# Conclusion

## Etat d’avancement de l’application – Aboutie ?

1. Amélioration possibles

## Tests effectués 🡪 Tableau de tests à mettre

1. Répartition des tâches
2. Conclusion
3. Etat de l’application :

L’application est aboutie et finie. Tous les cas d’utilisations analysés ont été implémentés et sont fonctionnels.

Au lancement de l’application, 4 logiciels seront créés par défaut au lancement du script SQL de la base de données. Ceux-ci sont : Windows, Claroline, Nutrilog, GLOBAL. Le logiciel GLOBAL ne sera vu au niveau de l’application, il est uniquement utilisé pour l’ajout d’un administrateur uniquement au niveau de la base de données.

1. Amélioration possible

Au niveau des améliorations, du côté de la base de données, certaines contraintes de syntaxe pourraient être ajoutées. Syntaxe du mail d’un utilisateur : ‘[A-Za-z0-9.\_%-]+@[A-Za-z0-9.-]+[.][A-Za-z]{2,4}', syntaxe d’un matricule si définit au préalable. Une gestion des casses au niveau des noms avec l’utilisation de la fonction lower().  
De plus, il faudrait alléger certains traitements effectués par la base de données et les mettre au niveau du serveur. Notamment, lors de l’ajout/suppression des logiciels à un profil qui peut être trop lourd si trop de logiciels dans la base de données.

Au niveau du Back-End, on pourrait ajouter un test au cas où deux étudiants possèderaient le même nom et le même prénom. Avec l’application, ceux-ci auraient le même login, ce qui n’est pas accepté par la base de données, un login est unique, un utilisateur sera donc perdu dans ce cas-si. Le fichier CSV qu’on va importer ne pourra comporter des erreurs et devra traiter le cas des doublons.

En terme de sécurité, améliorer les vérifications des données reçues par le Front-End qui sont, pour le moment, gérées en grande partie par Sequelize.

Au niveau du client, une fonctionnalité qu’on pourrait ajouter mais futile, est l’impression du document de l’étudiant directement via un bouton d’impression sur le site ou une ouverture automatique de ce fichier sous format Pdf dans un nouvel onglet.

Pour les exports des données par logiciel, on est bloqué aux trois logiciels par défaut. Si, par la suite l’administrateur ajoute d’autres logiciels, il devra ajouter par lui-même le code correspondant à cet export. Ce n’est pas généré automatiquement.

Au niveau des formulaires, lors des envoies des informations, si l’envoie échoue, les données entrées dans le formulaire sont supprimées. Il faudrait garder les données déjà entrées pour une meilleure ergonomie.

Enfin, lors d’un rafraichissement d’une page, il faudrait garder l’endroit où l’utilisateur naviguait. Pour le moment, l’utilisateur est renvoyé vers la page d’accueil.

1. Tests
2. Répartition des tâches :

Front-End :

* Html : Olivier - Maxime
* CSS : Maxime
* Javascript : Maxime – Olivier

Back-End :

* Nodes.js : Corentin, Cédric

Base de données :

* Maxime – Olivier

Rapports, analyse :

* Olivier, Cédric

1. Conclusion :

Pour conclure, au niveau des outils que nous avions pensé utiliser, un seul n’a pas été jugé nécessaire pour le bon suivi du projet : Trello.

Suite à l’analyse du projet, on a pu assigner de manière claire, toutes les différentes tâches que chacun devait réalisées.   
 A la place, nous avons organisé des réunions chaque matin avec les différentes tâches qui allaient être réalisées pendant la journée pour chaque personne et une réunion avant de partir, pour les tâches à terminer pour le lendemain.  
On n’a eu peu, voir aucun conflits au niveau du code grâce aux contenus en permanence mis à jour sur Github et un bon suivi de celui-ci par chaque membres du groupe.

Au niveau des améliorations, les ajouts proposés sont pour la plupart futiles