

Maison des Ligues

Mission 2 – Rapport de tests

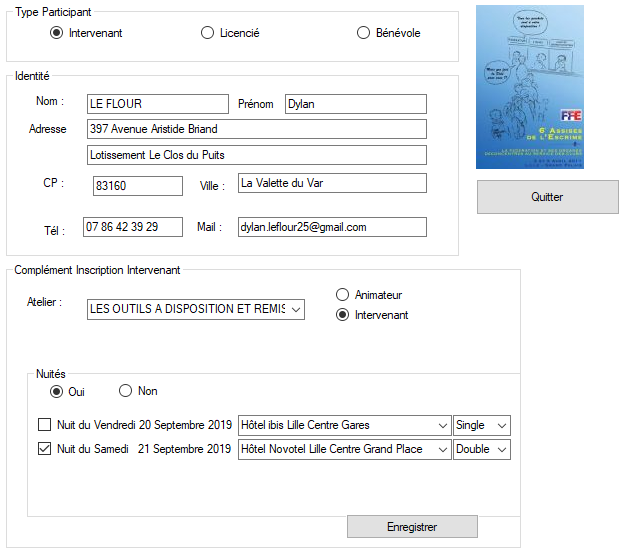
# 

# 

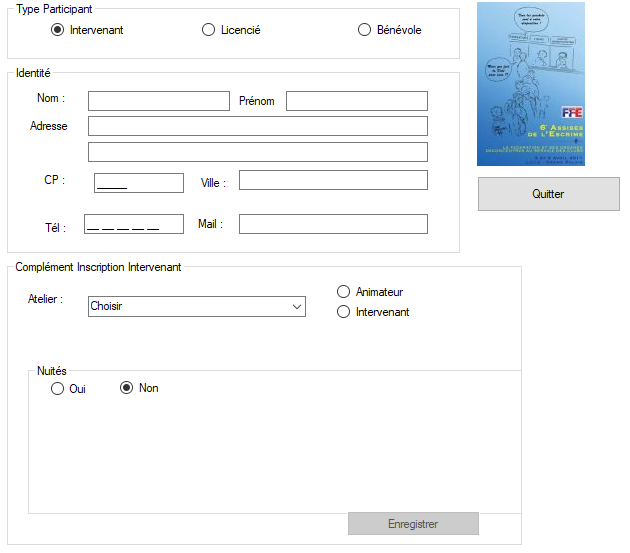
# **Suppression du contenu des zones de saisie et de sélection et envoi du mail.**

X - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

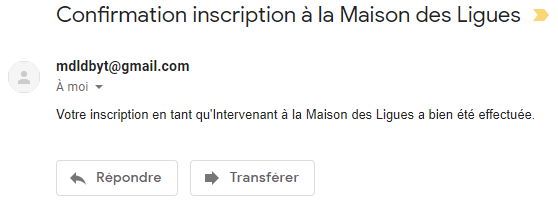
*Ajout d’un Participant.*



*Les champs se vident pour une nouvelle saisie.*



*Envoi d’un mail de confirmation au Participant.*



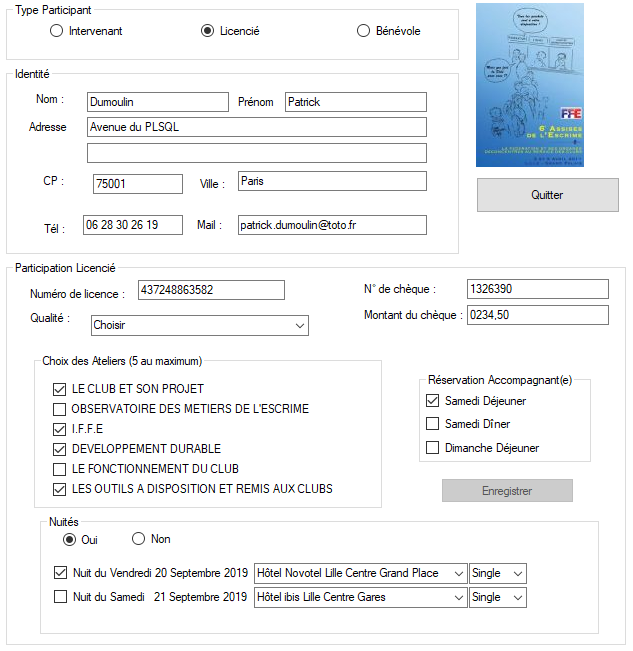
L'envoi de mail via la base de données est fiable, et facilement pris en charge. Le contenu d'un mail peut être le résultat d'une requête ou même des fichiers de n'importe quelle ressource du réseau. De plus, oracle pourra informer l'administrateur si un traitement ne s'est pas déroulé correctement. Cependant la base de données nous rend dépendant vis-à-vis d'elle. Nous n’avons pas choisi cette solution-là.

Pour l'envoi sous C#, nous avons utilisé l'espace de noms System.Net.Mail qui contient les classes utilisées pour envoyer des messages électroniques à un serveur de protocole SMTP. Nous avons choisi cette solution pour sa fiabilité et sa facilité à mettre en œuvre et pour les exceptions qui sont immédiatement affichées dans une MessageBox. Cette solution permet d'être indépendant vis à vis la base de données.

# **Inscription d’un licencié dans la base**

X - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

*Interface d’inscription du Licencié*



*Surcharge de la procédure NOUVEAULICENCIE en base*

|  |  |
| --- | --- |
| procedure NOUVEAULICENCIE(  pNom participant.nomparticipant%type,  pPrenom participant.prenomparticipant%type,  pAdr1 participant.adresseparticipant1%type,  pAdr2 participant.adresseparticipant2%type,  pCp participant.cpparticipant%type,  pVille participant.villeparticipant%type,  pTel participant.telparticipant%type,  pMail participant.mailparticipant%type,  pLicence benevole.numerolicence%type,  pQualite qualite.id%type,  pLesAteliers tids,  pNumCheque paiement.numerocheque%type,  pMontantCheque paiement.montantcheque%type,  pTypePaiement paiement.typepaiement%type  );    procedure NOUVEAULICENCIE(  pNom participant.nomparticipant%type,  pPrenom participant.prenomparticipant%type,  pAdr1 participant.adresseparticipant1%type,  pAdr2 participant.adresseparticipant2%type,  pCp participant.cpparticipant%type,  pVille participant.villeparticipant%type,  pTel participant.telparticipant%type,  pMail participant.mailparticipant%type,  pLicence benevole.numerolicence%type,  pQualite qualite.id%type,  pLesAteliers tids,  pNumCheque paiement.numerocheque%type,  pMontantCheque paiement.montantcheque%type,  pTypePaiement paiement.typepaiement%type,  plescategories tchars1,  pleshotels tchars4,  plesnuits tids  ); | procedure NOUVEAULICENCIE(  pNom participant.nomparticipant%type,  pPrenom participant.prenomparticipant%type,  pAdr1 participant.adresseparticipant1%type,  pAdr2 participant.adresseparticipant2%type,  pCp participant.cpparticipant%type,  pVille participant.villeparticipant%type,  pTel participant.telparticipant%type,  pMail participant.mailparticipant%type,  pLicence benevole.numerolicence%type,  pQualite qualite.id%type,  pLesAteliers tids,  pNumCheque paiement.numerocheque%type,  pMontantCheque paiement.montantcheque%type,  pTypePaiement paiement.typepaiement%type ,  plescategories tchars1,  pleshotels tchars4,  plesnuits tids,  pLesAccompagnants tids  );    procedure NOUVEAULICENCIE(  pNom participant.nomparticipant%type,  pPrenom participant.prenomparticipant%type,  pAdr1 participant.adresseparticipant1%type,  pAdr2 participant.adresseparticipant2%type,  pCp participant.cpparticipant%type,  pVille participant.villeparticipant%type,  pTel participant.telparticipant%type,  pMail participant.mailparticipant%type,  pLicence benevole.numerolicence%type,  pQualite qualite.id%type,  pLesAteliers tids,  pNumCheque paiement.numerocheque%type,  pMontantCheque paiement.montantcheque%type,  pTypePaiement paiement.typepaiement%type,  pLesAccompagnants tids  ); |

# **Gestion de la contrainte empêchant un bénévole d’avoir des nuitées.**

X - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

*Création et modifications de triggers*

|  |  |
| --- | --- |
| CREATE OR REPLACE TRIGGER TRGBIU\_CONTENUHEBERGEMENT  BEFORE INSERT OR UPDATE ON CONTENUHEBERGEMENT  FOR EACH ROW  DECLARE  nbPartiHerb integer:=0;  estBenevole exception;  BEGIN  select count(\*) into nbPartiHerb from benevole  where idbenevole=:new.idparticipant;  if nbPartiHerb >0 then  raise estBenevole;  end if;  EXCEPTION  when others then  raise\_application\_error(-20125 ,'Le participant est bénévole, il ne peut donc pas avoir d hébergement');  END; | create or replace TRIGGER trgbiu\_benevole before insert on benevole  for each row  declare  nb integer;  nbPartiHerb integer :=0;  hebergement exception;  begin  select 1 into nb from dual where not exists(select numerolicence from benevole  where numerolicence= :new.numerolicence) ;  select count(\*) into nbPartiHerb from contenuhebergement  where idparticipant=:new.idbenevole;  if nbPartiHerb>0 then  raise hebergement;  end if;  exception  when hebergement then  raise\_application\_error(-20120 ,'Le participant possedé un hébergement, il ne peut donc pas être bénévole.');  when no\_data\_found then  raise\_application\_error(-20110, 'Bénévole déjà inscrit, \n vous devez faire une modification de bénévole');  when others then  raise\_application\_error(-20002, 'Erreur à l''enregistrement');  end; |

La première contrainte concerne les relations Animer et Intervenir, avec un pivot sur les entités Intervenant et Atelier, c'est une contrainte d’exclusion totale. C'est-à-dire qu’un intervenant ne peut pas animer et intervenir sur un atelier en même temps, mais il doit faire au moins un des deux.

Cette contrainte est gérée par une entité "Statut", qui nous dit si l'intervenant anime ou intervient dans un atelier.

|  |
| --- |
|  |

La deuxième contrainte se situe entre les relations Participer et Inscrire et a pour pivot les entités Licencié et Atelier. C'est une inclusion puisque tout licencié qui s’inscrit à un atelier doit y participer.

La contrainte se fait entre les deux associations transformées en tables qui indique que l’idParticipant doit être égal à l’idLicencie et que l’idAtelier doit être le même.

|  |
| --- |
|  |

La troisième contrainte s’établit sur les relations Inclure et EtrePresent et a comme pivot l'entité Participant. Cette exclusion montre qu'un participant étant bénévole ne peut pas être hébergé.

Afin de s'assurer qu'un bénévole ne puisse pas être hébergé, nous avons décidé d’ajouter des triggers qui vérifient avant chaque insertion si l’id du participant n’est pas présent dans table contenuhebergement .

Par Exemple : lors de l’inscription d’un bénévole, on va simplement vérifier que l’idBenevole soit différent de tous les idParticipant de la table contenuhebergement.