APO~~calypse~~théose day

Pourquoi faire ?

Le socle, permet d'acquérir du *savoir*.

La spé, permet d'acquérir du *savoir-faire* :

ok, j'ai des connaissances, mais en plus je sais vraiment m'en servir parce que j'ai passé 20 jours sur un seul truc.

Le projet, c'est le *savoir-être* :

je code propre, je code avec git, je sais bosser sur un code que je n'ai pas écrit, je respecte les deadlines, je suis des process, je m'entends avec mon équipe 🙂

Déroulement

De quand à quand ?

Début de l'apothéose : aujourd'hui mercredi 31/05
Fin de l'apothéose : le mercredi 28/06 après les démos de projets

Quel rythme ?

L'apothéose dure 4 semaines divisées en 4 sprints, du mercredi au mercredi.

Sprint 0 : du 31/05 au 07/06

Sprint 1 : du 07/06 au 14/06

Sprint 2 : du 14/06 au 21/06

Sprint 3 : du 21/06 au 28/06

Les horaires

9h à 15h + 2h libres

vous devez rester facilement joignables sur Slack

Rendez-vous hebdomadaires

- 1 RDV avec l'équipe d'encadrement
  - à peu près à mi-sprint
  - (prise de RDV via Doodle ou Google Agenda)
- 1 rétrospective le mercredi

Rétrospectives

C'est quoi ?

Réunion sur Slippers et sur Discord, afin que chaque groupe présente ce qu'il a fait lors du précédent sprint (partage d'écran + son)

Pourquoi ?

Voir ce que les autres groupes ont fait, les problèmes qu'ils ont rencontrés...
Se préparer pour les démos
S'entraîner à présenter un projet devant un client/une équipe

Déroulement

Chaque membre du groupe doit présenter le projet au moins une fois. Possibilité de présenter à plusieurs, mais attention à bien se coordonner et au timing

Durée : 15 minutes réparties en 8 minutes de présentation et 7 minutes de questions

Conseils pour la présentation

- Recontextualisation du projet : ça s'appelle comment, c'est quoi le but
- Ce qui était prévu de faire durant le sprint
- Ce qui a été fait durant le sprint (ce qui fonctionne)
- Les difficultés rencontrées
- Ce qu'il est prévu de faire lors du prochain sprint
- S'entraîner avant : prévoir un temps en groupe pour se préparer (le mardi après-midi par exemple)

Les 11 ~~commandements~~ conseils pour bien vivre son apothéose

1. **Communiquez !** Ne codez pas dans votre coin. Ne restez pas des heures sans parler avec votre équipe (surtout back/front). Manque de communication = malentendus = groupe qui prend feu ! 😱🔥
2. Ne restez pas trop longtemps (des heures) sur le même problème. Essayez de le résoudre à plusieurs.
3. Tentez de trancher assez vite sans perdre trop de temps dans des débats interminables.

4. Avancez par micro-objectifs, par itérations : c'est bon pour le moral et ça permet de voir qu'on avance. Si votre objectif est trop lointain ("faire un site"), vous allez avoir l'impression de stagner.
5. **Entraidez-vous**, dans le groupe, et entre groupes. 🤗
6. Prenez du recul, réévaluez vos objectifs : c'est juste un mois et c'est un entraînement, **vous n'avez rien à perdre mais tout à gagner**.

7. **Désamorcez les conflits au plus tôt** : si le ton monte, penser à prendre l'air, ne pas se faire dépasser par sa pensée (et derrière un écran c'est facile).
8. Travaillez vos **hard skills** (savoir-faire) mais aussi vos **soft skills** (savoir-être)
9. Vous avez du mal à travailler avec telle personne ? Ca ne vas durer qu'un mois et c'est comme en entreprise : on ne choisit pas ses collègues ni les projets sur lesquels on travaille.

10. Ne codez pas 12 heures d'affilée. Dormez, les déclics se feront plus facilement ! 💤
11. Hydratez-vous et prenez des douches, ça détend 😌

Les 7 ~~péchés capitaux~~ conseils d'organisation

1. **Ne codez jamais sur master** mais sur des branches à part. Exemple : une branche front et une branche back, ou une branche par personne, **ou une branche par fonctionnalité.** Décidez du process ensemble.
2. Faites des **petits commits réguliers et explicites**, et **en anglais**. Tout le monde !
3. **Commentez votre code** au maximum : pour vous et pour votre groupe, pour le TP. **En anglais** aussi de préférence.

4. Faites des **Pull Request** et des **Code Reviews**
    - *Pas pour les profs (vous êtes autonomes)*
    - Vérifier qu'il n'y a pas de conflits
    - Vérifier que votre code est compris par les autres
    - Pour que tout le monde puisse voir l'avancée du projet
5. Faites du **pair-programming** quand vous en ressentez le besoin.

6. Faites une **réunion de quelques minutes tous les matins**

- **"Stand-up meetings"** pour coller à la méthode Scrum ~ 5 minutes chacun.
- C'est l'occasion de demander de l'aide, de réévaluer les objectifs et de remplir les carnets de bord.
- Chacun fait le point sur :
  - Ce qu'il avait prévu de faire la veille
  - Ce qu'il a fait/terminé, les victoires qu'il a accomplies
  - Les difficultés qu'il a rencontrées - Ce qu'il compte faire aujourd'hui
- NB : Des fiches récap existent sur les méthodes agiles et la méthode Scrum.

7. Créez et **définissez des conventions** que chaque membre du groupe devra respecter : nommages de fichiers/fonctions, titres de commits etc. pour homogénéiser le projet.

En parlant conventions...

Tout le monde change son pseudo sur Discord en suivant la nomenclature : **(NomProjet) Prenom-PremiereLettreDuNom**
Rien de plus simple et ce sera d'une grande utilité pour vous retrouver et notamment pour les rétrospectives, rdv de suivi et démos ^^

Une petite pause ?!

Sprint 0

Le quoi 0 ?

- Il commence cet après-midi et c'est **le plus important** !
- Il sert à définir les bases, l'organisation et la structure du projet.
- Il va falloir **produire un certain nombre de documents** et les **déposer dans le dossier Drive du projet**. Ces documents sont **obligatoires pour le TP** : eBook Titre Pro DWWM

ON NE CODE PAS

- durant ce sprint
- du moins pas tant que l'équipe péda n'a pas validé explicitement les documents.

On commence à penser nos choix techniques

- Coté PHP, on laisse de côté API platform etc., on garde EasyAdmin et ses copains en bonus !
- **Il faut comprendre ce que l'on fait durant cette période de projet, pas de magie !**
- **Que ce soit back ou front...**

SP0 : Les documents à fournir

1. Le cahier des charges

Il sert à apporter un maximum d'informations sur le projet (besoins, objectifs, fonctionnalités...)
Il est modifiable au fil du temps (pas figé après le Sprint 0) pour réévaluer/redéfinir les objectifs.
Il devra contenir :

Le CDC : contenu

- La **présentation** du projet
- La définition des **besoins** (problèmes auxquels répond le projet) et des **objectifs** (solutions qu'apporte le projet) du projet

Le CDC : contenu

- Les **fonctionnalités** du projet (spécifications fonctionnelles)
  - Le **MVP (Minimum Viable Product)** : image
    - Le projet va faire l'objet de plusieurs étapes : l'idée c'est qu'**à chaque étape le produit fonctionne, même si ce n'est pas avec toutes les fonctionnalités qui étaient prévues**
    - Il faut se poser la question : pour que mon projet marche, est-ce que telle fonctionnalité doit marcher (MVP) ou être une fonctionnalité annexe ?
  - Les **évolutions potentielles** (ce qui ne sera pas terminé) : tout ce qui est prévu mais ne fait pas partie du MVP

Le CDC : contenu

- La liste des **technololgies** utilisées pour le projet (spécifications techniques)
- La définition de la **cible** du projet (le public visé) : ça permettra entre autres choses d'anticiper certaines contraintes visuelles/ergonomiques/techniques
- Les **navigateurs** compatibles
- L'**arborescence** de l'application (le chemin de l'utilisateur)

Le CDC : contenu

- La liste des **routes** prévues
- La liste des **User stories** : micro-scénarios, en tant que tel utilisateur, je dois pouvoir effectuer telle action depuis tel endroit (chaque action sera redécoupée en différentes fonctionnalités)
- La liste des **rôles de chacun**

CDC : Les rôles individuels

Chaque membre du groupe doit **s'attribuer un ou plusieurs rôles (en plus de celui de développeur)**.

Product Owner

Connaît le produit et représente les intérêts et les besoins du client/des utilisateurs purement fonctionnels (pas techniques)
Tranche en cas de conflits fonctionnels (pas techniques)
En général c'est le porteur du projet (s'il est dans le groupe)

Scrum Master

Garant de la méthode du projet : il gère le respect des conventions définies dans le groupe
S'assure que toutes les tâches sont bien attribuées, suivies, accomplies
Assure la communication au sein du groupe : il vérifie que tout le monde a les bonnes informations
Gère l'outil de suivi du projet
Anime la réunion du matin et gère l'avancement du projet

Lead dev

Front / Back

Choisit les orientations importantes, choix techniques importants

S'assure du bon fonctionnement de sa partie du projet

Référents techniques : git

Git master

Garant du bon fonctionnement du versionning avec Git.

Responsable du bon fonctionnement du versionning, vérifie les PR et merge, gère les conflits etc.

Référents techniques : techno

Référent par librairie/techno particulière (exemples : Bootstrap, Google Maps...)
S'informe, se documente sur cette techno, sa sémantique, son utilisation.
Restitue les informations au groupe

2. Les documents relatifs à la BDD

Le **MCD** et éventuellement le **MLD**
Le **dico de données**
Pensez à regarder les **fiches récap**

3. Les Wireframes

- Représentation des éléments de vos pages et de leur agencement (zoning)
- Ce ne sont pas des maquettes graphiques : ils ne doivent pas contenir de logos, ni de couleurs, ni d'images.
- Il faut **légender les wireframes** : tout le monde n'a pas la même perception d'une convention implicite. Le but est donc d'expliciter. Il faut donc indiquer le type d'élément, ce qu'il fait et à quelle condition/action.

Exemple : 3 lignes parallèles = menu burger pour vous, mais pas forcément pour tout le monde.

- Une vue doit contenir au moins deux wireframes (version desktop et version mobile).

- **Chaque membre du groupe doit réaliser 2 wireframes minimum** : une page en version mobile et en version desktop.
- **En bonus**, vous pouvez faire des maquettes et une charte graphique. Exemples d'outils :

- Mockflow
- Figma
- Wireframe.cc
- Whimsical

4. Les documents de veille

Ils sont **propres à chaque membre du groupe**. C'est comme une **bibliographie** où vous devez noter :

- Les recherches effectuées (mots-clés)
- Les liens vers les ressources utilisées pour résoudre les problèmes
- Bugs rencontrés et solutions mises en place

L'intérêt ici est d'anticiper les dossiers qu'il faudra préparer pour le TP

5. Les carnets de bord

- Il y en a 2 types :
  - Le **journal d'équipe** : il est rempli par le scrum master.
  - Le **journal personnel** : il est propre à chaque membre du groupe.
- Les carnets de bord servent à avoir une traçabilité au jour le jour du projet, et ainsi de votre progression durant l'apothéose.

Chaque jour on y met

- La date et le Sprint actuel
- Un résumé de ce qui a été fait la veille
- Ce que l'on compte faire aujourd'hui
- Les difficultées rencontrées
- Les solutions / contournements apportés
- Exemple de carnet de bord

Soyez direct

On essaie d’être le plus simple, concis et le plus clair possible. .
N'importe qui doit pouvoir comprendre le contenu de ces carnets.
Le mieux c'est de les remplir en même temps que les réunions du matin.
Important pour le TP pour l'aspect "remontée dans le temps"

6. Un outil de gestion et suivi de projet

Outils

Exemples :

- Trello
- Github Project
- Clickup
- Monday.com
- Kanban Tool
- Notion
- Etc.

Il existe un tas d'outils, à vous d'en choisir un.

Soyez cohérents

Veillez à avoir des documents toujours cohérents entre eux.
Vous allez être amenés à les mettre à jours plusieurs fois tout au long du projet.
Si vous en modifiez un, vérifiez que les autres sont à jour également.
Vous avez à disposition un channel dédié sur Slack, un Drive et un ou deux repo Github par projet.

Sprints 1 & 2

On code (enfin) ! 🎉

On fait avancer le projet.

On redéfinit les enjeux au fur et à mesure de l'avancée du projet :

- on ajoute ou on retire des fonctionnalités
- on revoit ses objectifs
- on réadapte les documents

Sprint 3

Assurer le projet

- On ne code plus de nouvelles fonctionnalités.
- On termine celles qui ne sont pas terminées.
- On teste, on corrige les derniers bugs.
- On fait du refactoring : on revoit le code, on homogénéise, on nettoie.
- On se prépare et on s'entraîne pour la démo.

A chaque sprint

Toutes les semaines

Rétrospectives (on peut/doit se préparer/s'entraîner avant)
On déploie le projet (sauf Sprint 0) **AVANT chaque rétrospective**
Point hebdomadaire avec les encadrants.

Déploiement

Serveurs O'clock disponibles

On le fait dès que possible pour se débarrasser des bugs fréquents et éviter les surprises de dernière minute

2 solutions pour le déploiement :

- Un serveur par projet : création de virtual hosts pour dire front branché sur ce dossier, back sur ce dossier.
- Un serveur pour le front et un serveur pour le back : moins de configurations DocumentRoot, par contre 2 serveurs différents donc 2 installations à faire (Apache, PHP, MySQL etc.).

Déploiement

Si on n'y arrive pas, ce n'est pas grave, on présente en local.
Le but c'est d'essayer pour avoir une idée de ce que ça implique et se préparer pour les démos.
On ne perd pas un temps fou là-dessus (2/3h par sprint).

Gestion des problèmes

Si l'immeuble prend feu, si je suis face à un mur, je fais quoi ?

1. On cherche la réponse sur les internets : RTFM, Google, Stackoverflow...
2. On cherche en équipe
3. On demande sur Slack dans #entraide
4. On va regarder les issues des autres groupes : ils ont très probablement déjà eu le même problème
5. D'où le fort intérêt de poster dans votre Slack et de rester sur le Discord O'clock (on peut voir et se rendre compte de potentiels problèmes !)

6. En dernier recours, on ouvre une issue sur le repo Projects

- On met un titre le plus explicite possible 👮 Pas de "ça marche pas", il parait qu'un chat meurt à chaque fois qu'on dit ça 😭
- On suit les étapes du template : on décrit le problème, comment le reproduire, et on montre du code
- On pense à mettre son code entre trois backticks (ALT GR + 7) suivis du nom de la techno : ```js
- On ajoute un label à l'issue pour lui attribuer un/des sujet.s
- Si on a trouvé comment la résoudre : on indique ce qui n'allait pas, les étapes de résolution et on ferme l'issue.
- On ne fait pas de demandes de type MP ou "Je suis dans le salon XXX sur Discord" : les profs n'étant plus disponibles, ils répondront en asynchrone quand ils auront le temps.
- On n'hésite pas à répondre aux issues des autres groupes 😀 (sans les polluer ^^).

Les droits

Concernant les images et médias que vous utiliserez pour votre projet

Ne les récupérez pas au hasard, n'importe où sur internet
Vous aurez des problèmes
Nous aurons des problèmes (c'est déjà arrivé 😱)

Prenez vos propres photos ou choisissez des banques d'images libres de droits telles que :

- Pixabay
- Pexels
- Freephotos.cc
- Etc.

Fiches récap

Pensez à consulter les fiches récap présentes sur Kourou
Notamment celle sur une liste d'outils qui pourront vous être utiles 😏

Et...

Franchement...

Voilà !

That's all folks!

Souvenez-vous, pendant cette période, **vous n'avez rien à perdre et tout à gagner**