PROJET IOT (Internet des objets)

### L'IT Akademy nous a demandé de réaliser un projet IOT, nous avons donc (toute la classe) écrit sur des petits papiers des idées.

### Après avoir lu et noté la pertinence des idées, nous avons voté pour celles qui nous intéressaient le plus et ensuite essayé de créer des équipes de 4.

### J'ai personnellement choisi de participer à la création d'une gamelle pour chat connectée à son application web et mobile ainsi qu'à son site marchand.

### Dans ce projet, la première étape a été la création de l'équipe. J'ai eu la chance de travailler avec d'excellents coéquipiers :

### - Ryan, qui possède une logique impressionnante et une aisance remarquable en algorithmie,

### - Mel, qui a un goût prononcé pour le front-end et une réflexion hors du commun,

### - Enfin, Yannis, qui possède une expérience solide en programmation, un grand sens de l'humour, ainsi qu'une source de connaissances et de motivation très appréciable pour l'apprentissage.

### C:\Users\delarream\AppData\Local\Microsoft\Windows\INetCache\Content.Word\PresentationEquipe.png

### Lorsque nous nous sommes concertés pour déterminer comment le projet allait se dérouler, il leur est apparu évident que ma place serait celle de chef de projet.

### En effet, mes expériences passées m'ont fait apparaître comme un leader à leurs yeux, et ils m'ont donc confié cette responsabilité sans même me laisser d'autre choix.

### Cela m'a permis de remplir plusieurs tâches cruciales du projet, notamment :

### - Arbitrer les décisions en cas de litiges ou d'indécisions,

### - Collaborer avec l'équipe pour définir l'organisation et les objectifs (utilisation de Scrum et définition de l'objectif final),

### - Gérer les petits conflits qui peuvent survenir,

### - Organiser le groupe afin de minimiser les éventuels ralentissements,

### - Évaluer les forces et les faiblesses de chacun afin d'équilibrer la répartition du travail et de garantir au maximum que le projet puisse être mené à bien dans les délais impartis.

### La deuxième étape a été de déterminer comment nous allions réaliser une gamelle connectée pour chat intégrée à une application web. Cette phase a soulevé plusieurs questions essentielles :

1. Quel langage de programmation devons-nous utiliser ?

2. Quelle architecture convient le mieux ?

3. Quelles technologies devons-nous intégrer ?

4. Comment construire le dispositif en lui-même ?

5. Quelles sont les exigences de l'école à ce sujet ?

6. Comment pouvons-nous travailler de manière efficace en équipe ?

7. Qui sera responsable de quelles tâches ?

### Nous avons abordé chacune de ces questions en groupe, et j'ai suggéré une idée supplémentaire : la nécessité de documenter l'ensemble du processus, en créant une présentation.

### En fin de compte, nous avons pris les décisions suivantes :

1. Langages de programmation : Python pour la partie embarquée, React pour le front-end, et PHP pour le back-end.

### C:\Users\delarream\AppData\Local\Microsoft\Windows\INetCache\Content.Word\logoPython.png C:\Users\delarream\AppData\Local\Microsoft\Windows\INetCache\Content.Word\LogoSymfony.png



### Et l'interaction entre ces langages nécessitait une vue claire :

### 

### C:\Users\delarream\AppData\Local\Microsoft\Windows\INetCache\Content.Word\LangagesIotServeurClient.png

2. Architecture MVC (Modèle-Vue-Contrôleur) :

### C'est un modèle de conception logicielle qui divise une application en trois composants interconnectés : le modèle, la vue et le contrôleur;

### - Le modèle représente la logique de l'application et les données. Dans notre projet, le modèle est responsable de la gestion des données de la gamelle connectée, telles que les informations sur la nourriture, les horaires d'alimentation, le nom du chat, etc.

### - La vue est responsable de l'interface utilisateur. Dans notre cas, il s'agit de l'interface web que les utilisateurs utiliseront pour interagir avec la gamelle connectée.

### - Le contrôleur agit comme un intermédiaire entre le modèle et la vue. Il traite les demandes de l'utilisateur et met à jour le modèle en conséquence.

### Par exemple, lorsqu'un utilisateur planifie l'alimentation de son chat via l'application, le contrôleur s'assure que ces informations sont enregistrées dans le modèle et que la vue est mise à jour pour refléter les changements.

### L'architecture MVC offre une séparation claire des fichiers du code ainsi qu'une logique plus facile à entrevoir, ce qui facilite la maintenance et l'évolutivité de l'application.

3. Utilisation d'un Raspberry Pi ainsi qu'une puce pour le suivi du chat :

### Le Raspberry Pi est un petit ordinateur monocarte abordable qui peut être utilisé dans une variété d'applications, y compris les projets IoT.

### Dans notre projet, nous avons choisi d'utiliser un Raspberry Pi pour plusieurs raisons :

### - Il est compact et économe en énergie, ce qui en fait un choix idéal pour un dispositif embarqué.

### - Il dispose de ports d'entrée/sortie (GPIO) permettant de connecter des capteurs et des actionneurs.

### - Il peut exécuter une version de Python, ce qui facilite la programmation de la partie embarquée de notre gamelle connectée.

### Le Raspberry Pi sera responsable de la collecte de données à partir de capteurs sur la gamelle, comme le poids de la nourriture, et de l'envoi de ces données au serveur back-end via une connexion Internet. Cela permet aux utilisateurs d'accéder à ces informations via l'application web.

4. Nous avons élaboré plusieurs plans individuellement pour ne pas influencer les idées des autres membres, puis nous avons fusionné nos meilleures idées pour créer un prototype.



5. L'école nous avait imposé certains langages, mais nous avons pu justifier notre choix d'utiliser Python.

### Elle est également de fait notre client. (L'IT Akademy nous avait demandé de réaliser un projet IoT (Internet des objets) ainsi qu'un "Fil Rouge" qui consistait initialement en la création séparée d'une application web et d'un objet IoT.

### Cependant, nous avons négocié avec l'école la possibilité de fusionner ces deux objectifs en un seul projet, en développant la gamelle connectée ainsi que son application web associée.)

6. Nous avons adopté la méthode agile Scrum pour travailler ensemble de manière efficace tout au long de notre aventure.

La méthodologie Scrum est un cadre de travail agile qui favorise la collaboration, la transparence et l'adaptabilité dans le développement de logiciels. Dans notre projet, nous avons choisi d'adopter Scrum pour plusieurs raisons :

### - Scrum nous permet de travailler de manière itérative et incrémentale, ce qui signifie que nous développons notre projet par étapes successives et fonctionnelles.

### - Les "sprints" de Scrum sont des périodes de développement définies, généralement de deux à quatre semaines (dans notre cas, nous faisions un nouveau sprint toutes les deux semaines), au cours desquelles nous nous concentrons sur la réalisation d'objectifs spécifiques.

### - Scrum encourage la communication régulière au sein de l'équipe, ce qui est essentiel pour résoudre rapidement les problèmes et s'assurer que le projet avance sans problèmes.

### Au fil des jours, il m'est apparu évident que notre rythme et notre manière de travailler ensemble étaient encore optimisables. J'ai donc décidé de tenir une brève réunion tous les soirs, d'une durée de 15 à 20 minutes, pour résumer la journée en trois points :

# **- Qu'as-tu accompli aujourd'hui ?**

# **- Quels problèmes as-tu rencontrés ?**

# **- Qu'as-tu prévu de faire demain ?**

### Cela présentait plusieurs avantages :

### - Chacun d'entre nous savait où en étaient les autres,

### - Nous échangions sur nos problèmes, et si l'un d'entre nous avait une proposition, le problème était résolu facilement le lendemain matin, ce qui représentait un gain de temps et d'énergie,

### - Le maintien des échanges contribuait à maintenir une atmosphère positive au sein de l'équipe.

7.

Nous avons constaté que nous étions parfaitement équilibrés avec deux membres spécialisés en front-end et deux en back-end, avec Ryan et Yannis pour le back-end, et Mel et moi pour le front-end.

### En parallèle, j'ai commencé sur Canva une potentielle présentation.

### Mes collègues ont débattu sur l’utilité de tout noter, tout enregistrer, coucher sur papier ou Canva tous les détails de notre avancée.

### J’ai préféré être prudent et continuer malgré tout à mettre une quantité non exagérée de données de côté (pour le cas où).

# **LES DÉBUTS DU TRAVAIL D'ÉQUIPE**

## Nous nous sommes répartis les tâches de la manière suivante, en vue de réaliser notre projet :

## ***1.Notion*** ***:***

## Nous avons créé une section sur Notion. Notion est une plateforme de gestion de projet et de collaboration en ligne qui permet aux utilisateurs de créer, organiser et partager des informations de manière flexible.

## Elle combine des fonctionnalités de prise de notes, de gestion de tâches, de création de bases de données et de collaboration en temps réel au sein d'une seule application.



## ***2.Wireframes*** ***:***

## Les wireframes sont des schémas de conception initiale de l'interface de l'application.

## Ils nous ont permis de visualiser à quoi ressemblerait l'application avant de passer à la phase de développement.

## 



## ***3. Maquettes :***

## Les maquettes sont des représentations plus détaillées de l'interface, montrant les éléments de design, les couleurs et les emplacements des éléments de l'application.

## Elles nous ont permis d'avoir une idée précise du rendu final de l'application.

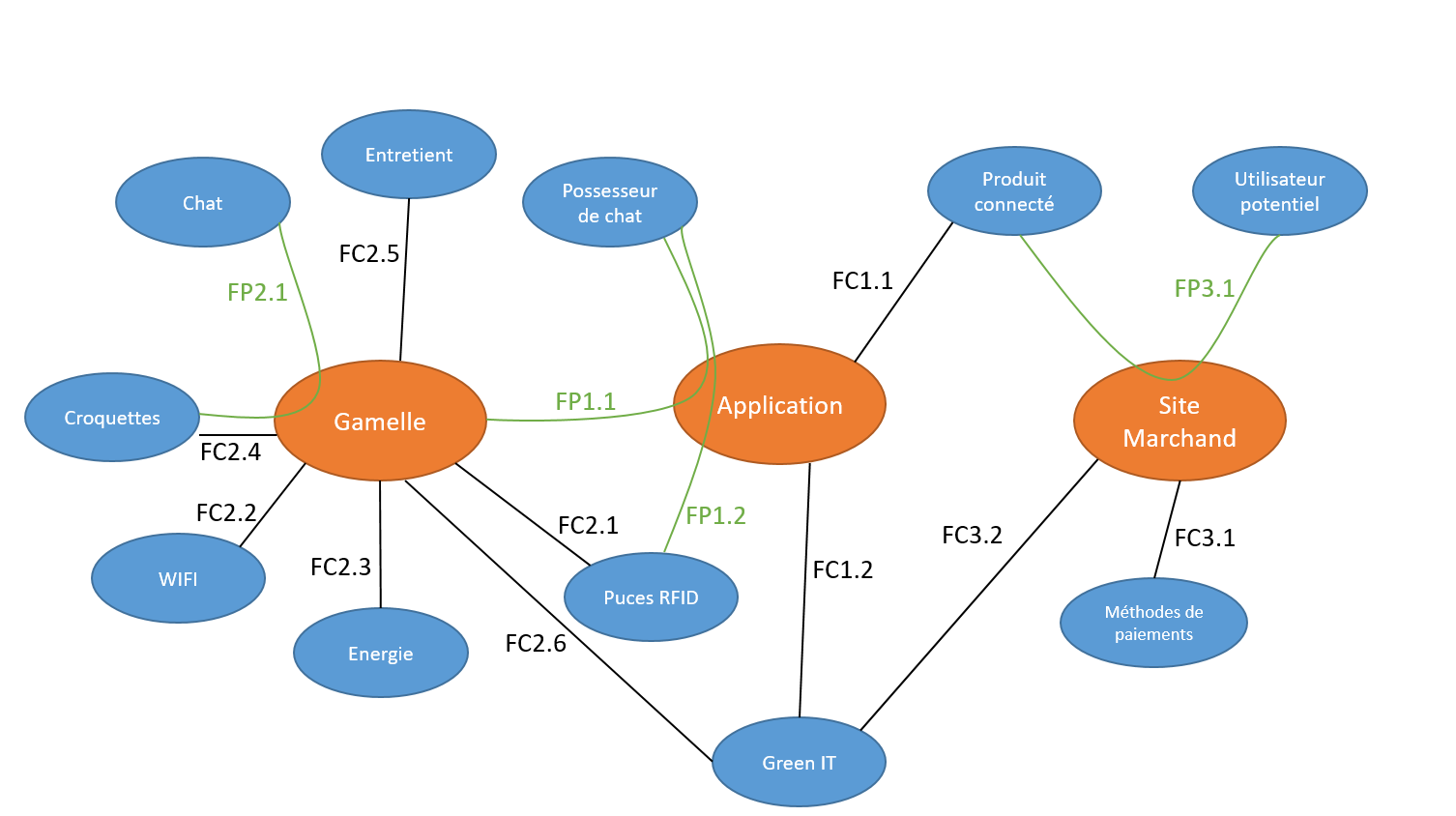
## Elles ont également soulevées une question essentielle, la charte graphique.

## Nous avons choisi en couleur principale le bleu, et en couleur secondaire l’orange.

## C:\Users\delarream\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Maquettes.png

### ***4. UML du projet :***

### Nous avons utilisé l'UML (Unified Modeling Language) pour modéliser différents aspects du projet, notamment la spécification des besoins (MOA, Méthode d'Organisation et d'Analyse, et le modèle conceptuel de données (MCD), le modèle physique de données (MPD).



### C:\Users\delarream\AppData\Local\Microsoft\Windows\INetCache\Content.Word\MCD MPD.PNG

### Ces différentes étapes de planification et de documentation ont contribué à assurer la cohérence et la clarté de notre projet tout au long de son développement.

### Nous avons ensuite créé un document se nommant "Risques et Criticité".

### C:\Users\delarream\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Risques et criticite.png

### Un document "Risque et Criticité" est un outil de gestion de projet qui vise à identifier, évaluer et classer les risques potentiels associés à un projet. Voici une explication détaillée de ce qu'est ce document :

### **1. Identification des risques :** Dans ce document, on recense tous les risques possibles qui pourraient affecter la réussite du projet. Ces risques peuvent être de différentes natures, comme des retards, des problèmes techniques, des contraintes budgétaires, des conflits internes, etc.

### **2. Évaluation des risques :** Chaque risque est ensuite évalué en termes de probabilité de survenue (du pessimiste au plus optimiste) et d'impact potentiel sur le projet en cas de réalisation. On utilise souvent une échelle de notation pour ces deux critères.

### **3. Calcul de la criticité :** Pour déterminer la criticité de chaque risque, on applique généralement une formule mathématique, qui combine les estimations pessimistes et optimistes. Cette criticité permet de hiérarchiser les risques en fonction de leur importance pour le projet.

### Pour ce faire, nous avons donc appliqué cette formule mathématique simple :



### J'explique :

### - P représente l'estimation "Pessimiste", c'est-à-dire la probabilité que le risque se matérialise de la pire manière possible.

### - R signifie "Réaliste", et il s'agit de l'estimation moyenne ou d'une évaluation basée sur une perspective pragmatique.

### - O correspond à l'estimation "Optimiste", décrivant la probabilité que le risque se matérialise de la meilleure manière possible.

### En utilisant cette formule, en prenant en compte à la fois l'avis pessimiste et optimiste, puis en ajoutant le résultat à deux estimations réalistes, nous obtenons un indicateur qui permet d'évaluer la probabilité globale d'occurrence d'un risque.

### Nous divisons ensuite ce résultat par quatre pour normaliser l'évaluation et ainsi obtenir une mesure plus représentative de la probabilité du risque.

### Cette méthode permet d'obtenir une estimation de la probabilité de manière équilibrée en tenant compte des perspectives pessimistes, réalistes et optimistes. Elle nous a donc aidés à mieux évaluer et gérer les risques dans notre projet.

### **4. Plan d'action:** Une fois les risques évalués et classés, on élabore des plans d'action pour chaque risque critique. Ces plans décrivent les mesures préventives et correctives à prendre en cas de réalisation du risque, afin de minimiser son impact sur le projet.

### **5. Suivi et mise à jour :** Le document "Risque et Criticité" n'est pas statique. Il est régulièrement mis à jour tout au long du projet pour refléter l'évolution des risques. On peut ainsi anticiper les problèmes potentiels et prendre des mesures proactives pour les gérer.

### Pour résumer, le document "Risque et Criticité" que nous avons réalisé tous ensemble est un outil de gestion de projet essentiel qui permet d'identifier, évaluer et classer les risques potentiels, tout en élaborant des plans d'action pour minimiser leur impact sur la réussite du projet.

### \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

### **Chef de projet**

### Personnellement, ce fut un moment véritablement agréable d'échange entre coéquipiers.

### J'ai ainsi pu évaluer la psychologie de mes collègues de travail, ce qui m'a beaucoup aidé dans la gestion du projet.

### J'ai pu voir où étaient les points bloquants pour eux, où étaient les incertitudes/doutes, et où se situait leurs points forts, là où ils se sentaient le mieux ou ce qu'ils étaient sûrs de savoir/pouvoir faire.

### C'est en partie grâce à cet échange que j'ai pu établir une stratégie à court et moyen terme pour atteindre notre objectif.

### \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

### Nous avons donc créé le GitHub du projet IoT, ajouté les fichiers `**. Git ignore**` et `**README.md**`, ainsi que configuré le fichier `**package.json**` avec certaines dépendances, comme par exemple :

### "**axios": "^1.4.0**" ===> Bibliothèque HTTP pour les requêtes réseau.

### "**dotenv": "^16.3.1**" ===> Gestion des variables d'environnement.

### "**expo": "^48.0.9**", ===> Permet de voir via expo go sur notre tel mobile le résultat du code

### "**path": "^0.12.7**" ===> Utilitaire pour la gestion des chemins de fichiers.

### "**react": "^18.2.0**" ===> Langage React avec sa dernière version

### Ensuite, nous avons commencé à coder nos premières lignes dans `App.js`, qui est la pierre angulaire de notre application.

### C:\Users\delarream\AppData\Local\Microsoft\Windows\INetCache\Content.Word\AppJs.png

### Nous avons également créé plusieurs règles en suivant au mieux ce qui nous a été enseigné, notamment :

### ***- DevOps :*** Le DevOps est une méthode de développement logiciel qui vise à améliorer la collaboration et la communication entre les équipes de développement (Dev) et les équipes d'exploitation (Ops).

### Cette approche met l'accent sur l'automatisation, la mesure et le partage des informations pour atteindre un déploiement plus rapide et plus fiable des logiciels.

### ***- CI/CD :***

### La CI (Intégration Continue) est une pratique de développement où les développeurs intègrent fréquemment leur code dans un référentiel partagé.

### Le CD (Livraison Continue) va plus loin en automatisant la mise en production du code intégré et testé.

### Une fois que le code a passé les tests automatisés, il est prêt à être déployé en production sans étapes manuelles supplémentaires.

### ***- Création de branches localisées*** en fonction du travail à faire, selon la logique suivante :

### - Actions à faire, par exemple => **feature**: Ajout d’une nouvelle fonctionnalité

### **Bugfix**: Correction d’un bug

### 

### - Travail ciblé, par exemple => **ModificationsBoutons**, **AjoutHeader**

### 

### Cela donne un résultat qui ressemble à :

### **- feature/ModificationsBoutons**

### **- bugfix/header**

### Avant de faire une pull-request, il fallait que le/la coéquipier/ère accepte le push de l'autre.

### Nous avons donc ajouté la règle de faire une revue de code ensemble (code review) et de vérifier si tout était exactement comme il fallait, tant au niveau de la propreté du code, de l'efficacité énergétique (green IT), de la sécurité, des commentaires insuffisants, etc.

### Et c'est seulement une fois validé que nous pouvions pull le code et continuer à avancer.

### De plus, nous nous sommes également efforcé de ne pas toucher au code de l'autre.

### J'ai essayé de pratiquer le Scrum en répartissant le travail de la manière la plus efficace possible au sein de l'équipe,

### - en déléguant certaines tâches,

### - en définissant l'objectif à atteindre et

### - en maintenant les sprints agiles des deux équipes (l'une front, l'autre back).

### Je tiens à préciser que je ne suis pas Scrum Master, mais plutôt, pour le moment, un Scrum But (dit autrement, un débutant).

### Yannis s'occupait de la partie "industrielle" de l'objet,

### Ryan a paramétré le Raspberry pour qu'il fasse le lien machine/back,

### Mel a organisé, en tant que frontLead, les composants de l'application et ajouté les taches à faire sur Notion, et moi, je me suis concentré sur les demandes des clients (restrictions budgétaires pour l'objet, vérification des obligations linguistiques, vérification des contraintes, etc.) et j'ai commencé à réaliser quelques composants obligatoires, comme le header, les pages de connexion ectect.



### Il y avait plusieurs détails à prendre en compte, le premier étant l'emplacement (en haut), le deuxième étant la couleur.

### Nous avons donc appliqué notre charte graphique.

### Les couleurs ont été ajoutées en tant que variables dans le SCSS, que nous avons nommées "**PrimaryColor**" (pour le bleu) et "**SecondaryColor"** (pour l'orange).

### Si le bouton du header est cliqué et/ou actif, alors il devient orange. Sinon, son état "naturel" est le bleu.

### La cloche des notifications, si elle est cliquée, déroule une liste de notifications et s'ouvre lentement de la gauche vers le centre de l'écran.

### Le bouton burger, lui, doit afficher un menu avec à l'intérieur des textes cliquables qui redirigent vers la page voulue.

**Composant Header.js**

### C:\Users\delarream\AppData\Local\Microsoft\Windows\INetCache\Content.Word\header.js.png

**Header.scss**



### Nous avons avancé ainsi pendant plusieurs jours, en nous adaptant aux différentes nécessités du moment.

### Les réunions devenaient de plus en plus précises et habituelles, le dialogue entre nous s'est installé de manière naturelle, et la progression était très satisfaisante.

### **C:\Users\delarream\AppData\Local\Microsoft\Windows\INetCache\Content.Word\avance notion mel marc.png**

### **Mais un jour, un mardi matin (pour être précis),**

### **tout fut chamboulé.**

### L'école nous a envoyé un mail pour nous demander une .... Oui oui, une présentation !

### Et là, je ne vous cache pas que je me suis retenu au maximum, mais la phrase

### **"AHAH Je vous l'avais bien dit !!"** a été sortie.

### Et quel bonheur d'avoir préparé, noté et illustré cette présentation Canva tout du long du projet, à chaque nouvelle image, chaque nouvelle étape, tout était mis de côté pour le cas où.

### Ce qui s'est avéré être au final, ce que l'on appel dans le jargon geek, un "Preshot de dingue" (PDD pour les intimes).

### Car il faut le savoir, la date limite pour la présentation était de seulement deux jours.

### Heureusement pour nous, environ 80% du travail avait déjà été réalisé, il nous restait principalement à définir le contenu verbal, à décider qui prononcerait quel discours, et à ajouter quelques slides supplémentaires pour plus de détails.

### 

### **Exemple de slides ajoutées:**

### C:\Users\delarream\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Etude marche Strategie marketing Marketing digital.png

### Pour compléter notre présentation, nous avons donc ajouté une étude de marché, la stratégie marketing de notre marque ***« Purrpaws »*** et quelques éléments supplémentaires.

### **Et le Canva:**

### C:\Users\delarream\AppData\Local\Microsoft\Windows\INetCache\Content.Word\presentation canva.png

### Nous avons fait notre présentation et tout s’est bien passé.

### Une fois cette épreuve achevée, j'ai repris la suite de notre Notion c'est à dire, le bouton des notifications.

### C:\Users\delarream\AppData\Local\Microsoft\Windows\INetCache\Content.Word\componentNotificationButton.png

### Nous nous sommes rendu compte qu’il fallait que l’onglet des notifications se ferme si on clique à côté dudit onglet, (**handleClickOutside**), et en mettant un léger délai a l’ouverture/fermeture, cela rend le tout plus interactif, l’**UX** est beaucoup plus agréable donc, idée conservée (**UX** signifie Expérience utilisateur).

### Puis le burger bouton, les pages de connexions ectect...

### Et arrive donc, l'heure du Stage en entreprise !

### \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

### RESUME DES CHOSES A FAIRE ET MONTRER

### 

### Images => mise a jour fréquentes

### Gestion projet => fait

### Front => fait

### Codes => fait en front, pas en back

### Back => voir cours ou projet

### Securite => voir les cours RootMe, owasp

### Reseaux => voir les cours