Table des matières

[1. Enoncé du projet 2](#_Toc512953956)

[1.1. Questions concernant l’énoncé 2](#_Toc512953957)

[2. Planning des réunions 3](#_Toc512953958)

[3. Tableau des erreurs et consignes 7](#_Toc512953959)

[4. Diagrammes et schémas 9](#_Toc512953960)

[4.1. Axe fonctionnel 9](#_Toc512953961)

[4.1.1. Diagramme de contexte statique 9](#_Toc512953962)

[4.1.2. Diagrammes des cas d’utilisation 9](#_Toc512953963)

[4.1.2.1. Cuisinier amateur 10](#_Toc512953964)

[4.1.2.2. Voisin 10](#_Toc512953965)

[4.1.2.3. Administrateur 11](#_Toc512953966)

[4.1.2.4. Global 11](#_Toc512953967)

[4.1.3. Diagrammes de séquence système 12](#_Toc512953968)

[4.1.3.1. Un utilisateur s’enregistre sur le site 12](#_Toc512953969)

[4.1.3.2. Cuisinier amateur encode une nouvelle recette 15](#_Toc512953970)

[4.1.3.3. Un voisin commande un plat 17](#_Toc512953971)

[4.1.3.4. Un administrateur gère les utilisateurs 18](#_Toc512953972)

[4.2. Axe statique 20](#_Toc512953973)

[4.2.1. Diagrammes de classes partielles 21](#_Toc512953974)

[4.2.2. Diagramme de classes 23](#_Toc512953975)

[4.3. Axe dynamique 24](#_Toc512953976)

[4.3.1. Diagramme de séquence analytique 24](#_Toc512953977)

[4.4. Schéma conceptuel 24](#_Toc512953978)

[5. Conclusions 25](#_Toc512953979)

[5.1. Conclusion de Bisconti Flavian 25](#_Toc512953980)

[5.2. Conclusion de Pire Nathan 25](#_Toc512953981)

[5.3. Conclusion de Quentin Zaretti 25](#_Toc512953982)

[5.4. Conclusion du groupe 25](#_Toc512953983)

# Enoncé du projet

Le travail porte sur la gestion d’une plateforme de rencontre entre cuisiniers amateurs et voisinage (proches, amis, voisin, …). L’objectif de l’application est d’automatiser un maximum de tâches.

* Chaque mois, la plateforme propose un thème (asiatique, italien, poisson, ...)
* Les cuisiniers amateurs s’inscrivent sur la plateforme et proposent un ou plusieurs plats correspondant au thème et appartenant à une certaine catégorie (entrées, dessert, …)
* Les voisins intéressés commandent les plats en précisant le nombre et le jour désirés
* La commande est confirmée, ainsi que le jour, l’heure et le lieu de livraison (domicile du cuisinier) et est transmise au cuisinier
* Le jour de l’enlèvement des plats, le cuisinier doit signaler si le voisin a bien emporté son repas.
* Après l’évènement, la plateforme permet aux voisins de laisser des appréciations ou commentaires sur les plats qu’ils ont dégustés.
* Le prix payé pour chaque plat est fonction du prix des matières premières
* Les meilleures recettes du mois seront éditées sur la plateforme

## Questions concernant l’énoncé

* Serait-il possible d’implémenter au site des méthodes de payement externe au site ? Ex. : la plateforme Paypal.
* Quelle méthode mettre en place pour la gestion des commentaires ?
* Afin de créer et gérer la base de données du site Web, utiliserons-nous *Entity Framework* ou l’établirons-nous nous-mêmes ?
* Comment s’assurer que les plats proposés correspondent au thème ?
* Quelles sont les limites à mettre en place au niveau du prix des plats ?
* Comment, lors de l’authentification, connaitre l’identité de l’utilisateur ? (Cook, Neighbor ou Admin ?)

# Planning des réunions

|  |  |  |  |
| --- | --- | --- | --- |
| Date et lieu | Objectifs | Répartitions des tâches | Problèmes et solutions |
| Le 07/03/18  à 10 :15  00h15  Ecole Condorcet   |  |  |  | | --- | --- | --- | | F | Q | N | | Définir des moyens de répartition des tâches et de contact | Flavian –  Définir la structure et la mise en page du dossier  Quentin –  Créer une page Web *Trello* pour le projet | Rester en contact, se répartir les tâches et se tenir à jour du travail personnel –  Création d’une page Web *Trello* (Site permettant à des groupes de travail de rester en contact et de rester à jour sur l’avancement d’un projet et de s’organiser plus efficacement) |
| Le 08/03/18  à 13h00  00h20  Ecole Condorcet   |  |  |  | | --- | --- | --- | | F | Q | N | | Mettre en commun l’avancement et présentation du squelette du dossier | Tous les membres –  Relire l’énoncé et mettre en commun nos questions |  |
| Le 12/03/18  à 12h20  00h05  Ecole Condorcet   |  |  |  | | --- | --- | --- | | F | Q | N | | Organiser des réunions pour l’établissement des diagrammes de contexte statique et des cas d’utilisation | Tous les membres –  Etre présents durant le temps de midi pour travailler en groupe le 13, 14 et 15/03/18 |  |
| Le 13/03/18  à 12h40  00h50  Ecole Condorcet   |  |  |  | | --- | --- | --- | | F | Q | N | | Etablir le diagramme de contexte statique et des cas d’utilisation  Idée lors de la programmation du site : Mettre en place une page *Github* pour le partage et la mise en commun du code | Flavian et Nathan –  Définir les utilisateurs susceptibles d’interagir avec l’application ainsi que les cas d’utilisation  Nathan –  Retranscrire les éléments sur *StarUML*  Flavian –  Prendre note des évènements pour le dossier. Corriger l’orthographe des éléments des diagrammes et l’esthétisme des liens | Utiliser adéquatement les cardinalités pour le diagramme de contexte statique –  Effectuer une comparaison avec les diagrammes des exercices corrigés au cours  Traduire les éléments des diagrammes en Anglais –  Utiliser le traducteur en ligne *Linguee* |
| Le 14/03/18  à 9h30  00h45  Ecole Condorcet   |  |  |  | | --- | --- | --- | | F | Q | N | | Obtenir l’opinion du professeur sur les diagrammes et apporter les corrections éventuelles | Tous les membres –  Mettre à jour les diagrammes et développer de nouvelles fonctionnalités  Flavian –  Créer un diagramme spécifique à l’acteur « Amateur cook », « Neighbor » et « Administrator » | Diagramme des cas d’utilisation trop complexe –  Réaliser des sous-diagrammes par rapport aux acteurs et ajouter l’acteur « Guest » |
| Le 19/03/18  à 11h50  00h40  Ecole Condorcet   |  |  |  | | --- | --- | --- | | F | Q | N | | Remettre en commun le dossier et les corrections apportées au diagramme de contexte statique et des cas d’utilisation. Ainsi que le prototype d’un diagramme de séquence | Chaque membre –  Etablir les diagrammes de séquence du usecase « Login system » ensemble et choisir un usecase chacun dont les trois scénarios devront être écris et représentés en diagramme de séquence  Flavian –  Prendre note des corrections mentionnées par le professeur et mettre à jour le dossier | Trouver un créneau dans notre horaire pour établir en groupe les diagrammes de séquence du usecase « Login System », se répartir un usecase pour établir les diagrammes de séquence personnels et définir les priorités –  Le mardi 20/03/18 durant le temps de midi |
| Le 20/03/18  à 12h30  01h00  Ecole Condorcet   |  |  |  | | --- | --- | --- | | F | Q | N | | Améliorer les diagrammes des cas d’utilisation avec les nouvelles relations vues au cours et choisir un usecase par membre pour la réalisation des diagrammes de séquence | Tous les membres –  Améliorer et corriger les diagrammes des cas d’utilisation  Flavian –  Mettre à jour les diagrammes dans le dossier |  |
| Le 26/03/18  à 10h30  00h20  Ecole Condorcet   |  |  |  | | --- | --- | --- | | F | Q | N | | Mettre en commun de tous les diagrammes de séquence personnels | Tous les membres –  Ajouter les diagrammes de séquence et la description du scénario en Français dans le dossier  Flavian –  Mettre à jour les diagrammes dans le dossier | Difficultés personnelles à la réalisation des diagrammes –  Travailler en groupe sur les diagrammes concernés le mercredi 28 mars lors des heures de pause  Format de création des diagrammes de séquence différents selon le membre –  Trouver un format commun pour harmoniser la présentation des diagrammes dans le dossier |
| Le 28/03/18  à 10h15  2h15  Ecole Condorcet   |  |  |  | | --- | --- | --- | | F | Q | N | | Mettre à jour les diagrammes des cas d’utilisation, définir les diagrammes de séquence du usecase « Login system » et déterminer les priorités | Tous les membres –  Créer les diagrammes de séquence du usecase « Login system », répartir les priorités et développer les diagrammes des cas d’utilisation avec la fonctionnalité de gestion d’un agenda | Implémenter un système de gestion et de consultation d’un agenda de manière cohérente |
| Le 11/04/18  à 13h30  01h30  Canal *Discord*   |  |  |  | | --- | --- | --- | | F | Q | N | | Définir le diagramme de classes | Tous les membres –  Définir les classes, leurs attributs, méthodes et liens  Nathan –  Retranscrire le diagramme en UML  Créer la page *Github* pour le partage de code  Flavian –  Prendre des notes pour le dossier | Trouver un système de communication autorisant le partage d’écrans –  Utilisation du programme *Discord* |
| Le 16/04/18  à 11h00  01h30  Ecole Condorcet   |  |  |  | | --- | --- | --- | | F | Q | N | | Définir les itérations, priorités, écrans et diagrammes de classes partielles | Flavian et Nathan –  Définir les itérations et leur priorité  Imaginer les différents écrans afin d’établir les classes et leurs attributs  Flavian –  Prendre des notes pour le dossier  Retranscrire les diagrammes de classes partielles en UML  Nathan –  Retranscrire les écrans en WPF | Définir en premier les diagrammes de classes partielles en fonction des itérations au lieu d’un diagramme de classe global |
| Le 23/04/18  à 10h15  00h20  Ecole Condorcet   |  |  |  | | --- | --- | --- | | F | Q | N | | Ajouter les cardinalités et confirmer les liens entre les classes | Tous les membres –  Se mettre d’accord au niveau des associations entre les classes |  |
| Le 26/04/18  à 13h50  01h40  Ecole Condorcet | Valider le diagramme de classes et répartir les tâches de programmation | Tous les membres –  Améliorer la gestion des commentaires, du catalogue et de l’agenda  Nommer les liens  Flavian –  Classes liées à la création des recettes   * « Cook » * « Ingredient » * « Recipe »   + Page de connexion  Nathan –  Classes liées à la commande   * « Recette » * « Order » * « Schedule » * « Comment »   Quentin –  Classes liées à la gestion des utilisateurs   * « Neighbor » * « User » * « Admin » * « Catalogue » | Répartir les tâches de programmation de manière cohérente et aisément réalisable –  Création de classes partielles pour que chaque membre concerné écrive sa portion de code pour la classe |
| Le 30/04/18  à 12h10  00h20  Ecole Condorcet   |  |  |  | | --- | --- | --- | | F | Q | N | | Mettre en commun la correction des diagrammes de classes partielles | Tous les membres –  Etablir chacun un diagramme de séquence analytique |  |
| Le 07/05/18  à 10h30  02h00  Ecole Condorcet   |  |  |  | | --- | --- | --- | | F | Q | N | | Mettre en commun l’avancement dans la programmation de l’application | Tous les membres –  Progresser dans la conception des pages et des classes | Envoyer des tableaux et listes de string d’une vue à un contrôleur |

# Tableau des erreurs et consignes

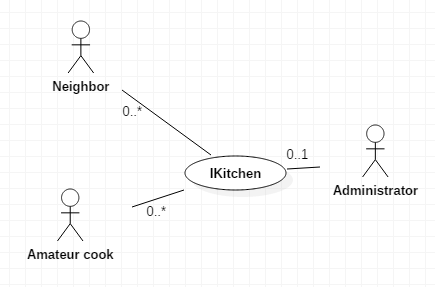
|  |  |  |
| --- | --- | --- |
| Date | Consignes | Corrections éventuelles |
| Mercredi  14/03/18 | Définir le diagramme de contexte statique et des cas d’utilisation | Diagramme contexte statique –  Supprimer les acteurs internes au site, définir un nom d’application plus compréhensible, correction des cardinalités en fonction du nombre d’utilisations potentielles et simultanées de l’application (acteurs connectés en même temps)  Diagramme des cas d’utilisation –  Ajout de nouvelles fonctionnalités, faciliter la lecture du diagramme (point de vue du client) par la création d’un diagramme par acteur |
| Lundi  26/03/18 | Améliorer le diagramme des cas d’utilisation avec les nouvelles liaisons vues au cours  Etablir un diagramme de séquence (les trois scénarios pour chaque usecase) pour le groupe et un par membre du groupe | Diagramme des cas d’utilisation –  Supprimer les liens d’inclusion et de généralisation du diagramme global.  Ajouter la gestion d’un agenda pour le cuisinier amateur  Diagrammes de séquence « Booking dishes » de Pire Nathan –  Modifier les diagrammes en prenant en compte l’agenda des cuisiniers amateurs |
| Lundi  16/04/18 | Terminer les diagrammes de séquences personnels  Définir les itérations et les priorités  Mettre en place un brouillon de diagramme de classes | Définir des diagrammes de classes partielles par rapport aux itérations |
| Jeudi  19/04/18 | Mettre en commun les diagrammes de classes partielles pour former le diagramme de classes général | Modifier le sens des liens et ajouter les cardinalités  Définir les méthodes de base |
| Lundi  23/04/18 | Retranscrire les itérations, priorités, écrans en WPF et diagrammes de classes partielles en UML  Corriger les diagrammes de classes |  |
| Jeudi  26/04/18 | Terminer et affiner le diagramme de classe | Modifier la gestion des commentaires –  Un voisin ne peut commenter que les recettes qu’il a commandées, un lien doit donc être créé entre la classe « Comment » et « Order » dans le diagramme de classe  Créer un seul catalogue qui contient toutes les recettes au lieu d’un catalogue par mois  Donner un nom aux liens  Améliorer la gestion de l’agenda –  Gérer la quantité maximale de recettes pouvant être commandées liée à des dates de disponibilité établie par le cuisinier |
| Lundi 30/04/18 | Mettre à jour le diagramme de classe et le dossier |  |
| Lundi 07/05/18 | Etablir les diagrammes de séquence analytique et définir le schéma conceptuel de la base de données |  |
| Mardi 22/05/18 | Remettre l’application |  |

# Diagrammes et schémas

## Axe fonctionnel

### Diagramme de contexte statique

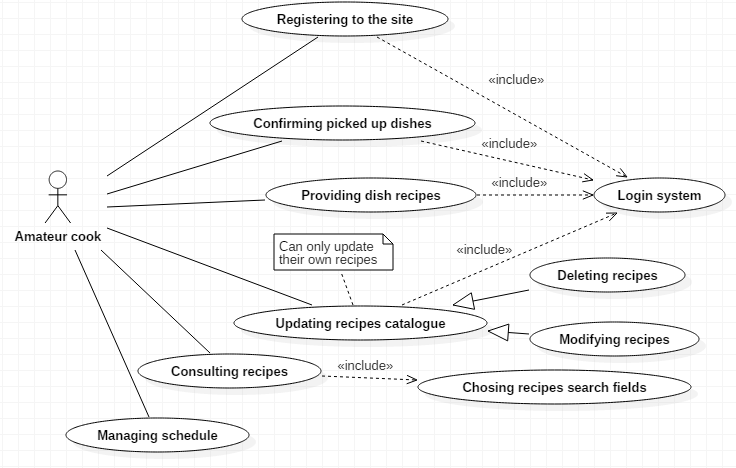
* Utilisateurs susceptibles d'interagir avec l'application
* Cardinalités
  + 0.1 : Uni-Utilisateur
  + 0.n : Multi-Utilisateur
* Acteurs
  + Primaires : Cuisinier amateur, voisin.
  + Secondaires : Administrateur.



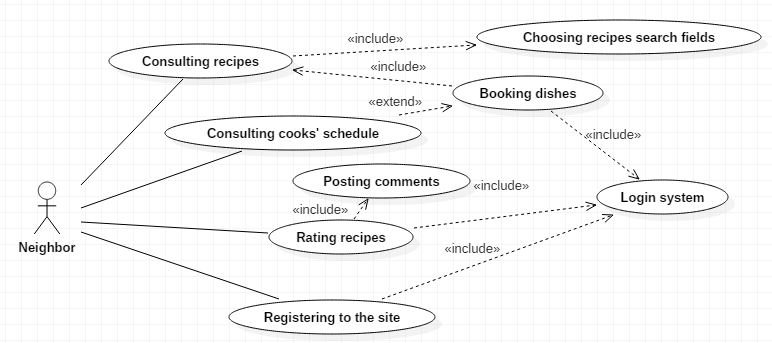
### Diagrammes des cas d’utilisation

* Définir les fonctions de l'application et les différents services disponibles pour chaque utilisateur. Outils de communication pour le client.

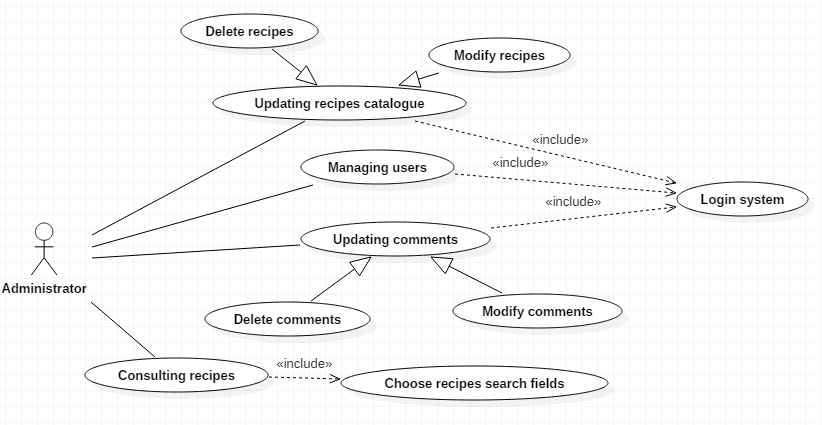
### Cuisinier amateur



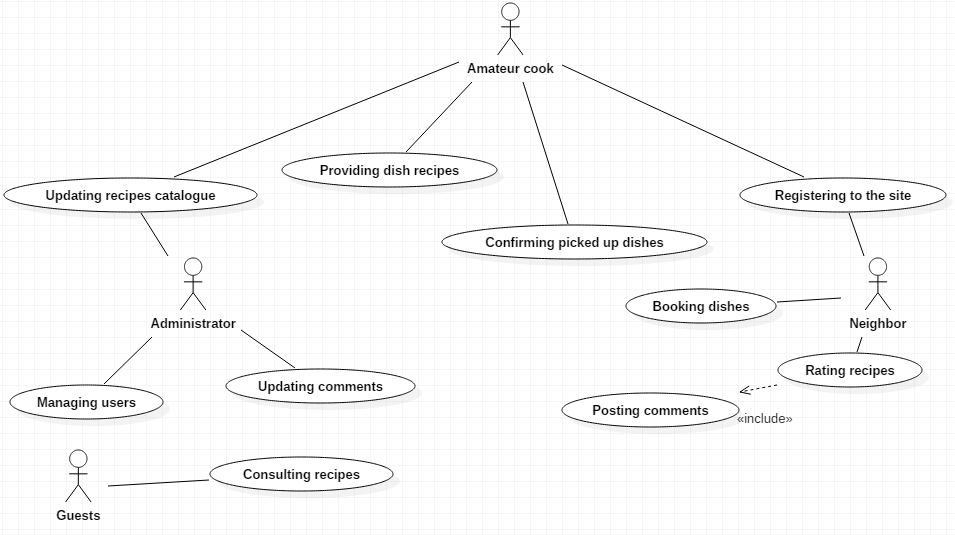
### Voisin



### Administrateur



### Global



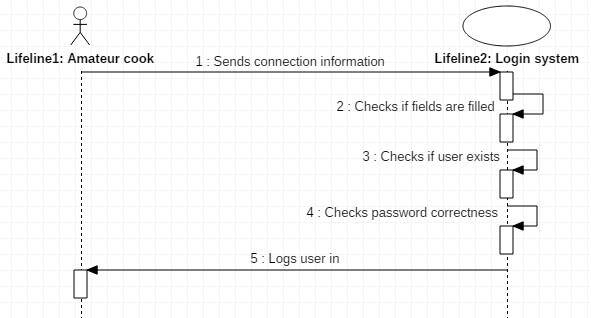
### Diagrammes de séquence système

* + - Répartition de la réalisation des diagrammes
      * Groupe : cas d’utilisation « Login System »
      * Bisconti Flavian : cas d’utilisation « Providing dish recipe »
      * Pire Nathan : cas d’utilisation « Booking dishes »
      * Zaretti Quentin : cas d’utilisation « Managing users »

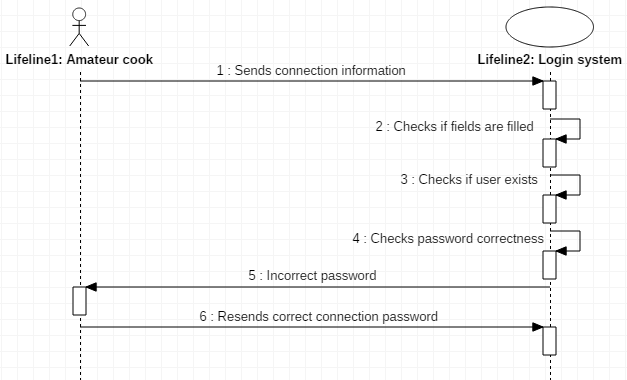
### Un utilisateur s’enregistre sur le site

Usecase « Login system »

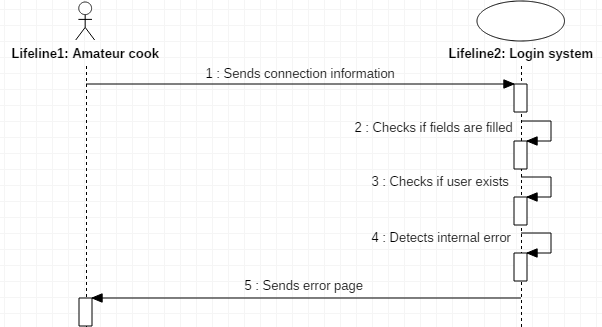
1. Séquence nominale – Enchainement des communications entre l’utilisateur et l’application lorsque l’utilisateur se connecte à sa session sans rencontrer d’erreur
   1. L’utilisateur envoie ses informations de connexion (pseudo et mot de passe)
   2. L’application vérifie que tous les champs sont remplis
   3. L’application vérifie que le pseudo correspond à un compte
   4. L’application vérifie si le mot de passe correspond au pseudo entré
   5. L’application connecte l’utilisateur à sa session



1. Séquence alternative – Enchainement des communications entre l’utilisateur et l’application lorsque l’utilisateur entre un mot de passe incorrect lors de sa tentative de connexion
   1. L’utilisateur envoie ses informations de connexion (pseudo et mot de passe)
   2. L’application vérifier que tous les champs sont remplis
   3. L’application vérifie que le pseudo correspond à un compte
   4. L’application vérifier si le mot de passe correspond au pseudo entré. Le mot de passe est incorrect.
   5. L’application affiche un message d’erreur : mot de passe incorrect
   6. L’utilisateur rentre un mot de passe correct
   7. (Retour point 2. Séquence nominale)



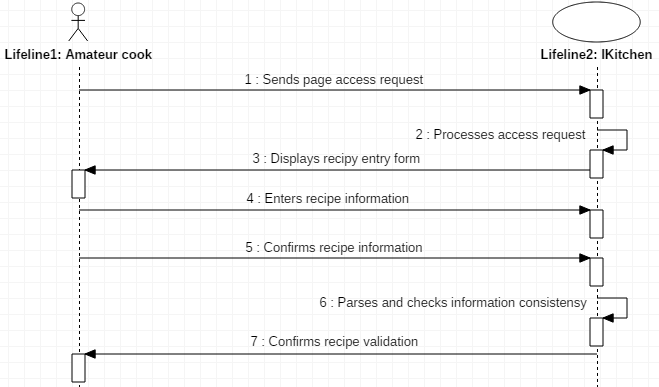
1. Séquence d’exception – Enchainement des communications entre l’utilisateur et l’application lorsqu’une erreur fatale est rencontrée lors d’une tentative de connexion (Ex. : base de données de l’application temporairement inaccessible)
   1. L’utilisateur envoie ses informations de connexion (pseudo et mot de passe)
   2. L’application vérifier que tous les champs sont remplis
   3. L’application vérifie que le pseudo correspond à un compte
   4. L’application rencontre une erreur interne
   5. L’application affiche un message d’erreur : erreur fatale, impossibilité de se connecter pour le moment



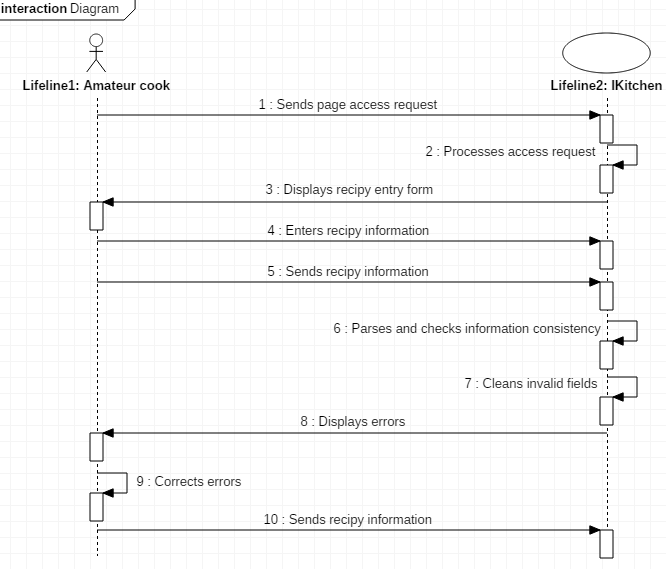
### Cuisinier amateur encode une nouvelle recette

Usecase « Providing dish recipe »

1. Séquence nominale – Enchaînement des communications entre le cuisinier amateur et l’application lorsque celui-ci propose une nouvelle recette sur le site sans rencontrer d’erreur.
   1. Le cuisinier amateur envoie une requête d’accès à la page
   2. L’application traite la requête
   3. L’application affiche le formulaire d’ajout d’une nouvelle recette
   4. Le cuisinier amateur entre les informations de la recette
   5. Le cuisinier amateur confirme (envoie) les informations de sa recette à l’application
   6. L’application vérifie que les informations sont cohérentes et ne présentent aucune erreur de syntaxe
   7. L’application affiche au cuisinier amateur un message confirmant la nouvelle recette entrée



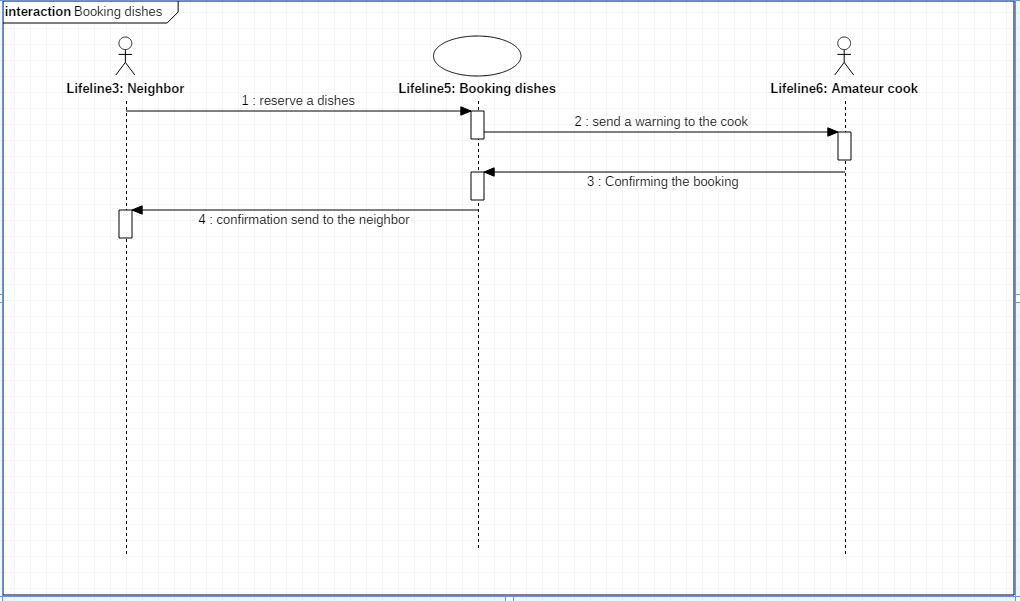
1. Séquence alternative – Enchaînement des communications entre le cuisinier amateur et l’application lorsque celui-ci propose une nouvelle recette sur le site mais commet une erreur lors de l’encodage des informations.
   1. Le cuisinier amateur envoie une requête d’accès à la page
   2. L’application traite la requête
   3. L’application affiche le formulaire d’ajout d’une nouvelle recette
   4. Le cuisinier amateur entre les informations de la recette
   5. Le cuisinier amateur confirme (envoie) les informations de sa recette à l’application
   6. L’application vérifie que les informations sont cohérentes et ne présentent aucune erreur de syntaxe : une ou plusieurs informations sont erronées
   7. L’application restaure les champs comportant des informations erronées
   8. L’application avertit l’utilisateur sur les champs erronés
   9. L’utilisateur corrige les erreurs
   10. L’utilisateur renvoie le formulaire sans erreur
   11. (Retour point 6. Séquence nominale)



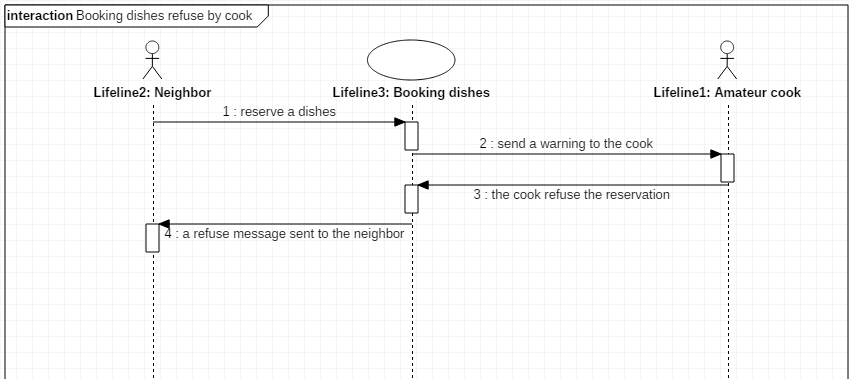
### Un voisin commande un plat

Usecase : « Booking dishes »

1. Séquence nominale – Enchaînement des communications entre le système et le cuisinier amateur ainsi que le voisin sans rencontrer d’erreur.
   1. Le voisin réserve un repas
   2. Un avertissement au cuisinier est envoyé
   3. Celui-ci donne une réponse favorable
   4. La confirmation et envoyé au voisin



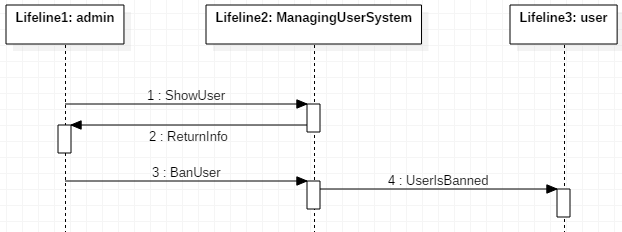
1. Séquence Alternative – Enchaînement des communications entre le système et le cuisinier amateur ainsi que le voisin mais le cuisinier refuse la commande.
   1. Le voisin réserve un repas
   2. Un avertissement au cuisinier est envoyé
   3. Celui-ci donne une réponse non favorable
   4. Un refus est envoyé au voisin



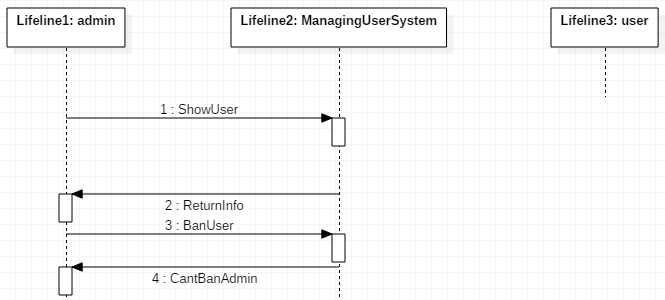
### Un administrateur gère les utilisateurs

Usecase : « Managing users »

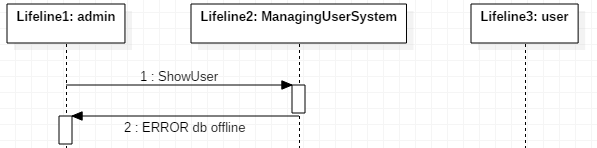
1. Séquence nominale – Enchaînement des communications entre le système et l’administrateur lorsque celui-ci banne un utilisateur
   1. L’’admin demande la liste des utilisateurs
   2. Le système renvoie les données
   3. L’admin demande le bannissement
   4. Le système bannit l’utilisateur



1. Séquence alternative – Enchaînement des communications entre le système et l’administrateur lorsque celui-ci tente de bannir un autre administrateur (Un administrateur ne peut pas bannir un autre administrateur)
   1. L’admin demande la liste des utilisateurs
   2. Le système renvoie les données
   3. L’admin demande le bannissement
   4. Le système refuse le bannissement



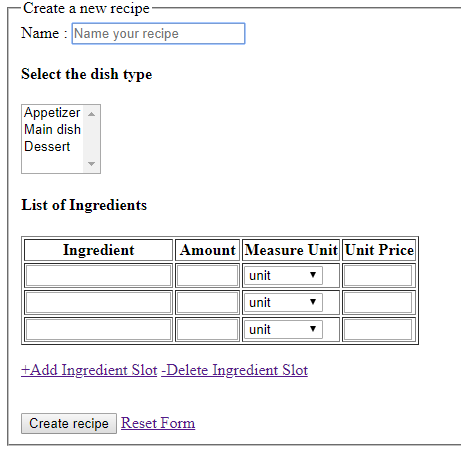
1. Séquence d’exception – Enchaînement des communications entre le système et l’administrateur lorsque celui-ci tente de bannir un utilisateur mais une erreur de communication à la base de données est rencontrée
   1. L’admin demande la liste des utilisateurs
   2. Le système renvoie une erreur de communication à la base de données



## Axe statique

* **Itérations et priorités :**

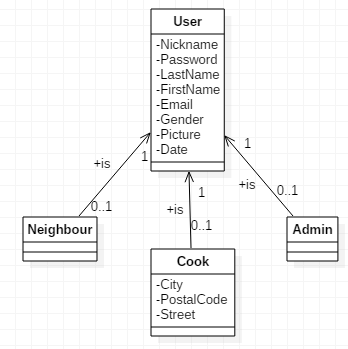
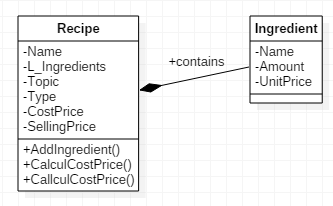
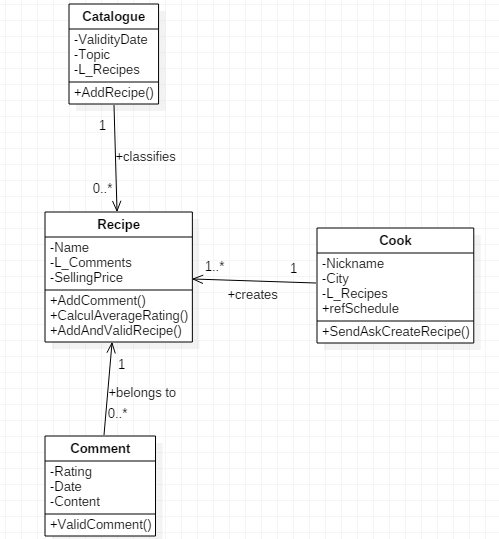
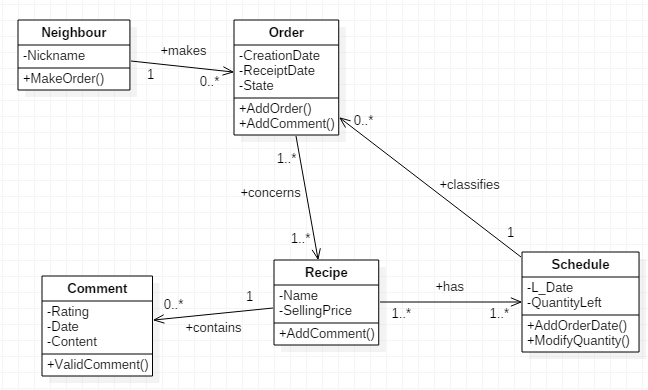
1. Inscription
   1. « Registering to the site »
2. Recettes
   1. « Providing dish recipes »



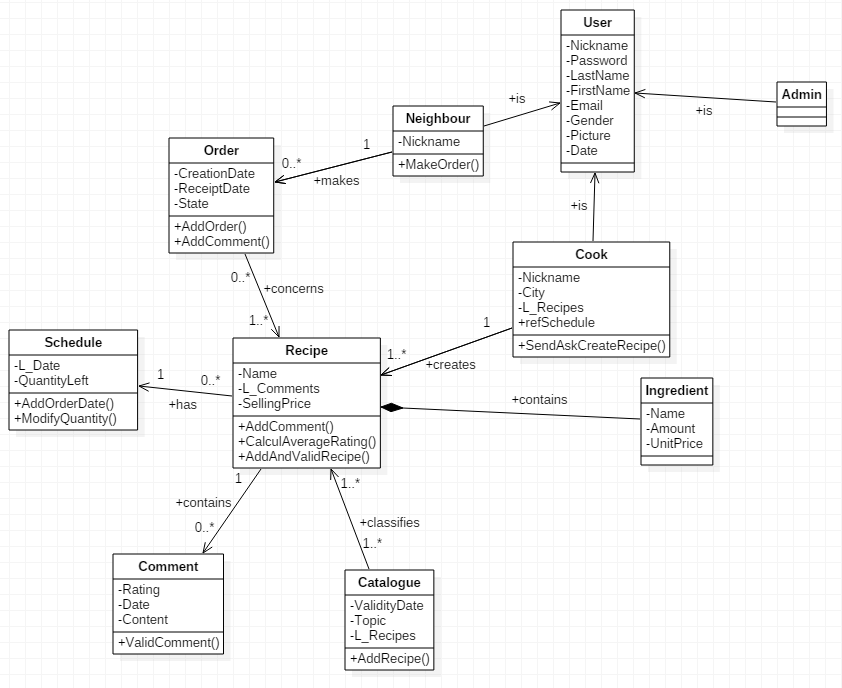
* 1. « Updating dish recipes »

1. Catalogue
   1. « Consulting recipes »
   2. « Displaying recipes information »
   3. « Choosing recipes search fields »
2. Commandes
   1. « Booking dishes »
   2. « Confirming picked up dishes »
   3. « Rating recipes »

### Diagrammes de classes partielles

1. Diagramme des classes partielles concernant l’itération « Inscription » 
2. Diagramme des classes partielles concernant l’itération « Recettes » 
3. Diagramme des classes partielles concernant l’itération « Catalogue » 
4. Diagramme des classes partielles concernant l’itération « Commandes » 

### Diagramme de classes



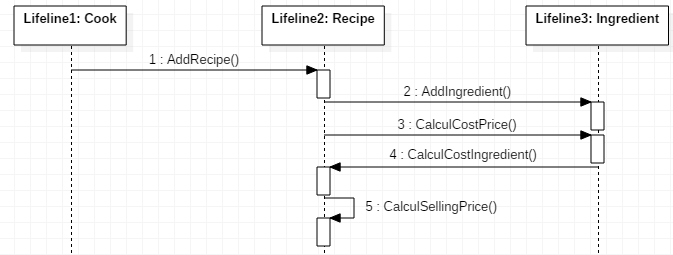
## Axe dynamique

* + - Répartition de la réalisation des diagrammes
      * Bisconti Flavian : cas d’utilisation « Providing dish recipe »
      * Pire Nathan : cas d’utilisation « Booking dishes »
      * Zaretti Quentin : cas d’utilisation « Managing users »

### Diagrammes de séquence analytique

Usecase « Providing dish recipe »

1. Le cuisinier manifeste son désir d’ajouter une nouvelle recette. Il introduit correctement dans le formulaire les informations concernant le plat et les envoie
2. La recette est créée et ainsi que chacun de ses ingrédients. Elle les récupère et les ajoute à sa liste d’ingrédients
3. Automatiquement, la recette calcule son prix de revient. Pour cela, elle envoie un message aux ingrédients pour qu’ils calculent leur coût individuel
4. Les ingrédients renvoient à la recette leur coût et celle-ci calcule son prix de revient.
5. La recette établit son prix de vente à partir de son prix de revient



## Schéma conceptuel

# Conclusions

## Conclusion de Bisconti Flavian

## Conclusion de Pire Nathan

## Conclusion de Quentin Zaretti

## Conclusion du groupe