Point central avant de détailler ce que les profs nous ont ressorti du projet.

Pour l’instant il y a deux choses cruciales à implémenter. Le risque de taux (taux d’intérêt variable dans le temps) n’est pas à couvrir. La première chose à implémenter est le test du P&L que j’appelerai ici le **ForwardTest**. Il faut à tout prix que l’on ai un forward test du même type que celui du projet .NET vraiment c’est le point névralgique du projet et celui sur lequel on sera principalement noté le jour du rendu. Il faudrait aussi faire des backward test. En fait le pricer C++ est vraiment la fusion des deux projets. La partie web est exclusivement la partie **« Vue »** et ne fais aucun calcul, elle affiche simplement des valeurs retrouvés par le pricer avec comme entrée seul et uniquement le cours des actifs et des taux d’intérêt.

La deuxième chose à implémenter est le risque de change, avec des taux fixes rentrés à la main il nous faut le prix d’un call quanto, tout ça sera effectué dans **QuantoCall**.

Bien à présent, parlons de ce que les profs nous ont dit sur le rendu :

* Page 3 erreur sur la date du 11/03/2014 qui est plutôt 11/04/2014 dans le calcul du flux final
* Etre plus précis lorsque l’on parle des « performances »
* Erreur à la page 4 dans les performances semestrielles d’une action vous évoquiez le risque du taux de change. Le risque du taux de change est présent pour l’investisseur !!! Or à la fin vous dites qu’il n’y en a pas.
* Le pseudo-code est complètement à revoir. Dès le départ les INPUT et OUPUT doivent bien être identifié, celui du cours des actions publiés les 11, 14 et 15 avril 2014 doit être un seul et unique input, ça ne doit pas être un calcul. L’algorithme est trop compliqué, il faut que ça soit un PSEUDO-CODE, les fonctions doivent avoir des noms précis et les mots clés comme **for, while, if then else** doivent apparaître en gras.
* PARTIE RISQUE :
  + Comme on avait dit de le faire, il faut séparé les risques que l’on souhaite géré ou que l’on ne souhaite pas géré. Ou au moins ajouter à la fin un tableau récapitulatif pour dire les risques que l’on gère que l’on ne gère pas et les risques gérés dans une extension (exemple risque des taux)
  + La partie risque n’est pas assez détaillé il faut ajouter des exemples et/ou des schémas illustratifs pour que l’on comprenne les risques.
  + Le risque de marché s’appelle le RISQUE DE LIQUIDITE !!! Voir le Hull, chacun des risques y est exprimé. (PS : je vous conseille vraiment d’aller le voir, ça explique tout)
* LA GRANDE PARTIE MANQUANTE , LA PARTIE MATHEMATIQUES :
  + LE PEPS et un ENSIMAG regroupe trois catégories :
    - FINANCE (Analyse des risques)
    - Mathématiques(La partie manquante)
    - Informatique(L’analyse technique)

Le problème c’est qu’il n’y a pas de lien entre l’analyse financière et l’analyse technique. Il manque l’analyse mathématique qui en est la jointure. Par exemple lorsque vous parlez du risque liés aux actions, vous mentionnez la couverture par les grecques(ici le delta) mais vous n’expliquez pas ce que c’est et d’où cela sort. Il faut réellement indiquer avec calcul mathématiques à l’appui le rôle mathématique et financier du delta.

De même pour le taux de change (qui n’est pas indiqué dans l’analyse technique), il faut indiquer comment on couvre se risque avec le modèle mathématiques à l’appui, pensez au Call Quanto et la démonstration mathématiques !

Pour le risque de taux également, il faut plus développer et donner des exemples concrets, pour le modèle mathématique on utilisera notre modèle stochastique de Vasicek ! Donc il faudra l’expliciter. ( Ce risque fait partie des risques d’extension dans le tableau récapitulatif mais doit absolument être identifié)

Pour le risque de liquidité, future ou option. Il faut détailler la formule mathématiques qui les calcule.

* Bien le cahier des charges et l’architecture sont bien selon le prof attention cependant à ne pas copier exactement le projet c++ mais plutôt s’en inspirer. Car par exemple la calibration des paramètres doit se faire dans une classe à part et non pas être pris en tant que paramètre des fonctions. **Comme on l’a mentionné dans notre rapport il y a une classe** CorrelationCalibration **qu’on aurait dû plutôt appeler ParameterCalibration qui, à partir d’une matrice path, calcule tous** l**es paramètres dont nous avons besoins comme la volatilité et la matrice de corrélation.** Plus tard on en aura également besoin pour la calcule des paramètres de Vasicek.

Dans l’ordre d’étape des choses à faire.

* Implémenter le backwardtest et le forwardtest sur des options vanilles :Pour cela il faut gérer à tout prix **la notion de temps**, pour l’instant je n’ai pas d’idée de comment gérer ça. Difficulté 5/5
* Implémenter la classe ParameterCalibration qui nous permettra de calibrer nos paramètres et dans tous les cas cette classe sera utile au backward et forwardtest. Difficulté : 2/5
* Implémenter la classe QuantoCall qui nous premettra donc de tester nos risques de changes. Difficulté 3/5
* Modifier le rapport, pour ajouter ce que l’on a dit, avec vraiment tout ce qu’on a dit. Difficulté 1/5
* MOINS PRIORITAIRES
* Faire des extensions : VasicekModel, gérer dans delta et P&L la gestion du paramètre d’actualisation. Et donc également ajouté la calibration des paramètres de Vasicek. Difficulté 3/5
* Faire le parallélisme de montecarlo avec Open MP, en reprenant le TP2 ou TP3 ça se fait d’un coup donc Difficulté 1/5