Startup à taille humaine.  
Ambiance très familiale.

Crise laitière.  
**3,5 millions de bovins laitiers en France dans 92000 exploitations**.  
**Licences dans les écoles véterinaires** (Toulouse, Maison Alfort, Lyon et Nantes). Logique commerciale Microsoft.

Ordinateur sous Linux.  
**Visual Studio Code (plutôt que PHPStorm).** Microsoft . DEVELOPPER  
**DataGrip** (plutôt que PHPMyAdmin). JetBrains. DEVELOPPER **(oracle sql server, mySQL…)**  
Google Chrome (et sa fameuse **console**). DEVELOPPER  
GIT (on en repara dans un instant).  
Discord (pourquoi plutôt que Slack). Gamers  
GanttProject (pour les plannings).

Git créé en 2005 par Linus Torvalds.  
Git permet le travail collaboratif ainsi que le travail à distance.  
**Très simple à utiliser : git commit -am "message", git push**  
**Très fiable contrairement à SVN** (utilisé à l’ENI pendant la formation).

Pour information, Milkup est le nom du chat de Gaëtan.  
PHP 7 est très performant (beaucoup plus que les versions précédentes).  
MySQL (possibilité de migrer très simplement sue MariaDB).  
JavaScript : peut être le langage le plus moderne et le plus flexible (en tout cas l’un des plus demandés sur le marché du travail).  
jQuery, Bootstrap, Moment, DataTables.  
Ajax (Ajax n’est pas à proprement parler une librairie).  
Mais aussi des librairies propriétaires comme Vtext pour l’internationalisation et d’autres pour les graphiques statistiques, pour les SMS, pour les formulaires.

Une multitude de frameworks libres. (Synfony, Laravel, Zend, Cake, CodeIigniter, Yii…  
Choix du maison : **Connaissance parfaite du framework : essentiellement par le créateur du framework.  
Pas de doc en ligne, pas de bouquin, pas de communauté.   
Pas de plug-in IDE.**

Réformer une vache n’est pas un choix facile pour un éleveur.  
**En fonction des anomalies de la vache (mamites, boiteries… une quinzaine), de son taux de production de lait, de son âge, de la qualité globale de son lait (taux buryteux, taux protéique).**  
Pouvoir voir instantanement (statistiques) si la vache n’est plus rentable.

Pas trop de choix : logiciel existant, bdd existante.  
Module indépendant pouvant être vendu sous forme d’option).  
ORM pour faciliter la maintenance.

Pas facile d’établir un planning pour une durée si courte sans connaitre ni le framework ni l’ORM

La base de données est en MySQL.  
Elle comporte autant de tables que d’objets : 98 tables actuellement.  
Comme c’est la mode aujourd’hui, on utilise un ORM (couche d’abstraction à la base de données). Pas DOCTRINE.  
Pas de docs (3 pages très sommaires)

Institut de l’élevage : Idele (web service soap)  
**Données des contrôles laitiers. Utiles (taux protéiques, taux butéryques…)** pour beaucoups de calculs dont les miens.

98 tables sans clés étrangères (d’après Simon, sur un moteur de BDD InnoDB, les clés étrangères ralentissent les requêtes, il vaut mieux gérer ça dans l’ORM).  
J’ai ajouté les liaisons sur ce schéma pour plus de compréhension.   
J’utilise 6 tables de cette BDD.  
La table principale (à tout le logiciel) : cow.  
**Les tables anomaly et cow\_lactation auront des liaisons de types many to one.  
Et il y a beaucoup de calculs de dates sur ces deux tables.**Heureusement, l’ORM gère les fonctions DATEDIFF, CURDATE, DATE\_ADD, DATE\_SUB…

Montrer l’efficacité de l’ORM :  
Ça ne ressemble à rien de connu, même pas à Doctrine.  
Très efficace et agréable pour les jeunes développeurs qui sont un peu hostile à SQL.

Voici les principales requêtes que j’ai réalisé

Oui, pour quelqu’un comme moi qui aime coder et qui aime le SQL (aussi) mais ne se sent pas l’âme d’un graphiste, coder, c’est du bonheur.

Chaque objet se trouve dans le répertoire objects (BO)  
Le métier se trouve dans views/page (BLL)  
Pour ma part, j’utilise les pages Scrap

Exemple de code.  
**un formateur que j’ai beaucoup apprécié m’a qualifié un jour de « vieux développeur PHP autodidacte » : c’est là que je m’éclate**.  
Pour ce genre de développement, rien n’est impossible : toute l’aide imaginable est disponible en ligne, ne serait-ce que dans la doc PHP.

**Les variables dynamiques**.  
Explication du code.  
N’existe (à ma connaissance) **qu’en PHP et en Python (pas en JAVA ni en C#).**  
Simon et Florian ne connaissaient pas.  
J’ai souvent utilisé par le passé.  
Générér des variables à la volée (dans une boucle).

Milkup n’est pour le moment qu’en français, mais…  
Utilisation de Vtext. **Si le Vtext demandé n’existe pas, il est créé**, et affiché dans la vue avec le nom du Vtext.  
Une interface permet de modifier ou d’ajouter des traductions.   
Si les constantes sont nommées correctement, pas de problème pour la lisibilité du code.

Une fois le JavaScript écrit pour l’affichage de la page, le reste est enfantin (HTML/Bootstrap).

**Tableaux** :  
On envoie du HTML  
On récupère dans la colonne corespondant (on définit uniquement les en-têtes, le framework gère le reste).

**Hors tableau** :  
On envoie du HTML  
On récupère dans la vue avec du JavaScript (si, c’est du JavaScript).

On obtient des vues dans un pur style Bootstrap.  
Bootstrap +/- (clair, mais comme tout le monde)  
Pas de rechargement de page grace à Ajax.

Ajax permet un site « ONE PAGE », utile pour une prochaine intégration Cordova.

La performance est une priorité pour Milkup.  
Pour cela , la Console de Google Chrome (l’un des meilleurs amis du développeur web).  
Onglet Network pour tester les requêtes.  
Onglet Performance pour tester les ressources des boucles et autres calculs.   
Ces tests on permis d’optimiser (par 2) le chargement de la page initiale en réglant des problèmes de boucles.