(01) In our documentation we trust

[Image : gazon] https://www.pexels.com/fr-fr/photo/arriere-plan-brillant-champ-d-herbe-clairiere-413195/

But de la présentation : - Expliquer la démarche engagée par l'équipe Publisher concernant la documentation - Adoption de l'engineering - Prendre du recul par rapport à la documentation - Fil rouge des slides est analogie entre les légumes et la documentation. - Histoire de Kelkoo romancée sous le spectre de la documentation - 20 minutes de présentation, 10 minutes de démo

(02) Presentation DEV / EM / PO

[Image : Feuilles rouges] Photo Alaska, G.R.

- Dev: doit fournir de la documentation sur les fonctionnalités développées
- EM: doit fournir de la documentation haut niveau sur l'état des systems
- PO de theGardener
    - Cette démarche s'appuie sur un outil theGardener qui est déployée sur 3 instances en production pour des audiences différentes

(03) Avant l'agilité

[Image : Plante qui pousse] https://www.pexels.com/fr-fr/photo/centrale-grandir-jardiner-plante-1214394/

- Parler du cycle en V pour demander à chaque fois.
    - A quoi ressemblait la documentation?

(04) Documentation en Cycle en V ou WaterFall

[Image : Feuilles mortes] https://pixabay.com/photos/leaf-leaves-fall-sewer-drain-grid-3791022/

- La documentation fonctionnelle et technique est écrite avant toute implementation
    - Ultra détaillées et Exaustive
- Seulement le produit ne correspond pas forcément au besoin, la documentation technique associée a toutes les chances d'être également en décalage

=> A quoi ressemblait la documentation ? - des piles de documents avec des specifications détaillées imprimés qui correspondent à une application qui n'a jamais été écrite.

(05) Documentation en mode agile: aucune

[Image : Désert] https://www.pexels.com/photo/green-bushes-on-desert-998653/

- Rappel sur l'agilité: itérations, développement d'une application correspondant aux attentes
- Un grand néant dans certain cas:
    - Le passage à l'agilité a eu pour conséquence que l'écriture de la documentation est passé hors

scope.
- La documentation fonctionnelle se perd dans les tickets de representant les stories.

=> A quoi ressemblait la documentation ? - Des user stories terminées qui sont dans perdues dans Jira.

(06) Documentation en mode agile: le bordel

[Image : Jungle] https://www.pexels.com/photo/green-leaf-trees-904807/

- Un Wiki/Confluence interne est un bordel sans nom:
    - mal organisé
    - bcp de données
        - melange de minutes de meeting Produit / Technique
        - decisions d'architecture dont on ne sait pas la valeur aujourd'hui
        - des estimations pour les GPR
        - des investigations
        - des tests de performances
        - des informations liées à l'agilités ou à l'organisation d'équipe
        - information sur les entrées sorties des systemes... mais jamais exhaustive. du moins ne le sais pas vraiment
    - compliqué de trouver quelque chose d'utilisable
- Documentation de type Java doc hyper precise mais ne donnant aucun contexte. Ne sert que lorsque l'on fait de la completion dans un éditeur, ne sert pas pour comprendre comment utiliser une brique technique.

=> A quoi ressemblait la documentation ? - Confluence avec tout et n'importe quoi

(07) Documentation en mode agile: peu de confiance

[Image : 2 Champignons venimeux] https://www.pexels.com/fr-fr/photo/amanite-tue-mouches-champignon-champignons-dangereux-2443/

- Une documentation pour les utilisateurs externes tres peu modifiées par les developpeurs internes => désynchronisation entre les services que l'on présente et la réalité => perte potentielle liées par non adhésion
- Manque de confiance /
    - Documentation fonctionnelle
    - Documentation technique

=> A quoi ressemblait la documentation ? - Explications détaillées sur le MLR dans le Search. - Lequel ne fonctionne plus depuis des années - Comment utilisé Product Search sur Kelkoo Developer Network

=> - Effectuée de maniere itérative, donc à moins de chance d'être en décalage - Mais - la documentation fonctionnelle et technique est perdue - dans outil de type Jira/Tulip avec une pile de ticket DONE. - dans une wiki/confluence - l'architeture se résume simplement à un diagramme d'architecture de l'état courant

=> On a gagné en efficacité de développement mais on a perdu en terme de documentation => On n'a pas

confiance en notre documentation....

Les symptomes:

Question auquelle on n'arrive pas à répondre pas facilement : - Où est la liste des WebService disponibles sur tel System ? - Pouquoi on a fait ce choix d'architecture, ca n'a aucun sens ? - Quelle est le niveau robustess de ce Web service ? - Dans quel context je peux utiliset cette fonctionnalité ?

Moyen que l'on trouve pour y répondre : - C'est dans un mail de machin de l'été dernier. - Va discuter avec truc c'est lui qui sait. - Fait un recherche full text sur le Wiki tu devrais trouver. - Je vais regarder dans le code et je te dis.

Comment on en est arrivé là et quelles sont les actions pour retrouver cette confiance ?

(08) Digression, sur les tests

[Image : Heolienne] rien à voir https://www.pexels.com/photo/clouds-dawn-dusk-electricity-157039/

- **A un moment donné les tests étaient chiants,**on ne voyait pas l'intérêt, ça faisant perdre un temps précieux et puis d'abord tester c'était douter.... et il y a eu l'avènement du TDD et **d'un seul coup, les tests sont devenus excitants à écrire par le développeur**, on a trouvé un grand intérêt pour la couverture de test, la non régression et j'en passe....

[Diagramme : Cycle de dev sans les tests]

- TDD, en substance c'est mettre le TEST dans le cycle de DEV

[Diagramme : Boucle du TDD]

- DEV est tres guidé Test / Main / Refactoring
- Gros intéret pour la validation de la story, non regression

=> Adoption

(09) TDD => BDD

[Diagramme : Scenarios au centre]

- Rappel :
    - Specification fonctionnelle lisible par le Product Owner et le DEV + implémentatable
    - Scenario Given / When / Then
    - Test et Documentation en même temps
    - Documentation validée par le code
    - TDD gros grain au niveau fonctionnel

(10) Projet innovation pour adresser ce besoin

[Image : Source] https://pixabay.com/photos/source-water-natural-fresh-drink-4381156/

Developement d'un projet pour servir notre documentation tout en restant agnostique de nos technologies. => Open source

- Créer un projet Open source
- Project Innovation

(11) Origine du nom

[Image : Comcombre] https://pixabay.com/photos/carrots-onion-cucumber-vegetables-155715/ - Origine du nom

(12) theGardener

[Image : theGardener icon]

- People

(13) theGardener: fonctionnement générale

[Image : diagramme sur l'overview sur le projet Writer -> Git Repo -> theGardener -> Reader]

(14) Génération de documentation: inexploitable

[Image : Moissoneuse batteuse] https://www.pexels.com/fr-fr/photo/cereale-charrue-clairiere-cultures-arables-163752/

Oui, c'est exaustif, à jour, précis et donne des examples Impression d'avoir atteint le but de produite une documentation fonctionnelle exaustive et à jour

(15) Génération de documentation: inexploitable

[Image : Bottes de foin] https://www.pexels.com/fr-fr/photo/agriculture-aliments-balles-de-foin-ballot-289334/

=> A quoi ressemblait la documentation ? - Des piles de scenarios techniques incomprehensibles pour un non initiés - Manque de contexte.... - Trop trop d'information

(16) Pourquoi on a du mal à écrire de la documentation ?

[Image : Personne faisant la grimace en mangeant un fruit] https://www.pexels.com/fr-fr/photo/nourriture-aliments-sain-personne-3214277/

- Hors du cycle de developpement, contexte switch, oubli de la tache
- Peu de guide, le contour est flou
- On n'en voit pas la valeur immédiate, à quoi bon
- On est souvent obliger de répeter des choses qui sont dans le code =>
    - on n'a jamais envie de se répéter
    - c'est dangereux de se répéter

(17) Pourquoi ne pas inclure l'écriture de cette documentation dans le cycle de DEV ?

[Image : diagramme sur la documentation dans le cycle de DEV]

Principales idées du projet : - le developpeur ne sort jamais de son IDE. Il écrit de la documentation en markdown au même endroit que son source. - le developpeur peut inclure des élements issus du code: - les scenarios BDD - les models et les endpoint OpenAPI - Markdown est minimaliste: impose de structurer sa pensée
- le developpeur peut avoir un rendu immédiat dans son IDE (Markdown est tres répandu dans les IDE) et il pourra le visualiser dans l'outil.

=> Developpé lors de l'EPIC Technical Documentation for Publisher

(18) Brainstroming sur la documentation technique

[Image : Réunion brainstroming sur la documentation] [Image : Post it] [Image : Sapin de noêl] https://www.pexels.com/fr-fr/photo/a-l-interieur-arbre-de-noel-cadeaux-celebration-1708601/

Liste du père noël

(19) Documentation Trust Agreement

[Image : Riz en étages] https://www.pexels.com/fr-fr/photo/agriculture-aliments-arbre-arrondissement-235648/

Nouvel accronyme

[< SLIDE] - Definir - Quand: DEV, DOD, Sprint - Comment: Text, Auto, Diagram - Où: confluence, theGardener, - Qui: DEV, EM - à Qui: Audience - Quoi: Niveau Platforme, Niveau Projet, Niveau Fonctionnalité

[> SLIDE]

(20) Documentation Trust Agreement

[Image : Potager bien rangé] https://www.pexels.com/fr-fr/photo/agriculture-aliments-arbre-arrondissement-235648/

- Engagement de l'équipe sur le contenu de la documentation que l'on peut produire. Auquel on s'engage à produire et à maintenir.

(21) Snow landscape

[Image : Neige] https://www.pexels.com/fr-fr/photo/alpin-aventure-colline-complexe-869258/

Presentation pour le SnowCamp

(22) Demo

[Image : Allumette]

-> http://backyard.corp.kelkoo.net/tools -> http://thegardener.corp.kelkoo.net

- Montrer la DTA
  - Guidelines
  - Revue rapide du contenu
- Montrer la documentation au Niveau platforme
- Montrer les guides
- Montrer la documentation du PMWS
- Montrer la documentation du LeadService
- Montrer la documentation Public du LeadService passage sur https://developers.kelkoogroup.com
- Ouvrir IntelliJ, dans PMWS
- Ouvrir le repertoire Documentation
- Montrer la relation entre l'achitecture des répertoires et des pages dans IntelliJ et dans theGardener
- Editer documentation/Features/Constraints/Overview.md
- Montrer le rendu dans IntelliJ et dans theGardener en selectionnant la branche feature/demo
- Ajouter les scenarios de selection de contraintes au niveau Profile
  - /constraints/GetConstraintsAtProfileLevel.feature
  - uniquement @documentation
- S'appuyer sur la documentation presente pour ecrire l'inclusion
- Commit push sur la branche
- Montrer le rendu sur la theGardener refraichissant simplement la page

---

Contenu propre à la presentation du SnowCamp

REX : notre cheminement à Kelkoo par rapport à la documentation de nos composants techniques

[Image : Feuilles rouges d'Alaska]

- vous pourrez en retenir des enseignements dans votre propre contexte
- les recettes présentées ne sont pas forcément adaptables partout, ce sont nos recettes pour notre contexte
- ceci dit on doit avoir des préoccupations similaires
- pour comprendre nos recettes, je vais faire un historique sur notre situation
- situation qui peut avoir de large raisonance chez vous
- libre interprétation basée sur des faits réels, je romance (image, la documentation est un plante végétale) ;)

Le contexte

[Image : Evolution d'une plante au cours du temps] https://www.pexels.com/fr-fr/photo/agriculture-centrale-croissance-feuilles-1072824/

- Kelkoo a 20 ans
- Tres nombreux services et process developpés en interne. Utilisé en interne et en externe.
- Passage du cycle en V à l'agilité il y a 12 ans.
- Passage à la pratique intensive du test, il y a 10 ans.
- Passage à la pratique intensive du BDD depuis 4 ans.