

**BREVET DE TECHNICIEN SUPÉRIEUR
SERVICES INFORMATIQUES AUX ORGANISATIONS**

Option : Solutions logicielles et applications métiers

**U5 – PRODUCTION ET FOURNITURES DE
SERVICES INFORMATIQUES**

SESSION 2020

Durée : 4 heures

Coefficient : 5

Matériel autorisé :

Aucun matériel ni document est autorisé.

Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Le sujet comporte 18 pages, numérotées de 1/18 à 18/18
(sans compter la page de garde).

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS	SESSION 2020
U5 : Production et fourniture de services informatiques	Durée : 4 heures
Code sujet : SI5SLAM	Page 0 sur 18

Cas Coopain

Le sujet compte 18 pages dont 11 pages de documentation.

Le sujet est constitué de quatre dossiers qui peuvent être traités de façon indépendante.

Présentation du sujet	2-7
Dossier documentaire	8-18

Dossier documentaire

Document 1 : Schéma de données partiel de la base actuelle	8
Document 2 : Requête <i>SQL</i>	9
Document 3 : Informations imprimées sur une paillette.....	9
Document 4 : Description des cas d'utilisation des nouvelles fonctionnalités de l'application <i>MobiSemin</i>	10
Document 5 : Maquette des écrans pour la gestion des tournées	11
Document 6 : Schéma de la base de données locale <i>SQLite</i>	11
Document 7 : Classe <i>MobiDb</i> permettant de créer la base de données <i>SQLite</i> sous système <i>Android</i> ...	12
Document 8 : Types de données en <i>SQLite</i> (extrait de documentation).....	12
Document 9 : Diagramme des classes métier de la gestion des tournées	13
Document 10 : Description textuelle des classes métier de la gestion des tournées (vue partielle)	13
Document 11 : Implémentation partielle de la classe <i>GestionTournee</i> et sa classe de test <i>GestionTourneeTest</i>	15
Document 12 : Exemple d'utilisation de la classe technique <i>ArrayList</i>	16
Document 13 : Scénario pour la réalisation des tests unitaires de la classe <i>GestionTournee</i>	16
Document 14 : Documentation de l'outil <i>JUnit</i>	17
Document 15 : Retour sur investissement (<i>ROI</i> en anglais) : rappels	18
Document 16 : Éléments du devis pour le calcul du coût de mise en place des nouvelles fonctionnalités de <i>LogiSemin</i> et de <i>MobiSemin</i>	18

Barème

Mission 1	Gestion des demandes d'insémination	40 points
Mission 2	Nouvelles fonctionnalités de l'application <i>Android MobiSemin</i>	15 points
Mission 3	Gestion des tournées des inséminateurs	35 points
Mission 4	Calcul du retour sur investissement	10 points
	Total	100 points

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS	SESSION 2020
U5 : Production et fourniture de services informatiques	Durée : 4 heures
Code sujet : SI5SLAM	Page 1 sur 18

Présentation du contexte Coopain

L'organisation cliente : Coopain

Coopain (Coopérative Agricole Pour l'Insémination en Normandie) est une coopérative agricole spécialisée dans la reproduction des bovins, à la fois dans la sélection bovine des races Prim'Holstein et Normande, et dans l'insémination artificielle.

Chiffres clés

La coopérative compte 5 000 adhérents éleveurs de bovins. Son chiffre d'affaires annuel est de 20 millions d'euros. Elle est administrée par 24 administrateurs élus (12 femmes et 12 hommes) et compte 180 salariés. Elle est présente sur trois sites en Normandie. Le siège social administratif et technique est situé à Caen.

L'insémination artificielle bovine

Coopain emploie 160 inséminateurs qui pratiquent l'insémination artificielle bovine chez les adhérents, permettant ainsi une amélioration génétique de leur cheptel.

Dans un premier temps, les inséminateurs collectent la semence de taureaux sélectionnés pour leurs qualités génétiques. La semence récoltée est alors conditionnée en lots de doses pour la commercialisation. Chaque dose est appelée paillette. Dans un second temps, sur demande d'un adhérent, un inséminateur utilise une paillette pour féconder une vache, qui donnera ainsi naissance à un ou plusieurs veaux. Cette technique de reproduction assistée est appelée insémination artificielle.

L'activité d'insémination animale est de fait le cœur de métier de Coopain avec 450 000 actes par an.

Système d'information de la coopérative

Coopain utilise un logiciel métier nommé *LogiSemin*, développé par un prestataire, AgriDev, spécialisé dans le développement de logiciels pour l'agriculture et l'élevage.

Les fonctionnalités principales du logiciel métier *LogiSemin* sont :

- gestion des adhérents ;
- gestion du stock de paillettes ;
- gestion de la tarification ;
- suivi des inséminations ;
- bilan de fécondité ;
- suivi de reproduction ;
- gestion de la facturation aux adhérents.

Le service informatique de Coopain, basé au siège à Caen, compte un technicien support, une technicienne en charge de l'administration des bases de données (BDD) et du paramétrage du logiciel métier, un apprenti qui aide à l'administration et la maintenance de l'architecture réseau, et enfin le responsable du service.

L'infrastructure réseau de Coopain, sur les 3 sites, est composée de :

- 90 postes clients ;
- 3 serveurs supportant le logiciel métier *LogiSemin* et sa base de données ;
- 2 sites *Web* hébergés en interne ;
- 1 serveur pour le logiciel de comptabilité et de paie ;
- 1 réseau *VPN (virtual private network)* pour la communication inter sites.

Chaque inséminateur dispose :

- d'un PC avec accès au logiciel *LogiSemin*, via internet ;
- d'un ordiphone (*smartphone*) *Android* muni d'un logiciel *MobiSemin* pour le suivi de la reproduction, le suivi des adhérents et le suivi du stock de paillettes.

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS	SESSION 2020
U5 : Production et fourniture de services informatiques	Durée : 4 heures
Code sujet : SI5SLAM	Page 2 sur 18

L'organisation prestataire : AgriDev

AgriDev est un éditeur de solutions numériques innovantes pour les PME du monde agricole. Cette société compte une trentaine de salariés, dont 25 travaillent sur des projets de génie logiciel ou d'infrastructure réseau.

De nombreuses coopératives d'insémination animale, françaises ou étrangères, dont Coopain, utilisent l'application *LogiSemin* proposée par AgriDev.

AgriDev a, par ailleurs, développé le site *Web* de Coopain ainsi que l'application mobile *MobiSemin* qui s'appuie sur les mêmes données que *LogiSemin*, l'outil installé au siège.

Nouveaux besoins pour le contrôle des tournées des inséminateurs

Les administrateurs de Coopain ont décidé de mettre en place un suivi régulier des déplacements des inséminateurs. En effet, chaque inséminateur dispose d'un véhicule pour faire ses tournées d'insémination. Il peut être amené à réaliser d'autres catégories d'actes vétérinaires, appelées prestations, comme des constats de gestation, des transplantations, du suivi de reproduction, des actes de formation, ou des actes commerciaux, dans le secteur auquel il est affecté. Les véhicules sont des véhicules dédiés ; ils sont équipés d'une cuve d'azote liquide, non amovible, dans laquelle sont stockées les paillettes. Au total, c'est un parc de 150 véhicules que doit gérer Coopain.

L'enjeu principal est de réduire les coûts de déplacement, notamment en optimisant les tournées journalières des inséminateurs.

Ce suivi passe par le développement de fonctionnalités nouvelles dans l'application *LogiSemin* d'une part, et sur l'application *MobiSemin*, d'autre part.

Les **nouvelles fonctionnalités** de l'application métier *LogiSemin* à développer, sont :

- la gestion des tournées d'insémination ;
- le bilan inséminateur (contrôle des ratios : nombre d'inséminations, ou autres actes par kilomètre parcouru).

Les **nouvelles fonctionnalités** de l'application *MobiSemin* à développer, sont :

- la saisie à partir de l'ordiphone (*smartphone*) des tournées des inséminateurs ;
- la remontée journalière de ces informations par synchronisation via un serveur *Web*, vers la BDD.

Au sein de la société AgriDev, après un premier temps de travail pour comprendre l'existant, vous prenez part aux développements des nouvelles fonctionnalités des applications *LogiSemin* et *MobiSemin*.

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS	SESSION 2020
U5 : Production et fourniture de services informatiques	Durée : 4 heures
Code sujet : SI5SLAM	Page 3 sur 18

Mission 1 – Gestion des demandes d'insémination

Documents à utiliser : 1, 2 et 3

IMPORTANT : la ou le candidat(e) présentera les évolutions de la structure de la base de données en adoptant le formalisme de son choix (schéma entité association, diagramme de classes, ou encore schéma relationnel).

Votre chef de projet vous a fourni des éléments du système d'information de gestion des collectes de semence sur les taureaux. Il vous demande de compléter la documentation de l'application existante.

Question 1.1 : Déduire de la requête SQL fournie dans le dossier documentaire, la règle de gestion utilisée par Coopain pour choisir une paillette lors d'une demande d'insémination.

Question 1.2 : Indiquer pour chaque information à imprimer sur une paillette, la ou les données nécessaires en précisant pour chacune d'où elle provient dans le schéma existant et si elle nécessite une transformation.

Le chef de projet vous demande ensuite d'écrire deux requêtes, relatives à la gestion des demandes d'insémination, qui doivent pouvoir être réalisées avec la base de données existante.

Question 1.3 : Écrire la requête permettant d'obtenir pour le taureau FR0103015562 les tarifs des types de paillette (libellé du type de paillette, prix de la paillette).

Question 1.4 : Écrire la requête permettant d'obtenir la liste des taureaux (identifiant national, nom, stock total restant de paillettes), le tout trié de façon croissante par stock total restant.

Enfin, il vous demande de modifier le schéma de données en prenant en compte l'ensemble des nouvelles règles de gestion exprimées ci-dessous.

Le choix du taureau et du type de paillette

Les adhérents peuvent consulter le catalogue de Coopain qui présente les taureaux pour lesquels Coopain commercialise des paillettes.

Lorsqu'une de ses vaches est prête à être fécondée, l'adhérent contacte Coopain pour demander une insémination. La société Coopain doit être très réactive pour effectuer l'insémination au plus tard le lendemain. Une demande est alors créée pour laquelle on enregistre la date de la demande ainsi que les informations fournies par l'adhérent : l'identifiant de l'adhérent, l'identifiant du taureau choisi, le type de paillette souhaité et l'identifiant de la vache à inséminer. Chaque vache porte un nom et est identifiée par un numéro d'identification national inscrit sur une boucle agrée fixée à l'oreille.

Les inséminateurs

Chaque inséminateur :

- possède un numéro de licence obligatoire pour pratiquer les inséminations ;
- est affecté sur un secteur géographique, chaque secteur possédant un nom, une description et regroupant un certain nombre d'adhérents ;
- est doté d'un véhicule équipé d'une cuve d'azote liquide à -196°, toujours le même, mais qu'il peut partager avec plusieurs inséminateurs ;
- est doté d'un ordiphone (*smartphone*) avec abonnement téléphonique.

Chaque véhicule est identifié par son numéro d'immatriculation. Il est nécessaire de connaître sa marque, son modèle et sa date d'achat.

L'insémination

L'inséminateur prend connaissance des demandes qui lui ont été affectées et s'organise à partir des adresses des adhérents. Il choisit et transfère dans la cuve d'azote de son véhicule, les paillettes nécessaires aux différentes inséminations prévues. Lorsqu'il arrive chez un adhérent, il procède aux inséminations et complète chaque demande en précisant le lot de la paillette utilisée, la date et l'heure d'insémination, la température ambiante et un compte rendu détaillé. Ces informations serviront ultérieurement pour le suivi de la reproduction, suivi qui sort du cadre de cette mission.

Question 1.5 : Présenter le schéma de données nécessaire à la gestion des demandes d'insémination en reprenant les éléments nécessaires du modèle existant.

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS	SESSION 2020
U5 : Production et fourniture de services informatiques	Durée : 4 heures
Code sujet : SI5SLAM	Page 4 sur 18

Mission 2 – Nouvelles fonctionnalités de l'application *Android MobiSemin*

Documents à utiliser : 4, 5, 6, 7 et 8

Lors de la réunion de travail entre AgriDev et Coopain concernant les adaptations de l'application mobile *MobiSemin*, il a été évoqué de faire saisir quotidiennement aux inséminateurs les kilomètres qu'ils ont parcourus. Cette nouvelle activité nécessitera une synchronisation des données saisies et enregistrées dans une base de données *SQLite* locale, avec la base de données distante. Pour cela il est prévu d'ajouter une option « Gestion des tournées » à l'application mobile *MobiSemin*.

Cette nouvelle option « Gestion des tournées » nécessitera un menu à trois items : « Saisie tournée », « Synchronisation » et « Quitter ». Les items « Saisie tournée » et « Synchronisation » permettront d'accéder respectivement aux fonctionnalités du même nom, et « Quitter » permettra de revenir sur les activités principales de l'application *MobiSemin*.

L'item « Saisie tournée » permet la saisie des kilomètres des tournées.

L'inséminateur s'authentifie puis il obtient un formulaire permettant de renseigner la date de tournée ainsi que le véhicule utilisé lors de la tournée. Par défaut, ces deux informations contiennent respectivement la date du jour et le véhicule affecté à l'inséminateur. Ce dernier valide les valeurs par défaut fournies ou peut les changer. Par défaut, le kilométrage de début de tournée sera le même que celui de fin de la dernière tournée saisie pour le véhicule. Les informations saisies seront stockées en local sur l'ordiphone (*smartphone*).

Le dossier documentaire comprend une partie du rapport présentant le diagramme des cas d'utilisation et une partie du scénario de la nouvelle fonctionnalité « **Saisir une tournée** ».

Question 2.1 : Compléter les scénarios d'erreur du cas d'utilisation « Saisir une tournée » concernant la saisie des kilomètres d'une tournée.

On s'intéresse à la partie de l'application mobile permettant l'enregistrement des kilomètres saisis par l'inséminateur dans une table nommée *histoKm* de la base locale *MobiDb* du mobile *Android*. Cette base est gérée par le système de gestion de bases de données *SQLite* intégré dans chaque appareil *Android*. *SQLite* propose les fonctionnalités standards des SGBD relationnels. Le schéma relationnel de cette base est donné dans la documentation.

Le dossier documentaire vous propose une partie du code de la classe *MobiDb*, dérivée de la classe *SQLiteOpenHelper*. La classe *SQLiteOpenHelper* est une classe utilitaire qui permet de créer une base de données et les tables qu'elle contient, notamment en donnant accès à un objet *SQLiteDatabase*.

Question 2.2 : Compléter la méthode *onCreate()* de la classe *MobiDb*.

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS	SESSION 2020
U5 : Production et fourniture de services informatiques	Durée : 4 heures
Code sujet : SI5SLAM	Page 5 sur 18

Mission 3 – Gestion des tournées des inséminateurs

Documents à utiliser : 9, 10, 11, 12, 13 et 14

IMPORTANT : le candidat ou la candidate peut choisir de présenter les éléments de code à l'aide du langage de programmation de son choix ou de pseudo-code algorithmique.

Au cours de leurs tournées, les inséminateurs sont amenés à effectuer diverses prestations vétérinaires.

La majeure partie des classes développées pour la gestion des tournées sera utilisée indifféremment dans *MobiSemin* et/ou *LogiSemin*.

Pour des raisons d'efficacité dans le développement, et pour permettre une plus grande évolutivité du code, la persistance des données est gérée par l'outil *Hibernate*, qui gère la persistance des objets dans une base de données relationnelle (*ORM object-relational mapping*). Son utilisation permet d'établir la correspondance entre une table de la base de données et une classe du modèle objet métier.

Les classes métier (visibles sur le diagramme de classes) seront ainsi générées automatiquement à partir de cet outil de mise en correspondance (*mapping*) objet-relationnel. Elles sont qualifiées de *POJO* (*Plain Old Java Object* – « bon vieux objet Java ») et se limitent chacune à définir :

- un constructeur sans paramètre (parfois implicite) ;
- des membres de données privés ;
- des accesseurs à ces membres en lecture et en écriture (*get*, *set*).

La génération automatisée de ces classes (qu'elle soit partielle ou totale) peut être à nouveau invoquée dans le cadre du développement, en cas de modification de la structure de la base de données relationnelle associée.

Pour permettre cette génération automatique sans prendre le risque de perdre du code déjà implémenté, les fonctionnalités à coder pour gérer les tournées le seront dans des "classes de gestion" encapsulant des objets métier. Ainsi la classe *GestionTournée* sera utilisée pour gérer les tournées.

Question 3.1

3.1.1 Expliquer le mécanisme qui permet de différencier les deux méthodes de même nom (*CATournée*) de la classe *GestionTournée*.

3.1.2 Écrire la méthode *getCoordGPS()* de la classe *Adherent* qui renvoie les coordonnées GPS (latitude et longitude) sous la forme d'une chaîne de caractères.

3.1.3 Écrire le code de la méthode *getAdherents()* de la classe *GestionTournée*.

3.1.4 Écrire le code de la méthode *montantAFacturer()* de la classe *Visite*.

Le chef de projet vous demande ensuite de travailler sur les tests unitaires de la classe *GestionTournée* selon le scénario qu'il vous a fourni et avec l'atelier de développement (*framework*) de test *JUnit*.

Question 3.2 : Compléter les tests unitaires correspondants à la méthode *CATournée()* de la classe *GestionTournée* en respectant le scénario fourni par le chef de projet et en respectant la syntaxe utilisée.

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS	SESSION 2020
U5 : Production et fourniture de services informatiques	Durée : 4 heures
Code sujet : SI5SLAM	Page 6 sur 18

Mission 4 – Calcul du retour sur investissement

Documents à utiliser : 15 et 16

Coopain souhaite calculer le retour sur investissement de la mise en place du suivi des déplacements des inséminateurs :

- à partir de l'application *MobiSemin* installée désormais sur les ordiphones (*smartphones*) *Android* des inséminateurs ;
- à partir de l'application *LogiSemin*, et des données issues de *MobiSemin* pour la gestion des kilomètres (saisie et synchronisation).

Un tableau est établi, permettant de calculer une estimation des coûts de l'analyse, des développements et des tests.

À la fin de la première année d'utilisation de ces nouvelles fonctionnalités, on constate que le nombre de kilomètres déclaré par les inséminateurs a diminué de 5 %. Cette diminution est considérée comme stable dans la durée.

Le coût d'un kilomètre dépend du véhicule utilisé et du nombre de km. Le coût moyen par véhicule est équivalent à 0,4 €/km.

Les 150 véhicules avaient parcouru au total 2 000 000 km la dernière année avant la mise en place des nouvelles fonctionnalités de *LogiSemin* et *MobiSemin*.

Question 4.1 : Calculer le retour sur investissement (*ROI*) de la mise en place des nouvelles fonctionnalités ; donner et expliquer le détail du calcul.

La durée de rentabilité (point mort) a été calculée, le résultat obtenu est de 92 jours.

Question 4.2 : Justifier dans une courte note la pertinence de cet investissement, connaissant la durée de rentabilité.

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS	SESSION 2020
U5 : Production et fourniture de services informatiques	Durée : 4 heures
Code sujet : SI5SLAM	Page 7 sur 18

Document 1 : Schéma de données partiel de la base actuelle

Schéma entité-association

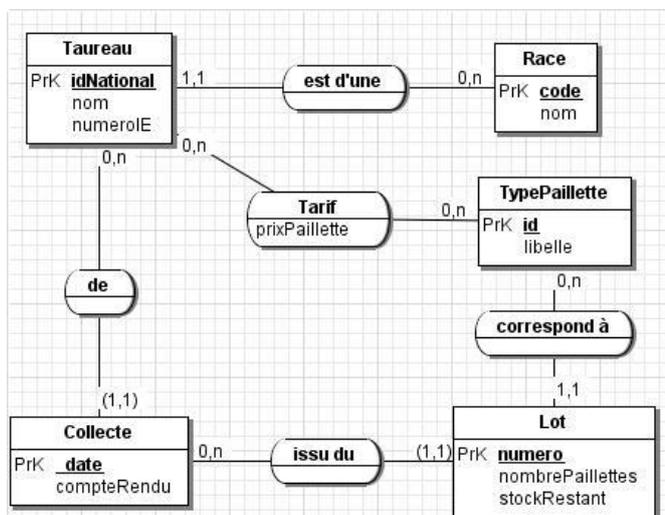
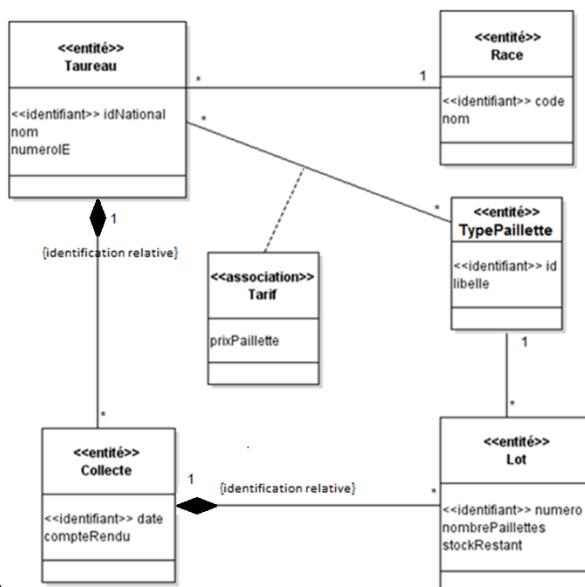


Diagramme de classes



Description textuelle du schéma relationnel

Race (code, nom)

- Clé primaire : code

Taureau (idNational, nom, numerolE, codeRace)

- Clé primaire : idNational
- Clé étrangère : codeRace en référence à code de Race

Collecte (idNationalTaureau, date, compteRendu)

- Clé primaire : idNationalTaureau, date
- Clé étrangère : idNationalTaureau en référence à idNational de Taureau

TypePaille (id, libelle)

- Clé primaire : id

Lot (idNationalTaureau, date, numero, nombrePailles, stockRestant, idTypePaille)

- Clé primaire : idNationalTaureau, date, numero
- Clé étrangère : (idNationalTaureau, date) en référence à (idNationalTaureau, date) de Collecte
- Clé étrangère : idTypePaille en référence à id de TypePaille

Tarif (idNationalTaureau, idTypePaille, prixPaille)

- Clé primaire : idNationalTaureau, idTypePaille
- Clé étrangère : idNationalTaureau en référence à idNational de Taureau
- Clé étrangère : idTypePaille en référence à id de TypePaille

Informations complémentaires

- Le contenu de la table TypePaille est fixe :

id	libelle
1	paille conventionnelle
2	paille sexée femelle
3	paille enrichie
4	paille sexée mâle

- Le prix d'une paille dépend du type de paille et du taureau dont est issue la semence.
- Lors d'une collecte de semence de taureau, il est possible de créer plusieurs lots de pailles.
- Toutes les pailles d'un lot seront du même type de paille.

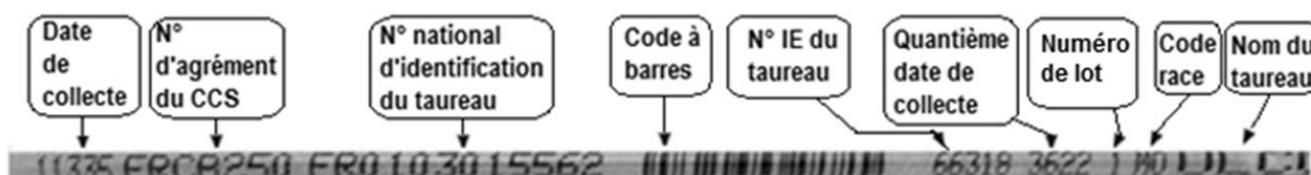
Document 2 : Requête SQL

Cette requête permet d'obtenir les informations nécessaires pour choisir une paillette enrichie du taureau FR0103015562 :

```
SELECT date, numero
FROM Lot
WHERE idNationalTaureau = 'FR0103015562'
AND idTypePaillette = 3
AND date = (SELECT MIN(date)
            FROM Lot
            WHERE idNationalTaureau = 'FR0103015562'
            AND idTypePaillette = 3
            AND stockRestant >= 1)
```

Document 3 : Informations imprimées sur une paillette

Une des activités de Coopain est de procéder à la collecte de semence de taureaux. Un taureau peut être collecté deux à trois fois par semaine à des jours différents. Chaque collecte va permettre de remplir environ 200 paillettes qui seront ensuite réparties en lots et stockées dans de l'azote liquide. Pour répondre à une obligation de traçabilité, chaque paillette est dotée d'informations claires sous une forme normalisée à l'échelle internationale qui permettent de tracer le lot. Chaque paillette d'un même lot contiendra les mêmes informations :

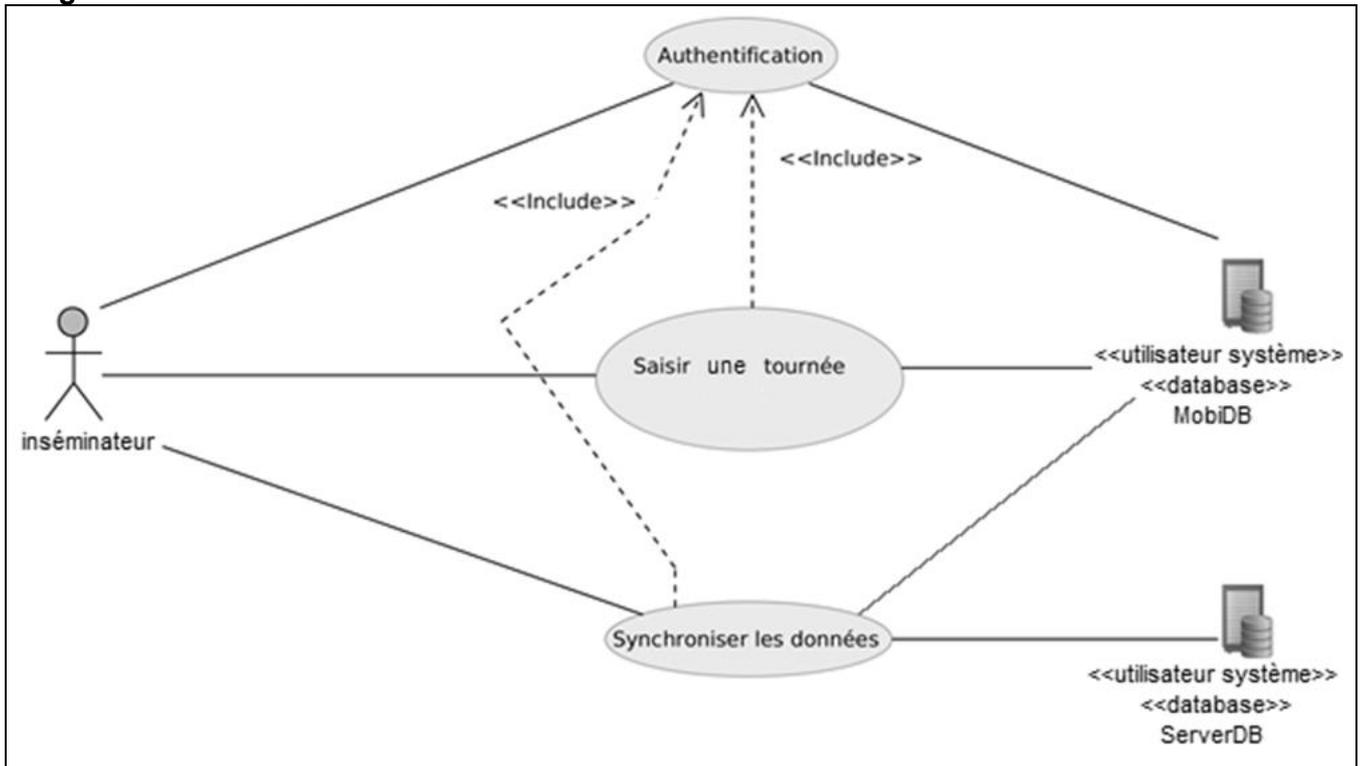


- Date de collecte (sous la forme AAJJJ). Dans l'exemple, 11335 signifie 335^{ème} jour de l'année 2011 (1^{er} décembre) ;
- N° d'agrément du CCS : identification du centre de collecte de semence (le numéro attribué à Coopain) ;
- N° national d'identification du taureau. Dans l'exemple FR0103015562 :
 - FR représente le pays de naissance du taureau (France),
 - 0103015562 est le numéro national du taureau (sur 10 positions) ;
- Code à barres (de type 128 CRT numérique sur 10 positions) est composé :
 - du n° d'IE (institut de l'élevage) du taureau (5 positions),
 - de la date de collecte exprimée en quantième depuis le 1^{er} janvier 2002 (sur 4 positions),
 - du numéro de lot de prélèvement ;
- N° IE du taureau : identifiant du taureau inscrit dans la base de données nationale de l'institut de l'élevage (sur 5 positions) ;
- Quantième date de collecte : nombre de jours (quantième) écoulés entre le 01 janvier 2002 et la date de collecte (pour information le 01 janvier 2002 est la date de référence internationale à partir de laquelle le dispositif de traçage des lots a été mis en place) ;
- Numéro de lot de prélèvement ;
- Code de la race : par exemple MO est le code de la race Montbéliard ;
- Nom du taureau (imprimé en gras).

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS	SESSION 2020
U5 : Production et fourniture de services informatiques	Durée : 4 heures
Code sujet : SI5SLAM	Page 9 sur 18

Document 4 : Description des cas d'utilisation des nouvelles fonctionnalités de l'application *MobiSemin*

Diagramme du cas d'utilisation



Fiche descriptive du cas d'utilisation (use case) « Saisir une tournée »

Pré-condition

L'utilisateur devra être authentifié.

Scénario nominal

1. Le système présente un formulaire avec un calendrier initialisé avec la date du jour, la liste des véhicules disponibles et son véhicule affecté par défaut.
2. L'utilisateur met éventuellement à jour les données (choix d'un autre véhicule et/ou d'une autre date) et clique sur le bouton valider du formulaire.
3. Le système vérifie les données saisies et présente un formulaire de saisie des kilomètres.
4. L'utilisateur renseigne le kilométrage au compteur de fin de tournée et clique sur le bouton valider du formulaire.
5. Le système vérifie les données saisies, enregistre toutes les données, informe l'utilisateur que les données ont été enregistrées et renvoie sur la page d'accueil.

Scénarios alternatifs

- 2.a L'utilisateur abandonne la saisie. Fin.
- 4.a L'utilisateur abandonne la saisie. Fin.

Scénarios d'erreur

- 3.a La date saisie est supérieure à la date du jour, le système en informe l'utilisateur et retourne à l'étape 1.

... // À compléter sur votre copie

Post-condition

Les données d'une tournée sont enregistrées en local sur le mobile.

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS	SESSION 2020
U5 : Production et fourniture de services informatiques	Durée : 4 heures
Code sujet : SI5SLAM	Page 10 sur 18

Document 5 : Maquette des écrans pour la gestion des tournées



Document 6 : Schéma de la base de données locale SQLite

inseminateur(id, nom, prenom)

- Clé primaire : id

vehicule(id, immat, marque, modele, dateAchat)

- Clé primaire : id

histoKm (id, date, kmDebut, kmFin, idInsemin, idVehicule)

- Clé primaire : id
- Clé étrangère : idInsemin en référence à id d'inseminateur
- Clé étrangère : idVehicule en référence à id de vehicule

Remarque : date est la date de la tournée, kmDebut et kmFin représentent les relevés du kilométrage au compteur.

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS	SESSION 2020
U5 : Production et fourniture de services informatiques	Durée : 4 heures
Code sujet : SI5SLAM	Page 11 sur 18

Document 7 : Classe MobiDb permettant de créer la base de données SQLite sous système Android

```
package com.mobisemin.projet.sqliteMobiDb;
import android.content.ContentValues;
...

//class MobiDb : classe pour la création de la base de données locale SQLite et des tables

public class MobiDb extends SQLiteOpenHelper {
    private static final android.content.Context context = null;
    // Constructeur
    public MobiDb(Context context, String name, SQLiteDatabase.CursorFactory factory, int version){
        super(context, name, factory, version);
    }

    // Méthode pour créer les tables inseminateur, vehicule et histoKm
    @Override // Surcharge d'une méthode héritée
    public void onCreate(SQLiteDatabase db) {
        String strReq= "CREATE TABLE inseminateur(id INTEGER PRIMARY KEY AUTOINCREMENT,
nom TEXT, prenom TEXT)";
        db.execSQL(strReq);

        String strReq= "CREATE TABLE vehicule(id INTEGER PRIMARY KEY AUTOINCREMENT,
immat TEXT, marque TEXT, modele TEXT, dateAchat TEXT)";
        db.execSQL(strReq);



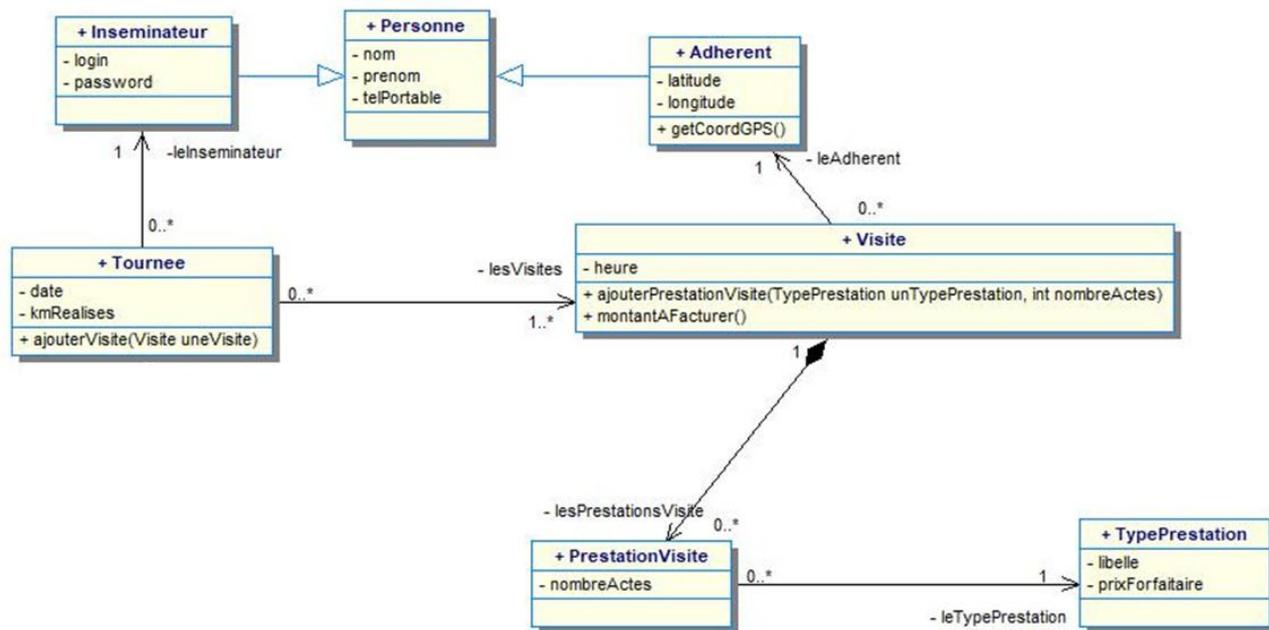
// À compléter sur votre copie


    }
}
```

Document 8 : Types de données en SQLite (extrait de documentation)

Type de donnée	Signification
NULL	Représente la valeur NULL.
INTEGER	Pour les nombres entiers.
REAL	Pour les nombres à virgule flottante.
TEXT	Pour les chaînes de caractères ou les dates.
BLOB	Pour les données brutes, octets, images... stockées au format binaire.

Document 9 : Diagramme de classes métier de la gestion des tournées



Remarque : Les constructeurs et accesseurs ne sont pas représentés sur ce diagramme

Document 10 : Description textuelle des classes métier de la gestion des tournées (vue partielle)

```

public class Personne {
    private String nom;
    private String prenom;
    private String telPortable;
    public Personne(String unNom, String unPrenom, String unTelPortable){...}
    public String getNom() {...}
    public String getPrenom() {...}
    ...
}

public class Adherent extends Personne {
    private String latitude;
    private String longitude;
    public Adherent(String unNom, String unPrenom, String unTelPortable, String uneLatitude,
        String uneLongitude){...}
    public String getCoordGPS() {
        // À compléter sur votre copie //renvoie une chaîne de caractères
        //composée de la latitude et de la
        //longitude séparées par un espace
    }
    ...
}

public class Inseminateur extends Personne {
    private String login;
    private String password;
    public Inseminateur(String unNom, String unPrenom, String unTelPortable){...}
    ...
}
    
```

```

public class Tournee {
    private Date date;
    private int kmRealises; // nombre de km effectués pendant la tournée
    private Insemineur leInsemineur;
    private ArrayList<Visite> lesVisites;
    public Tournee(Date uneDate, Insemineur unInsemineur) {...}
    public Date getDate() {...}
    public void setDate(Date date) {...}
    public int getKmRealises() {...}
    public void setKmRealises(int kmRealises) {...}
    public Insemineur getLeInsemineur() {...}
    public ArrayList <Visite> getLesVisites() {...}
    public void ajouterVisite(Visite uneVisite) {...}
}

```

```

public class Visite {
    private Adherent leAdherent;
    private String heure; // sous la forme hh:mm par exemple "09:00"
    private ArrayList<PrestationVisite> lesPrestationsVisite;
    public Visite(Adherent unAdherent, String uneHeure) {...}
    public Adherent getLeAdherent() {...}
    public String getHeure() {...}
    public ArrayList<PrestationVisite> getLesPrestationsVisite() {...}
    public ajouterPrestationVisite(TypePrestation unTypePrestation, int unNombreActes) {...}

    // Retourne le montant total à facturer pour la visite
    // le montant d'une prestation de la visite s'obtient
    // en multipliant le nombre d'actes réalisés par le prix forfaitaire du type de la prestation
    public float montantAFacturer(){

|                                       |
|---------------------------------------|
| <i>// À compléter sur votre copie</i> |
|---------------------------------------|


    }
}

```

```

public class PrestationVisite {
    private TypePrestation leTypePrestation; // le type de prestation effectuée durant la visite
    private int nombreActes; // nombre d'actes
    public TypePrestation getLeTypePrestation() {...}
    public int getNombreActes() {...}
    ...
}

```

```

public class TypePrestation { // type de prestation
    private String libelle;
    private float prixForfaitaire; // prix forfaitaire du type de prestation (prix forfaitaire d'un acte)
    public TypePrestation(String libelle, float prixForfaitaire) {...}
    public float getPrixForfaitaire() {...}
    ...
}

```

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS	SESSION 2020
U5 : Production et fourniture de services informatiques	Durée : 4 heures
Code sujet : SI5SLAM	Page 14 sur 18

Document 11 : Implémentation partielle de la classe GestionTournee et sa classe de test GestionTourneeTest

```
public class GestionTournee {
    private Tournee laTournee;
    // Constructeur
    public GestionTournee(Tournee uneTournee) {
        this.laTournee = uneTournee;
    }
    // Ajoute la visite uneVisite à la liste des visites de la tournée
    public void ajouterVisite(Visite uneVisite) {
        laTournee.ajouterVisite(uneVisite);
    }
    // Retourne la liste des adhérents visités sur la tournée
    // un adhérent n'est visité qu'une seule fois au cours de la tournée
    public ArrayList<Adherent> getAdherents() {
        // À compléter sur votre copie
    }
    // Retourne le montant total à facturer pour l'ensemble des visites de la tournée
    public float CATournee() {
        // Cette méthode fera l'objet d'un test unitaire
    }
    // Retourne le montant total à facturer pour l'ensemble des visites de la tournée
    // correspondant au type de prestation passé en paramètre
    public float CATournee(TypePrestation unTypePrestation) {...}
}
```

```
public class GestionTourneeTest {
    private Inseminateur ins1;
    private Adherent adh1, adh2;
    private Visite v1, v2;
    private TypePrestation tp1, tp2;
    private Tournee t1;
    private GestionTournee gt;

    @Before
    public void setUp() throws Exception { // Méthode setup() automatiquement exécutée
        // avant chaque appel de la méthode de test
        tp1 = new TypePrestation("Insémination", 100);
        tp2 = new TypePrestation("Echographie", 10);
        ins1 = new Inseminateur("Petit", "Ferdinand", "0600000100");
        adh1 = new Adherent("Duboeuf", "Georges", "0600000020", "48.5331", "7.72");
        adh2 = new Adherent("Cow", "Marguerite", "0600000003", "48.5833", "7.75");
        t1 = new Tournee(new SimpleDateFormat("dd-MM-yyyy").parse("06-05-2020"), ins1);
        gt = new GestionTournee(t1);

        // Préparation des visites
        // À compléter sur votre copie
    }

    @Test
    public void testCATournee() {
        // À compléter sur votre copie
    }
}
```

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS	SESSION 2020
U5 : Production et fourniture de services informatiques	Durée : 4 heures
Code sujet : SI5SLAM	Page 15 sur 18

Document 12 : Exemple d'utilisation de la classe technique ArrayList

```
ArrayList<Visite> lesVisites ; // Déclaration d'une collection d'instances de la classe Visite
lesVisites = new ArrayList<Visite>() ; // Instanciation d'une collection d'instances de Visite
Visite uneVisite = new Visite(...) ;
lesVisites.add( uneVisite ) ; // Ajout d'une visite dans la collection
System.out.println( lesVisites.get(0) ) ; // Affichage du premier élément
System.out.println( lesVisites.size() ) ; // Affichage du nombre d'éléments

for( Visite uneVisite : lesVisites ){ // Parcours de la collection (foreach dans d'autres langages)
    System.out.println(uneVisite.getHeure()) ; // Affichage de la date de chaque visite
}
```

Document 13 : Scénario pour la réalisation des tests unitaires de la classe GestionTournee

Le scénario de la tournée de tests a été élaboré en concertation avec le chef de projet.

L'inséminateur M. Ferdinand Petit effectue une tournée le 06 mai 2020. Il commence chez M. Georges Duboeuf à 9 heures chez lequel sont pratiquées 3 inséminations et 1 échographie. La tournée se poursuit à 10h30 chez Mme Marguerite Cow afin de réaliser 1 insémination.

Tests de la méthode CATournee() à effectuer :

- Vérification que le montant total à facturer initial est nul ;
- Vérification du montant total à facturer à l'issue de la première visite ;
- Vérification du montant total à facturer final de la tournée.

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS	SESSION 2020
U5 : Production et fourniture de services informatiques	Durée : 4 heures
Code sujet : SI5SLAM	Page 16 sur 18

Document 14 : Documentation de l'outil *JUnit*

L'outil *JUnit* utilise les annotations pour définir les méthodes de test et les configurer. Le tableau suivant fournit la liste des annotations utilisées dans le projet :

Annotation	Description
@Test	marque une méthode comme étant une méthode de test
@Before	exécutée avant chaque test, elle est utilisée pour préparer l'environnement de test (par exemple : lecture de données, initialisation d'objets, etc.)

Assertions

L'outil *JUnit* fournit des méthodes statiques pour tester certaines conditions via la classe `Assert`. Les méthodes d'assertion commencent toutes par `assert`. Elles permettent de spécifier le message d'erreur, le résultat attendu et le résultat effectif. Une méthode d'assertion compare la valeur effective retournée par un test à la valeur attendue, levant une exception de type `AssertionException` si la comparaison échoue.

Le tableau suivant fournit un aperçu des méthodes utilisables. Les paramètres entre crochets [] sont facultatifs et de type `String`.

Statement	Description
<code>assertTrue([message,] boolean condition)</code>	Checks that the boolean condition is true.
<code>assertFalse([message,] boolean condition)</code>	Checks that the boolean condition is false.
<code>assertEquals([message,] expected, actual)</code>	Tests that two values are the same. Note: for arrays the reference is checked not the content of the arrays.
<code>assertNull([message,] object)</code>	Checks that the object is null.
<code>assertNotNull([message,] object)</code>	Checks that the object is not null.

Exemple de classe de test avec utilisation d'assertions

```
public class MyClassTest {  
  
    private MyClass objet; // MyClass est testée  
  
    @Before{  
    public void setUp()           // Méthode setup() automatiquement exécutée  
        objet = new MyClass();   // avant chaque appel de la méthode de test  
    }  
  
    @Test  
    public void multiplicationOfZeroIntegersShouldReturnZero() {  
        // assert statements  
        assertEquals("10 x 0 doit retourner 0", 0, objet.multiplier(10, 0));  
        assertEquals("0 x 10 doit retourner 0", 0, objet.multiplier(0, 10));  
        assertEquals("0 x 0 doit retourner 0", 0, objet.multiplier(0, 0));  
    }  
}
```

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS	SESSION 2020
U5 : Production et fourniture de services informatiques	Durée : 4 heures
Code sujet : SI5SLAM	Page 17 sur 18

Document 15 : Retour sur investissement (ROI en anglais) : rappels

Le retour sur investissement est un ratio qui permet de calculer le pourcentage de gain à attendre d'un investissement par rapport à la mise de départ.

On rappelle la formule générale du retour sur investissement :

$$ROI = (gain - coût de l'investissement) / coût de l'investissement$$

La durée de rentabilité est le temps nécessaire pour que la mise de départ puisse être récupérée soit un retour sur investissement égal à zéro.

Document 16 : Éléments du devis pour le calcul du coût de mise en place des nouvelles fonctionnalités de *LogiSemin* et de *MobiSemin*

1. Prix d'une journée facturée à Coopain par qualification

- a. Chef de projet / Analyste / Expert (CAE) 600 € HT
- b. Analyste programmeur (AP) 400 € HT

2. Estimations du temps nécessaire pour le développement des fonctionnalités de *LogiSemin* et *MobiSemin*

Applications	Travaux à effectuer	Qualification requise	Nombre de jours
LogiSemin et MobiSemin	Analyse fonctionnelle et reporting	CAE	4
LogiSemin	Modification de la base de données	CAE	2
LogiSemin	Développement Véhicule-Catégorie	AP	1
LogiSemin	Développement Véhicule-Suivi du kilométrage	AP	2
LogiSemin	Développement Véhicule-Parc	AP	2
LogiSemin	Développement Véhicule-Saisies	AP	1
LogiSemin	Développement Bilan inséminateur	AP	2
MobiSemin	Développement Gestion des kilomètres	AP	4
MobiSemin	Développement script de synchronisation	AP	1
LogiSemin et MobiSemin	Tests unitaires	AP	3