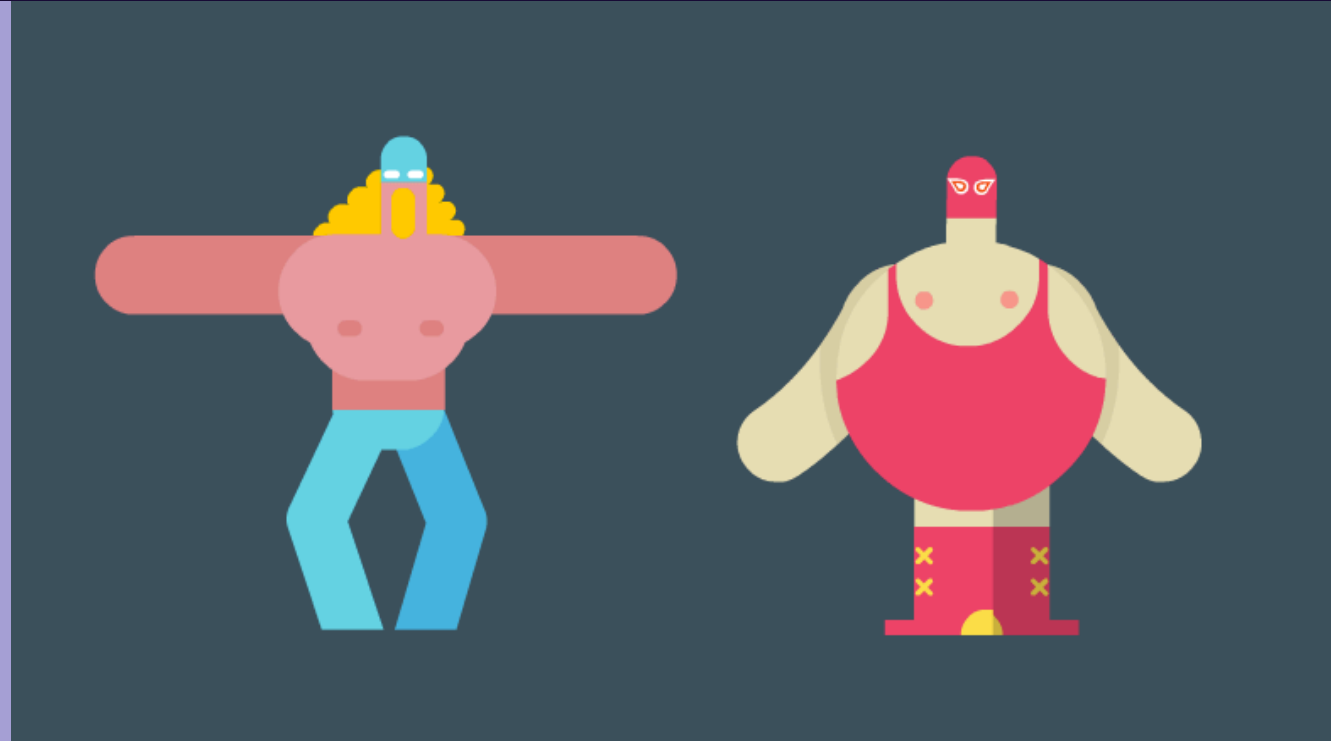
RECHERCHE CREATIVE

LE MOODBOARD/AMBIANCE



RECHERCHE CREATIVE

LE MOODBOARD/CHARACTER DESIGN



PARCOURS UTILISATEUR

WIREFRAMES

Ecran de démarrage



Tutoriel



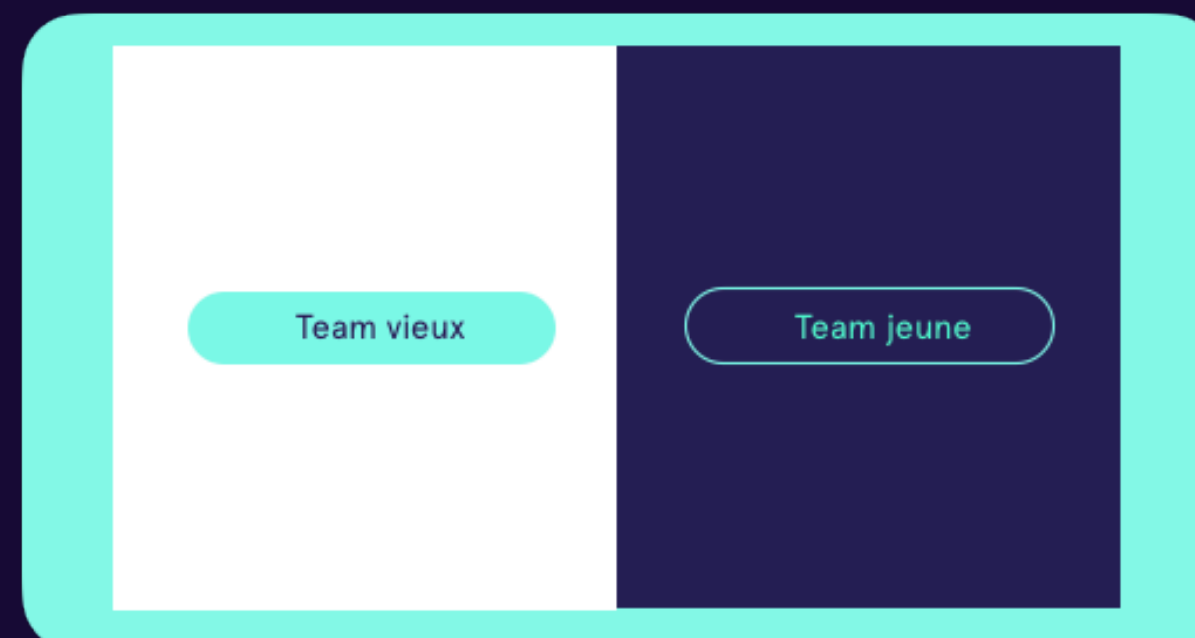
Menu principal



Configuration de partie



Ecran de choix d'équipe



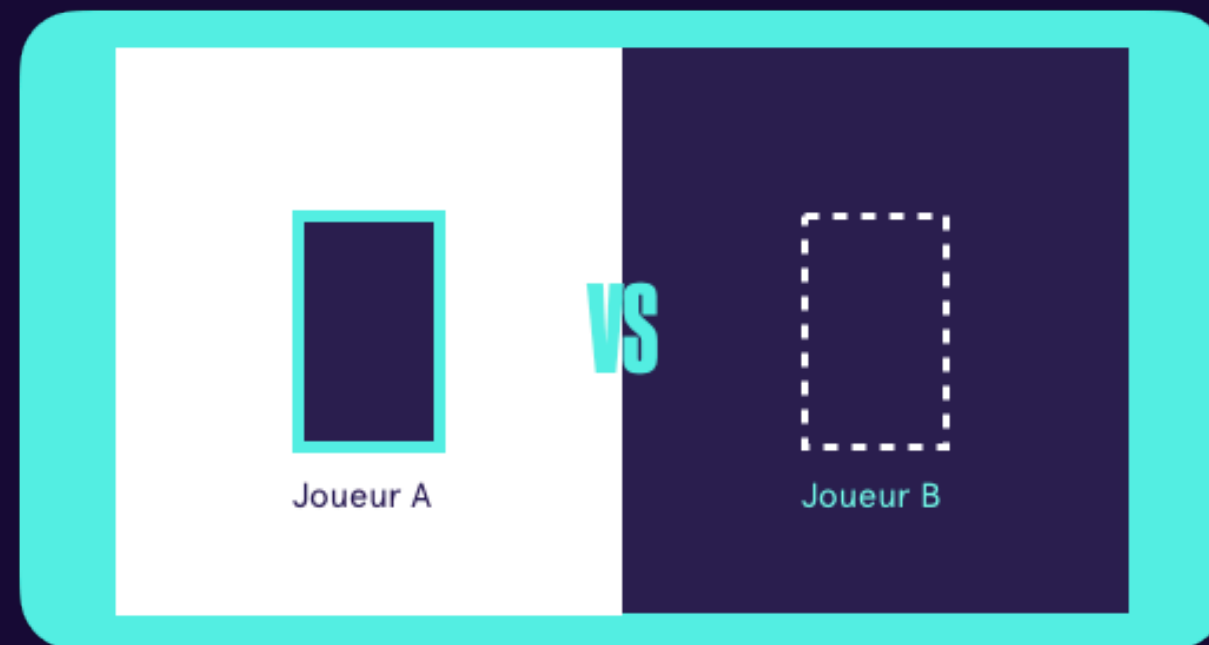
Ecran de choix de personnage



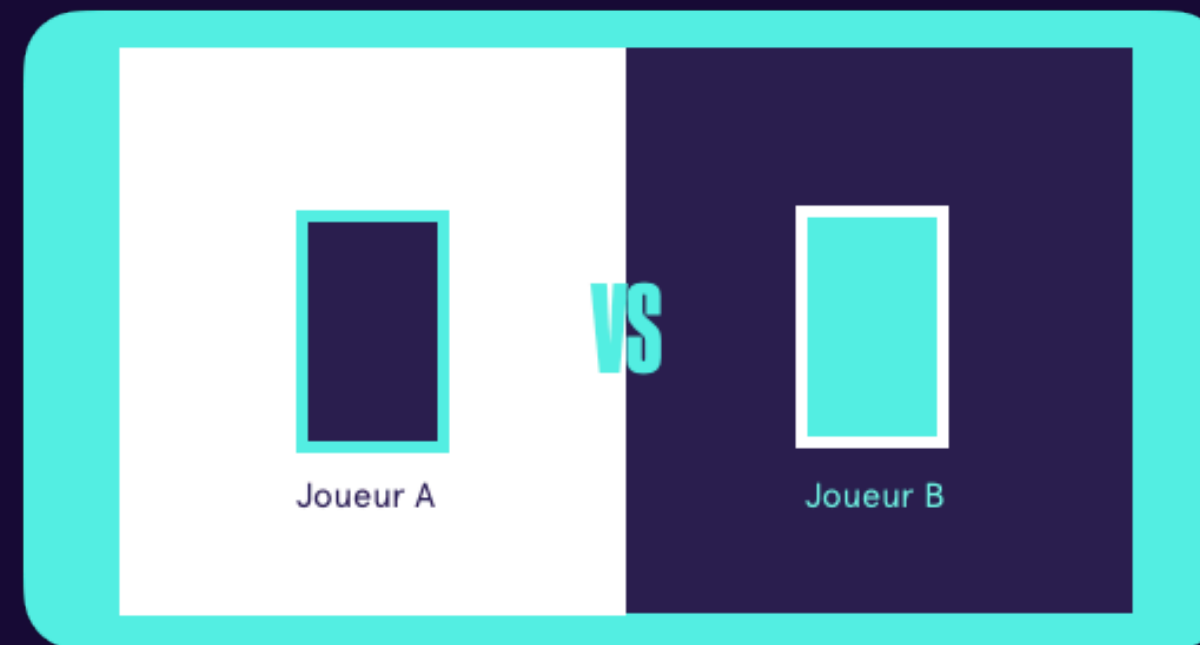
PARCOURS UTILISATEUR

WIREFRAMES

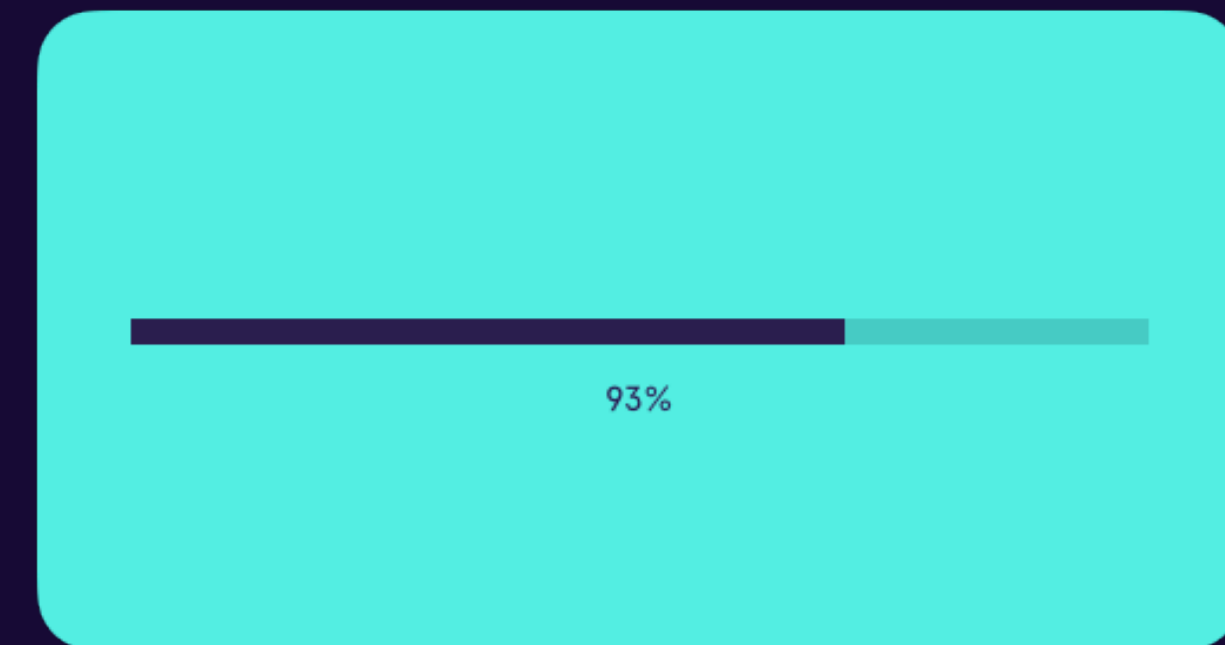
Ecran salle d'attente



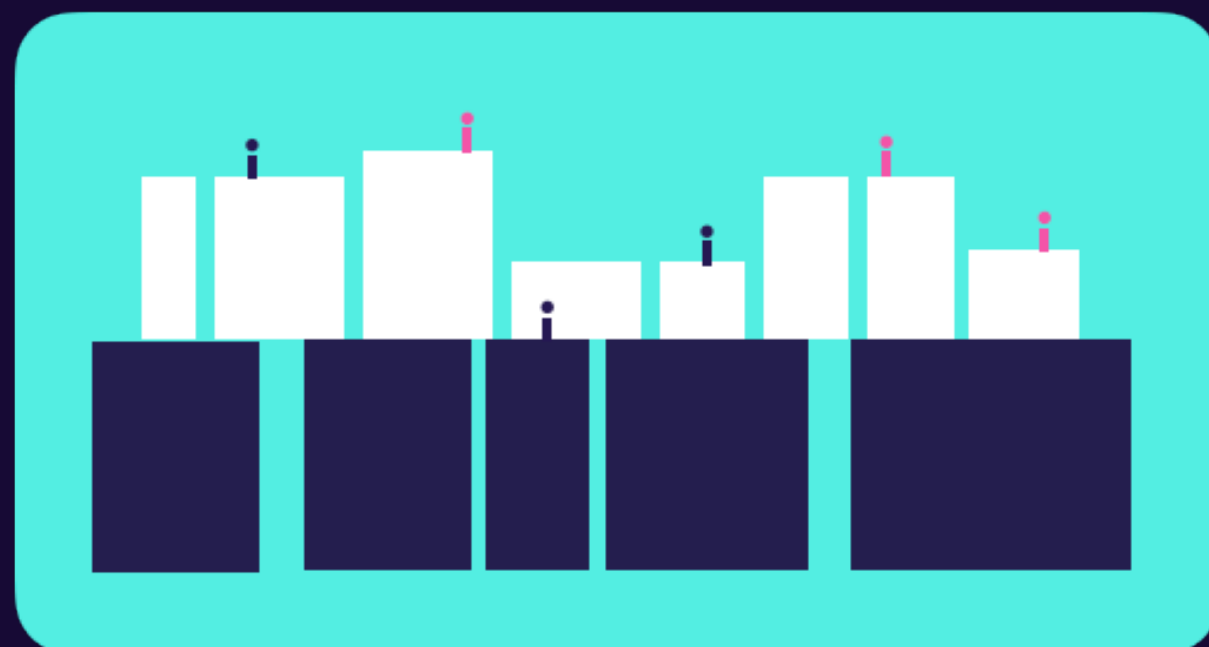
Ecran salle d'attente



Loading



Jeu



Ecran resultat



Menu principal



IDENTITE GRAPHIQUE

LA CHARTE GRAPHIQUE

Le logotype



Les boutons



Les typos

Drak Cond Super
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz

Les couleurs



Les personnages



Aperçu

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz

Les éléments de décors

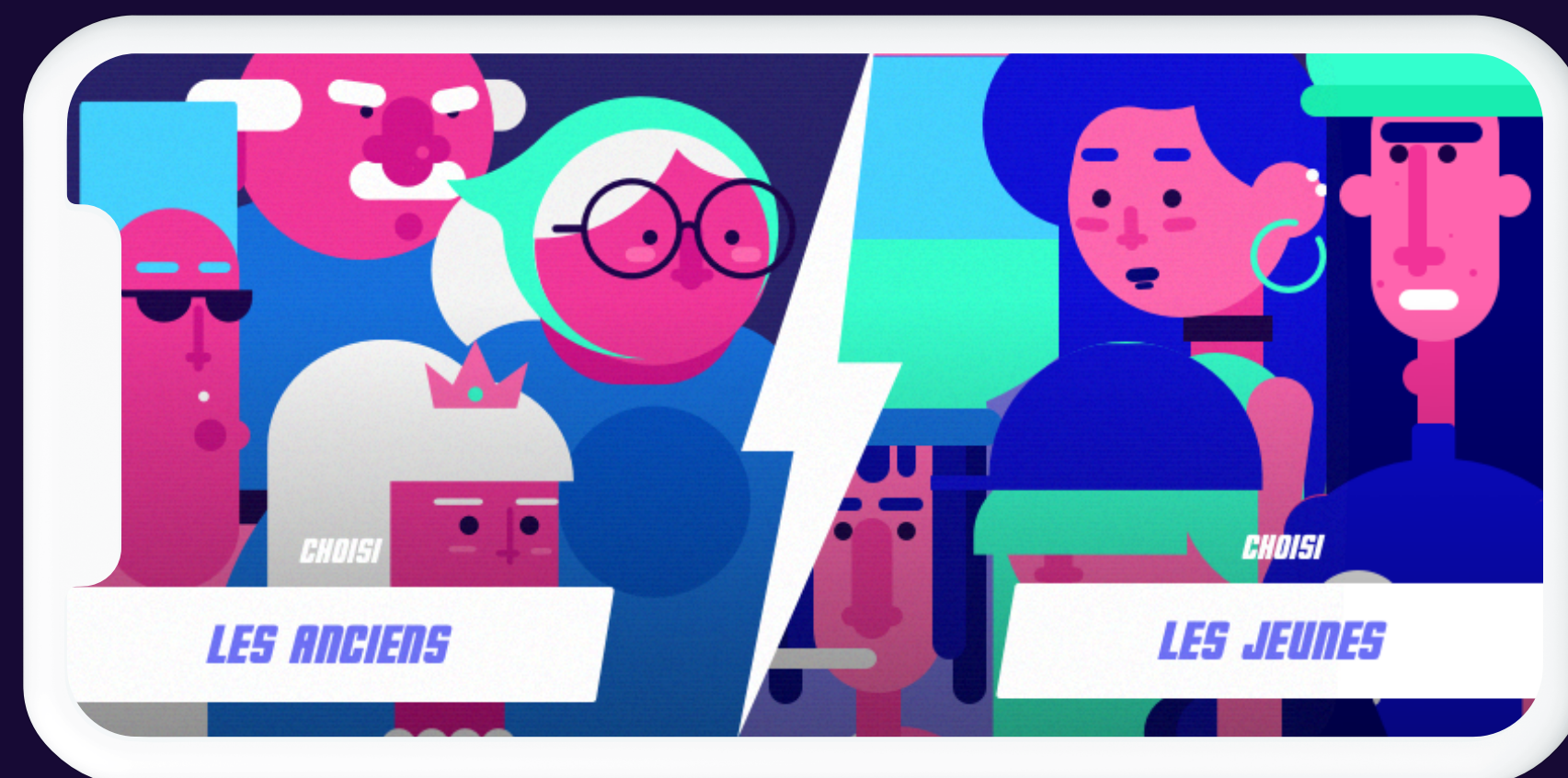
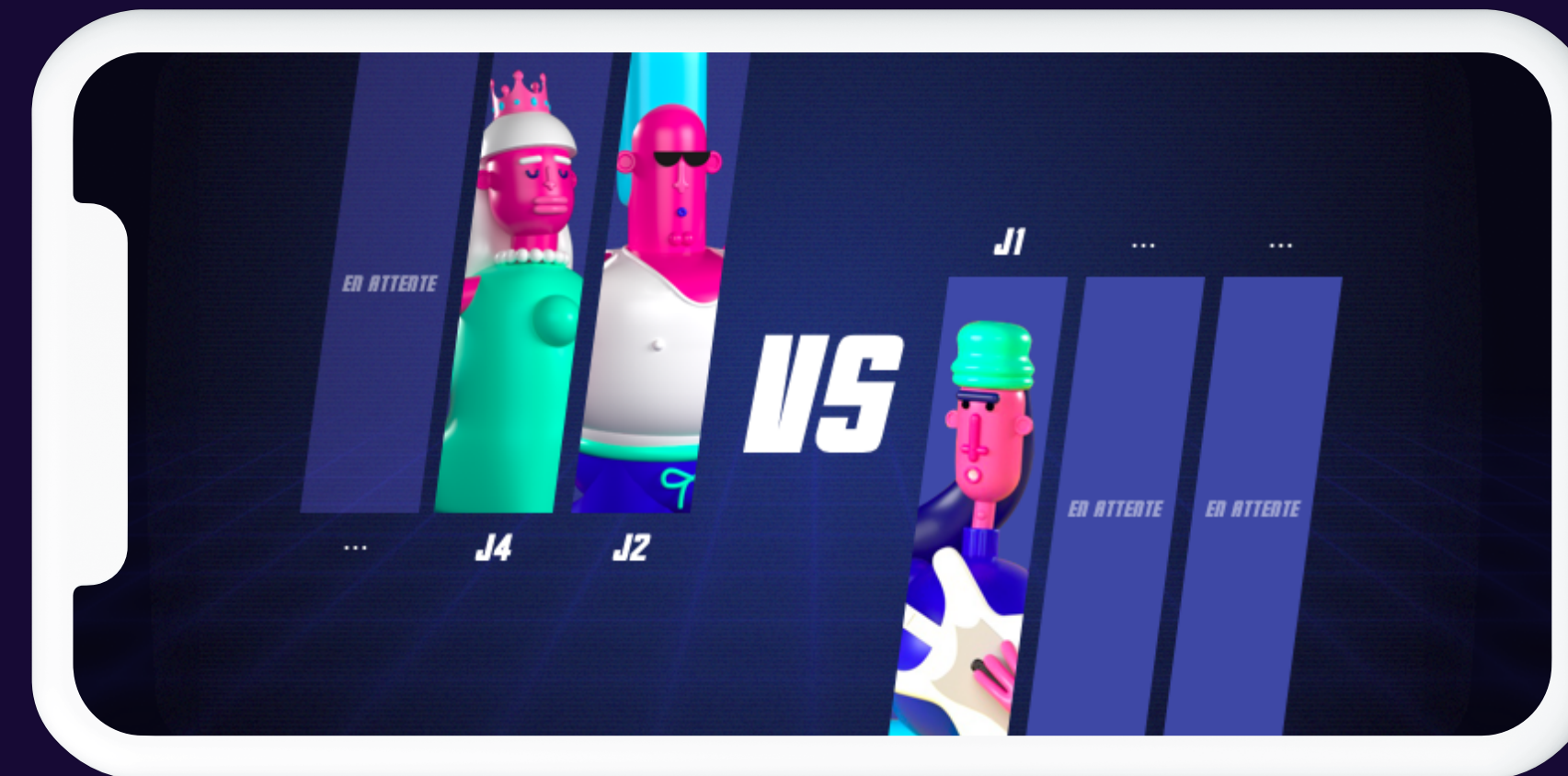
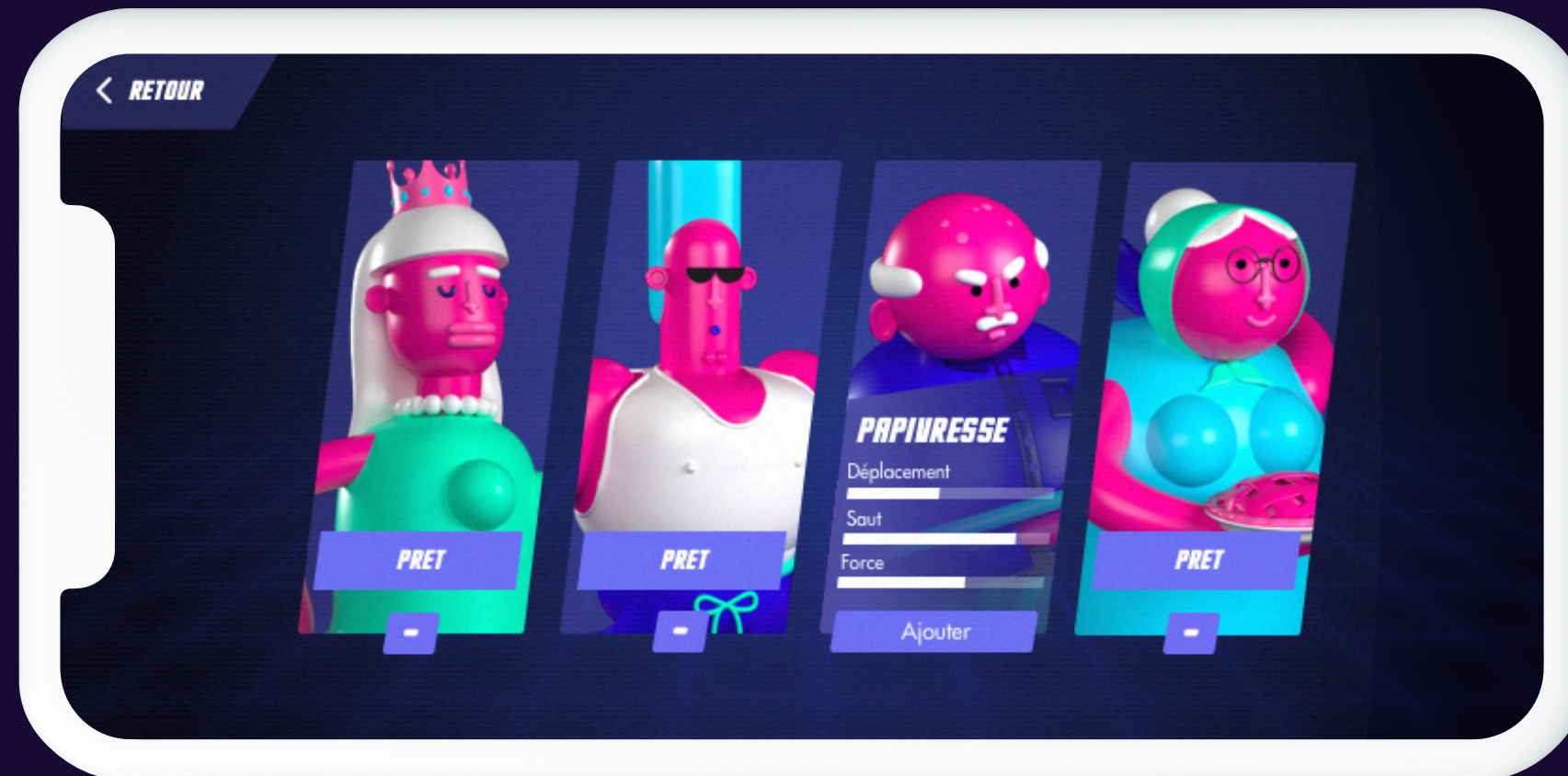


IDENTITE GRAPHIQUE

LA MAP



INTERFACE



SPECIFICATIONS FONCTIONNELLES

TECHNIQUES



INTRO

Notre équipe technique, loin des fruits de la contingence, trouve son origine dans un souhait partagé de collaboration, motivé par nos affinités quant à l'organisation, l'architecture, et les conventions que nous adoptons pour produire une application.

Ce document saura en témoigner en synthétisant nos ambitions techniques ainsi que les contraintes et intérêts qui les justifient. Ce sera également un outil précieux pour évaluer, a posteriori, la pertinence de nos décisions.

Avant de détailler nos choix techniques, rappelons brièvement la nature de ce que nous nous apprêtons à produire: un jeu mobile multijoueurs en ligne dont la mécanique s'apparente à celle du célèbre jeu Worms en 2 dimensions, à savoir un combat entre deux équipes qui éliminent leurs adversaires en lançant des projectiles pour les faire tomber en dehors de l'arène.

TECHNOLOGIE

Notre réflexion autour des technologies à utiliser s'oriente principalement vers deux bibliothèques que sont PhaserJS et Pixi.js. Phaser étant un framework, il constitue une solution clé en main pour tous les besoins de création de jeux.

Comprenant notamment un gestionnaire d'inputs et un gestionnaire de son, Phaser est tout à fait pertinent lorsqu'il s'agit d'abstraire une grande partie de la logique de jeu. À vrai dire, la comparaison entre Phaser et Pixi n'est pas tout à fait équitable, Pixi ne s'attèle à répondre qu'aux problématiques de rendu (Phaser utilise d'ailleurs Pixi comme moteur de rendu).

Un autre avantage apporté par Phaser par rapport à Pixi est celui de la gestion de la physique. Phaser inclut plusieurs moteurs physiques répondant chacun à des

SPECIFICATIONS FONCTIONNELLES

TECHNIQUES

probablement le moteur dit arcade car c'est un moteur assez simpliste, peu gourmand en ressources, suffisant pour le type de jeu que nous souhaitons créer.

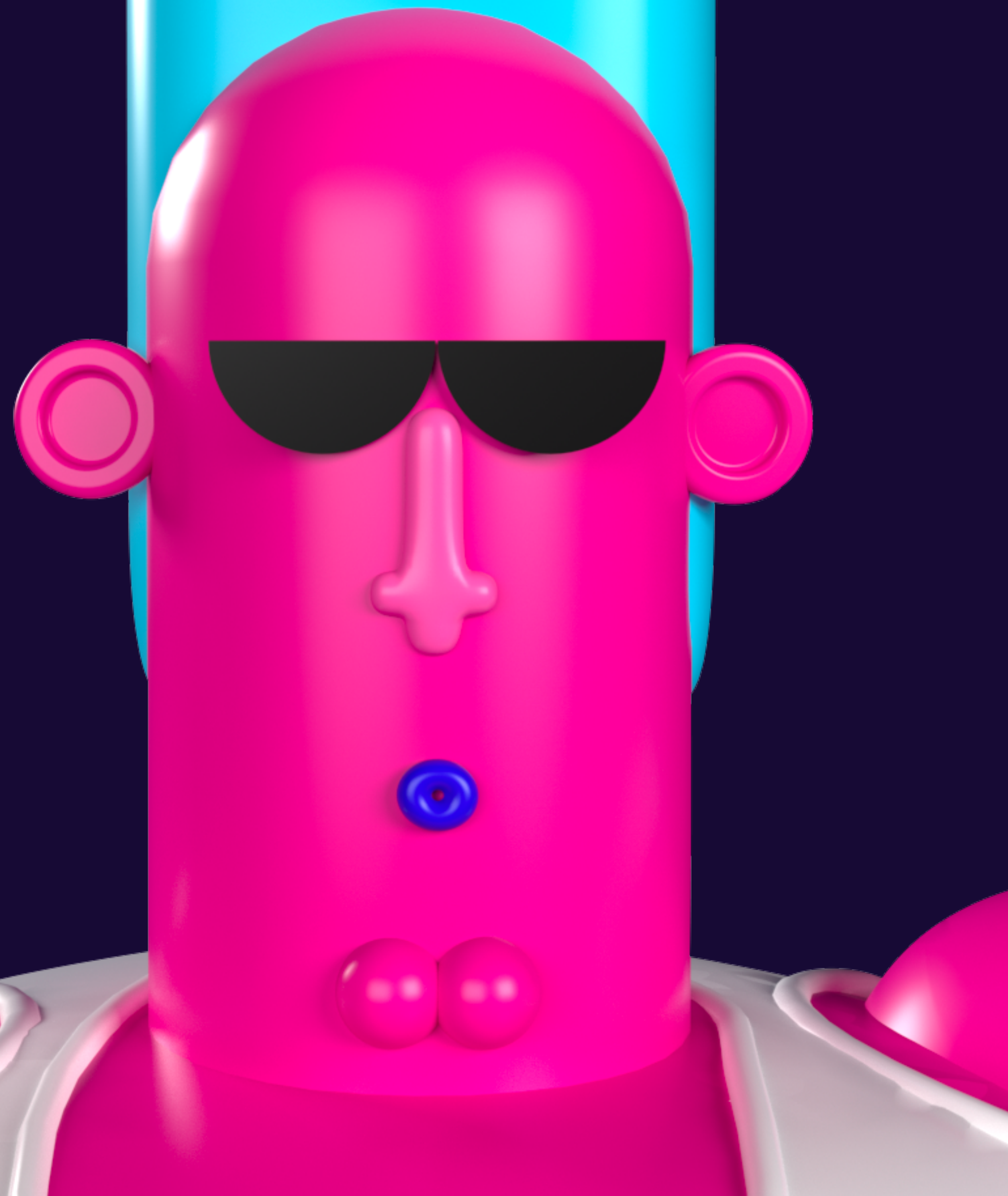
Depuis la version 3 de Phaser, le framework dispose d'un fichier de définition Typescript et nous permet donc d'utiliser ce préprocesseur à typage statique dans l'intégralité de notre projet.

Ce choix technique est assez important puisqu'il conditionne l'écriture de tout notre code source, il est aussi très avantageux: Typescript optimise notre organisation, la qualité de notre code, il permet une syntaxe plus explicite comme par exemple l'exposition de champs de classes avec les mots clés public / protected / private.

Typescript permet aussi de limiter les erreurs potentielles du projet en vérifiant la cohérence de notre typage avant même l'exécution du code.

Pour la mise en réseau de notre jeu, nous nous orientons vers les WebSockets en utilisant la librairie socket.io, facilitant la gestion de WebSockets, notamment en proposant directement un système de rooms indispensable pour ce projet.

Le build côté client est assuré par les scripts de Vue (exploitant notamment en arrière-plan Webpack, Babel, Typescript). La librairie Vue elle-même nous permet de bénéficier de tous les avantages des données réactives, d'un router et d'un store, pour construire notre interface orientée composant.



SPECIFICATIONS FONCTIONNELLES

TECHNIQUES



ARCHITECTURE

Notre projet s'appuie sur un monorepo. Ce type d'architecture est pertinent lorsqu'il est nécessaire de partager de mêmes logiques à travers différents paquets interdépendants.

Dans notre cas, il s'agit de partager la logique du jeu lui-même entre le client et le serveur. Nous évitons ainsi beaucoup de duplications, tout en dissociant proprement deux applications (client et serveur) différentes par leur nature mais aussi par leur environnement d'exécution (le client est destiné à un moteur de navigateur tandis que le serveur s'exécute avec node).

Nous utilisons Lerna pour organiser notre monorepo. Lerna facilite la gestion de ce type d'architecture en exploitant le système de dépendances et les workspaces de yarn ou npm.

Le fonctionnement de notre application donne autorité au serveur pour tout le déroulement du jeu et les calculs physiques. Avec cette approche, nous n'avons pas à arbitrer d'éventuelles informations contradictoires des différents clients.

La physique du jeu étant relativement simple (déplacement et gestion de projectiles au tour par tour), elle ne devrait pas être une grande source de stress pour le serveur, d'autant plus que nous pourrions limiter la fréquence de calcul en ajustant le tick.

SPECIFICATIONS FONCTIONNELLES

TECHNIQUES

WORKFLOW

Nous avons opté pour l'outil Github Project permettant une gestion de projet intégrée à Github, et qui propose un board type kanban (TODO, DOING, DONE) synchronisé avec nos issues, pull requests et commits.

Nous bénéficions ainsi de tous les avantages d'un workflow avec git: nous travaillons par features organisées en branches (ex: feat/base-socket.io) que nous pouvons ensuite soumettre avec des Pull Requests qui permettent une revue de code avant fusion avec la branche master.

Ce flux permet une grande transparence (le projet étant open-source) en exposant nos réflexions et échanges entre développeurs lors des revues de code. De cette façon nous

sommes toujours au fait des évolutions de notre code et pouvons facilement critiquer et éditer ses modifications. Ce workflow apporte aussi un bénéfice en ce qu'il promeut l'échange de connaissances entre développeurs.

Ce flux de travail s'étend jusqu'aux processus de déploiement: La branche master est la source de code pour les applications en production. Il s'agit donc d'une organisation en développement continu où toute modification ou fusion de la branche master déclenche automatiquement une mise en production chez Heroku pour le serveur, et chez Netlify pour le client.

Le code déployé est alors compilé et les services sont relancés sans nécessiter aucune



SPECIFICATIONS FONCTIONNELLES

TECHNIQUES

action de notre part. En cas d'erreurs lors de l'exécution des processus de déploiement, nous sommes automatiquement notifiés par email, mais normalement ce genre de situation ne peut pas arriver car la mise en production est automatiquement testée avant déploiement.

En effet, à la création d'une Pull Request, une instance de staging est créée à la volée, nous permettant de vérifier le bon fonctionnement de nos modifications.

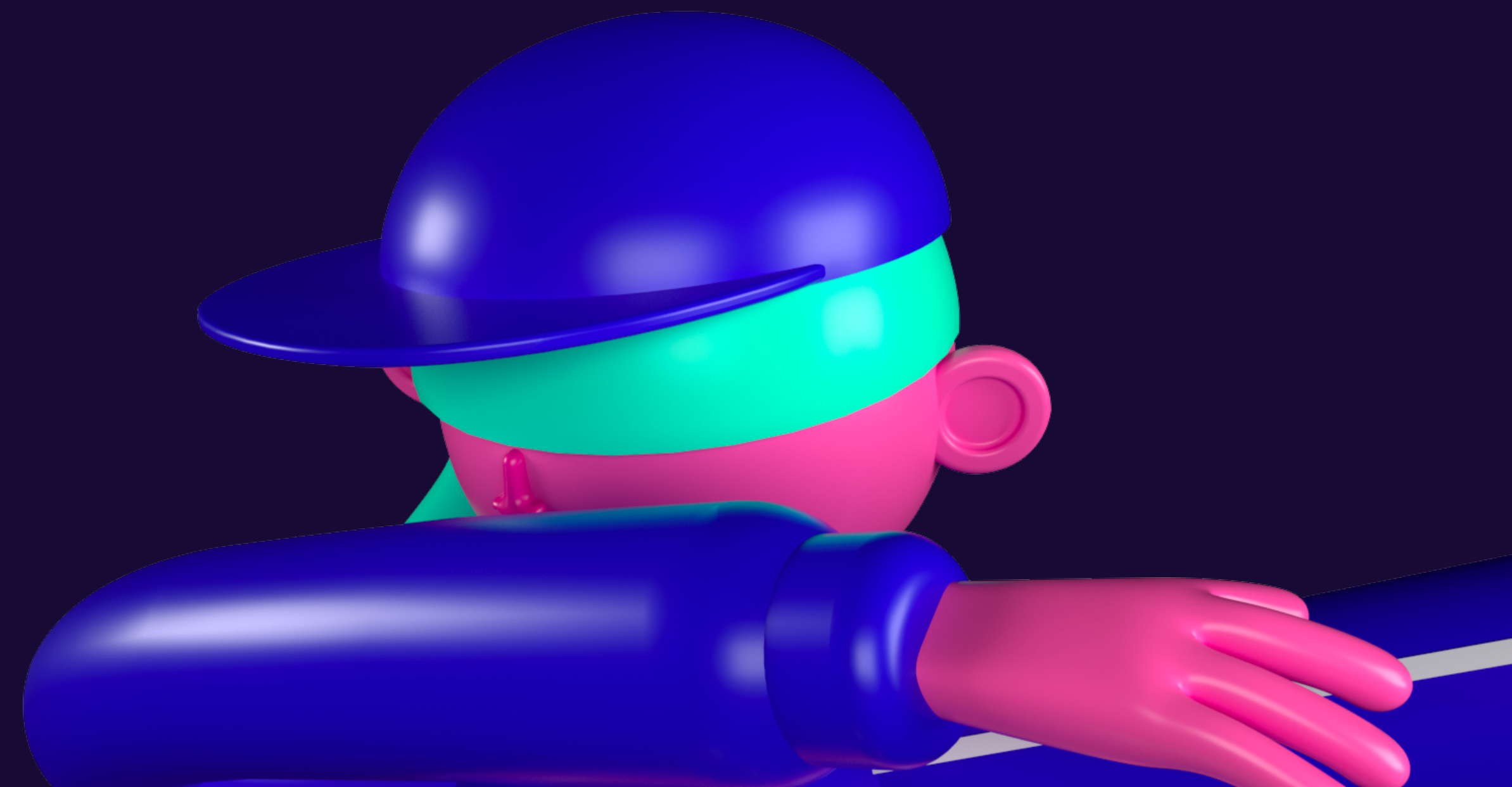
Enfin, nous avons aussi décidé d'utiliser un channel slack réservé au développement dans lequel nous avons intégré le bot Github for Slack. Cela nous permet de suivre, d'un seul endroit, toute les actions liées au développement du projet.

Nous échangeons principalement via Messenger pour la communication en temps réel entre designers et développeurs. Nous avons aussi réalisé un planning sur Teamweek afin de coordonner les rendus à date butoire. Cela nous permet d'avoir une vision globale sur l'avancement du projet, mais aussi de prioriser certaines tâches.

Comme nous réalisons un jeu en 2 dimensions, il a fallu définir au plus tôt des normes de nommage et d'export de fichiers. Nos assets (sprites, atlases...) doivent être cohérents et partager une échelle commune. Il faut aussi qu'ils soient optimisés au maximum. Une partie du workflow lié aux assets est confiée à des scripts dédiés avec webpack et un texture packer qui assurent automatiquement l'optimisation et la génération de sprites.

En confiant ce type de tâches rébarbatives à des processus automatisés, nous absorbons les éventuelles erreurs humaines.

Le partage de fichiers entre designers et développeurs se fait à l'aide de Google Drive avec un drive d'équipe.



PLANNING

RÉPARTITION DES TÂCHES

	1ère semaine du 15 avril	Weekend 1	2ème semaine du 22 avril	Weekend 2	3ème semaine du 29 avril	Weekend 3	4ème semaine du 6 mai	5ème semaine du 13 mai	6ème semaine du 20 mai	7ème semaine du 27 mai	8ème semaine du 3 juin	9ème semaine 10 juin	Jury 12 juin
<i>Liste générale</i>	<p>Décors :</p> <ul style="list-style-type: none"> - Illustrations du 1er plan par biom - Unifier les couleurs et second plan <p>Interface :</p> <ul style="list-style-type: none"> - Logo - Typo - Couleurs (nuancier) 	<p>Checking</p> <p>Décors plus unification en vu d'appliquer les couleurs dominantes par biom plus les lumières et les ombres</p>	<p>Décors 2 :</p> <ul style="list-style-type: none"> - Lumières et ombres - Post prod - Sortie des assets <p>Interface :</p> <ul style="list-style-type: none"> - Logo - Typo 	<p>Checking</p> <p>Propositions de logo en vu d'une micro anim logo plus check des assets sorties pour les dev si tout est ok (avec le paralaxe)</p>	<p>Character design :</p> <ul style="list-style-type: none"> - Personnages 2d simplifiés - Préparation pour les 3D <p>Personnages 3D :</p> <ul style="list-style-type: none"> - Modélisation - Lumières et textures 	<p>Checking</p> <p>Tout les personnages 2D en vu des animations plus les (kit) pour les 3D</p>	<p>Animations :</p> <ul style="list-style-type: none"> - Déplacements - Armes / explosions - Fixe - Réception des coups 	<p>Animations 2 :</p> <ul style="list-style-type: none"> - Chutes - Sauts - Atterrissage / réception - Mort 	<p>Interface :</p> <ul style="list-style-type: none"> - Fond start - Splash screen - Boutons - Configue partie - Cartes personnages - Ecran pause - Victoire / Défaite - Anim tour du joueur - Stick directionnel - Envoi de l'arme (flèche directionnelle) 	<p>Sound Design :</p> <ul style="list-style-type: none"> - Ambiance - Effets spéciaux - Voix off (supermarché) - Onomatopées 	<p>Présentation :</p> <p>Keynote de présentation Ecrans de la présentation Affiche / Com Trailer</p>	<p>Présentation :</p> <p>Préparation du jury Oral entrainement</p>	
Bastien	<p>Divertissement et caisses</p> <p>Réfléchir au logo, couleurs, typo etc.</p>				<p>Modélisation 3D de tous les personnages</p>								
Fany	<p>Décors Jouets et plantes</p> <p>Réfléchir au logo, couleurs, typo etc.</p>				<p>Persos Babos, Bimbo et Geek</p> <p>(Version 2D simple et 2D "détaillée")</p>								
Iris	<p>Poissonerie et fruits et légumes</p> <p>Réfléchir au logo, couleurs, typo etc.</p>				<p>Persos Mamie gateau, influenceuse et Papi raciste</p> <p>(Version 2D simple et 2D "détaillée")</p>								