

M
P
I
C
Y
Z

DOSSIER DE SYNTHESE

Concepteur développeur web

Sommaire

DOSSIER DE SYNTHESE	2
I. Présentation du projet	5
II. Etude concurrentielle.....	6
1. Première étude	6
2. Deuxième étude.....	7
3. Conclusion Étude concurrentielle	8
II. Règles adopté de l'Ux et Ui	9
III. Recherche et développement	11
1. Les outils utilisés pour la recherche et le développement ont été :	11
2. Liste des technologies utilisées :	11
3. Tools de design :	14
4. Environnement de travail	15
5. Outil de stockage des données	15
4. Choix des plateformes de partage.....	16
5. Management avec JiraSoftware.....	17
IV. Charte graphique	20
V. Maquette Adobe xD.....	22
Type mobile First	22
Type desktop	23
VI. Persona	24
Persona Primaire.....	24
Persona Secondaire.....	25
Ux des utilisateurs.....	26
VII. Zoning des fonctionnalités	27
Fonctionnalité du site.....	27
Zoning fonctionnel de l'upload.....	28
VIII. Méthodologie de d'upload des vidéos	29

1.	Méthode de poste avec Twitch	29
2.	Méthode de poste avec Twitter	34
3.	Méthode de poste avec YouTube.....	36
IX.	Prototypage Bureau et Mobile final	38
X.	Présentation base de données MongoDB	39
XI.	Modules.....	41
XII.	Accounts management	43
	Mise en place de la connexion via Account-Google.....	43
	Récupérations des données utilisateurs avec la connexion Gmail.....	43
XIII.	Sécurisation	44
1.	Utilisation de méthodes.....	44
2.	Les publications	44
3.	Fichier servi.....	44
4.	List de contrôle de sécurité.....	45
XIV.	Contraintes techniques	46
XVI.	CSS	47
1.	Variable.....	47
2.	Normalisation Naming	48
3.	Structure CSS	48
XV.	Projection du site	49
1.	Description des principales fonctionnalités dans l'évolution.....	49
2.	Modération d'accès	50
XVI.	Conclusion	51
XVII.	Remerciements.....	52
XVIII.	Annexe.....	53

I. Présentation du projet

Mon projet de fin d'étude est une future application côté client. Elle est développée en Javascript avec la librairie React et le framework Meteor est utilisé Node.js et MongoDB comme gestionnaire de données. Cette plateforme permet à l'utilisateur de pouvoir publier des vidéos sur le thème des jeux vidéos vers des plateformes spécifiques telles que Twitch, Youtube, Twitter simultanément. Un espace personnel est développé autour de l'utilisateur. Pour débuter ce projet, nous utiliserons la plus simple des plateformes et la plus représentatives : Twitch

La fonctionnalité principale est de pouvoir poster une vidéo sur plusieurs réseaux aux types et format adapté à celui-ci. Un des facteurs principaux du projet est la persistance de donnée avec un hébergement indirect.

Ce projet est un tremplin vers des successions d'activités qui nous permettront avec les accords des différents distributeurs comme Facebook, Instagram, Snapchat pour créer des Clips de publications avec des outils d'éditions.

Les contraintes techniques pour ces plateformes sont juridique.

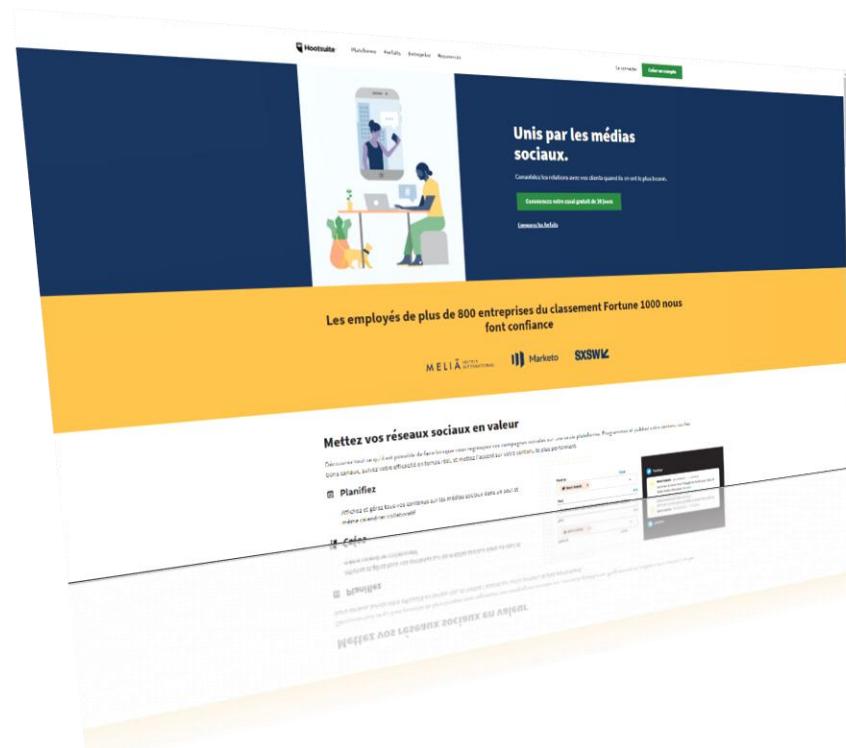
Puis avec le temps nous agrémenterons de plateforme et de fonctionnalité. Avec l'ambition de créer un outil de management de médias pour le monde professionnel et public du jeu vidéo ou des activités médiatiques qui permettre aux différents acteurs de pouvoir organisés sont plannings de publications mais aussi de les différer.

II. Etude concurrentielle

1. Première étude

Le site web ciblé : <https://hootsuite.com/fr/>, ce site permet de prévoir les postes et l'organisation de partage de médias sur les réseaux sociaux. Son utilisation principale reste la gestion de partage de vidéo et de management de son planning de partage. Cette étude concurrentielle sur le site hootsuite a mis en évidence le besoin du partage des média simultanément sur différentes plateformes avec une grande simplicité.

Etude concurrentielle		
POINTS SUR LE SITE HOOTSUITE		
	Positif	Négatif
Une page d'accueil claire, simple et précise	✓	✗
Des textes lisibles	✓	✗
Un affichage rapide	✓	✗
Un plan de page lisible	✓	✗
Des menus accessibles et intuitifs	✓	✗
Une hiérarchie visuelle des éléments	✓	✗
Un accès en 3 clics	✓	✗
Un accès optimisé sur les différents supports	✓	✗



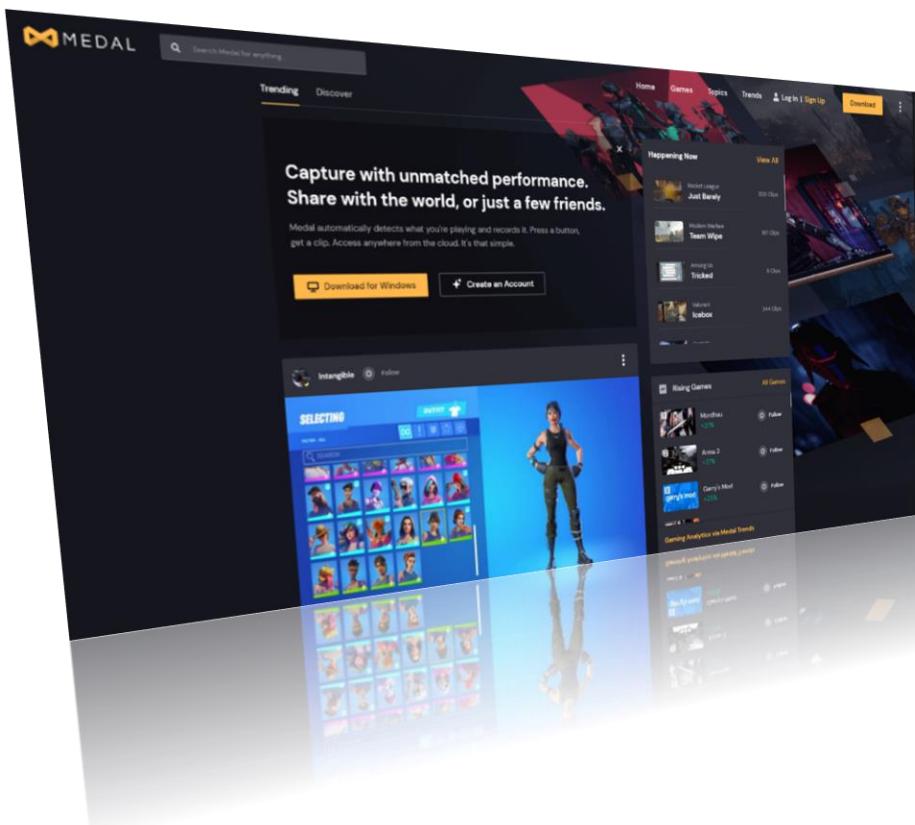
2. Deuxième étude

Le site web ciblé : <https://medal.tv/> est le meilleur moyen d'enregistrer, de regarder et de partager des clips de jeu avec le plus faible impact FPS possible. Sur ce site, vous pouvez synchroniser les clips sur un cloud pour libérer de l'espace personnel sur votre ordinateur et créer votre profil. Il est possible de visualiser les clips sur le site, mais nous devons installer un logiciel pour la publication.

Etude concurrentielle

A la suite de cette démarche nous dégagerons les points négatifs et positifs qui nous permettront de faire une évaluation du site dans sa globalité.

POINTS SUR LE SITE MEDAL.TV	Positif	Négatif
Une page d'accueil claire, simple et précise	✓	
Des textes lisibles	✓	
Un affichage rapide	✓	
Un plan de page lisible	✓	
Des menus accessibles et intuitifs	✓	
Une hiérarchie visuelle des éléments	✓	
Un accès en 3 clics	✓	
Un accès optimisé sur les différents supports	✓	



3. Conclusion Étude concurrentielle

LES POINTS A RETENIR SUR L'ÉTUDE

LE SITE MEDAL

- Le site est très ergonomique
- Tres bien organisé
- Vitesse de chargement rapide
- Il reste en partie un site vitrine
- Dépendent de télécharger le logiciel pour publier des vidéos

LE SITE HOOTSUITE

- Manque de pertinence sur la première navigation
- Plan de page abstrait
- Mise en valeur des services proposés en pieds de page
- Utilisation des outils directement sur le site

Les deux sites traitent des sujets bien précis et se complètent. J'ai souligné plusieurs points intéressants sur l'organisation du site MEDAL.TV en gardant les principes de Hootsuite qui est de tout avoir disponible directement sur l'application Web. Je tiens aussi à souligner la simplicité de navigation des deux sites que je garde comme ligne de conduite dans les domaines fonctionnels et visuels.

II. Règles adoptées de l'UX et UI

Une page d'accueil claire, simple et précise :

C'est la page qui aura bien souvent le plus fort trafic. Il est donc essentiel que les visiteurs aient tout de suite envie d'en savoir plus.

Contrairement à ce que l'on pourrait penser, ce n'est pas la beauté du design, qui poussera les visiteurs à l'action mais, la simplicité de trouver une information.

Le premier objectif est de leur simplifier la vie, toutes les pages de l'application devront également respecter cette règle.

Des textes lisibles :

Lire un texte sur un ordinateur fatigue rapidement les yeux alors il convient d'éviter les colonnes de texte trop larges ou trop fines, les polices illisibles ou trop exotiques, les textes sur fond gris ... Par mon vécu, j'ai pris la décision d'utiliser une typographie spécialisée pour les dyslexiques ou les personnes avec des troubles de la lecture. Je souhaite apporter des facilités et intégrer des outils de lecture.

Voici les points à privilégier :

- ❖ Texte foncé sur fond claire ou inverse
- ❖ Une taille de police standard (pas de police microscopique ou macroscopique)
- ❖ Une police spécifique
- ❖ Le soulignage des liens exclusivement
- ❖ Des plans de page découpés en paragraphes avec titres et sous-titres
- ❖ Chaque page doit avoir un titre et être aérée par une mise en page agréable.
- ❖ Utilisation de sous-titres, de paragraphes courts et concis.

Un affichage rapide :

Les internautes sont des gens pressés en règle générale. Je ne souhaite pas que l'internaute ressorte aussi vite qu'il est entré, j'optimiserais le poids des pages.

Des menus accessibles et intuitifs :

Les visiteurs ne devront pas passer plus de 10 secondes à tenter de comprendre comment fonctionne le site. Les menus doivent être clairs et accessibles au premier coup d'œil.

Une hiérarchie visuelle des éléments :

Afin de capter l'œil de nos visiteurs, il faudra jouer sur les contrastes des zones les plus importantes du site. Mais attention de ne pas en abuser au risque de leur donner le tournis. Le test à suivre : flouter légèrement une page avec Photoshop et constater où se porte le regard en priorité.

Un accès en 3 clics :

L'utilisateur pourra en trois clics avoir accès à tout le contenu du site depuis la page principale.

Un accès optimisé sur les différents supports :

À l'heure où plus de la moitié des navigations s'effectuent sur un appareil mobile, votre site doit offrir une expérience optimale sur chacun de ces supports. Pour se faire, le responsive design permettra de faire varier automatiquement le mode d'affichage des pages en fonction de la taille de l'écran.

III. Recherche et développement

1. Les outils utilisés pour la recherche et le développement ont été :

- ❖ Une maquette Adobe xD
- ❖ Personna (<https://uxpressia.com/>)
- ❖ Ux Maps (<https://uxpressia.com/>)
- ❖ JiraSoftware management

2. Liste des technologies utilisées :

Les logiciels de développement :



Microsoft Visual Studio code est une suite de logiciels de développement pour Windows et mac OS conçue par Microsoft. Elle facilite le développement d'applications pour toutes les plates-formes et tous les langages. La dernière version s'appelle Visual Studio code 2019.

Les langages utilisés :



HTML5 dernière révision majeure du HTML finalisée le 28 octobre 2014.



CSS les feuilles de style en cascade, généralement appelées CSS de l'anglais Cascading Style Sheets, forment un langage informatique qui décrit la présentation des documents HTML et XML. Les standards définissant CSS sont publiés par le World Wide Web Consortium (W3C). Introduit au milieu des années 1990, CSS devient couramment utilisé dans la conception de sites web. Dans les années 2000 CSS a été pris en charge par les navigateurs Web.



Javascript un langage de programmation de scripts principalement employé dans les pages web interactives, mais aussi pour les serveurs avec l'utilisation de Node.js

Framework et librairie :



React.js souvent appelé React ou ReactJS est une bibliothèque JavaScript responsable de la construction d'une hiérarchie de composants de l'interface utilisateur ou, en d'autres termes, responsable de la génération des composants de l'interface utilisateur. Il fournit un support pour front-end et côté serveur. Rappelez-vous que React.js est le V du pattern MVC, c'est donc une bibliothèque uniquement destinée à générer vos vues.



Meteor permet de développer avec le même langage (en Javascript ou dans un langage compilant vers Javascript) Ce choix d'architecture permet de déplacer facilement un traitement du serveur vers le client (et réciproquement) voire de le dupliquer par exemple dans le cas de la validation d'un formulaire.

Dans cette logique, Meteor inclut un système de gestion de base de données côté client, fonctionnalité originale du framework. Il est ainsi possible d'effectuer des requêtes même en étant déconnecté du serveur. Cela permet notamment à Meteor d'inclure par défaut, des mécanismes de *compensation de latence*. Par exemple, l'envoi d'un message dans un chat sera

instantanément ajouté au fil des messages au clic sur le bouton "Envoyer", tandis que la vérification du message se fera en arrière-plan côté serveur. Ce mécanisme permet l'utilisation de la programmation réactive côté client.



Node.js Meteor s'appuie sur Node.js qui permet d'exécuter du code JavaScript sur le serveur. Cela signifie que nous allons utiliser le même langage de programmation sur le client et sur le serveur, un confort indéniable pour le programmeur. Meteor pousse même l'idée un peu plus loin en unifiant les API utilisées sur le client et sur le serveur, on parle de JavaScript isomorphique, un terme un peu pédant pour désigner ce concept qui vous semblera bientôt naturel. Prenez par exemple la méthode Meteor.startup qui permet d'exécuter du code au démarrage du client ou du serveur. Du côté serveur la fonction est exécutée dès que le processus Node.js est lancé, et du côté client elle l'est dès que le DOM est prêt à être modifié. Il s'agit donc d'une méthode disponible sur les deux environnements pour le même usage, exécuter du code au démarrage, mais implémentée de manière différente sur le client et sur le serveur.



Mongo DB c'est dans cet esprit que Meteor propose également une base de données complète accessible du côté client. À l'usage, vous verrez qu'il est très commode de pouvoir faire des requêtes directement dans le navigateur avec la même API que sur le serveur. Il s'agit de l'API de MongoDB qui est la base de données utilisée par défaut avec Meteor. Un développeur utilisera donc les mêmes méthodes sur deux environnements différents pour exprimer la même idée (« stocker des données ») bien que l'implémentation de ces méthodes diffère largement en fonction des contraintes imposées à l'environnement.

Avantage de React, Meteor, Node.js

DOM (document Object model) est un compromis des vues sur les entrées et sorties de données. Le DOM virtuel de React est plus rapide que le modèle de rafraîchissement complet conventionnel, puisque le DOM virtuel ne rafraîchit

que certaines parties de la page. Ce qui est intéressant, c'est que l'équipe de Facebook n'était pas consciente qu'une actualisation partielle d'une page se révélerait plus rapide. Facebook cherchait juste un moyen de réduire leur temps de reconstruction et le rafraîchissement partiel du DOM était juste une bonne solution. En finalité, cela augmente les performances et accélère la programmation. Vous pouvez réutiliser des composants de code dans React JS, ce qui vous fait gagner beaucoup de temps. La génération complète de vos pages, du serveur au navigateur, améliorera le référencement de votre application web.

Il améliore la vitesse de débogage facilitant ainsi la vie de votre développeur.

Même pour ceux qui ne sont pas familiers avec React, il est facilement lisible. De nombreux Framework exigent que vous appreniez une longue liste de concepts qui ne sont utiles que dans le Framework. React s'efforce de faire le contraire.

Vous profitez de toutes les avancées du langage Java et de son écosystème.

3. Tools de design :



PhotoShop est un logiciel de retouche, de traitement et de dessin assisté par ordinateur. Édité par Adobe, il est principalement utilisé pour le traitement des photographies numériques, mais sert également à la création. J'ai pu donc faire mûrir les idées avec cet outil et j'ai pu commencer à intégrer des items graphiques.



Adobe xD est un outil vectoriel développé et publié par Adobe Inc. pour la conception et le prototypage de l'expérience utilisateur pour les applications Web et mobiles. Il m'a permis de faire un prototypage proche de la réalité et de tester la navigation.



Draw.io pro est une application de création de diagrammes compatibles avec Google que j'ai utilisé afin de mettre en place le zonning des fonctionnalités.

4. Environnement de travail :

- ❖ Mon ordinateur personnel contient :
 - **MSI Z170A KRAIT GAMING 3X**
 - **Intel Core i5-6600K (3.5 GHz)**
 - **MSI GeForce GTX 980 Ti GAMING 6 Go**
 - **DDR4 HyperX Savage Black, 2 x 8 Go, 3000 MHz**
 - **SSD Samsung Série 850 EVO, 250 Go**
 - **BenQ 24" LED - XL2411Z 144Hz**
 - **Corsair RGB k70 lux**
 - **BenQ ZOWIE FK1+ ⇔ Dpi 400**
 - **HeadPhone HyperX cloud I**

- ❖ Responsivité avec le simulateur navigateur Google mode Smartphone 5S.
- ❖ Mon téléphone personnel S8+.
- ❖ Mise en ligne avec Réseau Orange fibre (800mb/s débit descendant 300mb/s débit montant).

5. Outil de stockage des données :

Base de données MongoDB dont les appels sont faits et gérés par Meteor.

Héberger chez **Scalingo** qui propose plusieurs services comme :

- Déploiement de code instantané.
- Tous les langages sont supportés pour les projets web.
- Bases de données à la demande. Provisionnez MongoDB en un clic.

6. Choix des plateformes de partage



Twitch sera mon choix par défaut, l'application utilise un build React et une API d'upload est disponible sur le site de développement avec des requêtes URL de type JSON. Il est donc possible de pouvoir une fois connecté (OAuth) et pouvoir créer une vidéo, de la modifier ou d'en supprimer une.



Twitter est aussi une plateforme build sous React. Il est possible aussi d'upload des médias via une API avec le même format de requête URL de type JSON.



YouTube est une plateforme qui utilise un procédé de scripte Python2 ou Python3 avec des requêtes URL de format JSON.

7. Management

Jira Software

Jira Software est un outil de gestion de projet Agile qui prend en charge toute méthodologie Agile, qu'il s'agisse de Scrum, Kanban. Grâce à un seul outil, nous pouvons : planifier, suivre, gérer nos projets de développement et tableaux aux rapports.

Les réunions de planification de sprints définissent les tâches d'une équipe qu'elle doit accomplir dans le sprint à venir, qui sont consignées dans le backlog, où répertorient toutes les tâches à réaliser. Jira Software place notre backlog au centre des réunions de planification de sprint.

Gestion des versions

Nous pouvons suivre les versions, les fonctionnalités et la progression en un coup d'œil. Cliquer sur une version pour consulter l'état complet, notamment les tickets, les données de développement et les éventuels problèmes.

Préparation du backlog en toute simplicité

On redéfinit la priorité des user stories et bugs en toute facilité. En un ou plusieurs tickets, puis glissez les dans le backlog pour les réorganiser. Mais aussi créer des filtres rapides pour remonter les tickets affichant des attributs importants.

Planification du sprint

Il est possible de placer le backlog au centre des réunions de planification de sprint. Avec le reste d'une équipe, faire des estimations des stories, ajuster le périmètre du sprint, vérifier la vitesse et redéfinisse la priorité des tickets en temps réel.

Story points

Effectuer des estimations et le suivi, générer des rapports sur les story points afin qu'une équipe puisse gagner en précision dans les sprints futurs. Utiliser des story points, des heures optimales ou leurs propres méthodes d'estimation.

Voici le contexte global de mon utilisation au sein du développement de mon projet.

The screenshot shows a Jira Kanban board for the project "PFE-2020". The board is organized into four columns: BACKLOG, SÉLECTIONNÉ POUR LE DÉVELOPPEMENT, EN COURS, and TERMINÉ. Each column contains several items, each with a title, a progress bar, and a unique ID (P2-1 to P2-18). The sidebar on the left provides navigation for the project, and the top bar shows various system and application icons.

Column	Item Title	ID
BACKLOG	Présentation du projet	P2-1
	Définir le projet dans le PFE final	P2-21
	Persona mise en page.	P2-8
	Présentation base de données	P2-10
	Sécurisation de l'application	P2-15
	Sécurisation des données personnelles	P2-16
	Sécurisation et droit de connexion dans l'environnement.	P2-17
	CSS	P2-18
	Création de toutes les animations JS	
SÉLECTIONNÉ POUR LE DÉVELOPPEMENT	Spécifications techniques	P2-5
	Zoning des fonctionnalités	P2-9
	Description des principales fonctionnalités du site.	P2-13
	Contraintes techniques	P2-14
EN COURS	P2-3 Recherche et développement	
	Création UX maps avec les personnes	P2-28
	Charte graphique avec Logo	P2-6
	P2-7 Maquette Adobe XD	
TERMINÉ	Recherche et développement	P2-3
	Maquette Adobe XD	P2-7
	Développement type mobile	P2-12



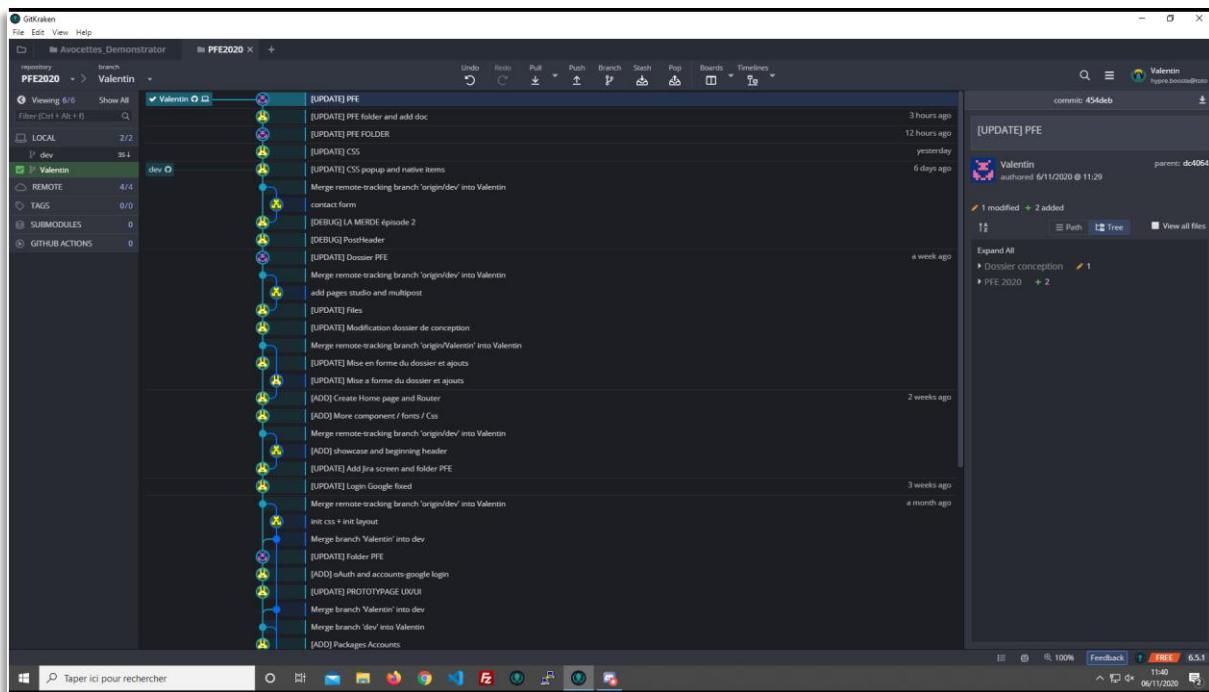
Git

Est un logiciel de gestion de versions (Version Control System) qui suit l'évolution des fichiers sources et garde les anciennes versions de chacun d'eux sans rien écraser. ... Il fournit une interface visuelle pour gérer localement des projets avec les contrôles de version.



Gitkraken

J'ai utilisé un client graphique pour Git.



IV. Charte graphique

Polices / Fonts

La police intégrée dans le projet est de Google font « Monospace »

Tailles natives

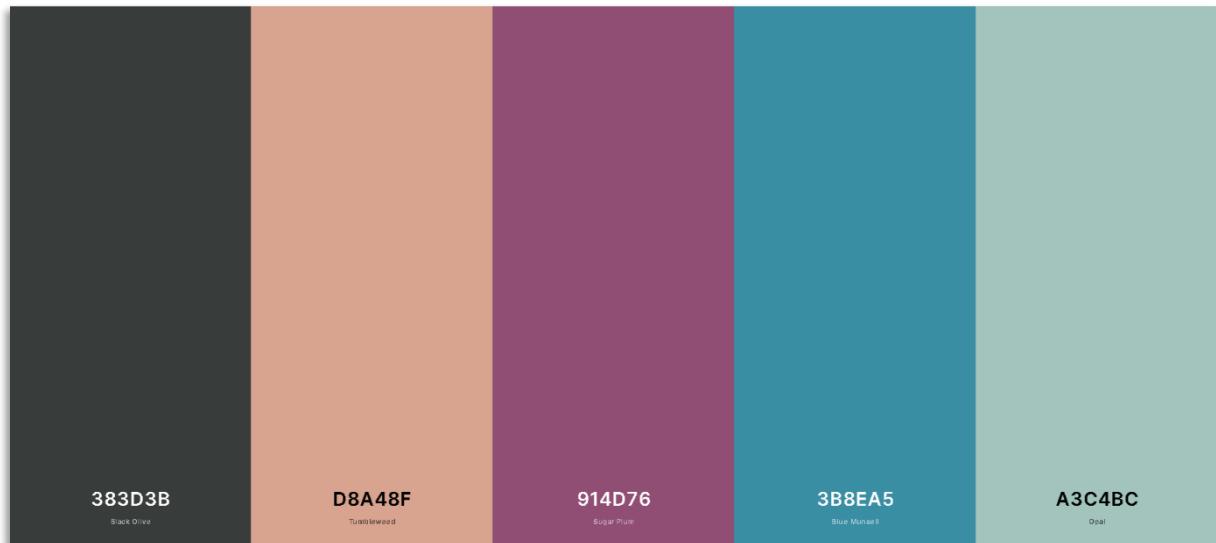
Les tailles natives du projet sont sur des chiffres ronds.

Dans le cas des références en REM elles seront présentées de 0.5 en 0.5

Couleurs

coolors

Les couleurs choisies sur coolors.co avec un algorithme pour le choix et les contrastes.



Logos

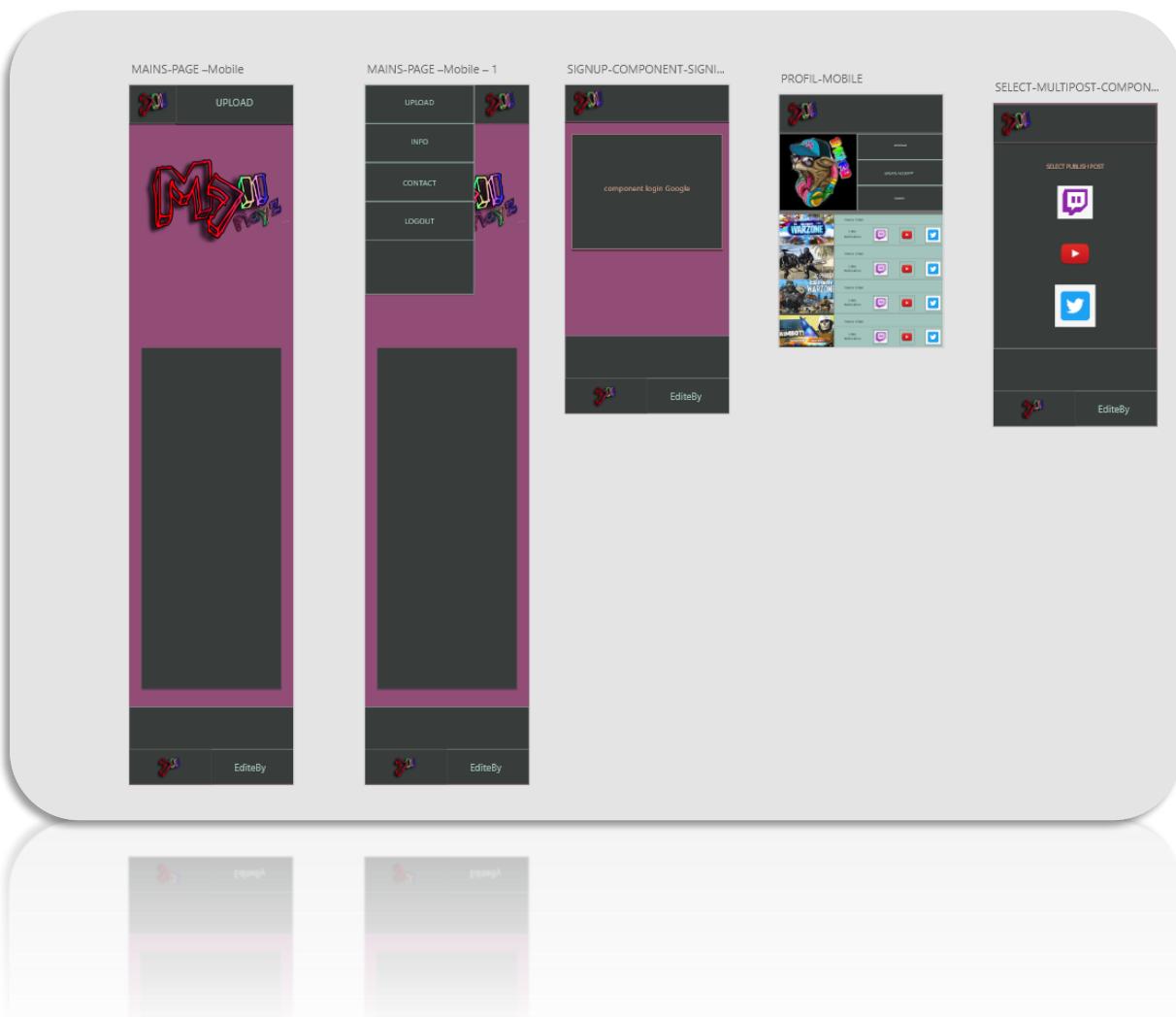


V. Maquette Adobe xD

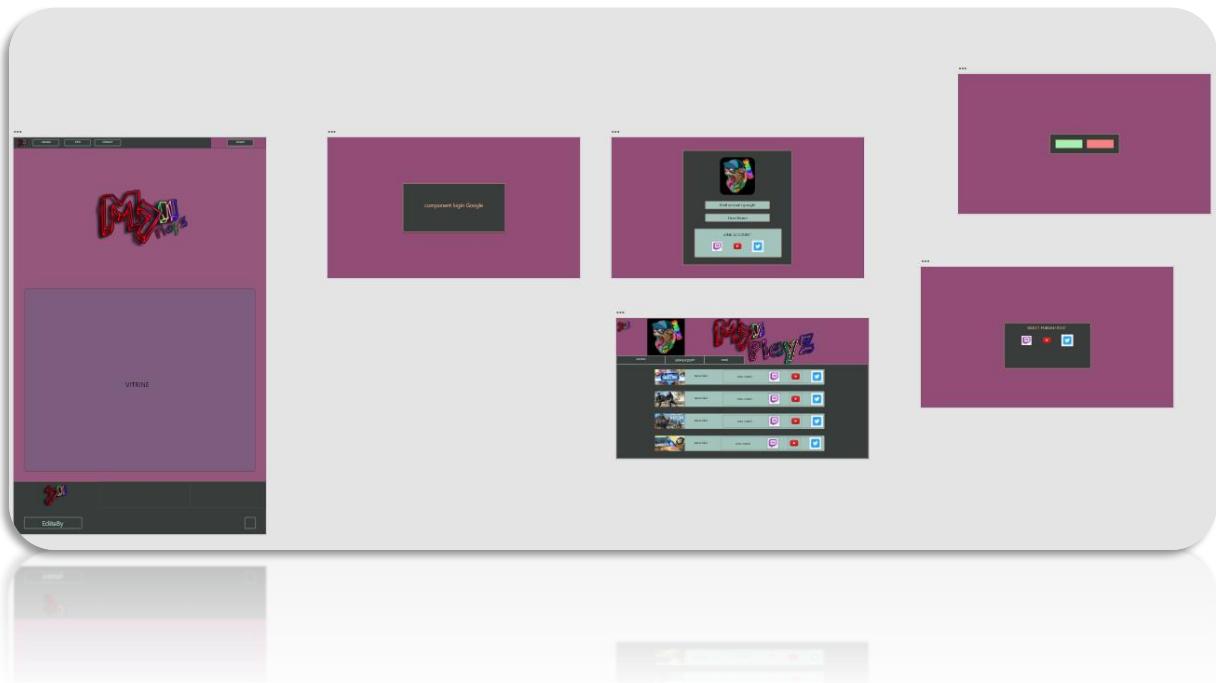
Xd

Pour l'étude UX UI, nous allons nous appuyer sur une première vision du site avec une ébauche de maquette.

Type mobile First :



Type desktop :



Nous allons mettre en place les persona et ainsi les faire naviguer sur cette maquette afin de produire un User expérience.

VI. Persona

Persona Primaire

PROJECT: Persona Gull DESPTER PERSONA: Harry DOPTER

NAME
Harry DOPTER

Il recherche à partager ses moments de jeux vidéo.

Qui est harry ?	Expérience de jeux	Relationnel avec le digital
Canada	<ul style="list-style-type: none">Il est droitier / clavier AZERTYPossède un ordinateur portable de type Gamer.Il joue depuis 5 ans aux FPS.	<ul style="list-style-type: none">Parcours souvent le web depuis son smartphone.Il aime parcourir les événements à venir autour des FPS.Il a envie d'échanges, et de partages de connaissances autour du E-sport.
Préférence de jeux	Jeux FPS	Outil Digital
Sans salariat	0 25 50 75 100	
Colocataire	0 25 50 75 100	
Avec véhicule	0 25 50 75 100	
Passionné de jeux vidéo	0 25 50 75 100	
Etudiant	Jeux MMO RPG	
Célibataire	Jeux stratégie	
21 ans	0 25 50 75 100	
Passionné de jeux vidéo	0 25 50 75 100	

Il fait des études dans le développement et la conception de série type Netflix.

Persona Secondaire

PROJECT: Persona Gull DESPTER PERSONA: Gull DESPTER

NAME
Gull DESPTER



Qui est gull ?

L'union
51 ans
Marié
Ingenieur
Salarié
Propriétaire
Avec véhicule
Passionné de jeux vidéo

Il recherche à partager ses moment de jeux MMO-RPG

Expérience de jeux

- Il est droitier / clavier AZERTY
- Possède un ordinateur de type Gamer.
- Il joue depuis 20 ans aux MMO-RPG.
- Il a un bureau dédié à son monde.

Préférence de jeux

Jeux FPS

0 25 50 75 100

Jeux MMO RPG

0 25 50 75 100

Jeux stratégie

0 25 50 75 100

Relationnel avec le digital

- Parcours souvent le web depuis son smartphone.
- Il aime parcourir les événements à venir autour des MMO-RPG.
- Il a envie d'échanges, et de partages de connaissances autour du MMO-RPG.

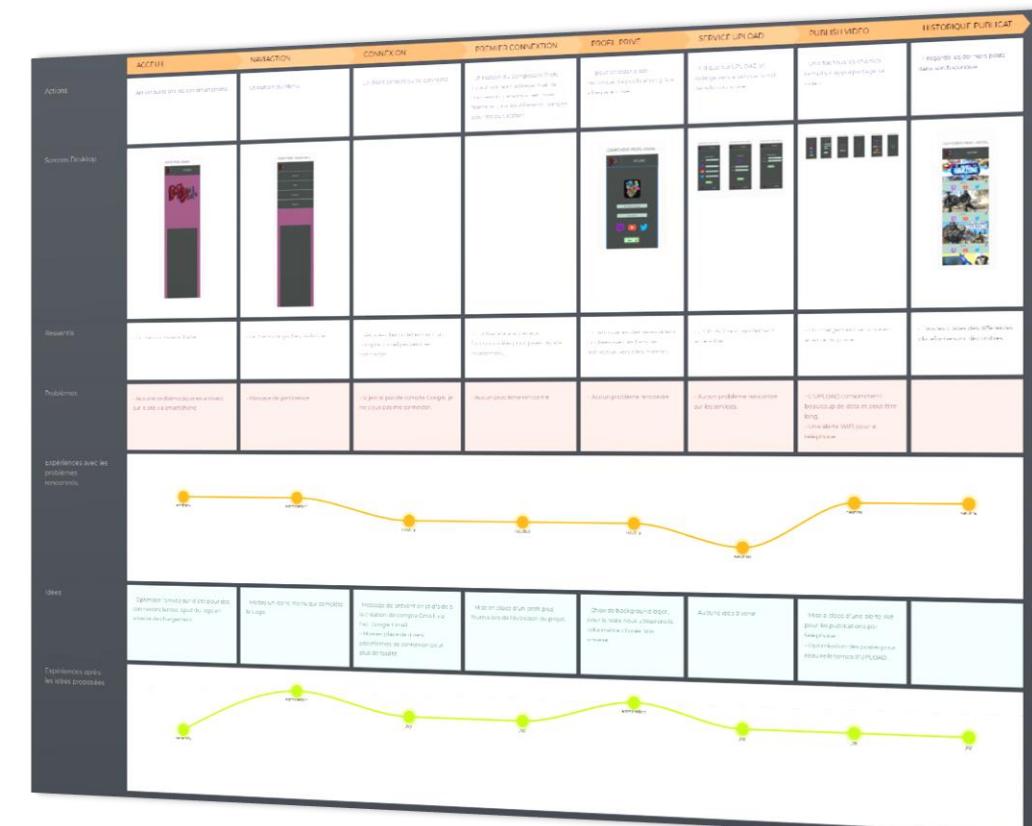
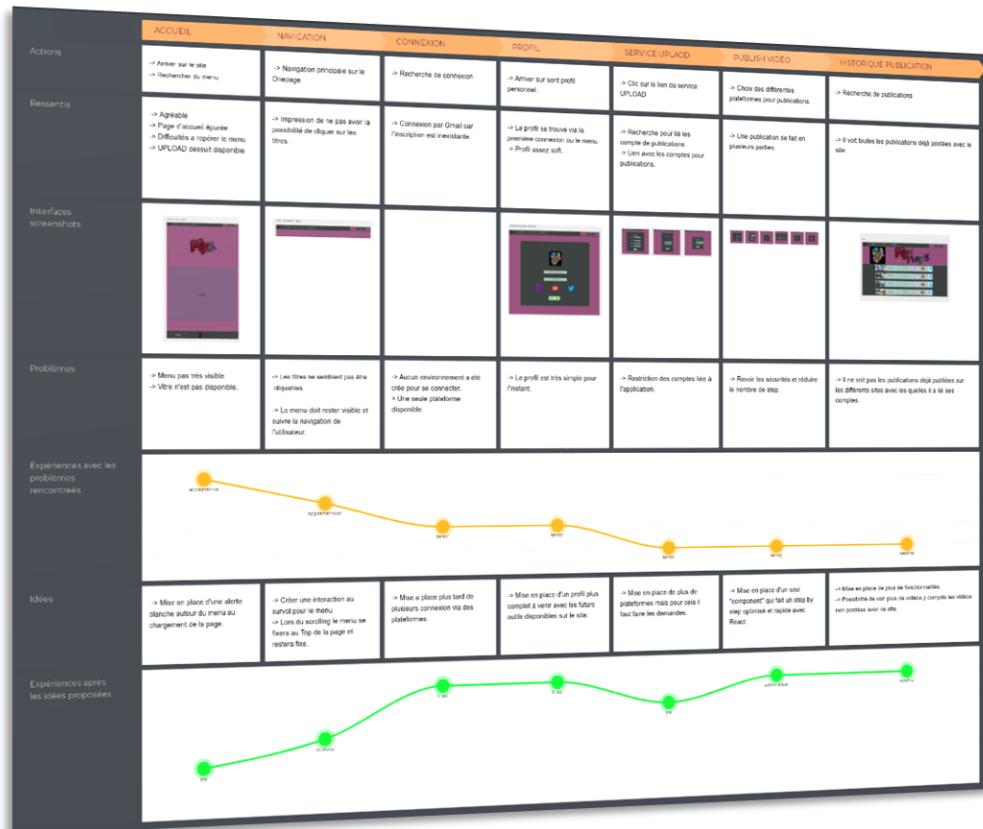
Outil Digital



Navigateur

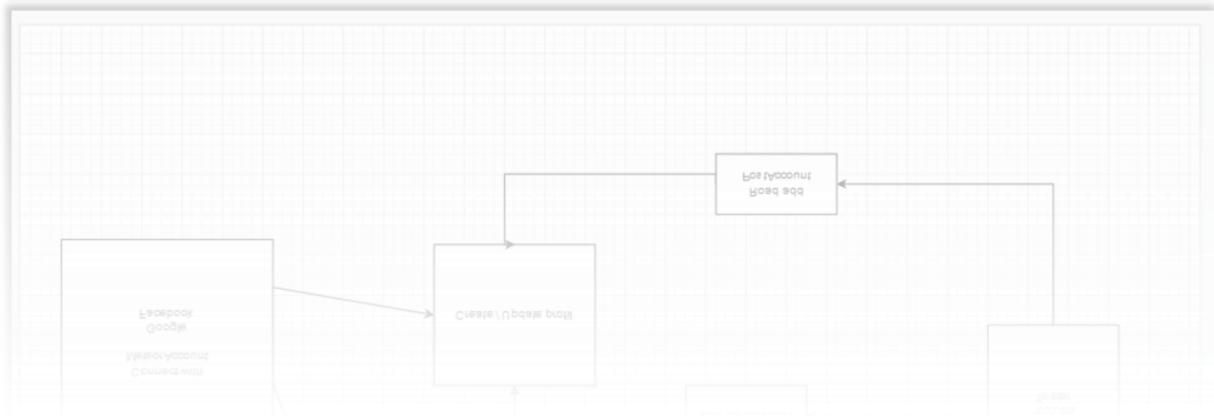
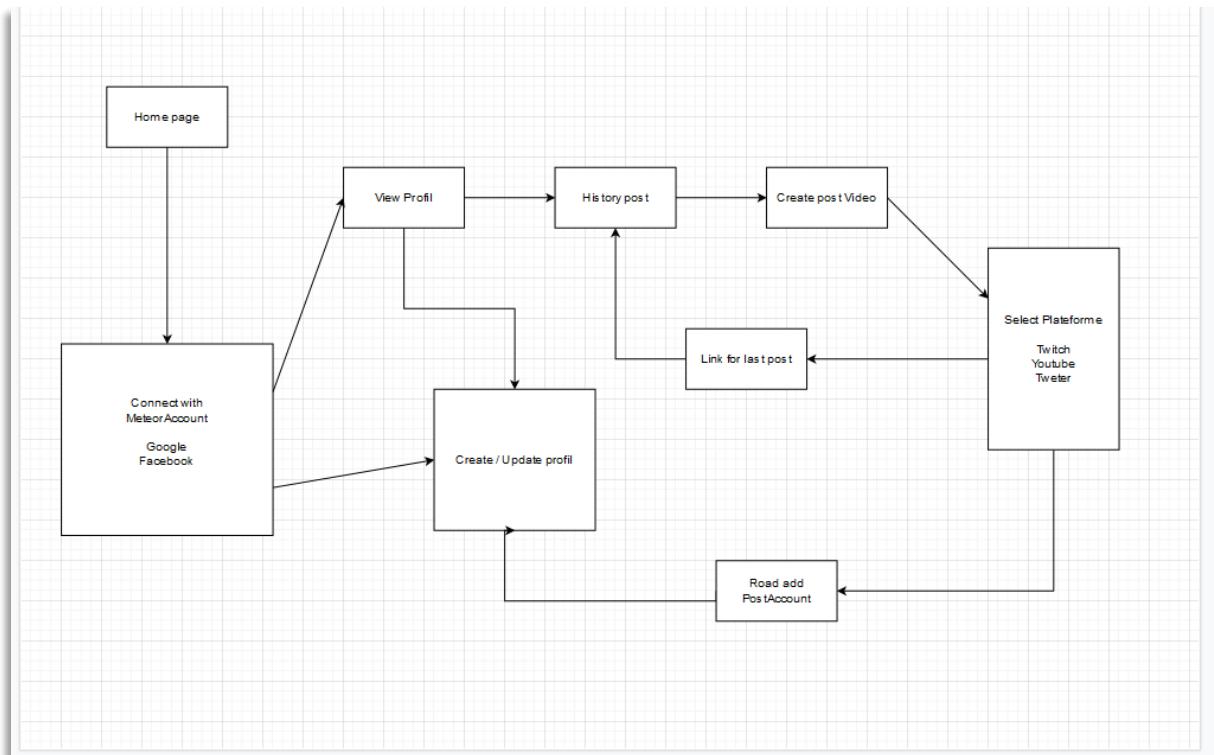


UX des utilisateurs

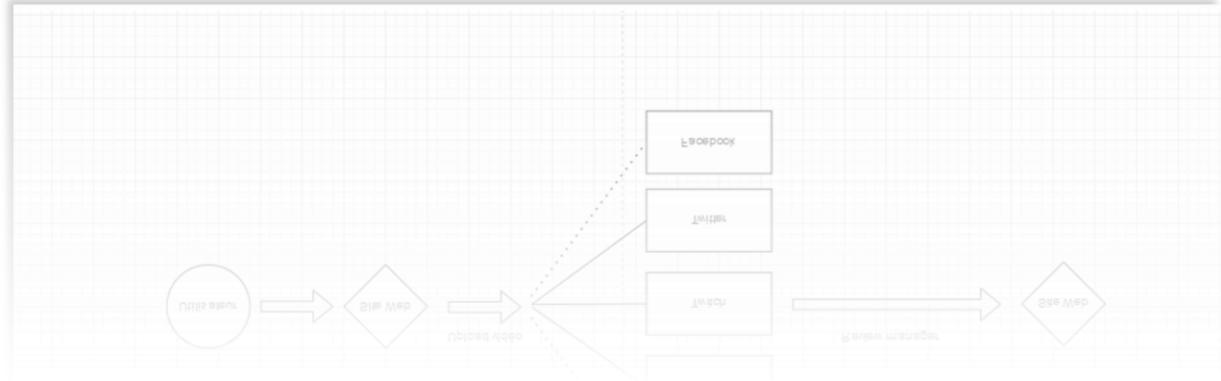
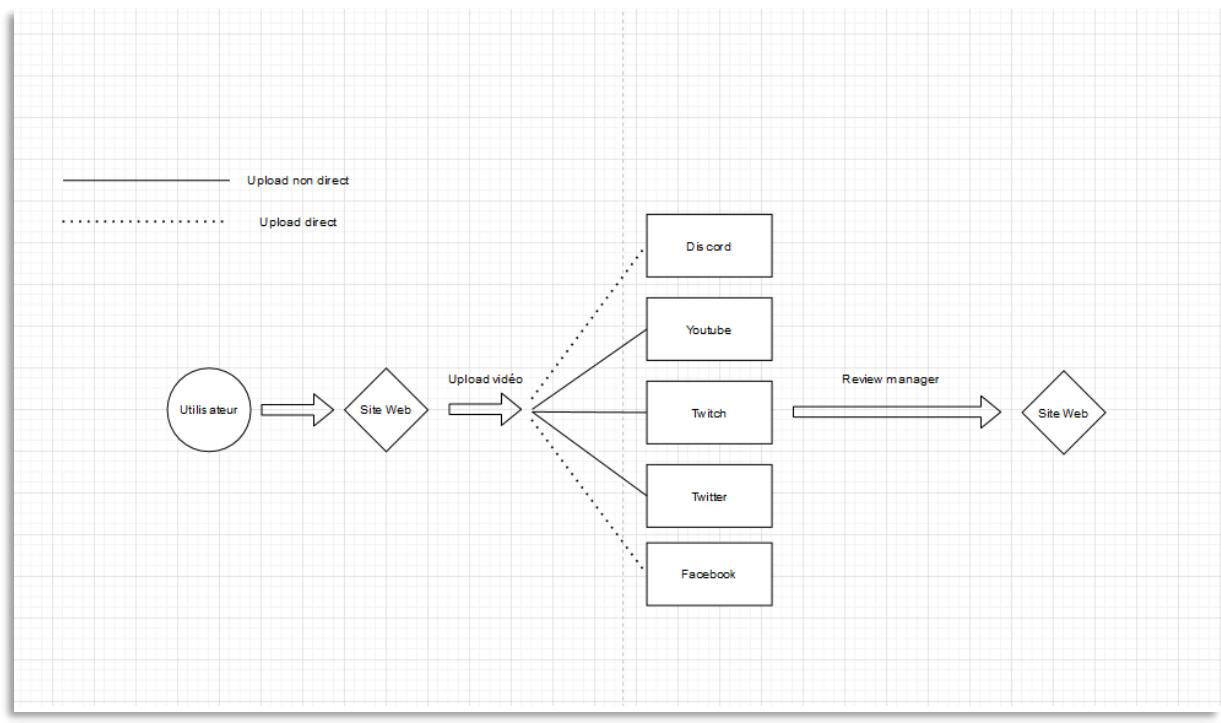


VII. Zoning des fonctionnalités

Fonctionnalité du site



Zoning fonctionnel de l'upload



VIII. Méthodologie de d'upload des vidéos

1. Méthode de poste avec Twitch

Nous allons faire appel à une API. Dans cette API les formats suivants sont acceptés :

- ❖ Formats de fichiers MP4, MOV, AVI et FLV
- ❖ Audio AAC
- ❖ Codec h264
- ❖ Débit binaire jusqu'à 10Mbps
- ❖ Jusqu'à 1080p / 60FPS

Une limite de taux de 5 UPLOAD simultanés par utilisateur avec un maximum de 100 téléchargements en 24 h.

Tout d'abord, le premier step est de pouvoir se connecter à l'API avec OAuth.

Faire appel à l'API, pour cela, nous avons besoin d'un ID client. Pour en recevoir un, nous devons nous connecter à la console des développeurs Twitch. Nous utiliserons l'accès URI de redirection avec OAuth. Une fois identifié, nous recevons un Tokken d'identification, que nous allons réutiliser pour une requête de type :

URL POST

`https://api.twitch.tv/kraken/videos?channel_id=<channel ID>&title=<video title>`

Exemple de requête :

```
curl -H 'Accept: application/vnd.twitchtv.v5+json' \
-H 'Authorization: OAuth cfabdegwdoklmawdzdo98xt2fo512y' \
-H 'Client-ID: uo6dggojyb8d6soh92zknwmi5ej1q2' \
-X POST
'https://api.twitch.tv/kraken/videos?channel_id=44322889&title=Test video')
```

Typage et description de donnée :

Nom	Type	La description
<i>description</i>	String	Brève description de la vidéo.
<i>game</i>	String	Nom du jeu dans la vidéo.
<i>language</i>	String	Langue de la vidéo (par exemple, en).
<i>tag_list</i>	String	Liste de balises séparées par des virgules décrivant la vidéo (par exemple, « avions, effrayant »). Maximum: 100 caractères par balise, 500 caractères pour toute la liste
<i>viewable</i>	String	Spécifie qui peut voir la vidéo. Valeurs valides: public (la vidéo est visible par tout le monde) ou private (la vidéo n'est visible que par le propriétaire et les éditeurs de la chaîne). Par défaut: public.
<i>viewable_at</i>	RFC3339 date	Date à laquelle la vidéo deviendra publique. Cela ne prend effet que si <i>viewable</i> =private.

Format UPLOAD de type JSON :

```
11 "upload": {
12     "token": "7_STl2gsKwDy1mHj2k951d3aodD13E_NzzQdS1N9110p6Ct0xFhvr05bWu7tGUKEH5jtD_hjB9q3X4vT0Q000B1UcmMxduT30FuHhAmFYgYf7DoJE9PVvGRZqk9W
13     QAR2ZGphUpj_237smnjE2gMoaQ--",
14     "url": "https://uploads.twitch.tv/upload/132561385"
15 },
16 "video": [
17     {
18         "title": "Test video",
19         "description": "",
20         "description_html": "",
21         "broadcast_id": 1,
22         "broadcast_type": "upload",
23         "status": "created",
24         "tag_list": "",
25         "views": 0,
26         "url": "https://www.twitch.tv/videos/132561385",
27         "language": "en",
28         "created_at": "2017-03-31T22:59:00Z",
29         "viewable": "public",
30         "viewable_at": null,
31         "published_at": null,
32         "_id": "v106400740",
33         "recorded_at": "2017-03-31T22:59:00Z",
34         "game": "",
35         "length": 0,
36         "preview": {
37             "small": "https://vod-secure.twitch.tv/_404/404_processing_80x45.png",
38             "medium": "https://vod-secure.twitch.tv/_404/404_processing_320x180.png",
39             "large": "https://vod-secure.twitch.tv/_404/404_processing_640x360.png",
40             "template": "https://vod-secure.twitch.tv/_404/404_processing_{width}x{height}.png"
41         },
42         "animated_preview_url": "https://vod-storyboards.twitch.tv/dallas/132561385/1d283591-a423-4f57-9656-a332d3949b55/storyboards/
43         132561385-strip-0.jpg",
44         "thumbnails": {
45             "small": [],
46             "medium": [],
47             "large": [],
48             "template": []
49         },
50         "fps": {},
51         "resolutions": {},
52         "channel": {
53             "_id": "44322889",
54             "name": "dallas",
55             "display_name": "dallas"
56         }
57     }
58 }
```

Procédé de mise à jour les informations sur la vidéo qui ont été créée.

Tous d'abord l'authentification est nécessaire avec OAuth 2.0

Scope requis visé : channel_editor

URL PUT

`https://api.twitch.tv/kraken/videos/<video ID>`

Les paramètres de requête facultatifs :

Nom	Type	La description
<code>description</code>	String	Brève description de la vidéo.
<code>game</code>	String	Nom du jeu dans la vidéo.
<code>language</code>	String	Langue de la vidéo (par exemple, en).
<code>tag_list</code>	String	Liste de balises séparées par des virgules décrivant la vidéo (par exemple, « avions, effrayant »). Maximum : 100 caractères par balise, 500 caractères pour toute la liste.
<code>title</code>	String	Titre de la vidéo. 100 caractères maximum.

Exemple de requête :

```
curl -H 'Accept: application/vnd.twitchtv.v5+json' \
-H 'Authorization: OAuth cfabdegwdoklmawdzdo98xt2fo512y' \
-H 'Client-ID: uo6dggojyb8d6soh92zknwmi5ej1q2' \
-X PUT 'https://api.twitch.tv/kraken/videos/106400740?title=Updated test video'
```

Procédé de suppression d'une vidéo spécifiée :

La vidéo peut être de n'importe quel type.

L'authentification est nécessaire avec OAuth 2.0

Scope requis visé : channel_editor

URL DELETE

<https://api.twitch.tv/kraken/videos/<video ID>>

Aucun paramètres facultatifs, mise en place d'une validation « self DELETE »

Exemple de requête :

```
curl -H 'Accept: application/vnd.twitchtv.v5+json' \
-H 'Authorization: OAuth cfabdegwdoklmawdzdo98xt2fo512y' \
-H 'Client-ID: uo6dggojyb8d6soh92zknwmi5ej1q2' \
-X DELETE 'https://api.twitch.tv/kraken/videos/106400740'
```

Réponse attendue :

```
200 OK
{"ok": "true"}
```

2. Méthode de poste avec Twitter

Les formats

Les vidéos Twitter doivent répondre aux exigences minimales de la plate-forme de médias sociaux.

- ❖ La limite de taille de la vidéo Twitter est de 512 Mo (asynchrone) / 15 Mo (synchronisation)
- ❖ La résolution vidéo minimale sur Twitter est de 32 x 32
- ❖ La résolution vidéo maximale sur Twitter est de 1 920 x 1 200 et 1 200 x 1 900
- ❖ Les vidéos Twitter doivent respecter les proportions suivantes : 1 : 2,39 - 2,39: 1
- ❖ La fréquence d'images maximale pour les vidéos est de 40 ips (images par seconde)
- ❖ Le débit binaire maximal est de 25 Mb/s (mégabits par seconde)
- ❖ Twitter prend actuellement en charge les formats vidéo suivants : formats MOV et MP4 pour les applications mobiles
- ❖ La durée minimale de la vidéo Twitter est de 0,5 secondes et la durée maximale de la vidéo Twitter est de 2 minutes 20 secondes (140 secondes)
- ❖ Twitter prend en charge les formats vidéo Web suivants : téléchargements vidéo MP4 au format H264 et audio AAC.

Requête

Les requêtes doivent être au format multipart/form-data ou application/x-www-form-urlencoded POST.

Le domaine de ce point de terminaison est upload.twitter.com

URL de la ressource

<https://upload.twitter.com/1.1/media/upload.json>

Informations sur les ressources

Formats de réponse	JSON
Nécessite une authentification ?	Oui (contexte utilisateur uniquement)
Rate limited ?	Oui

Paramètres

Nom	Obligatoire	La description
Médias	Obligatoire	Le contenu du fichier binaire brut en cours de téléchargement. Ne peut pas être utilisé avec media_data.
media_category	Optionnel	La catégorie qui représente la façon dont le média sera utilisé. Ce champ est obligatoire lors de l'utilisation du média avec l'API Ads

POST

https://upload.twitter.com/1.1/media/upload.json?media_category=tweet_video

Exemple de réponse

```
{  
  "media_id": 710511363345354753,  
  "media_id_string": "710511363345354753",  
  "media_key": "3_710511363345354753",  
  "size": 11065,  
  "expires_after_secs": 86400,  
  "video": {  
    "video_type": "video/mp4",  
    "w": 800,  
    "h": 320  
  }  
}
```

3. Méthode de poste avec YouTube

Les formats

Cette méthode prend en charge le téléchargement multimédia. Les fichiers téléchargés doivent respecter ces contraintes :

- ❖ Taille maximale du fichier : 128 Go
- ❖ Types MIME : Médias `video/*` compatibles ,`application/octet-stream`

Mise en place de l'autorisation de connexion

Mise en place de OAuth 2.0 pour pouvoir accéder à l'API de développement.

Vidéos

Les requêtes pour les différentes actions possibles présentées par la documentation YouTube.

Méthode	Requête HTTP	La description
URI relatif à <code>https://www.googleapis.com/youtube/v3/</code>		
<code>insert</code>	POST /videos	Télécharge une vidéo sur YouTube et définit éventuellement les métadonnées de la vidéo.
<code>list</code>	GET /videos	Renvoie une liste de vidéos correspondant aux paramètres de la requête API.
<code>delete</code>	DELETE /videos	Supprime une vidéo YouTube.
<code>update</code>	PUT /videos	Mets à jour les métadonnées d'une vidéo.

Les ressources de représentation sont grandes, mais sont disponibles sur la documentation :

<https://developers.google.com/youtube/v3/docs/videos#methods>

Exemple de requête :

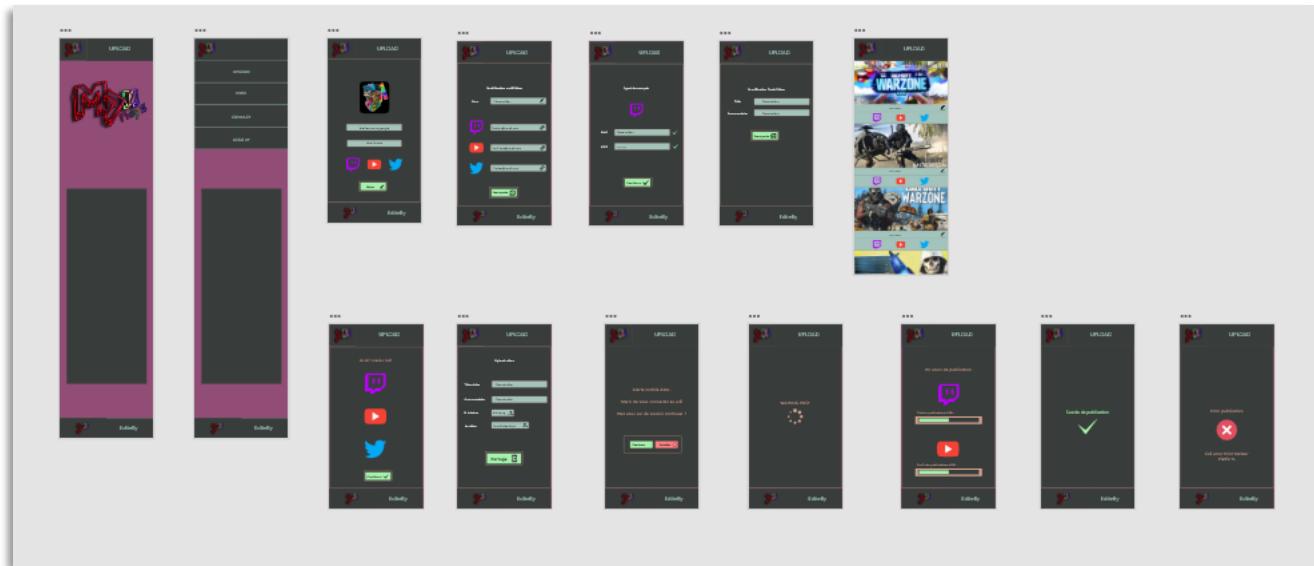
POST

[https://youtube.googleapis.com/youtube/v3/videos?part=snippet%2Cstatus&key=\[YOUR_API_KEY\]](https://youtube.googleapis.com/youtube/v3/videos?part=snippet%2Cstatus&key=[YOUR_API_KEY])

Exemple de data :

```
{  
  "kind": "youtube#video",  
  "etag": etag,  
  "id": string,  
  "snippet": {  
    "publishedAt": datetime,  
    "channelId": string,  
    "title": string,  
    "description": string,  
    "thumbnails": {  
      (key): {  
        "url": string,  
        "width": unsigned integer,  
        "height": unsigned integer  
      }  
    },  
    "channelTitle": string,  
    "tags": [  
      string  
    ],  
    "categoryId": string,  
    "liveBroadcastContent": string,  
    "defaultLanguage": string,  
    "localized": {  
      "title": string,  
      "description": string  
    },  
    "defaultAudioLanguage": string  
  },  
}
```

IX. Prototypage Bureau et Mobile final



X. Présentation base de données MongoDB

Modèle de données pour un utilisateur :

```
{  
  username: {  
    type: String,  
  },  
  services: {  
    type: Object,  
    blackbox: true  
  },  
  emails: {  
    type: Array,  
    optional: true  
  },  
  "emails.$": {  
    type: Object,  
    optional: true,  
    blackbox: true  
  },  
  coloreyes:{  
    type: String,  
    optional: true  
  },  
  createdAt: Date,  
}
```

XI. React hooks

Les **Hooks** sont des fonctions qui permettent de « se brancher » sur la gestion d'état local et de cycle de vie de React depuis des fonctions composants. Les Hooks ne fonctionnent pas dans des classes : ils nous permettent d'utiliser React sans classes.

Voici un exemple d'utilisation dans mon projet :

```
1 import { Meteor } from 'meteor/meteor';
2 import { useTracker } from 'meteor/react-meteor-data';
3 import React, {
4   useState, useMemo, useCallback, useEffect,
5 } from 'react';
6 import { toast } from 'react-toastify';
7
8 const ComLog = () => {
9
10   const handleClick = (e) => {
11     Meteor.loginWithGoogle({
12       requestPermissions: ['email'],
13     }, (error) => {
14       if (error) {
15         toast.error(error.reason);
16       } else {
17         toast.success('Bienvenue !');
18       }
19     });
20
21   const connected = useTracker(() => Meteor.userId(), []);
22
23   const logout = useCallback(() => {
24     Meteor.logout();
25   }, []);
26
27   const connectionLink = useMemo(() => {
28     if (connected) {
29       return (
30         <button onClick={logout}>Déconnexion</button>
31       );
32     }
33     return (
34       <button onClick={handleClick}>SIGNIN</button>
35     );
36   }, [connected, logout]);
37
38   return (
39     <>
40       {connectionLink}
41     </>
42   );
43 }
44
45 export default ComLog;
```

XII. Modules

React-router-dom

React Router est une extension à React qui permet de gérer les routes d'une application côté client. Il permet de synchroniser (d'associer) des composants graphiques React à des urls. React Router tire profit de l'interface Windows.history pour manipuler les URL et les associer à des composants React.

Commande d'installation dans PowerShell à la racine du dossier :

```
$ npm install --save react-router-dom
```

Exemple router-dom :

```
imports > ui > App.jsx > App
  1 import React from 'react';
  2 import { BrowserRouter as Router, Switch, Route } from 'react-router-dom';
  3 import { ToastContainer } from 'react-toastify';
  4
  5 import Home from './components/pages/Home';
  6 import AccountProfil from './components/pages/AccountProfil';
  7 import SelectMultipost from './components/pages/SelectMultipost';
  8 import StudioProfil from './components/pages/StudioProfil';
  9 import ContactPopup from './components/ContactPopup';
 10
 11 import 'react-toastify/dist/ReactToastify.css';
 12 import '../../../../../public/src/css/layout/normalize.css';
 13 import '../../../../../public/src/css/layout/header.css';
 14 import '../../../../../public/src/css/layout/layout.css';
 15
 16 import '../../../../../public/src/css/components/showcase.css';
 17 import '../../../../../public/src/css/components/accept.css';
 18
 19 import '../../../../../public/src/css/utilities/_colors.css';
 20 import '../../../../../public/src/css/utilities/_fonts.css';
 21
 22
 23 const App = () => {
 24   return(
 25     <>
 26       <ToastContainer />
 27       <ContactPopup />
 28       <Router>
 29         <Switch>
 30           <Route exact path="/" component={Home} />
 31           <Route path="/accountprofil" component={AccountProfil} />
 32           <Route path="/upload" component={SelectMultipost} />
 33           <Route path="/studio" component={StudioProfil} />
 34         </Switch>
 35       </Router>
 36     </>
 37   );
 38 };
 39
 40
 41
 42 export default App;
```

React-toastify

Les toasts sont des notifications légères conçues pour imiter les notifications push qui ont été popularisées par les systèmes d'exploitation mobiles et de bureau.

Commande d'installation dans PowerShell à la racine du dossier:

```
$ npm install --save react-toastify
```

```
const handleClick = (e) => {
  Meteor.loginWithGoogle({
    requestPermissions: ['email'],
  }, (error) => {
    if (error) {
      toast.error(error.reason);
    } else
      toast.success('Bienvenue !');
  });
};
```

XIII. Accounts management

Mise en place de la connexion via Account-Google

Google dispose d'une interface de programme d'application ("API") intégrée que l'on peut utiliser pour permettre à notre application Web d'accéder à un service Google protégé par le compte Google d'un utilisateur avec l'autorisation du propriétaire du compte Google.

Utilisation du settings.json

```
PS D:\travail\ecole\PFE_2020\PFE_Valentin_ACEBES\Mplayz> meteor --settings settings.json  
[[[[[ ~\D\travail\ecole\PFE_2020\PFE_Valentin_ACEBES\Mplayz ]]]]]  
  
=> Started proxy.  
=> Meteor 1.11.1 is available. Update this project with 'meteor update'.  
=> Started MongoDB.  
=> Started your app.  
  
=> App running at: http://localhost:3000/  
Type Control-C twice to stop.
```

Settings de l'API Google

```
{
  "private": {
    "clientId": "130232473888-  
vh19v85v2pmik9iblo0vk5igvpkjn9r3.apps.googleusercontent.com",
    "secret": "-uLIIcVeZYY_7rVc8x1uUV21"
  }
}
```

Récupérations des données utilisateurs avec la connexion Gmail

XIV. Sécurisation

1. Utilisation de méthodes

Toutes les données qui arrivent via les arguments de méthode doivent être validées et les méthodes ne doivent pas renvoyer de données auxquelles l'utilisateur ne devrait pas avoir accès.

Nous retrouvons :

- ❖ Validation de tous les arguments avec « check » ou « npm-schema »
- ❖ Ne pas transmettre userId du client
- ❖ Une seule méthode par action
- ❖ Limitation du débit de connexion

2. Les publications

Nous devons s'assurer qu'aucun des fichiers de code source ou de configurations servis au client ne contient de données secrètes.

Nous retrouverons :

- ❖ La mise en place d'un code serveur secret
- ❖ La sécurisation des clés API
- ❖ Paramètre sur le client
- ❖ Clés API pour Oauth

3. Fichier servis

Nous devons assurer qu'aucun des fichiers de code source ou de configurations servis au client ne contient de données secrètes.

Nous retrouverons :

- ❖ La mise en place d'un code serveur secret
- ❖ La sécurisation des clés API
- ❖ Paramètre sur le client
- ❖ Clés API pour Oauth

4. List de contrôle de sécurité

- ❖ S'assurer que l'application ne dispose pas des packages 'insecure' ou 'autopublish'
- ❖ Valider tous les arguments de méthode et de publication.
- ❖ Utilisons des méthodes au lieu d'insérer/mettre à jour/supprimer coté client.
- ❖ Utiliser les sélecteurs spécifiques et champs de filtre dans les publications
- ❖ D'assurer que les clés API et les mots de passe secrets ne figurent pas dans le code source
- ❖ Sécuriser les données par l'interface utilisateur.
- ❖ Ne pas faire confiance aux ID utilisateurs transmis par le client. Utiliser this.userId à l'intérieur des méthodes et des publications.

XV. Contraintes techniques

La mise en place de l'UPLOAD vers les différents prend du temps, car chaque plateforme a son propre langage de développement. La prospection des API est longue et fastidieuse.

Une gestion de format de vidéo devra être proposée, car sur Twitter, les durées de vidéos sont plus courtes que sur Twitch. La plateforme Twitch a atteint le record de 200 h de flux non-stop lors d'un live sur son application.

Les limitations des plateformes, car les démarches juridiques pour les API sont longues et des dossiers doivent être montés pour les structures de type Facebook afin de bien respecter les modalités de publication. Car lui-même surveille beaucoup les données entrantes.

Mon environnement de travail a été complètement instable sur les derniers mois donc j'ai dû suivre les cours dans un contexte compliqué et aussi le travail sur mon projet personnel, extrême difficile. J'ai perdu mon disque dur et aussi mon processeur en pleine année, j'ai pu récupérer mon projet sur GIT, mais pour le processeur, j'ai dû rester dans la difficulté pendant pas mal de temps.

XVI. CSS

1. Variable

Les propriétés personnalisées CSS (custom properties en anglais, aussi parfois appelés variable CSS) sont des entités définies par les développeurs ou les utilisateurs d'une page Web, contenant des valeurs spécifiques utilisables à travers le document.

The screenshot shows a GitHub Gist with the following CSS code:

```
Valentin, 7 days ago | 2 authors (Valentin and others)
anonymize, 25 days ago • [ADD] showcase and beginning header
:root {
    --black-olive: #rgb(56, 61, 59);
    --black-olive-smoke: #rgba(56, 61, 59, 0.7);
    --tumbleweed: #rgb(216, 164, 143);
    --sugar-plum: #rgb(145, 77, 118);
    --sugar-plum-light: #rgba(145, 77, 118, 0.80);
    --blue-munsell: #rgb(59, 142, 165);
    --opal: #rgb(163, 196, 188);
}

.box-shadow-windows{
    box-shadow: 0 1px 1px #rgba(0,0,0,0.12),
    0 2px 2px #rgba(0,0,0,0.12),
    0 4px 4px #rgba(0,0,0,0.12),
    0 8px 8px #rgba(0,0,0,0.12),
    0 16px 16px #rgba(0,0,0,0.12);
}

.box-shadow-windows-up{
    box-shadow: 0 1px 1px #rgba(0,0,0,0.12),
    0 2px 2px #rgba(0,0,0,0.12),
    0 4px 4px #rgba(0,0,0,0.12),
    0 8px 8px #rgba(0,0,0,0.12),
    0 16px 16px #rgba(0,0,0,0.12);
}

.native-radius{
    border-radius: 3px;
}

}
padding-top: 3px;
outline: 1px solid black;
border: 1px solid black;
background-color: white;
border-radius: 3px;
width: 100px;
height: 100px;
display: flex;
flex-direction: column;
align-items: center;
justify-content: center;
font-size: 14px;
font-weight: bold;
color: black;
text-align: center;
margin: auto;
margin-top: 20px;
border: 1px solid black;
border-radius: 50% / 50% 50% 50% / 50% 50% 50% 50%;
```

2. Normalisation Naming

Syntaxes globales

Langage	Modèle	Syntaxe
FOLDER / FILE	Underscore case	Nom_du_fichier

Syntaxes HTML

Modèle	Syntaxe	Commentaire
CSS (id , class , data)	Kebab Case	Exemple: sujet-action-com

Syntaxes JS

Modèle	Syntaxe	Commentaire
Namespace / Class	UpperCamelCase	
Methode	lowerCamelCase	
Variable	lowerCamelCase	
Constante	UPPER_TEXT	
CSS (id , class , data)	Kebab Case	Exemple: sujet-action-com

Syntaxes CSS

Modèle	Syntaxe	Commandaire
CSS (id , class , data)	Kebab Case	Exemple: sujet-action-com

3. Structure CSS

Les fichiers CSS sont spécifiques à chaque page pour l'optimisation sauf les fichiers de type layout que l'on retrouve partout. Donc ils sont intégrés dans le fichier racine APP. On les reconnaît à leurs underscore devant l'intitulé du fichier.

XVI. Projection du site

1. Description des principales fonctionnalités dans l'évolution

Mise en place de plusieurs plateformes pour une connexion au site

La mise en place d'une connexion autre que Gmail avec plateforme tels que Facebook ou tweeter seraient l'idéal pour toucher le plus d'utilisateurs dans la simplicité de créations de comptes et de connexions aux sites.

Plus de plateformes de publications

Des plateformes tels que Facebook ou snap sont difficiles d'accès, car les restrictions de publications sont juridiquement complexes, mais seraient un atout dans le module vidéo short clip qui sera en futur développement.

Gestionnaire de publication

La gestion de publications pourra permettre de supprimer/modifier une vidéo directement sur le site qui lui-même.

Agenda de publication différé

Permettre aux utilisateurs de poster des vidéos préchargées sur un planning défini sur le site.

Éditeur de vidéo et outils digital

2. Modération d'accès

Free accès

La mise place d'un accès free pourrait permettre un nombre de postes et un accès aux outils limités.

Forfait Premium

Le forfait premium est à définir, mais il comprend dedans un nombre de postes illimité sur toutes les plateformes disponibles.

Un accès aux nouveaux outils de gestion de poste et de management.

XVII. Conclusion

Le projet MplayZ est la première version de l'application, on peut la considérer comme une pré-alpha. Il permet lui-même de tester toutes les possibilités et réalisations pour les versions futures. J'ai développé tous les points dans leur généralité pour ensuite travailler sur des points précis afin de solidifier les différents domaines que présente ce dossier. Un projet de cette envergure nécessite un développement plus important ainsi qu'une gestion de projet conséquente. J'ai énormément appris sur différents domaines de dev, comme les librairies et frameworks React, Meteor, Node.js mais aussi de mettre en place un projet le plus rapidement possible avec les bons outils. Ces outils m'ont permis de répartir les charges de travail, et grâce à eux, j'ai eu un meilleur management et une meilleure rapidité d'exécution.

Mon souhait pour le début de mon projet a été d'utiliser la plus simple des plateformes et la plus représentative, mais par la suite, j'agrémenterais de plateformes et de fonctionnalités.

J'ai comme ambition pour la suite de créer un outil de management de médias pour le monde professionnel et public du jeu vidéo ou des activités médiatiques permettront aux différents acteurs de pouvoir organiser leurs plannings de publications ainsi que différer leurs vidéos.

XVIII. Remerciements

Tout d'abord, je tiens à remercier mes professeurs et en particulier Olivier MANGIN qui a su m'épauler et me guider durant cette formation,

Je remercie mes camarades Yann METIER et Dorian VERDON qui ont su m'épauler tout au long de ce projet.

Je remercie également Manu GUCEMAS qui m'a permis d'intégrer THALES et de ce fait de découvrir le monde de l'aéronautique.

Pour finir, merci à ma famille de m'avoir soutenu tout au long de mon apprentissage. Je tiens à souligner que cette période a été très difficile due au confinement et à la pandémie.

XIX. Annexe